## Tilburg University

## Integrality and cutting planes in semidefinite programming approaches for combinatorial optimization

de Meijer, Frank
DOI:
10.26116/dpxe-ha35

Publication date:
2023

Document Version
Publisher's PDF, also known as Version of record

Link to publication in Tilburg University Research Portal

Citation for published version (APA):
de Meijer, F. (2023). Integrality and cutting planes in semidefinite programming approaches for combinatorial optimization. [Doctoral Thesis, Tilburg University]. CentER, Center for Economic Research.
https://doi.org/10.26116/dpxe-ha35

## General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal


## Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

## Integrality and Cutting Planes in Semidefinite Programming Approaches for Combinatorial Optimization

FRANK DE MEIJER


TILBURG UNIVERSITY

# Integrality and Cutting Planes in Semidefinite Programming Approaches for Combinatorial Optimization 


#### Abstract

Proefschrift ter verkrijging van de graad van doctor aan Tilburg University op gezag van de rector magnificus, prof. dr. W.B.H.J. van de Donk, in het openbaar te verdedigen ten overstaan van een door het college voor promoties aangewezen commissie in de Aula van de Universiteit op


vrijdag 24 november 2023 om 10:00 uur

> door

Frank Johannes Jacobus de Meijer
geboren te Zundert

| Promotores: | prof. dr. ir. Renata Sotirov (Tilburg University) |
| :--- | :--- |
|  | prof. dr. Dion Gijswijt (Delft University of Technology) |
| Promotiecommissie: | prof. dr. Miguel Anjos (University of Edinburgh) |
|  | prof. dr. ir. Christoph Helmberg (Chemnitz University of Technology) |
|  | prof. dr. Etienne de Klerk (Tilburg University) |
|  | prof. dr. Monique Laurent (Tilburg University) |
|  | dr. mr. Sven Polak (Tilburg University) |

Integrality and Cutting Planes in Semidefinite Programming Approaches for Combinatorial Optimization

Copyright (C) 2023 Frank de Meijer, The Netherlands. All rights reserved. No parts of this thesis may be reproduced, stored in a retrieval system or transmitted in any form or by any means without permission of the author. Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd, in enige vorm of op enige wijze, zonder voorafgaande schriftelijke toestemming van de auteur.

## Acknowledgements

Besides the joy I experienced during the overall PhD process, the past years have been quite a rollercoaster and it would not have been possible without the support of several people.

First and foremost, I would like to express my greatest gratitude to my supervisor, Renata Sotirov. It feels like a long time ago that we started working together, where you were the one who interested and encouraged me into the world of academics. Since that time, I have been blessed with your wide expertise, clear guidance, patience and stimulating attitude. Not to forget also the assistance in various side matters related to the PhD that you have provided me with. But above all, your positive approach made the time working together feel like a pure joy. I am already missing our lively office hours. Thank you so much for everything you have done over the years and I am sure we will continue on this path in whatever way in the future.

Also, I would like to express my sincere gratitude to my co-supervisor, Dion Gijswijt. Thanks for your valuable feedback on the thesis as a whole, and, in particular, on the part we worked on together. It has been a pleasure to study this exciting project alongside the 'regular' work and your clear feedback and knowledge were really appreciated. I am happy to continue as colleagues in my new job in Delft. Besides, I would also like to thank Miguel Anjos, Christoph Helmberg, Etienne de Klerk, Monique Laurent and Sven Polak for taking part in my PhD committee and providing me with lots of possible improvements. Given the extensive length of the thesis, an additional thanksgiving is in place.

A special thanks to my other co-authors: Angelika Wiegele, Shudian Zhao and Melanie Siebenhofer. These collaborations have been very educational and the visit to the Alpen-Adria-Universität Klagenfurt is among the highlights of the PhD phase. I would also like to thank Henry Wolkowicz and Tyler Weames for giving me a warm welcome during my brief visit to the University of Waterloo.

Another thanks is in place for Brian Boot, who designed the cover of this thesis. You did a great job in putting a dry mathematical object in a suitable and abstract design.

The working days on campus would not have been so sparkling without the presence of my office mates over the past years: Pepijn, Yequi, Gizem, Fatih and Marco. Thanks for all the coffee breaks, the laughs, the small talks and the share of complaints behind the doors of K422. It will be different no longer experiencing your company on my adjacent desks anymore.

Also thanks to my other fellow PhD students over the years: Andries, Bas, Daniel, Davide, Dirck, Ernst, Farzan, Felix, Hadi, Henk, Jop, Judith, Lennart, Lorenz, Martijn, Meike, Melissa, Riley, Stefan, Tijn, Valentijn, Wout and Wouter. It has been a pleasure to share a wing in the Koopmans building with you!

Of course, there are also many people to thank outside the university context. After a day of struggling with an unfinished proof or a malfunctioning piece of code, it were mostly these people who provided me with sufficient distraction. Among others, I would like to thank Alex, Alex, Bram, Chiel, Guus, Jelle, Jordy, Kris, Lex, Maarten, Nina, Rik, Robin, Ruben and Sven. In particular, I would like to mention Maartje, with whom I can always
share my stories and difficulties, while enjoying nice activities to forget them. Thanks for being a great friend! Without mentioning everyone, I would also like to thank the people I regularly play music or theatre with, as well as my friends from buurtschap Tiggelaar, with whom I like to spend lots of summer hours working on our float for the Zundert flower parade.

A person who I should definitely not forget in this list, is my main fellow companion along the PhD journey. Lieke, I don't even know where to start in addressing my gratefulness to you. Initiated by the pandemical situation, we rapidly transformed from colleagues into friends. Without you, my PhD time would not have been half so amusing as it has been. The calls, the random office visits, the laughs, the diners, the jokes, the deep conversations, the galas. Thanks for your thorough support in good and in bad times. I am sure we will enjoy lots of 'warmste weken' in future!

Finally, writing this thesis would have been totally impossible without the loving support of my family, in particular my brothers, Corné and Joost, my sisters-in-law, Léonie and Tessa, and above all my parents, Jos and Carina. Often you were concerned about not being able to help whenever I was in a negative mood due to some research-related issue. However, I can assure you that the opposite is true and that your unlimited support has been much more valuable than you realize. The mental support, as well as your willingness to help me with all kind of household, practical or other side activities, made my mind free to complete this dissertation. For that reason, I would like to dedicate this thesis to my parents and grandparents.

Frank de Meijer
Rotterdam, September 2023

## Contents

List of Figures ..... ix
List of Tables ..... xi
List of Algorithms ..... XV
Notation ..... xvii
1 Introduction ..... 1
1.1 Background ..... 1
1.2 Integer programming ..... 2
1.2.1 The (mixed)-integer program ..... 2
1.2.2 Integer hull and integer polyhedra ..... 4
1.2.3 The cutting-plane method ..... 5
1.2.4 Branch-and-bound ..... 6
1.2.5 Branch-and-cut ..... 7
1.2.6 Lagrangian relaxation ..... 8
1.3 Semidefinite programming ..... 9
1.3.1 Positive semidefinite matrices ..... 10
1.3.2 The semidefinite program ..... 11
1.3.3 Facial geometry of the PSD cone ..... 13
1.3.4 The Douglas-Rachford splitting method and the alternating direction method of multipliers ..... 14
1.4 Binary quadratic programs ..... 18
1.4.1 The quadratic traveling salesman problem ..... 18
1.4.2 The quadratic cycle cover problem ..... 20
1.4.3 The graph partition problem ..... 22
1.4.4 The linearization problem of binary quadratic programs ..... 24
1.5 Outline of the thesis ..... 25
1.6 Contributions to the literature ..... 29
2 The QCCP: special cases and efficient bounds ..... 31
2.1 Introduction ..... 32
2.1.1 Main results and outline ..... 32
2.2 The quadratic cycle cover problem ..... 33
2.3 The QCCP linearization problem ..... 36
2.4 Linearization-based bounds for the QCCP ..... 40
2.4.1 Construction of linearization-based bounds ..... 40
2.4.2 Preliminary results ..... 43
2.5 Reformulated LBB approach ..... 43
2.6 The Gilmore-Lawler type bound ..... 46
2.6.1 The classical GL type bound ..... 46
2.6.2 The best Gilmore-Lawler type bound ..... 49
2.6.3 Preliminary results ..... 51
2.7 Other bounds for the QCCP ..... 51
2.8 Computational results ..... 53
2.9 Conclusions ..... 57
3 SDP-based bounds for the QCCP via CP-ALM and RL ..... 59
3.1 Introduction ..... 60
3.2 The quadratic cycle cover problem ..... 61
3.3 SDP relaxations for the QCCP ..... 63
3.4 Graph-dependent facial reduction ..... 67
3.4.1 Strict feasibility by facial reduction ..... 68
3.4.2 A polynomial time algorithm for the transformation matrix ..... 70
3.5 A cutting-plane augmented Lagrangian approach ..... 73
3.5.1 The alternating direction method of multipliers and the Peaceman- Rachford splitting method ..... 73
3.5.2 ADMM versus PRSM: preliminary results ..... 76
3.5.3 Projection onto a single BQP Cut ..... 76
3.5.4 Semi-parallel Dykstra's projection algorithm ..... 79
3.5.5 A cutting-plane augmented Lagrangian method ..... 81
3.6 SDP-based upper bounds for the QCCP ..... 82
3.6.1 Best Euclidean approximation ..... 83
3.6.2 Randomized undersampling ..... 83
3.6.3 Randomized oversampling ..... 84
3.6.4 Sequential Q-learning ..... 86
3.6.5 Hybrid upper bounding algorithm ..... 88
3.7 Computational results ..... 88
3.7.1 Design of computational experiments ..... 88
3.7.2 Results on lower bounds ..... 92
3.7.3 Upper bounds and overall results ..... 100
3.8 Conclusions ..... 101
4 Partitioning through projections: strong SDP bounds for large graph partition problems ..... 105
4.1 Introduction ..... 106
4.2 The graph partition problem ..... 108
4.2.1 DNN relaxations for the $k$-equipartition problem ..... 109
4.2.2 DNN relaxations for the bisection problem ..... 111
4.3 A cutting-plane augmented Lagrangian algorithm ..... 116
4.3.1 The alternating direction method of multipliers ..... 116
4.3.2 Clustered Dykstra's projection algorithm ..... 119
4.3.3 The cutting-plane ADMM ..... 121
4.4 Valid cutting planes, their projectors and separators ..... 124
4.4.1 Triangle inequalities for the $k$-EP ..... 124
4.4.2 Independent set inequalities for the $k$-EP ..... 125
4.4.3 BQP inqualities for the GBP ..... 126
4.5 Computational results ..... 128
4.5.1 Computational results for the $k$-EP ..... 129
4.5.2 Computational results for the GBP ..... 132
4.6 Conclusions ..... 133
5 The Chvátal-Gomory procedure for ISDPs with applications in combin- atorial optimization ..... 141
5.1 Introduction ..... 142
5.1.1 Main results and outline ..... 143
5.2 The Chvátal-Gomory procedure for ISDPs ..... 144
5.2.1 The Chvátal-Gomory procedure ..... 144
5.2.2 The elementary closure of spectrahedra ..... 145
5.2.3 The Chvátal rank of bounded spectrahedra ..... 151
5.2.4 The elementary closure of spectrahedra and total dual integrality ..... 155
5.2.5 Strengthened Chvátal-Gomory cuts ..... 162
5.3 A CG-based branch-and-cut algorithm for ISDPs ..... 163
5.3.1 Generic branch-and-cut framework ..... 163
5.3.2 A separation routine for binary SDPs ..... 165
5.4 The Chvátal-Gomory procedure for ISDP formulations of the QTSP ..... 167
5.4.1 The quadratic traveling salesman problem ..... 168
5.4.2 ISDP based on algebraic connectivity in directed graphs ..... 169
5.4.3 Chvátal-Gomory cuts for the ISDPs of the QTSP ..... 174
5.5 Computational results ..... 175
5.5.1 Design of computational experiments ..... 175
5.5.2 Comparison of approaches ..... 179
5.6 Conclusions ..... 184
6 On integrality in SDP for combinatorial optimization ..... 187
6.1 Introduction ..... 188
6.2 Theory on discrete PSD matrices ..... 191
6.2.1 Theory on PSD $\{0,1\}$-matrices ..... 191
6.2.2 Theory on PSD $\{ \pm 1\}$-matrices ..... 199
6.2.3 Theory on PSD $\{0, \pm 1\}$-matrices ..... 200
6.3 Binary quadratic optimization problems ..... 203
6.3.1 Binary quadratically constrained quadratic programs ..... 203
6.3.2 Binary quadratic matrix programs ..... 206
6.4 Problem-specific formulations ..... 212
6.4.1 The QAP as a MISDP ..... 212
6.4.2 MISDP formulations of the graph partition problem ..... 214
6.4.3 MISDP formulations based on algebraic connectivity ..... 217
6.4.4 MISDP formulations beyond binarity ..... 219
6.5 Bounds by integer semidefinite programming duality ..... 222
6.5.1 Lagrangian duality theory for mixed-integer semidefinite programming ..... 222
6.5.2 A projected subgradient method for solving the Lagrangian dual ..... 225
6.5.3 Lagrangian dual bounds for binary SDPs and the max-cut problem ..... 227
6.5.4 Preliminary computational results for the max-cut problem ..... 231
6.6 Conclusions ..... 235
7 Exploiting symmetries in optimal quantum circuit design ..... 239
7.1 Introduction ..... 240
7.2 Nearest neighbour compliance problem ..... 245
7.2.1 Mathematical formulation of the NNCP ..... 245
7.2.2 The NNCP as a shortest path problem ..... 247
7.3 Symmetries in $X=(V, A)$ ..... 248
7.3.1 Automorphism group of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$ ..... 249
7.3.2 Automorphism group of $X$ ..... 251
7.3.3 Orbit and orbital structure of group action on $X$ ..... 254
7.4 Symmetry reduction for the NNCP ..... 260
7.4.1 Reduced LP formulation ..... 260
7.4.2 Reduced combinatorial formulation ..... 264
7.4.3 Symmetry-reduced NNCP algorithm ..... 266
7.5 Special coupling graphs ..... 266
7.5.1 Cycle graph $C_{N}$ ..... 267
7.5.2 Biclique graph $K_{M, N}$ and star graph $K_{1, N}$ ..... 268
7.5.3 Lattice graph $G_{N, N}$ ..... 269
7.6 Computational results ..... 270
7.6.1 Design of computational experiments ..... 270
7.6.2 Results on RevLib instances ..... 272
7.6.3 Results on random instances ..... 273
7.7 Conclusions ..... 277
A Appendices ..... 279
A. 1 Dykstra's parallel projection algorithm ..... 279
A. 2 Additional computational results for Chapter 4 ..... 281
A. 3 Derivation of subtour elimination constraints as CG cuts ..... 286
A.3.1 Ordinary subtour elimination constraint ..... 286
A.3.2 Cut-set subtour elimination constraints ..... 286
A.3.3 Merged subtour elimination constraint ..... 287
A.3.4 Strengthened subtour elimination constraints of size two ..... 287
A.3.5 Strenghtened subtour elimination constraints ..... 288
A. 4 The symmetric quadratic traveling salesman problem ..... 289
A. 5 Additional computational results for Chapter 5 ..... 290
A. 6 Quantum gates and their two-qubit decompositions ..... 296
Scientific summary ..... 301
Wetenschappelijke samenvatting ..... 303
Bibliography ..... 305

## List of Figures

1.1 Color chart of distribution of four research themes over the different chapters. ..... 26
2.1 Group consisting of $n-1$ cells corresponding to facility 1. ..... 34
2.2 Connection between two cells of group $i$ and $j$. ..... 35
2.3 Simplified overview of $G$ for $n=4$. ..... 36
3.1 Example of graph $G$ and its bipartite representation $B(G)$. ..... 71
3.2a Lower bounds for the ADMM (dashed) and the PRSM (solid) for full test set. ..... 77
3.2b Lower bounds for the ADMM (dashed) and the PRSM (solid) on average. ..... 77
3.3 Schematic decision tree corresponding to cycle-building phase of a single agent. ..... 87
3.4 Behaviour of the PRSM and the CP-ALM for different values of numCuts for instance ER_4. ..... 91
3.5 Normalized lower bounds (averaged over different number of cuts) for three instances with different numbers of arcs. Right figure shows zoomed plot including a threshold at 0.996. ..... 91
3.6 Boxplot showing the quality of lower bounds MILP, LBB1, RLT1 and $S D P_{S 2}$ on reload instances. ..... 93
5.1 Spectrahedra $P$ and $Q$ defined in Example 5.4. Their corresponding integer hulls are given by the dark gradient areas. ..... 147
5.2 Spectrahedron $P$ in $\mathbb{R}^{3}$ defined in Example 5.5. ..... 147
5.3 Spectrahedron $P$, its integer hull $P_{I}$ and its elementary closure $\mathrm{cl}_{C G}(P)$. ..... 151
5.4 Simplified example of strengthened separation routine on spectrahedron $P$ from Example 5.4. The dotted line shows an eigenvalue cut (5.26) separating $\hat{x}$ from $P$, the solid line shows a CG cut (5.27) separating $\hat{x}$ from $\mathrm{cl}_{C G}(P)$, where $S=\mathbb{Z}^{m}$. ..... 165
5.5 Optimal tours of two turn instances: the TSPLIB instance 'kn57' ( $n=57$ ) and the instance 'grid1' $(n=430)$. Each square in (b) represents a vertex in the grid graph. ..... 179
5.6 Computation times versus instance size for the bioinformatics classes 'bma' (top), 'map' (middle) and 'ml' (bottom). The computation times are given on a logarithmic scale. ..... 181
5.7 Boxplots showing the computation times for the reload instances for different values of $n$ and $p$, accumulated over the reload class and the number of colors $c$. We omit the results of SCIP-SDP, since these computation times are several magnitudes larger. ..... 182
6.1 Overview of various exact formulations of binary quadratic program and their relaxations. A double arrow ( $\Longleftrightarrow$ ) denotes equivalence between the formulations, while a solid arrow $(\rightarrow)$ denotes that the formulation is relaxed from the former to the latter. The sets $\mathcal{X}, \mathcal{X}_{M I L P}$ and $\mathcal{X}_{M I S D P}$ are defined by nonconvex integer constraints, while $\overline{\mathcal{X}}_{\text {MILP }}$ and $\overline{\mathcal{X}}_{\text {MISDP }}$ are convex relaxations.
6.2 Overview of average characteristics of Lagrangian dual bounds for 13 rudy instance classes of Biqmac Library [364]. ..... 235
7.1 Example of an (abstract) quantum circuit (left) with four gates that acts on four qubits. The quantum circuit cannot be compiled on the coupling graph $G$ (middle). By inserting an appropriate SWAP gate, the circuit can be compiled on $G$ (right). ..... 245
7.2 Graphical overview of orbital structure within a subgraph $H^{k}$. Each line type (solid, dotted, dashed and curled) corresponds to another orbital. Case I (left): $B_{\tau_{1}}$ is trivial. Case II (middle): $B_{\tau_{1}}$ is nontrivial and the orbital of $\sigma_{1}$ under $B_{\tau_{1}}$ contains $\sigma_{2}$. Case III (right): $B_{\tau_{1}}$ is nontrivial, but the orbital of $\sigma_{1}$ under $B_{\tau_{1}}$ only consists of $\sigma_{1}$. ..... 257
7.3 Four axes of reflection in $G_{N, N}$. ..... 269
7.4 Overview of total average computation times (construction + solution time) of random instances with respect to $n$ and $m$. Each data point displays the average over 5 randomly generated instances of that type. ..... 275
A. 1 Overview of commonly used two-qubit quantum gates. ..... 297
A. 2 Decomposition of multiple-control Toffoli gate with two controls and a single target qubit. ..... 297
A. 3 Decomposition of Peres gate on three qubits. ..... 298
A. 4 Decomposition of Fredkin gate (controlled swap gate) with one control qubit. ..... 298
A. 5 Decomposition of Fredkin gate (controlled swap gate) with two control qubits. ..... 298
A. 6 Decomposition of multiple-control Toffoli gate with three controls and a single target qubit. ..... 299
A. 7 Decomposition of multiple-control Toffoli gate with four controls and a single target qubit. ..... 299

## List of Tables

2.1 Polyhedral descriptions of the sets $\tau_{1}(Q), \tau_{2}(Q)$ and $\tau_{3}(Q)$. ..... 42
2.2 Bounds and computation times in seconds of linearization-based bounds on Erdős-Rényi instances. ..... 44
2.3 Bounds and computation times in seconds of GL type bounds on Erdős-Rényi instances. ..... 51
2.4 Bounds and computation times in seconds of $R L T 1, C G, L B B 1$ and $R G L$ on small Erdős-Rényi instances. ..... 55
2.5 Bounds and computation times in seconds of $R L T 1, C G, L B B 1$ and $R G L$ on small Manhattan instances. ..... 55
2.6 Bounds and computation times in seconds of $R L T 1, C G, L B B 1$ and $R G L$ on small Angle-distance instances. ..... 55
2.7 Bounds and computation times in seconds of $G L, M I L P, L B B 1$ and $L B B 1^{\text {skew }}$ on large Erdős-Rényi instances. ..... 56
2.8 Bounds and computation times in seconds of $G L, M I L P$ and $L B B 1$ on large Manhattan instances. ..... 56
2.9 Bounds and computation times in seconds of $G L, M I L P, L B B 1$ and $L B B 1^{\text {skew }}$ on large Angle-distance instances. ..... 57
3.1 Three types of valid squared linear constraints for $\mathcal{F}_{\text {basic }}$. ..... 64
3.2 Overview of operators and their definitions. ..... 75
3.3 Comparison of different bounds for reload instances. ..... 93
3.4 Computation times in seconds, average residuals and number of iterations for reload instances. ..... 94
3.5 Comparison of different bounds for Erdős-Rényi instances. ..... 96
3.6 Comparison of computation times (in seconds) for Erdős-Rényi instances. ..... 97
3.7 Comparison of residuals and total number of cuts for Erdős-Rényi instances. ..... 98
3.8 Comparison of different bounds for Manhattan instances. ..... 99
3.9 Comparison of computation times, average residuals and total number of ad- ded cuts for Manhattan instances. ..... 100
3.10 Statistics on performance of upper bounds and average gaps on total test set. ..... 101
3.11 Overview of best lower bounds, best hybrid and nonhybrid upper bounds and their relative gaps for all instances. ..... 103
4.1 Overview of operators and their definitions. ..... 119
4.2 Comparison of different lower bounds. ..... 131
4.3 Instances considered in [193, Table 9], $k=2$. ..... 135
4.4 Large $G$ and $U$ graphs from [228], $k=2$. ..... 136
4.5 Instances considered in [21], $k \in\{3,4,5,6\}$. ..... 136
4.6 Two-dimensional spinglass graphs, $k \in\{3,4,5\}$. ..... 137
4.7 Three-dimensional spinglass graphs, $k \in\{3,4,5\}$. ..... 137
4.8 Large $G$ and $U$ graphs from [228], $k=4$ ..... 138
4.9 Large $G$ and $U$ graphs from [228], $k=5$ ..... 139
4.10 Instances considered in [193, Table 9], $m=\left(m_{1}, n-m_{1}\right)$. ..... 139
4.11 Large $G$ graphs from [228] and two-dimensional spinglass graphs, $m=\left(m_{1}, n-\right.$ $m_{1}$ ). ..... 140
5.1 Five types of subtour elimination constraints for the QTSP that can be ob- tained as (strengthened) CG cuts of $\left(I S D P_{1}\right)$ and/or $\left(I S D P_{2}\right)$. The third column describes which (in)equalities and dual multipliers are used to con- struct the inequality. ..... 176
5.2 Summary table of the performance on the bioinformatics instances per setting and per instance type. The best performing setting per row is given in bold. ..... 180
5.3 Overview of average computation times for the reload instances. Each row provides averages of 30 instances, namely 10 random instances for each value of $c=5,10,20$. ..... 183
5.4 Computation times and number of branching nodes for the TSPLIB instances. ..... 184
5.5 Computation times and number of branching nodes for the grid instances. ..... 184
6.1 Average Lagrangian dual bounds and average computation times of various implementations of the projected subgradient algorithm for the rudy instances of the Biqmac Library [364]. Lagrangian dual bounds are shown for $p=5$. Each instance class consists of 10 randomly generated instances of the given type. ..... 234
6.2 Average Lagrangian dual bounds and average computation times of various implementations of the projected subgradient algorithm for the Beasley in- stances of the Biqmac Library [364]. Each instance class consists of 10 ran- domly generated instances of the given type ..... 236
7.1 Overview of the orders of quotients $V^{k} / G_{X}, A^{k} / G_{X}$ and $D^{k} / G_{X}$ in terms of the cardinality of $B_{\tau}$. ..... 259
7.2 Summary of NNCP symmetry reduction characteristics for a set of special coupling graphs. ..... 268
7.3 Results on the 'RevLib' instances on the cyclic coupling graph. We compare the performance of the base model $(S P P)$ with the reduced model $(R S P P)$. Times are clocktimes given in seconds. ..... 273
7.4 Results on the 'RevLib' instances on the star coupling graph. We compare the performance of the base model $(S P P)$ with the reduced model $(R S P P)$. Times are clocktimes given in seconds. ..... 274
7.5 Results on the 'RevLib' instances on the biclique coupling graph. We compare the performance of the base model $(S P P)$ with the reduced model $(R S P P)$. Times are clocktimes given in seconds. ..... 275
7.6 Results on the random instances on the star coupling graph. Each row shows the average values over 5 randomly generated instances. Times are clocktimes given in seconds. ..... 276
7.7 Results on the random instances on the biclique coupling graph. Each row shows the average values over 5 randomly generated instances. Times are clocktimes given in seconds. ..... 276
A. 1 Performance of CP-ALM using cyclic and parallel Dykstra on a test set of 6 Erdős-Rényi instances with maxIter $=1000$, maxTotalIter $=5000$, numCuts $=150$ and all other parameters the same as given in Section 3.7. ..... 280
A. 2 Large $G$ and $U$ graphs from [228] (Maxineq=3n). ..... 283
A. 3 Graphs considered in the paper of Hager, Phan and Zhang [193] (Maxineq=3n). ..... 285
A. 4 Computation times (s) for bioinformatics instances from the 'bma' class. ..... 291
A. 5 Number of branching nodes for bioinformatics instances from the 'bma' class. ..... 291
A. 6 Computation times (s) for bioinformatics instances from the 'map' class. ..... 292
A. 7 Number of branching nodes for bioinformatics instances from the 'map' class. ..... 292
A. 8 Computation times (s) for bioinformatics instances from the 'ml' class ..... 293
A. 9 Number of branching nodes for bioinformatics instances from the 'ml' class. ..... 293
A. 10 Computation times of the reload instances averaged over 10 generated in- stances for given values of $n, p$ and $c$. ..... 294
A. 11 Number of branching nodes for the reload instances averaged over 10 generated instances for given values of $n, p$ and $c$ ..... 295
A. 12 Computation times of SEC-simple and SEC on 72 additional reload instances for given values of $n, p$ and $c$. ..... 296

## List of Algorithms

1.1 Generic cutting-plane method ..... 5
1.2 Generic branch-and-bound method ..... 7
1.3 Generic branch-and-cut method with lazy constraints ..... 8
2.1 LBB Reformulation algorithm ..... 46
3.1 Computation of transformation matrix $W$ ..... 73
3.2 CP-ALM ..... 81
3.3 Randomized undersampling for the QCCP ..... 84
3.4 Randomized oversampling for the QCCP ..... 85
3.5 Sequential Q-learning for QCCP ..... 89
4.1 CP -ADMM for the GPP ..... 122
4.2 Probabilistic separation method for independent set inequalities ..... 126
5.1 CG-based B\&C algorithm for solving $\left(D_{I S D P}\right)$ ..... 165
6.1 Projected subgradient algorithm for solving (6.42) ..... 226
7.1 Symmetry-reduced NNCP algorithm ..... 267

## Notation

## Sets

$S \cup T \quad$ union of $S$ and $T$
$S \sqcup T \quad$ disjoint union of $S$ and $T$
$S \cap T \quad$ intersection of $S$ and $T$
$S \subseteq T \quad S$ is a subset of $T$
$S \subsetneq T \quad S \subseteq T$, but $S \neq T$
$|S| \quad$ cardinality of $S$
$\delta_{S} \quad$ indicator function of $S$, i.e., $\delta_{S}(x)=0$ if $x \in S$ and $\delta_{S}(x)=\infty$ otherwise
$[n] \quad$ set of integers $\{1, \ldots, n\}$
$\mathbb{R} \quad$ set of real numbers
$\mathbb{Q} \quad$ set of rational numbers
$\mathbb{Z} \quad$ set of integer numbers
$\mathbb{R}_{+} \quad$ the set of nonnegative real numbers (similar definitions for $\mathbb{Q}_{+}$and $\mathbb{Z}_{+}$)
$\mathbb{R}^{n} \quad$ the subspace of $n$-dimensional vectors of real numbers (similar definitions for $\mathbb{Q}^{n}$ and $\mathbb{Z}^{n}$ )
$\mathbb{R}^{n \times m} \quad$ the subspace of $n \times m$ matrices of real numbers (similar definitions for $\mathbb{Q}^{n \times m}$ and $\mathbb{Z}^{n \times m}$ )
$\mathbb{P}(S) \quad$ power set of $S$, i.e., collection of all subsets of $S$
$\mathcal{S}^{n} \quad$ subspace of $n \times n$ symmetric matrices
$\mathcal{S}_{+}^{n} \quad$ cone of $n \times n$ positive semidefinite matrices, i.e., $\left\{X \in \mathcal{S}^{n}: X \succeq \mathbf{0}\right\}$
$\mathcal{S}_{++}^{n} \quad$ cone of $n \times n$ positive definite matrices, i.e., $\left\{X \in \mathcal{S}^{n}: X \succ \mathbf{0}\right\}$
$\mathcal{N}_{+}^{n} \quad$ cone of $n \times n$ symmetric nonnegative matrices, i.e., $\mathcal{S}^{n} \cap \mathbb{R}_{+}^{n \times n}$
$\mathcal{D N N}_{+}^{n} \quad$ cone of $n \times n$ doubly nonnegative matrices, i.e., $\mathcal{S}_{+}^{n} \cap \mathcal{N}_{+}^{n}$
$\mathcal{C P}{ }_{+}^{n} \quad$ cone of $n \times n$ completely positive matrices, i.e., $\left\{X \in \mathcal{S}^{n}: X=B B^{\top}, B \geq\right.$ $0\}$
$\mathcal{K}^{*} \quad$ dual cone of cone $\mathcal{K}$
$\mathcal{N}_{S}(x) \quad$ normal cone of a set $S$ at point $x$
$\operatorname{conv}(S) \quad$ convex hull of elements in $S$
cone $(S) \quad$ conical hull of elements in $S$
$\operatorname{dim}(P) \quad$ dimension of a convex set $P$
$\operatorname{int}(S) \quad$ interior of a set $S$
$\operatorname{ri}(S) \quad$ relative interior of a set $S$
$\mathrm{cl}_{C G}(S) \quad$ elementary closure of closed convex set $S$
$\Pi_{n} \quad$ set of $n \times n$ permutation matrices

## Linear algebra

$\mathbf{I}_{n}$ or $\mathbf{I} \quad n \times n$ identity matrix or identity matrix of fitting size
$\mathbf{J}_{n}$ or $\mathbf{J} \quad n \times n$ matrix of ones of matrix of ones of fitting size
$\mathbf{1}_{n}$ or $\mathbf{1} \quad n \times 1$ column vector of ones or vector of ones of fitting size
$\mathbf{0}_{n}$ or $\mathbf{0} \quad n \times 1$ zero vector or zero vector of fitting size
$\mathbf{e}_{i} \quad i$ th column of identity matrix
$\mathbb{1}_{S} \quad$ binary indicator vector of set $S$, where entry $i$ is one if $i \in S$ and zero otherwise
$\mathbf{E}_{i j} \quad$ matrix $\mathbf{e}_{i} \mathbf{e}_{j}^{\top}$
$M_{i,:} \quad i$ th row of matrix $M$ as a row vector
$M_{:, i} \quad i$ th column of matrix $M$ as a column vector
$M[S] \quad$ principal submatrix of $M$ induced by indices in $S$
$\operatorname{Col}(M) \quad$ column space of matrix $M$
$\operatorname{Nul}(M) \quad$ null space of matrix $M$
$\mathcal{W}^{\perp} \quad$ orthogonal complement of linear subspace $\mathcal{W}$
$\operatorname{Span}\{S\} \quad$ linear subspace spanned by the elements in the set $S$
$M \oplus N \quad$ direct sum of $M$ and $N$
$M \otimes N \quad$ Kronecker product of $M$ and $N$
$M \circ N \quad$ Hadamard product of $M$ and $N$
$\operatorname{supp}(x) \quad$ support of vector $x$, i.e., the set of indices of nonzero elements in $x$
$\operatorname{tr}(M) \quad$ trace of square matrix $M$
$\operatorname{diag}(\cdot) \quad$ operator $\mathcal{S}^{n} \rightarrow \mathbb{R}^{n}$ that maps square matrix $M$ to a vector containing the diagonal elements of $M$
$\operatorname{Diag}(\cdot) \quad$ adjoint operator of $\operatorname{Diag}(\cdot)$
$\operatorname{vec}(\cdot) \quad$ operator $\mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m n}$ that maps a matrix to a vector by concatenation of its columns
$\operatorname{svec}(\cdot) \quad$ operator $\mathcal{S}^{n} \rightarrow \mathbb{R}^{\frac{1}{2}\left(n^{2}+n\right)}$ that maps a matrix to a vector containing the columnwise concatenation of its upper-triangular elements with weight two on the off-diagonal elements and weight one on the diagonal elements
$\operatorname{triu}(\cdot) \quad$ operator $\mathcal{S}^{n} \rightarrow \mathbb{R}^{\frac{1}{2}\left(n^{2}+n\right)}$ that maps a matrix to a vector containing the columnwise concatenation of its upper-triangular elements

## Semidefinite programming

$\langle M, N\rangle \quad$ trace inner product of matrices $M$ and $N$, i.e., $\operatorname{tr}\left(M^{\top} N\right)$
$\|M\|_{F} \quad$ Frobenius inner product of $M$, i.e., $\sqrt{\operatorname{tr}\left(M^{\top} M\right)}$
$M \succeq \mathbf{0} \quad M$ is positive semidefinite
$M \succeq N \quad$ shorthand for $M-N \succeq \mathbf{0}$
$M \succ \mathbf{0} \quad M$ is positive definite
$M \varsubsetneqq \mathbf{0} \quad M$ is positive semidefinite, but $M \neq \mathbf{0}$

## Graph theory

$G=(V, E) \quad$ undirected graph
$G=(N, A) \quad$ directed graph
$L_{G}$ or $L \quad$ Laplacian or weighted Laplacian matrix of $G$, where $G$ is omitted if the graph follows from the context
$B(G) \quad$ bipartite representation of a directed graph $G$

| $\delta^{+}\left(i, A^{\prime}\right)$ | set of arcs in $A^{\prime} \subseteq A$ leaving node $i$ in directed graph |
| :--- | :--- |
| $\delta^{-}\left(i, A^{\prime}\right)$ | set of arcs in $A^{\prime} \subseteq A$ entering node $i$ in directed graph |
| $\delta^{+}(i)$ | set of arcs leaving node $i$ in directed graph, i.e., shorthand for $\delta^{+}(i, A)$ |
| $\delta^{-}(i)$ | set of arcs entering node $i$ in directed graph, i.e., shorthand for $\delta^{-}(i, A)$ |
| $\delta^{+}(S, T)$ | set of arcs leaving $S$ and entering $T$ in directed graph |
| $\delta^{-}(S, T)$ | set of arcs leaving $T$ and entering $S$ in directed graph |
| $\delta^{+}(S)$ | set of arcs leaving $S$ and entering $A \backslash S$ in directed graph, i.e., shorthand for |
|  | $\delta^{+}(S, A \backslash S)$ |
| $\delta^{-}(S)$ | set of arcs leaving $A \backslash S$ and entering $S$ in directed graph, i.e., shorthand for |
|  | $\delta^{-}(S, A \backslash S)$ |
| $e^{+}$ | starting node of arc $e$ in directed graph |
| $e^{-}$ | ending node of arc $e$ in directed graph |
| $C_{n}$ | cycle graph on $n$ vertices |
| $K_{n}$ | complete graph on $n$ vertices |
| $K_{n, m}$ | complete bipartite graph or biclique graph on vertex sets of sizes $n$ and $m$ |
| $T_{n}$ | spanning tree on $n$ vertices |
| $G_{n, m}$ | lattice graph on $n \times m$ vertices |

## Group theory

$\mathrm{id}_{G}$ or id
$\operatorname{Sym}(S)$
$\mathbb{S}_{n}$
$\mathcal{D}_{2 n}$
$\tau(S)$
$\tau^{-1}(S)$
$\mathbb{S}_{n}(S)$
$x \circ g$
$g \circ x$
$\operatorname{Stab}(x)$
$\operatorname{Orb}(x)$
$X / G$
$X^{g}$
$C(H)$
$Z(G)$
identity element of group $G$
symmetric group of finite set $S$
symmetric group of $[n]$, i.e., $\operatorname{Sym}([n])$
dihedral group of order $2 n$, i.e., symmetries of the $n$-gon
image of set $S \subseteq[n]$ under permutation $\tau \in \mathbb{S}_{n}$, i.e., $\{\tau(s): s \in S\}$
preimage of set $S \subseteq[n]$ under permutation $\tau \in \mathbb{S}_{n}$, i.e., $\left\{\tau^{-1}(s): s \in S\right\}$
setwise stabilizer of $S$ under group action of $\mathbb{S}_{n}$, i.e., $\left\{\tau \in \mathbb{S}_{n}: \tau(S)=S\right\}$
image of right group action of $g$ on $x$
image of left group action of $g$ on $x$
stabilizer of $x \in X$ under (left) action of group $G$, i.e., $\{g \in G: g \circ x=x\}$
orbit of $x \in X$ under (left) action of group $G$, i.e., $\{g \circ x: g \in G\} \subseteq X$ quotient of $X$ under $G$, i.e., set of all orbits of $X$ under the (left) action of $G$ set of fixed points of $X$ under (left) action of $g$, i.e., $\{x \in X: g \circ x=x\}$ centralizer subgroup of subset $H$ of group $G$, i.e., $\{g \in G: g h=h g \forall h \in H\}$
center of a group $G$, i.e., $\{g \in G: g h=h g \forall h \in G\}$, i.e., centralizer of $G$

## Number theory

$\operatorname{gcd}(c) \quad$ greatest common divisor of the entries in integer vector $c \in \mathbb{Z}^{n}$
floor operator that maps a number to the largest integer smaller than its value
$\{ \pm 1\}$ ceil operator that maps a number to the smallest integer larger than its value the set $\{+1,-1\}$

## 1 <br> Introduction

### 1.1 Background

Given the optimized society we live in nowadays, the need for analytical methods for optimal decision-making has become indispensable. One of the most powerful tools in solving complex decision-making problems is mathematical optimization. Mathematical optimization is the subfield of mathematics concerned with finding the optimal values of decision variables such that an objective function is optimized under the presence of a set of constraints. Many real-life problems can be tackled using mathematical optimization, and hence it has played a groundbreaking role in many areas, including engineering, logistics and finance. The first modern literature on mathematical optimization finds its roots dating back to World War II and appeared on linear optimization problems, where both the objective function and the constraint functions are linear in terms of the decision variables. Since then, many other types of problems have been studied in the literature.

An important distinction within the field of mathematical optimization is that between continuous and discrete optimization problems. Whereas a continuous variable is allowed to take any value within a certain range, discrete variables may only take distinct countable values. Both variable types arise naturally in applications, e.g., temperature in physical models is measured as a continuous quantity, while the choice whether or not to open a warehouse in a supply chain model is best reflected by a discrete decision variable. This distinction has a great impact on the design of solution methods. Algorithms designed to solve purely continuous problems often rely on the use of first- or higher-order derivatives, whereas for discrete problems, these notions do not exist or are not directly applicable. Solution techniques for discrete problems, however, more than once exploit approaches from continuous optimization, resulting in both areas being highly entangled.

In this work we focus on combinatorial optimization problems, which deal with the optimization over a finite set of solutions. Combinatorial optimization problems are inherently discrete in nature. Most of these can be modeled as optimization problems over the set of integers, leading to so-called integer programs. Integer programming is a well-established
subfield of mathematical optimization, for which many solution approaches have been suggested. Fundamental techniques, such as the use of cutting planes and branching strategies, acquired a key role in most state-of-the-art optimization solvers.

Another subfield of mathematical optimization is semidefinite programming. Semidefinite programming deals with the optimization of a linear function over the cone of positive semidefinite matrices under the presence of affine constraints. Capturing linear programming as its subclass, semidefinite programming can be seen as the natural extension of linear programming, allowing for a richer source of theory and applications. At the same time, SDPs can be solved in polynomial time up to any fixed precision under some rather mild conditions, e.g., via interior-point methods [11, 290]. This combination between generality and efficiency makes semidefinite programming a fruitful research area that has drawn much attention in recent years. In particular, semidefinite programs have proven themselves useful in their ability to provide strong relaxations for hard combinatorial optimization problems.

In this thesis, we study the interplay between integer and semidefinite programming. A particular focus is put on SDP models derived from combinatorial optimization problems. Some of these problems play a primary role throughout this work.

### 1.2 Integer programming

In this section we briefly consider some characteristics of general integer programming problems and recap some of the established solution strategies. We restrict ourselves to the concepts that make their appearance in this thesis. Lacking fundamental approaches such as the Dantzig-Wolfe approach and the branch-and-price method, this chapter should therefore not be regarded as a comprehensive introduction to integer programming methods.

### 1.2.1 The (mixed)-integer program

Integer programming deals with mathematical optimization problems where the variables are restricted to be integer-valued. If only a part of the variables is integer-valued and the other variables are continuous, we refer to the problem as a mixed-integer programming problem. Besides the integrality restriction, an integer program (IP) or mixed-integer program (MIP) can embrace any type of structure in the objective or constraint functions. A general MIP can be cast as follows:

$$
\begin{align*}
\min & f(x) \\
\text { s.t. } & x \in F  \tag{1.1}\\
& x_{i} \in \mathbb{Z} \quad \forall i \in \mathcal{J}
\end{align*}
$$

where $x \in \mathbb{R}^{n}$ is a $n$-dimensional decision vector, $f: \mathbb{R}^{n} \rightarrow \mathbb{R}$ is the objective function, $F$ is a closed set in $\mathbb{R}^{n}$ and $\mathcal{J} \subseteq[n]$ contains the indices of the integer variables in the model. We further assume that the function $f$ is continuous and that the lower and upper bounds on the variables, if any, are included in the set $F$.

There exist many subclasses of mixed-integer programs. In its most general form, when $f$ is nonlinear and the set $F$ has no particular structure, we refer to (1.1) as a mixed-integer nonlinear program (MINLP). When $F$ is a convex set and $f$ is a convex function, the problem (1.1) is known as a convex mixed-integer nonlinear program (CMINLP) [51]. This does not imply that the CMINLP itself is convex, which is obviously not the case due to the integer variables, but the problem obtained after relaxing the integrality constraints is a
convex programming problem. Although convex programming is generally hard, i.e., there exist classes of convex programs that are still $\mathcal{N} \mathcal{P}$-hard, several important subclasses of convex programming can be solved efficiently up to any given precision [54]. A notable characterization of this dichomoty is given by the polynomial equivalence between the optimization problem and the separation problem of convex programs established by Grötschel et al. [183] (under some conditions). When the set $F$ in a CMINLP is conic representable, i.e., all nonlinearities in $F$ arise from the membership of the variables in a convex cone, we sometimes refer to it as a conic mixed-integer nonlinear program. Examples include the mixed-integer second-order cone program (MISOCP), see e.g., [107], and the mixed-integer semidefinite program (MISDP), see e.g., [156].

When the objective function $f$ is linear and the set $F$ is a polyhedron, (1.1) boils down to a mixed-integer linear program (MILP). Among all subclasses of integer programming described above, MILPs have undoubtly received most attention in the literature, pioneered by the early work of Gomory [179]. This has led to an enormous progress in the ability to solve MILPs to optimality. Many general-purpose software packages, such as Gurobi [189], CPLEX [220] and Mosek [284], are able to solve large-scale MILPs containing thousands of variables and constraints. Many excellent textbooks have been written on solution approaches and applications of MILPs, see e.g., Conforti et al. [81] and Schrijver [330]. MILPs are particularly useful in the formulation of combinatorial optimization problems. These problems aim to find an optimal solution among a finite set of (combinatorial) objects, e.g., paths in a network or schedules in an assignment problem. Many of these problems are $\mathcal{N} \mathcal{P}$-hard, implying that mixed-integer linear programming and mixed-integer nonlinear programming are $\mathcal{N} \mathcal{P}$-hard as well.

Other optimization problems that are extensively studied in the literature are the programs of the form (1.1) where the function $f$ is quadratic, i.e., of the form $f(x)=x^{\top} Q x+$ $c^{\top} x+d$ for some $Q \in \mathcal{S}^{n}\left(:=\left\{X \in \mathbb{R}^{n \times n}: X=X^{\top}\right\}\right), c \in \mathbb{R}^{n}$ and $d \in \mathbb{R}$. When $F$ is a polyhedron, the resulting program is denoted as a quadratic program (QP). When the set $F$ is of the form

$$
F=\left\{x \in \mathbb{R}^{n}: g_{i}(x) \leq 0 \quad \forall i \in[k]\right\},
$$

where $g_{i}: \mathbb{R}^{n} \rightarrow \mathbb{R}, i \in[k]$, are quadratic functions, we refer to it as a quadratically constrained quadratic program (QCQP). Such problems are in general $\mathcal{N} \mathcal{P}$-hard. If the functions $f, g_{1}, \ldots, g_{k}$ are convex, however, we can solve a QCQP as a semidefinite program.

In many practical applications, the integer variables are required to be in $\{0,1\}$ (or $\{ \pm 1\}$ ). Such problems are denoted as binary programming problems. Of main importance in this thesis is the binary quadratic program, which is considered in more detail in Section 1.4.

For the remaining part of this section, we assume (1.1) to be a convex integer nonlinear program with a linear objective function. That is, we consider a program of the form

$$
\begin{align*}
\min & c^{\top} x \\
\text { s.t. } & x \in F \quad  \tag{1.2}\\
& x_{i} \in \mathbb{Z} \quad \forall i \in[n],
\end{align*}
$$

where $c \in \mathbb{R}^{n}$ and $F \subseteq \mathbb{R}^{n}$ is a closed convex set. The majority of the problems considered in this thesis are of this form. Observe that the linearity of the objective function in a convex integer nonlinear program is nonrestrictive. Namely, if $f$ is convex in (1.1), we can embed the program into $\mathbb{R}^{n+1}$ by adding the variable $t$ and the constraint $f(x)-t \leq 0$,
where we minimize the linear function $f^{\prime}(x, t)=t$. We consider (1.2) to be purely integer to keep the representation simple. All concepts described in the next subsections can be easily generalized to the mixed-integer case. Finally, we assume that (1.2) is feasible and that the minimum to (1.2) and its continuous relaxation after dropping the constraints $x_{i} \in \mathbb{Z}, i \in[n]$, are attained.

### 1.2.2 Integer hull and integer polyhedra

Let us denote the feasible set of (1.2) by

$$
P:=\left\{x \in \mathbb{Z}^{n}: x \in F\right\} .
$$

Since the objective function of (1.2) is linear, optimizing over $P$ is equivalent to optimizing over its convex hull. The convex hull of the integer points in $F$ is also called the integer hull $F_{I}$ of $F$, i.e.,

$$
F_{I}:=\operatorname{conv}(P) .
$$

In case an explicit description of $F_{I}$ is known and optimizing over it is tractable, this can be exploited in practical algorithms to solve the integer problem.

One research stream in integer programming concerns the identification of facet-defining inequalities of $F_{I}$. Most effort in this direction is made in the case of integer linear problems. If $F$ is a rational polyhedron, it follows from Meyer's theorem [280] that $F_{I}$ is again a polyhedron. A polyhedron whose vertices are all integer is called an integer polyhedron. Obviously, if $F$ is an integer polyhedron, it must be equal to its integer hull. There exist several well-known sufficient conditions for a polyhedron to be integer, among which total unimodularity is a fundamental one.

Definition 1.1. A matrix $A \in \mathbb{R}^{m \times n}$ is called totally unimodular if any square submatrix of $A$ has determinant 0,1 or -1 .

A totally unimodular matrix $A$ by definition has entries in $\{0, \pm 1\}$. The following proposition provides a characterization of totally unimodular matrices related to integer polyhedra.

Proposition 1.2 ([209]). A matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular if and only if the polyhedron $\left\{x \in \mathbb{R}^{n}: A x \leq b, x \geq \mathbf{0}\right\}$ is integral for every $b \in \mathbb{Z}^{m}$.

An implication of Proposition 1.2 is that if $A$ is totally unimodular and $b \in \mathbb{Z}^{m}$, the polyhedron $\left\{x \in \mathbb{R}^{n}: A x \leq b\right\}$ is also integer, i.e., the nonnegativity of $x$ can be relaxed while keeping the integrality property. The perfect matching polytope in bipartite graphs is an example of a polytope that has a totally unimodular constraint matrix [210], and hence is integer. Therefore, the perfect matching problem in bipartite graphs can be solved as a linear programming problem.

Another common sufficient condition for the detection of integer polyhedra is total dual integrality [121].

Definition 1.3. Let $A \in \mathbb{Q}^{m \times n}, b \in \mathbb{Q}^{m}$. A system $A x \leq b$ is called totally dual integral (TDI) if for each $c \in \mathbb{Z}^{n}$, the dual of minimizing $c^{\top} x$ over $A x \leq b$, i.e., $\max \left\{b^{\top} y: y \leq\right.$ $\left.\mathbf{0}, A^{\top} y=c\right\}$ has an integer optimum solution $y$, if it has an optimal solution.

The following result states that total dual integrality implies that the polyhedron induced by $A x \leq b$ is integer if the vector $b$ has integer entries.

Proposition 1.4 ([121]). If $A x \leq b$ is totally dual integral and $b \in \mathbb{Z}^{m}$, then $\left\{x \in \mathbb{R}^{n}\right.$ : $A x \leq b\}$ is an integer polyhedron.

It has to be noted that TDI is an algebraic property rather than a geometric one. In fact, it is well-known that any rational polyhedron has a totally dual integral representation [171]. Moreover, it is also shown in [171] that each integer polyhedron has a totally dual integral representation $A x \leq b$ with $b$ integer.

Proposition 1.2 and 1.4 can be combined to show the following result on the relationship between total unimodularity and total dual integrality.
Proposition 1.5 ([209]). A matrix $A \in \mathbb{Z}^{m \times n}$ is totally unimodular if and only if the linear system $A x \leq b, x \geq \mathbf{0}$ is totally dual integral for each integral vector $b \in \mathbb{Z}^{m}$.

There is a wealth of literature on integer polyhedra and, more general, on polyhedral combinatorics, see e.g., the textbook by Schrijver [331] and the references therein.

Integer hulls in nonlinear convex sets have received much less attention in the literature. When the set $F$ is unbounded, it is not even known whether the convex hull of the integer points in $F$ can be described by a polyhedron. Sufficient conditions for this property are studied in [97]. Other results in this direction can be found in, e.g., [27, 65, 288].

### 1.2.3 The cutting-plane method

One of the most central tools in integer programming is the use of so-called cutting planes. These cutting planes can be used to cut off noninteger points from the geometric object that we are optimizing over.

In a natural first attempt to solve (1.2), one would drop the integrality constraints entirely and minimize $c^{\top} x$ with respect to $x \in F$. As indicated in Section 1.2.1, we assume this minimum to be attained. Let $x^{*}$ denote an optimal solution of this relaxed problem, and assume that $x^{*}$ is an extreme point of $F$. In case $x^{*}$ has integer entries, the point $x^{*}$ is also feasible to (1.2) and must therefore be optimal. If not, it follows from $x^{*}$ being an extreme point of $F$, that $x^{*} \notin F_{I}$. Now, our goal is to find an inequality that all integer solutions in (1.2) satisfy, but $x^{*}$ does not. Whenever $F_{I}$ is closed, we know that such an inequality exists due to the separating hyperplane theorem. Suppose we are in possession of an efficient separation routine that provides us a cut separating $x^{*}$ from $F_{I}$. By adding this cut to the constraint set of $F$ and solve $c^{\top} x$ over this renewed feasible set, we obtain a tighter relaxation of (1.2). This procedure is now repeated until $x^{*}$ is integer. This approach is known as a cutting-plane method, for which a generic framework is given in Algorithm 1.1.

```
Algorithm 1.1 Generic cutting-plane method
    Initialize \(\mathcal{F} \equiv F\).
    Solve \(\min \left\{c^{\top} x: x \in \mathcal{F}\right\}\) and let \(x^{*}\) denote an extreme point of \(\mathcal{F}\) that is optimal to this problem.
    If \(x^{*}\) is integer, Stop. Otherwise, find an inequality that separates \(x^{*}\) from the integer solutions in (1.2)
    and add this inequality to the constraint set of \(\mathcal{F}\). Return to Step 2.
```

It depends on the problem how the continuous problem in step 2 of Algorithm 1.1 is solved, e.g., by interior-point methods [11, 290] or the simplex method [92].

The cutting-plane algorithm requires a separation routine that for each point $x^{*}$ determines an inequality separating $x^{*}$ from the integer hull $F_{I}$. Based on the polynomial
equivalence between separation and optimization [183], we cannot always expect an efficient separation routine to exist. Different strategies have been proposed in the literature, including general-purpose cuts and problem-specific cuts.

General-purpose cutting planes are inequalities that can be constructed from the input data and can be applied to a wide variety of problems. A famous class of linear cutting planes for integer programs are the Chvátal-Gomory cuts [77, 179]. Suppose that $c^{\top} x \leq d$ is an inequality that is valid for $F$, i.e., $F \subseteq\left\{x \in \mathbb{R}^{n}: c^{\top} x \leq d\right\}$, with $c$ having integer entries such that $\operatorname{gcd}(c)=1$. In that case, we know that $c^{\top} x \in \mathbb{Z}$ for all integer points $x \in F$. If $d$ is noninteger, we can strengthen the cut by rounding down $d$ to the nearest integer. This leads to the cut

$$
c^{\top} x \leq\lfloor d\rfloor
$$

which is known as a Chvátal-Gomory cutting plane. By construction, $c^{\top} x \leq\lfloor d\rfloor$ contains all integer points in $F$, while it might cut off some fractional points in $F$. Gomory [179] was the first to apply these types of cutting planes in a framework for solving MILPs, and since then similar approaches have been applied to a wide variety of other problem classes, see e.g., $[68,88,89,98,359]$.

Complementary to these general-purpose cuts, many cutting-plane methods rely on the use of inequalities that follow from the structure of the underlying problem. An in-depth analysis of the structure of $F_{I}$ often leads to deeper cuts than general-purpose separation routines can provide. In the case of MILPs, much research has been performed on identifying facet-defining inequalities of the integer hull $F_{I}$ of certain problems. For example, an explicit polyhedral description on the perfect 2-matching polytope is derived by Edmonds [120] and succesfully exploited in a cutting-plane framework in [182].

Observe that the cuts that can be used in Algorithm 1.1 are not limited to linear cuts. Nonlinear cuts that have been considered in integer nonlinear programming include conic cuts [24] and quadratic cuts [68].

Remark 1.6. Instead of requiring that $x^{*}$ is an extreme point and checking whether $x^{*}$ is integer in Algorithm 1.1, we can also allow the optimal solution $x^{*}$ to be not necessarily an extreme point of $\mathcal{F}$ and check whether $x^{*} \in F_{I}$ in step 3 of Algorithm 1.1.

### 1.2.4 Branch-and-bound

Branch-and-bound (B\&B) is an approach in which the integer problem is divided into smaller subproblems, where a bounding procedure is used to prevent an exhaustive search on the entire search space to be performed. There exist many different implementations of the branch-and-bound method, see the survey of [283]. Here we present a generic branch-andbound framework for integer problems of the form (1.2). For a more detailed description of the branch-and-bound method, we refer to the textbooks [372, 373].

Let $U B$ denote an upper bound on (1.2), either derived by some heuristic method, or by simply setting $U B=\infty$. Let $\mathcal{L}$ denote a collection of feasible sets (i.e., subsets of $F$ ), each corresponding to a subproblem defined over that feasible region. Initially, we set $\mathcal{L}=\{F\}$. Now, we iteratively take a set $F^{i}$ from $\mathcal{L}$ and solve the program $\min \left\{c^{\top} x: x \in F^{i}\right\}$. Let $x^{*}$ be an optimal solution to this program, which we assume to be an extreme point of $F^{i}$. Obviously, $L B\left(F^{i}\right):=c^{\top} x^{*}$ provides a lower bound on the optimal objective value of all integer points in $F^{i}$. If $L B\left(F^{i}\right)>U B$, it follows that the optimal solution to (1.2) is not found in $F^{i}$, hence we disregard this subproblem. Otherwise, two scenarios are possible.

If $x^{*}$ is integer, then it is feasible for (1.2). We remove $F^{i}$ from $\mathcal{L}$, and, in case $x^{*}$ is the best solution found so far, we update $U B$ and let $x^{\text {best }}:=x^{*}$. If $x^{*}$ is fractional, we divide $F^{i}$ further into multiple subproblems. More precisely, we define $k$ sets $F_{1}^{i}, \ldots, F_{k}^{i} \subseteq F^{i}$ which are pairwise disjoint such that each integer point in $F^{i}$ is in exactly one of these sets. Now, we remove $F^{i}$ from $\mathcal{L}$, while we add to $\mathcal{L}$ the sets $F_{1}^{i}, \ldots, F_{k}^{i}$. We repeat the procedure until $\mathcal{L}$ is empty. An optimal solution to (1.2) is guaranteed to be found, since each integer solution in $F$ is either considered in one of the subproblems, or disregarded by a certificate for nonoptimality. In order to speed up the $B \& B$ procedure, primal heuristics are often implemented throughout the search, in order to obtain better upper bounds. Algorithm 1.2 shows a generic framework of the $\mathrm{B} \& \mathrm{~B}$ approach for integer problems.

```
Algorithm 1.2 Generic branch-and-bound method
    Obtain upper bound \(U B\) and initial feasible solution \(x^{\text {best }}\) to the problem. Initialize collection of
    feasible regions \(\mathcal{L}=\{F\}\).
    while \(\mathcal{L}\) is nonempty do
        Choose \(F^{i}\) from \(\mathcal{L}\) and remove it from \(\mathcal{L}\).
        Solve \(\min \left\{c^{\top} x: x \in F^{i}\right\}\) and let \(x^{*}\) denote an extreme point of \(F^{i}\) that is optimal to this problem.
        Compute the lower bound \(L B\left(F^{i}\right):=c^{\top} x^{*}\).
        if \(L B\left(F^{i}\right)>U B\) then
            Disregard the subproblem and continue to next iteration of while-loop.
        else
            if \(x^{*}\) is an integer solution then
                    Update \(U B\) and \(x^{\text {best }}\) if \(L B\left(F^{i}\right)<U B\).
            else
                    Divide \(F^{i}\) into \(k\) pairwise disjoint subsets \(F_{1}^{i}, \ldots, F_{k}^{i}\) with \(k \geq 2\), such that each integer
                    point in \(F^{i}\) is in exactly one set \(F_{1}^{i}, \ldots, F_{k}^{i}\).
                    \(\mathcal{L} \leftarrow \mathcal{L} \cup\left\{F_{1}^{i}, \ldots, F_{k}^{i}\right\}\).
            end if
        end if
        (Optional:) Run a primal heuristic on \(F^{i}\) to possibly improve \(U B\) and \(x^{\text {best }}\).
    end while
```

Many important implementation details are left out of Algorithm 1.2. Of major importance is the way to partition a subproblem. A common approach is to partition on a fractional entry in $x^{*}$. In other words, if $x_{i}^{*} \notin \mathbb{Z}$, we split $F^{i}$ into

$$
\begin{equation*}
F_{1}^{i}:=F^{i} \cap\left\{x \in \mathbb{R}^{n}: x_{i} \leq\left\lfloor x_{i}^{*}\right\rfloor\right\} \quad \text { and } \quad F_{2}^{i}:=F^{i} \cap\left\{x \in \mathbb{R}^{n}: x_{i} \geq\left\lceil x_{i}^{*}\right\rceil\right\} . \tag{1.3}
\end{equation*}
$$

It remains to decide on which entry $i$ to branch when $x^{*}$ has more than one fractional entry. Another decision choice concerns the order in which the sets in $\mathcal{L}$ need to be considered.

Similar to what is pointed out in Remark 1.6, instead of requiring that the optimal solution $x^{*}$ is an extreme point of $F^{i}$, we can relax that condition and check whether $x^{i} \in F_{I}^{i}$ rather than checking integrality.

### 1.2.5 Branch-and-cut

The branch-and-cut ( $\mathrm{B} \& \mathrm{C}$ ) approach can be seen as an extension of branch-and-bound, see e.g., [372, 373]. A typical branch-and-bound implementation starts from the relaxation $F$ in the root node of the branching tree and is only altered by the addition of the partition cuts of the form (1.3). A B\&C algorithm considers strengthenings of the relaxations in the subproblems by the utilization of cutting planes. Suppose a parent node in the branching tree is described by the feasible set $F^{i}$. Now, let $F_{1}^{i}=F^{i} \cap\left\{x \in \mathbb{R}^{n}: x_{i} \leq\left\lfloor x_{i}^{*}\right\rfloor\right\}$ be one of its child nodes. In the $\mathrm{B} \& \mathrm{~B}$ algorithm, one of the future iterations regards the minimization
of $c^{\top} x$ over $F_{1}^{i}$. In a B\&C algorithm, however, we perform an additional step and try to tighten the set $F_{1}^{i}$ by the addition of cutting planes that are valid for $\left(F_{1}^{i}\right)_{I}$ (but which might not be valid for all integer solutions in $F$ ). These cuts are transferred from parent to child nodes in the branching tree, thus augmenting the relaxations of all child subproblems. A strictly better lower bound is obtained when the added cutting plane(s) cut(s) off all the optimal solutions to $\min \left\{c^{\top} x: x \in F_{1}^{i}\right\}$, provided that these are noninteger. By adding deep cutting planes, $\mathrm{B} \& \mathrm{C}$ algorithms potentially show an accelerated performance compared to $\mathrm{B} \& \mathrm{~B}$ algorithms. Cuts that are added can be problem-specific or obtained from a generic separation routine.

A particular implementation of branch-and-cut relies on the use of so-called lazy constraints. These are inequalities that belong to the original problem formulation (1.2), but are only enforced as additional cuts during the solution process. Let us consider the following reformulation of (1.2):

$$
\begin{align*}
\min & c^{\top} x \\
\text { s.t. } & x \in F:=F_{0} \cap\left\{x \in \mathbb{R}^{n}: g_{j}(x) \leq 0 \quad \forall j \in[k]\right\}  \tag{1.4}\\
& x_{i} \in \mathbb{Z} \quad \forall i \in[n] .
\end{align*}
$$

Suppose that the inequalities $g_{j}(x) \leq 0, i \in[k]$, considerably complexify the optimization over $F$, e.g., because of their advanced structure or because there are very many of them. In that case, it might be advantageous to first ignore them in a branch-and-cut algorithm. Once an integer solution $x^{*}$ is found that is feasible for $F_{0}$, we check whether $g_{j}\left(x^{*}\right) \leq 0$ does hold for all $j \in[k]$. If so, the solution is in fact feasible for the entire problem (1.4). If not, we have identified a $j^{*}$ such that $g_{j^{*}}\left(x^{*}\right)>0$. Hence, we now add $g_{j^{*}}(x) \leq 0$ to the constraint sets of all the child subproblems that will follow. Whenever no more violated lazy constraints can be found and all subproblems have been considered, optimality is guaranteed.

Algorithm 1.3 shows a generic branch-and-cut method incorporating lazy constraints. The additional check of the lazy constraints is done in a so-called callback procedure, which is an independent subroutine that can be called during the search.

```
Algorithm 1.3 Generic branch-and-cut method with lazy constraints
    Initialize \(\mathcal{F}=F_{0}\).
    \(\mathbf{B \& C}\) procedure: Start or continue the branch-and-cut algorithm for solving
    \(\min \left\{c^{\top} x: x \in \mathcal{F} \cap \mathbb{Z}^{n}\right\}\) incorporating the callback function below at each node in the branching
    tree.
    Callback procedure:
    if an integer point \(x^{*} \in \mathcal{F}\) is found then
        if there exists \(j^{*} \in[k]\) with \(g_{j}\left(x^{*}\right)>0\) then
            Add \(g_{j^{*}}(x) \leq 0\) to the constraint set defining \(\mathcal{F}\).
        else
            Use \(x^{*}\) to possibly update \(U B\) and cut off other subproblems in the branching tree.
        end if
    end if
    Return to Step 2
```


### 1.2.6 Lagrangian relaxation

Branch-and-bound and branch-and-cut methods rely on the use of relaxations in the subproblems. Instead of using the relaxation $F$ of $P$ obtained by dropping all integrality constraints,
we here present an approach to obtain possibly tighter relaxations for integer problems based on Lagrangian duality theory. Lagrangian approaches in integer programming were first applied by Held and Karp [197, 198], see also [140] and the references therein.

Although the addition of integrality constraints to the set $F$ increases the complexity of solving (1.2), in some situations the integrality constraints are tractable with respect to some of the constraints in $F$. More precisely, suppose $\mathcal{A}: \mathbb{R}^{n} \rightarrow \mathbb{R}^{m}$ is an operator (not necessarily linear) and $b \in \mathbb{R}^{m}$ such that we can rewrite the set $F$ as follows:

$$
F:=F_{0} \cap\left\{x \in \mathbb{R}^{n}: \mathcal{A}(x) \leq b\right\} .
$$

Moreover, let $P_{0}$ denote the set of integer points in $F_{0}$, i.e.,

$$
P_{0}:=\left\{x \in \mathbb{Z}^{n}: x \in F_{0}\right\} .
$$

We assume that optimization over the set $P_{0}$ is tractable, e.g., because there is an available explicit description of the integer hull of $F_{0}$. However, adding the inequalities $\mathcal{A}(x) \leq b$ to the constraint set of $P_{0}$ makes the problem difficult to solve. We therefore dualize these constraints, by introducing a vector $\lambda \in \mathbb{R}_{+}^{m}$ of dual or Lagrange multipliers. This yields

$$
\begin{aligned}
g(\lambda):=\min & c^{\top} x+\lambda^{\top}(\mathcal{A}(x)-b) \\
\text { s.t. } & x \in P_{0} .
\end{aligned}
$$

For an optimal solution $x^{*}$ to (1.2), we have $c^{\top} x^{*} \geq c^{\top} x^{*}+\lambda^{\top}\left(\mathcal{A}\left(x^{*}\right)-b\right) \geq g(\lambda)$ for all $\lambda \in \mathbb{R}_{+}^{m}$. Therefore, the function $g(\lambda)$ provides a lower bound on the optimal value of (1.2) for all feasible dual multipliers. The Lagrangian dual problem is obtained by maximizing this lower bound:

$$
\begin{aligned}
z_{L D}:=\sup & g(\lambda) \\
& \text { s.t. }
\end{aligned} \quad \lambda \geq \mathbf{0} .
$$

Weak duality holds by construction, i.e., we have $z_{L D} \leq z_{I P}$, where $z_{I P}$ denotes the optimum to (1.2). In general, strong duality between (1.2) and its Lagrangian dual problem does not hold. For many problem classes, it can be shown that the Lagrangian dual provides a tighter relaxation than the standard continuous relaxation (i.e., optimizing over the set $F$ ). For example, in the case of integer linear programs, it is known that when $F \backslash\left(\operatorname{conv}\left(P_{0}\right) \cap\left\{x \in \mathbb{R}^{n}: \mathcal{A}(x) \leq b\right\}\right)$ is nonempty, then there exist objective functions $c^{\top} x$ such that $z_{L D}$ is strictly stronger than the standard LP relaxation, see [169]. This leads to successful implementations when incorporated in a branching setting, see e.g., [140].

Since the evaluation of $g(\lambda)$ might be time-consuming, typical methods to solve the Lagrangian dual problem involve decomposition approaches, e.g., subgradient algorithms [190], the Dantzig-Wolfe approach [250] and cutting-plane methods [268]. Most literature in this direction focuses on mixed-integer linear problems. For these problems, it can be shown that $z_{L D}$ is the solution to another linear programming problem with a large number of constraints, see [169].

### 1.3 Semidefinite programming

An introduction to the field of semidefinite programming (SDP) is provided in this section. Starting from basic results on positive semidefinite matrices, we present the semidefinite
program in its standard primal and dual form and consider the facial geometry of the cone of positive semidefinite matrices. We finalize the section by considering two first-order methods for solving semidefinite programs.

### 1.3.1 Positive semidefinite matrices

Let us recap the basic theory on positive semidefinite matrices. All of the results in this section are classical and can be found in many textbooks on matrix theory, e.g., the book by Horn and Johnson [213].

Let $\mathcal{S}^{n}$ be the vector space of symmetric matrices of order $n$. A fundamental property of symmetric matrices is that a symmetric matrix possesses a set of $n$ eigenvectors that form an orthonormal basis of $\mathbb{R}^{n}$. This result is known as spectral decomposition.

Proposition 1.7 (Spectral decomposition). A real symmetric matrix $X \in \mathcal{S}^{n}$ can be decomposed as:

$$
X=\sum_{i=1}^{n} \lambda_{i} u_{i} u_{i}^{\top},
$$

where $\lambda_{1}, \ldots, \lambda_{n} \in \mathbb{R}$ are the eigenvalues of $X$ and $u_{1}, \ldots, u_{n} \in \mathbb{R}^{n}$ are the corresponding eigenvectors. These eigenvectors form an orthonormal basis of $\mathbb{R}^{n}$. In matrix notation, we have $X=P D P^{\top}$ where $D$ is a diagonal matrix that contains the eigenvalues of $X$ on the diagonal and $P$ is an orthogonal matrix with columns $u_{i}$.

A matrix $X \in \mathcal{S}^{n}$ is positive semidefinite (PSD) if

$$
v^{\top} X v \geq 0 \text { for all } v \in \mathbb{R}^{n},
$$

which is denoted by $X \succeq \mathbf{0}$. A matrix $X \in \mathcal{S}^{n}$ is positive definite if

$$
v^{\top} X v>0 \text { for all } v \in \mathbb{R}^{n} \backslash\left\{\mathbf{0}_{n}\right\}
$$

which is denoted by $X \succ \mathbf{0}$. Using this notation, we write $X \succeq Y($ resp. $X \succ Y)$ if $X-Y \succeq \mathbf{0}$ (resp. $X-Y \succ \mathbf{0}$ ).

It is well-known that we have various equivalent statements for a matrix $X$ to be positive semidefinite.

Proposition 1.8. The following are equivalent:
(i) A matrix $X \in \mathcal{S}^{n}$ is positive semidefinite;
(ii) All eigenvalues of $X$ are nonnegative, i.e., $X=\sum_{i=1}^{n} \lambda_{i} u_{i} u_{i}^{\top}$ with $\lambda_{i} \geq 0$ for all $i \in[n]$;
(iii) All principal minors of $X$ are nonnegative;
(iv) There exist vectors $v_{1}, \ldots, v_{n} \in \mathbb{R}^{k}$ for some positive integer $k$ such that $X_{i j}=v_{i}^{\top} v_{j}$ for all $i, j \in[n]$, called the Gram representation of $X$.

The second criterion can be extended to the positive definite case. That is, we have $X \succ 0$ if and only if all the eigenvalues of $X$ are strictly positive.

We let $\mathcal{S}_{+}^{n}$ denote the set of all positive semidefinite matrices of order $n$, i.e.,

$$
\mathcal{S}_{+}^{n}:=\left\{X \in \mathcal{S}^{n}: X \succeq \mathbf{0}\right\} .
$$

This set is a closed convex cone, which is known as the positive semidefinite cone, or PSD cone in short. Similarly, the set $\mathcal{S}_{++}^{n}$ consists of all positive definite matrices. The PSD cone is full-dimensional in $\mathcal{S}^{n}$ and its relative interior equals $\mathcal{S}_{++}^{n}$.

A useful property of positive semidefinite matrices is the Schur complement lemma.
Lemma 1.9 (Schur complement lemma). Let the matrix $X \in \mathcal{S}^{n}$ be defined as

$$
X=\left(\begin{array}{cc}
A & B \\
B^{\top} & C
\end{array}\right)
$$

where $A \in \mathcal{S}^{p}, C \in \mathcal{S}^{n-p}$ and $B \in \mathbb{R}^{p \times(n-p)}$. If $A$ is nonsingular, then $X \succeq \mathbf{0}$ if and only if $A \succeq \mathbf{0}$ and $C-B^{\top} A^{-1} B \succeq \mathbf{0}$.

Moreover, we state the following well-known property regarding the kernel of positive semidefinite matrices.

Proposition 1.10. Let $X \in \mathcal{S}_{+}^{n}$ and $v \in \mathbb{R}^{n}$. Then $v^{\top} X v=0$ if and only if $X v=\mathbf{0}$.
For any square matrix $X \in \mathbb{R}^{n \times n}$, the trace is defined as the sum of the diagonal entries of $X$, i.e., $\operatorname{tr}(X):=\sum_{i=1}^{n} X_{i i}$. It is well-known that $\operatorname{tr}(X Y)=\operatorname{tr}(Y X)$ holds for any $X, Y \in \mathbb{R}^{n \times n}$. Moreover, if $X$ has eigenvalues $\lambda_{1}, \ldots, \lambda_{n}$, then $\operatorname{tr}(X)=\sum_{i=1}^{n} \lambda_{i}$. The trace can be used to define an inner product on $\mathbb{R}^{n \times m}$, called the trace inner product. For any two matrices $X, Y \in \mathbb{R}^{n \times m}$ the trace inner product is defined as $\langle X, Y\rangle:=\operatorname{tr}\left(X^{\top} Y\right)=$ $\sum_{i=1}^{n} \sum_{j=1}^{m} X_{i j} Y_{i j}$. The following proposition states a well-known property regarding the trace inner product of two positive semidefinite matrices.

Proposition 1.11. Let $X, Y \in \mathcal{S}_{+}^{n}$. Then, $\langle X, Y\rangle \geq 0$. Moreover, $\langle X, Y\rangle=0$ if and only if $X Y=\mathbf{0}$.

There exist several important subsets of $\mathcal{S}_{+}^{n}$. A matrix $X$ which is PSD and entrywise nonnegative is called doubly nonnegative (DNN). The cone of all doubly nonnegative matrices of order $n$ is denoted by

$$
\mathcal{D N N}_{+}^{n}:=\mathcal{S}_{+}^{n} \cap \mathcal{N}_{+}^{n}
$$

where $\mathcal{N}_{+}^{n}$ is the cone of entrywise nonnegative matrices of order $n$. A PSD matrix of the form $X=B B^{\top}$ with $B \geq \mathbf{0}$ is called completely positive. The cone of completely positive matrices of order $n$ is denoted by

$$
\mathcal{C P}{ }_{+}^{n}:=\left\{X \in \mathcal{S}^{n}: X=B B^{\top}, B \geq \mathbf{0}\right\} .
$$

### 1.3.2 The semidefinite program

In this section we provide a brief introduction to the field of semidefinite programming. Many textbooks on this topic have been written, see e.g., Anjos and Lasserre [15] and Ben-Tal and Nemirovski [40].

A standard semidefinite program (SDP) in primal form is an optimization problem in which we minimize a linear function (with respect to the trace inner product) over the cone of positive semidefinite matrices under the presence of affine constraints. Let $C, A_{1}, \ldots, A_{m} \in \mathcal{S}^{n}$
and $b \in \mathbb{R}^{m}$, then the standard semidefinite primal problem equals:

$$
\begin{align*}
p^{*}=\inf _{X \in \mathcal{S}^{n}} & \langle C, X\rangle \\
\text { s.t. } & \left\langle A_{i}, X\right\rangle=b_{i} \text { for all } i \in[m],  \tag{1.5}\\
& X \succeq \mathbf{0} .
\end{align*}
$$

The feasible set of (1.5) is the intersection of the PSD cone with an affine subspace, which is known as a spectrahedron. Note that we write infimum instead of minimum, since the optimal value of a semidefinite problem might not be attained. In general, we can have $p^{*} \in \mathbb{R} \cup\{ \pm \infty\}$, where $p^{*}=\infty$ if (1.5) is infeasible and $p^{*}=-\infty$ if it is unbounded. The standard dual formulation of the SDP (1.5) equals:

$$
\begin{align*}
d^{*}=\sup _{y \in \mathbb{R}^{m}} & b^{\top} y \\
\text { s.t. } & C-\sum_{i=1}^{m} A_{i} y_{i} \succeq \mathbf{0} . \tag{1.6}
\end{align*}
$$

Again, we can have $d^{*} \in \mathbb{R} \cup\{ \pm \infty\}$, where $d^{*}=-\infty$ if (1.6) is infeasible and $d^{*}=\infty$ if it is unbounded.

We can show that we always have $d^{*} \leq p^{*}$, which is denoted by weak duality.
Proposition 1.12 (Weak duality). Let $X \in \mathcal{S}^{n}$ be feasible for (1.5) and $y \in \mathbb{R}^{m}$ be feasible for (1.6). Then $b^{\top} y \leq\langle C, X\rangle$. In particular, we have $d^{*} \leq p^{*}$.
The difference $p^{*}-d^{*}$ is called the duality gap between (1.5) and (1.6). When $d^{*}=p^{*}$, we say that strong duality holds. Strong duality can be verified by an optimality condition which is called complementary slackness.

Proposition 1.13 (Complementary slackness). Let $X \in \mathcal{S}^{n}$ be feasible for (1.5) and $y \in \mathbb{R}^{m}$ be feasible for (1.6). Then, $d^{*}=p^{*}$ if

$$
\begin{equation*}
X\left(C-\sum_{i=1}^{m} A_{i} y_{i}\right)=\mathbf{0} \tag{1.7}
\end{equation*}
$$

Conversely, if $d^{*}=p^{*}$ and the optima to (1.5) and (1.6) are attained by $X$ and $y$, respectively, then $X$ and $y$ satisfy (1.7).

Unlike linear programming, strong duality does not always hold in semidefinite programming. Conditions under which strong duality holds are in the literature known as constraint qualifications [227]. Among the most commonly used constraint qualifications in convex programming is Slater's condition.

Definition 1.14 (Slater's condition). The primal SDP (1.5) satisfies Slater's condition if there exists a matrix $X$ that is feasible for (1.5) with $X \succ \mathbf{0}$. Similarly, the dual SDP (1.6) satisfies Slater's condition if there exists a feasible solution $y \in \mathbb{R}^{m}$ with $C-\sum_{i=1}^{m} A_{i} y_{i} \succ \mathbf{0}$.

An SDP satisfying Slater's condition is called Slater feasible or strictly feasible. Strict feasibility is a sufficient condition for strong duality, as the following result states.
Proposition 1.15 (Strong Duality). Let $p^{*}$ and $d^{*}$ be as defined in (1.5) and (1.6), respectively. Then:

- If the primal problem (1.5) is bounded from below ( $p^{*}>-\infty$ ) and strictly feasible, then strong duality holds. Moreover, the dual problem (1.6) attains its supremum.
- If the dual problem (1.6) is bounded from above ( $d^{*}<\infty$ ) and strictly feasible, then strong duality holds. Moreover, the primal problem (1.5) attains its infimum.

If strict feasiblity does not hold for an SDP, facial reduction can be applied to obtain an equivalent strictly feasible SDP. Facial reduction is a method proposed by Borwein and Wolkowicz [52] that projects the feasible region of an SDP (in the form (1.5)) to the minimal face of $\mathcal{S}_{+}^{n}$ that contains it. More prerequisites on the facial geometry of the cone of positive semidefinite matrices is provided in Section 1.3.3.

Since the outcome of an SDP can be irrational even if all data matrices are rationalvalued, one cannot hope for a polynomial time algorithm that solves it exactly. However, given that Slater's condition holds for both the primal and its dual SDP, it is known that semidefinite programs can be solved up to any given precision in polynomial time by interiorpoint methods [11, 290]. We refer here to polynomial time in terms of the number of arithmetic operations, i.e., with respect to the real model of computation.

Semidefinite programs have proven themselves useful in providing strong relaxations for many combinatorial optimization problems. The standard approach to provide such a semidefinite programming relaxation is introduced in Lovász and Schrijver [259] and Shor [336].

### 1.3.3 Facial geometry of the PSD cone

At several occasions throughout this thesis, we apply facial reduction in order to make a semidefinite program strictly feasible. In this subsection we review some basic definitions and results regarding the faces of general convex cones and, in particular, the PSD cone. For a more in-depth overview on this matter, we refer the reader to [108].

Let $\mathcal{K}$ be a convex cone in a vector space accompanied by an inner product mapping $\langle\cdot, \cdot\rangle$. A convex cone $F \subseteq \mathcal{K}$ is called a face of $\mathcal{K}$ if the following holds:

$$
x, y \in \mathcal{K} \text { with } x+y \in F \quad \Longrightarrow \quad x, y \in F
$$

It follows from this definition that $\mathcal{K}$ and $\emptyset$ are faces of $\mathcal{K}$. We call a face $F$ of $\mathcal{K}$ proper if $F$ is not equal to $\mathcal{K}$ or $\emptyset$. Every proper face of $\mathcal{K}$ is disjoint from $\operatorname{ri}(\mathcal{K})$, the relative interior of $\mathcal{K}$. It is easy to verify that the intersection of two faces of $\mathcal{K}$ is again a face of $\mathcal{K}$. This justifies the following definition.

Definition 1.16. The minimal face of a convex cone $\mathcal{K}$ containing a set $S \subseteq \mathcal{K}$ is the intersection of all faces of $\mathcal{K}$ containing $S$ and is denoted by face $(S, \mathcal{K})$.

Besides the general definition of a face of a cone, it is in some cases possible to provide a dual description of a face. From the dual cone

$$
\mathcal{K}^{*}:=\{x:\langle x, y\rangle \geq 0 \text { for all } y \in \mathcal{K}\}
$$

of $\mathcal{K}$, one can verify that the set $v^{\perp} \cap \mathcal{K}$ is a face of $\mathcal{K}$ for every $v \in \mathcal{K}^{*}$, where $v^{\perp}$ denotes the orthogonal complement of $v$. A face of this type is called an exposed face.

Definition 1.17. A set $F=v^{\perp} \cap \mathcal{K}$ with $v \in \mathcal{K}^{*}$ is called an exposed face of $\mathcal{K}$. The element $v$ is called an exposing vector of $F$.

In general not all faces of a convex cone are exposed. However, one can show that any point in the relative boundary of a convex cone is contained in some proper exposed face. A convex cone $\mathcal{K}$ is called facially exposed if all faces of $\mathcal{K}$ are exposed.

Let us now consider the case $\mathcal{K}=\mathcal{S}_{+}^{n}$. It is well-known that the PSD cone is self-dual, i.e., $\left(\mathcal{S}_{+}^{n}\right)^{*}=\mathcal{S}_{+}^{n}$. It can be shown that the faces of $\mathcal{S}_{+}^{n}$ correspond to linear subspaces of $\mathbb{R}^{n}$, see e.g., [32].

Proposition 1.18 ([32]). Let $\mathcal{R}$ be a linear subspace of $\mathbb{R}^{n}$. Then, the set

$$
\begin{equation*}
F_{\mathcal{R}}=\left\{X \in \mathcal{S}_{+}^{n}: \operatorname{Col}(X) \subseteq \mathcal{R}\right\} \tag{1.8}
\end{equation*}
$$

is a face of $\mathcal{S}_{+}^{n}$. Conversely, any face of $\mathcal{S}_{+}^{n}$ can be written in this way for some linear subspace $\mathcal{R}$ of $\mathbb{R}^{n}$.

Since $\operatorname{Col}(X)^{\perp}=\operatorname{Nul}(X)$ for all $X \in \mathcal{S}_{+}^{n}$, we can equivalently write

$$
\begin{equation*}
F_{\mathcal{R}}=\left\{X \in \mathcal{S}_{+}^{n}: \operatorname{Nul}(X) \supseteq \mathcal{R}^{\perp}\right\} \tag{1.9}
\end{equation*}
$$

Moreover, it can be shown that for all $X$ in the relative interior of $F_{\mathcal{R}}$ we have $\operatorname{Col}(X)=\mathcal{R}$ and $\operatorname{Nul}(X)=\mathcal{R}^{\perp}$. A useful fact is that for any matrix $W \in \mathbb{R}^{n \times k}$ with $\operatorname{Col}(W)=\mathcal{R}$, we have

$$
F_{\mathcal{R}}=W \mathcal{S}_{+}^{k} W^{\top}
$$

see e.g., $[32,108]$. This relation implies that $F_{\mathcal{R}}$ is isomorphic to the PSD cone of order $k$. Combining this with Proposition 1.18, it follows that all faces of $\mathcal{S}_{+}^{n}$ are isomorphic to a PSD cone, i.e., the PSD cone is self-replicating.

Finally, it is known that the PSD cone is facially exposed, i.e., every face of $\mathcal{S}_{+}^{n}$ is exposed, as stated by the following proposition.

Proposition 1.19 ([298]). All faces $F_{\mathcal{R}}$ of $\mathcal{S}_{+}^{n}$ are exposed. That is, $F_{\mathcal{R}}=\left(U U^{\top}\right)^{\perp} \cap \mathcal{S}_{+}^{n}$ where $U$ is such that $\operatorname{Nul}(U)=\mathcal{R}$.

Proposition 1.19 implies that if a matrix $U U^{\top}$ exposes a face of $\mathcal{S}_{+}^{n}$, then this face can be written as $W \mathcal{S}_{+}^{k} W^{\top}$, where the columns of $W \in \mathbb{R}^{n \times k}$ form a basis for $\operatorname{Nul}(U)$.

### 1.3.4 The Douglas-Rachford splitting method and the alternating direction method of multipliers

Interior-point methods (IPMs) are currently seen as the state-of-the-art in solving SDPs and are incorporated in several commercial solvers. It is known, however, that IPMs can only handle SDPs of moderate sizes, due to the large requirement of computation time and memory when the number of variables and constraints increase. Thus, aiming at solving large-scale SDPs in practice, there is a desire for alternative methods that are considerably cheaper in computation. In this section we consider two first-order methods to solve SDPs, namely the Douglas-Rachford splitting method (PRSM) and the alternating direction method of multipliers (ADMM), which are known to be equivalent under certain conditions.

We consider the SDP in standard primal form (1.5). By capturing the affine constraints
into a polyhedral set $\mathcal{Y}$, the program can be reformulated as

$$
\begin{align*}
\inf _{X \in \mathcal{S}^{n}} & \langle C, X\rangle \\
\text { s.t. } & X \in \mathcal{Y}:=\left\{X \in \mathcal{S}^{n}:\left\langle A_{i}, X\right\rangle=b_{i} \text { for all } i \in[m]\right\}  \tag{1.10}\\
& X \succeq \mathbf{0} .
\end{align*}
$$

By using indicator functions, we can further rewrite (1.10) as an unconstrained program. For a given set $S$, let $\delta_{S}$ denote an indicator function such that $\delta_{S}(x)=0$ if $x \in S$ and $\delta_{S}(x)=\infty$ otherwise. Now, let us define the closed, proper and convex functions $f: \mathcal{S}^{n} \rightarrow \mathbb{R} \cup\{\infty\}$ and $g: \mathcal{S}^{n} \rightarrow \mathbb{R} \cup\{\infty\}$ as

$$
f(X):=\langle C, X\rangle+\delta_{\mathcal{Y}}(X) \quad \text { and } \quad g(X):=\delta_{\mathcal{S}_{+}^{n}}(X),
$$

respectively. Then, the program (1.10) is equivalent to

$$
\begin{equation*}
\inf _{X \in \mathcal{S}^{n}} f(X)+g(X) \tag{1.11}
\end{equation*}
$$

Many applications of practical interest can be cast as the minimization of the sum of two convex functions, e.g., problems in distributed optimization, compressed sensing and statistical learning, see [177] and the references therein. Consequently, a large number of algorithms for problems of this form have been proposed, among which the Douglas-Rachford splitting method (DRSM). The DRSM is originally proposed in [106] and, starting from the seminal work of Lions and Mercier [255], inspired a widespread related literature. We consider here its application to solve (1.11).

Let $\beta>0$ be a positive penality parameter, then the DRSM applied to (1.11) starts from a matrix $W^{0} \in \mathcal{S}^{n}$ and iteratively updates:
(DRSM)

$$
\left\{\begin{aligned}
Y^{p+1} & :=\operatorname{prox}_{\frac{1}{\beta} f}\left(W^{p}\right), \\
X^{p+1} & :=\operatorname{prox}_{\frac{1}{\beta} g}\left(2 Y^{p+1}-W^{p}\right), \\
W^{p+1} & :=W^{p}+X^{p+1}-Y^{p+1},
\end{aligned}\right.
$$

where $\operatorname{prox}_{h}: \mathcal{S}^{n} \rightarrow \mathcal{S}^{n}$ denotes the proximal mapping of a function $h: \mathcal{S}^{n} \rightarrow \mathcal{S}^{n}$, i.e.,

$$
\operatorname{prox}_{h}(Z):=\arg \min _{X \in \mathcal{S}^{n}} h(X)+\frac{1}{2}\|X-Z\|_{F}^{2} .
$$

Equivalences between the DRSM and other methods have been studied over the past decades. It has been shown that the DRSM applied to the problem of minimizing the sum of two convex functions and its Fenchel dual problem provide identical sequences under certain conditions [116]. Moreover, the DRSM is well-known to be closely related to the ADMM, e.g., through equivalence of the DRSM applied to a problem with the ADMM applied to its Fenchel dual problem, see [35, 118, 119, 149]. Below we demonstrate how the DRSM update scheme can be rewritten to an equivalent scheme, resulting in the ADMM. The construction we follow is based on [300].

Suppose we perform the algorithm (DRSM), where in each iteration $p \geq 0$, we additionally construct $S^{p+1}:=\beta\left(Y^{p+1}-W^{p}\right)$. Exploiting the last line $W^{p+1}=W^{p}+X^{p+1}-Y^{p+1}$ from the Douglas-Rachford scheme, it follows that $W^{p+1}=X^{p+1}-\frac{1}{\beta} S^{p+1}$ for all integers $p \geq 0$. Assume moreover that $X^{0}$ and $S^{0}$ are initialized such that $W^{0}=X^{0}-\frac{1}{\beta} S^{0}$. Then,
for all integers $p \geq 0$, we can rewrite the steps in (DRSM) as follows:

$$
\begin{aligned}
Y^{p+1} & =\operatorname{prox}_{\frac{1}{\beta} f}\left(W^{p}\right)=\operatorname{prox}_{\frac{1}{\beta} f}\left(X^{p}-\frac{1}{\beta} S^{p}\right) \\
& =\arg \min _{Y \in \mathcal{S}^{n}} f(Y)+\frac{\beta}{2}\left\|Y-X^{p}+\frac{1}{\beta} S^{p}\right\|_{F}^{2} \\
& =\arg \min _{Y \in \mathcal{S}^{n}} f(Y)+\left\langle S^{p}, Y-X^{p}\right\rangle+\frac{\beta}{2}\left\|Y-X^{p}\right\|_{F}^{2} \\
S^{p+1} & =\beta\left(Y^{p+1}-W^{p}\right)=\beta\left(Y^{p+1}-X^{p}+\frac{1}{\beta} S^{p}\right) \\
& =S^{p}+\beta\left(Y^{p+1}-X^{p}\right) \\
X^{p+1} & =\operatorname{prox}_{\frac{1}{\beta} g}\left(2 Y^{p+1}-W^{p}\right)=\operatorname{prox}_{\frac{1}{\beta} g}\left(Y^{p+1}+\frac{1}{\beta} S^{p+1}\right) \\
& =\arg \min _{X \in \mathcal{S}^{n}} g(X)+\frac{\beta}{2}\left\|X-Y^{p+1}-\frac{1}{\beta} S^{p+1}\right\|_{F}^{2} \\
& =\arg \min _{X \in \mathcal{S}^{n}} g(X)+\left\langle S^{p+1}, Y^{p+1}-X\right\rangle+\frac{\beta}{2}\left\|Y^{p+1}-X\right\|_{F}^{2},
\end{aligned}
$$

where the $W$-update in the last line of (DRSM) is performed implicitly via the $S$-update. Without interfering with the result of the algorithm, we can shift the $X$-update to the next iteration. More precisely, we define $Z^{p+1}:=X^{p}$ and obtain the equivalent update scheme

$$
\begin{aligned}
Z^{p+1} & =\arg \min _{Z \in \mathcal{S}^{n}} g(Z)+\left\langle S^{p}, Y^{p}-Z\right\rangle+\frac{\beta}{2}\left\|Y^{p}-Z\right\|_{F}^{2} \\
Y^{p+1} & =\arg \min _{Y \in S^{n}} f(Y)+\left\langle S^{p}, Y-Z^{p+1}\right\rangle+\frac{\beta}{2}\left\|Y-Z^{p+1}\right\|_{F}^{2} \\
S^{p+1} & =S^{p}+\beta\left(Y^{p+1}-Z^{p+1}\right),
\end{aligned}
$$

for all $p \geq 0$, with the exception that $Z^{1}$ is initialized by $X^{0}$ and the algorithm is started from the first $Y$-update. Below we formalize the equivalence between the latter scheme and (DRSM), see Theorem 1.20.

The latter scheme in fact equals the update scheme of the ADMM applied to (1.11). The ADMM starts from viewing (1.11) as the constrained program $\inf _{Y, Z}\{f(Y)+g(Z): Y=Z\}$. Let $S \in \mathcal{S}^{n}$ denote the Lagrange multiplier with respect to $Y-Z=\mathbf{0}$ and consider the augmented Lagrangian function of this problem with penalty parameter $\beta>0$ :

$$
\begin{equation*}
\mathcal{L}_{\beta}(Z, Y, S):=f(Y)+g(Z)+\langle S, Y-Z\rangle+\frac{\beta}{2}\|Y-Z\|_{F}^{2} . \tag{1.12}
\end{equation*}
$$

Now, the ADMM alternatingly minimizes $\mathcal{L}_{\beta}$ with respect to one of the primal variables, while keeping the other fixed, after which the dual multiplier is updated via a stepsize update. Starting from a given $Y^{0}$ and $S^{0}$, the ADMM update scheme is as follows:
(ADMM)

$$
\left\{\begin{array}{l}
Z^{p+1}:=\arg \min _{Z \in \mathcal{S}^{n}} \mathcal{L}_{\beta}\left(Z, Y^{p}, S^{p}\right), \\
Y^{p+1}:=\arg \min _{Y \in \mathcal{S}^{n}} \mathcal{L}_{\beta}\left(Z^{p+1}, Y, S^{p}\right), \\
S^{p+1}:=S^{p}+\beta\left(Y^{p+1}-Z^{p+1}\right)
\end{array}\right.
$$

Based on the structure of the functions $f$ and $g$, the $Z$ - and $Y$-subproblem involve a projection
onto the sets $\mathcal{S}_{+}^{n}$ and $\mathcal{Y}$, respectively.
The following theorem formalizes the equivalence between the DRSM and the ADMM as presented above, see also [300].
Theorem 1.20 ([300]). The sequence $\left(\left(Z^{p+1}, Y^{p+1}, S^{p+1}\right)\right)_{p \geq 1}$ generated by (ADMM) starting from $Y^{0}$ and $S^{0}$ is identical to the sequence $\left(\left(\bar{X}^{p}, \bar{Y}^{p+1}, \beta\left(\bar{Y}^{p+1}-\bar{W}^{p}\right)\right)\right)_{p \geq 1}$ obtained from (DRSM) starting from $\bar{W}^{0}:=Z^{1}-\frac{1}{\beta} S^{0}$.

Observe that the sequences considered in Theorem 1.20 are identical for $p \geq 1$. For ease of notation, we exclude the boundary case $p=0$ from the statement, since $\bar{X}^{0}$ is not formally defined in (DRSM).

Based on the equivalence between the DRSM and the ADMM, convergence results for the ADMM follow from similar results for the DRSM. It is known that if the functions $f$ and $g$ are closed, proper and convex functions and the Douglas-Rachford operator $F(W):=W+\operatorname{prox}_{\frac{1}{\beta} g}\left(\operatorname{prox}_{\frac{1}{\beta} f}(W)-W\right)-\operatorname{prox}_{\frac{1}{\beta} f}(W)$ has at least one fixed point, then the sequence $\left(\bar{Y}^{p}\right)_{p \geq 0}$ obtained from (DRSM) converges to a minimizer of (1.11), see e.g., [300]. Under certain conditions, the convergence rate of the DRSM, and thus the ADMM, is known to be linear. The linear convergence rate of the DRSM was first studied by [255], and since then, linear convergence has been proven in various settings, see [173] and the references therein. Linear convergence rates of the ADMM are also established without exploiting the equivalence to the DRSM, see e.g., [212, 378].

Besides the DRSM, the ADMM has also strongly connections to other methods, like dual decomposition and the method of multipliers. Dual decomposition is a distributed version of the dual ascent method, which relies on the nonaugmented Lagrangian function, i.e., (1.12) with $\beta=0$, to solve (1.11). Although this distributed optimization technique is attractive, the conditions under which convergence is established are rather strong. To bring robustness to the dual ascent method, augmented Lagrangian methods are considered. Due to the addition of the final term in (1.12), the augmented Lagrangian is strictly convex for all $\beta>0$. Consequently, the dual function $\psi_{\beta}(s):=\inf _{Z, Y} \mathcal{L}_{\beta}(Z, Y, S)$ is differentiable under rather mild conditions. Iteratively minimizing the augmented Lagrangian with respect to $Z$ and $Y$ (jointly) and performing a dual update step is known as the method of multipliers. The method of multipliers converges under far more general conditions than dual ascent. Its drawback, however, is that due to the additional penality term in (1.12), we can no longer decompose the minimization of the augmented Lagrangian with respect to $Z$ and $Y$ into multiple subproblems. The ADMM combines the ideas behind dual decomposition and the method of multipliers. It can be seen as a variation of the method of multipliers, where the joint minimization of $\mathcal{L}_{\beta}$ with respect to $Z$ and $Y$ is replaced by a single Gauss-Seidel-type iterate [178]. For a more detailed analysis of the connections between these methods, see e.g., [53].

Several extensions and generalizations of (ADMM) have been studied. When the dual update is performed twice per iteration, i.e., also between solving the $Z$ - and the $Y$-subproblem, we obtain the symmetric ADMM, which is known to be equivalent to the Peaceman-Rachford splitting method (PRSM), see e.g., [195]. To speed up convergence, the final step of (ADMM) is often replaced by $S^{p+1}:=S^{p}+\gamma \cdot \beta\left(Y^{p+1}-Z^{p+1}\right)$ where $\gamma$ is a positive stepsize parameter, leading to the accelerated ADMM or the ADMM with larger stepsize [141].

Solving large-scale semidefinite programs by augmented Lagrangian methods or its variations has been investigated in several works, see [61, 308, 323, 350, 363, 382]. This has led to the ADMM and the DRSM being applied to solve SDPs of different flavours, see e.g., [195, 196, 216, 218, 275, 278, 292, 308, 339, 365, 382].

### 1.4 Binary quadratic programs

The problems that are considered in this thesis belong to a specific class of integer programming problems, namely the binary quadratic programs. In this section we consider these programs in more detail and introduce some of the problems that will play a major role in this thesis.

A binary quadratic program (BQP) is characterized by a quadratic objective function and affine constraints, where the variables are restricted to be binary, i.e., having entries in $\{0,1\}$. Let $Q \in \mathcal{S}^{n}, A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^{m}$ be given. The BQP is given by the following mathematical optimization problem:

$$
\begin{array}{cl}
\min & x^{\top} Q x  \tag{1.13}\\
\text { s.t. } & x \in P=\left\{x \in\{0,1\}^{n}: A x \leq b\right\} .
\end{array}
$$

Due to the binarity of the model, we have $x^{\top} Q x+c^{\top} x+d=x^{\top}(Q+\operatorname{diag}(c)) x+d$ for any quadratic form, hence the objective function as given in (1.13) is nonrestrictive. As (1.13) contains binary linear programs as a special case, BQPs are $\mathcal{N} \mathcal{P}$-hard. Consequently, solving problems of the form (1.13) is practically challenging, asking for advanced solution strategies.

Many combinatorial optimization problems can be modeled as (1.13), e.g., the quadratic assignment problem, the stable set problem and the graph coloring problem. As returning examples throughout this thesis, we consider the quadratic traveling salesman problem, the quadratic cycle cover problem and the graph partition problem. We describe these problems in more detail throughout the next subsections. After that, in Section 1.4.4, we consider the linearization problem associated with a BQP.

### 1.4.1 The quadratic traveling salesman problem

Let $G=(N, A)$ be a directed simple graph on $n:=|N|$ nodes and $m:=|A|$ arcs. A Hamiltonian cycle or tour in $G$ is a (directed) cycle that visits each node exactly once. Suppose we are given a cost matrix $Q=\left(q_{e f}\right) \in \mathbb{R}^{m \times m}$ that has its support on adjacent arcs. That is, $q_{\text {ef }}=0$ if arc $f$ is not a successor of arc $e$ in $G$, i.e., $f \notin \delta^{+}\left(e^{-}\right)$. The quadratic traveling salesman problem (QTSP) asks for a Hamiltonian cycle in $G$ such that the total quadratic cost among the successive arcs used in the tour is minimized. This version, where we only consider interaction costs among adjacent arcs, is sometimes referred to as the adjacent-only QTSP in the literature, see also [133, 134, 135, 224, 326]. The QTSP as introduced above where $G$ is directed and $Q$ not necessarily symmetric is also called the asymmetric QTSP in the literature. When $G$ is undirected and $Q$ symmetric, the problem is referred to as the symmetric QTSP, see Appendix A.4. In this thesis, where we mostly consider the asymmetric version of the problem, we simply use the abbreviation QTSP when the asymmetric version is meant. The linear counterpart of the QTSP, where we aim to find a Hamiltonian cycle in $G$ that minimizes the total linear arc costs, is known as the (asymmetric) traveling salesman problem (TSP) and is among the most studied combinatorial optimization problems in the literature.

A Hamiltonian cycle in $G$ can be characterized by a vector $x \in\{0,1\}^{m}$, where $x_{e}=1$ if arc $e$ is used in the tour and $x_{e}=0$ otherwise. Let $P_{\text {QTSP }}$ denote the set of characteristic vectors of Hamiltonian cycles in $G$. Now, the QTSP can be modeled as

$$
\begin{equation*}
\min \quad x^{\top} Q x \quad \text { s.t. } \quad x \in P_{\mathrm{QTSP}} . \tag{1.14}
\end{equation*}
$$

The set $P_{\text {QTSP }}$ can be written as

$$
P_{\mathrm{QTSP}}:=\left\{x \in\{0,1\}^{m}: \begin{array}{l}
\sum_{e \in \delta^{+}(i)} x_{e}=\sum_{e \in \delta^{-}(i)} x_{e}=1 \quad \forall i \in N  \tag{1.15}\\
\sum_{e \in \delta^{+}(S)} x_{e} \geq 1 \quad \forall \emptyset \neq S \subsetneq N
\end{array}\right\} .
$$

The first set of constraints are the so-called degree constraints, which manage that each node in $N$ has in- and out-degree equal to one. The second set of constraints prevent a solution $x$ to consist of multiple subtours, by requiring that for any subset $S \subseteq N$ with $1 \leq|S| \leq n-1$, there must be at least one arc leaving $S$. Indeed, if $x$ induces a closed subtour on a subset $S$ of the nodes, such an arc would not exist, preventing $x$ to be included in $P_{\text {Qtsp }}$. These constraints are known as the cut-set constraints and are originally proposed for the TSP by Dantzig et al. [91]. The authors of [91] show that these are equivalent to the so-called subtour elimination constraints

$$
\begin{equation*}
\sum_{\substack{e=\{i, j\} \\ i, j \in S}} x_{e} \leq|S|-1 \quad \forall \emptyset \neq S \subsetneq N . \tag{1.16}
\end{equation*}
$$

Since the set $P_{\text {QTSP }}$ does not depend on the quadratic nature of the problem, it is equal to the solution set of the (asymmetric) TSP. There exist many alternative formulations of the feasible set of the TSP, among which the ones by Dantzig et al. [91], Miller et al. [281] and Gavish and Graves [168] are the most well-known. For a comprehensive overview of formulations and solution approaches for the TSP, we refer the reader to book by Davendra [94].

The QTSP has been introduced by Jäger and Molitor [224], motivated by an important application in bioinformatics [135, 224]. Besides, the QTSP has also applications in telecommunication, precision farming, transportation, energy distribution networks and robotics, see e.g., $[6,127,370]$. The QTSP is shown to be $\mathcal{N} \mathcal{P}$-hard in the strong sense by a straightforward reduction from the asymmetric TSP [224], and is among one of the hardest combinatorial optimization problems to solve in practice.

Several special cases of the QTSP have been considered. When the nodes in $G$ are embedded in Euclidean space and the quadratic costs among two arcs is proportional to the induced turning angle, we call the problem the angular-metric TSP [6]. When the costs are a linear combination between the turning angles and Euclidean distances, we refer to the problem as the angular-distance-metric TSP. Another version of the problem follows from the reload cost model in graph optimization problems [370]. In this model, we assume the arcs in $G$ to be colored using some finite set $L$ of colors and $r: L \times L \rightarrow \mathbb{R}$ is a given reload cost function. Now, the quadratic costs among two successive arcs $e$ and $f$ is denoted by $r(s, t)$, where $s$ and $t$ are the colors of $e$ and $f$, respectively. The motivation behind this structure is that a cost is incurred whenever the underlying type of an arc changes, e.g., when the means of transportation along an arc changes in a cargo network. The QTSP under the reload cost structure is denoted as the minimum reload cost TSP. Besides transportation networks, the reload cost model has also applications in energy and telecommunication networks [12, 370]. Several other combinatorial optimization problems including these reload costs have been investigated, see e.g., $[12,151,160,180,370]$. The QTSP is also related to the class of covering tour problems with turn costs as considered in Arkin et al. [19]. Covering tour problems play an important role in, e.g., manufacturing, automatic inspection and spray painting operations. For a detailed overview of these problems and their applications, we
refer the reader to [127].
Several solution approaches for the QTSP or its related problems have been considered. Jäger and Molitor [224] consider two exact methods for solving the asymmetric QTSP: an integer programming formulation and a branch-and-bound method to enumerate over the set of tours. This work is extended by Fischer et al. [135, 136, 138], who compare three exact algorithms for the asymmetric version of the problem, namely a branch-and-bound algorithm, a branch-and-cut algorithm and an approach based on a reduction to the asymmetric TSP. The authors of [276] propose a different branch-and-cut algorithm for the QTSP, which exploits Chvátal-Gomory type cuts. These cuts follow from an integer semidefinite programming formulation of the problem that is based on algebraic connectivity, see also Chapter 5 in this thesis. Aichholzer et al. [7] consider exact solution approaches for the symmetric QTSP and the related angular-metric TSP. The polyhedral structure of a linearized formulation of the asymmetric QTSP is studied by Fischer [134], where several classes of facet-defining inequalities are introduced and analyzed. The same approach has been followed for the symmetric QTSP by Fischer and Helmberg [137], see also [133]. Rostami et al. [326] provide several lower bounding procedures for the asymmetric QTSP, including a column generation approach and a bound based on the reformulation-linearization technique followed by a Lagrangian relaxation. The linearization problem for the QTSP is studied by Punnen et al. [311]. Finally, several heuristic methods are considered in the literature. Aggarwal et al. [6] derive an $O(\log n)$ approximation algorithm for the angular-metric TSP. The first heuristics for the QTSP are proposed by [135, 224] and are based on similar approaches for the linear TSP. Woods and Punnen [374] provide different classes of neighbourhoods for the QTSP and related problems. Staněk et al. [347] discuss several heuristics for the QTSP in the plane, which are based on geometric arguments or auxiliary integer linear programs. Finally, Zhang et al. [379] adopt a deep reinforcement learning approach to tackle the angular-metric and the angular-distance-metric TSP.

### 1.4.2 The quadratic cycle cover problem

Given a directed graph $G=(N, A)$ and a cost matrix $Q=\left(q_{e f}\right) \in \mathbb{R}^{m \times m}$ with the same characteristics as described in Section 1.4.1, the quadratic cycle cover problem (QCCP) aims to find a set of node-disjoint cycles covering all the nodes in $G$, such that the total quadratic cost among the used successive arcs is minimized. The QCCP is closely related to the QTSP. Indeed, instead of finding one closed tour in $G$, we allow for the existence of multiple subtours which together visit all the nodes. In a similar fashion as described in Section 1.4.1, we distinguish between the symmetric and asymmetric QCCP. The linear version of the problem is called the cycle cover problem (CCP) and aims to find a minimal cycle cover under linear arc costs. The CCP is well-known to be solvable in polynomial time due to its equivalence with the bipartite perfect matching problem.

Let $x \in\{0,1\}^{m}$ denote the characteristic vector of a cycle cover in $G$ such that $x_{e}=1$ if arc $e$ is used in the cycle cover and $x_{e}=0$ otherwise. Let $P_{\mathrm{QCCP}}$ denote the set of characteristic vectors of cycle covers in $G$. Then, the QCCP can be modeled as

$$
\begin{equation*}
\min \quad x^{\top} Q x \quad \text { s.t. } \quad x \in P_{\mathrm{QCCP}} . \tag{1.17}
\end{equation*}
$$

The set $P_{\mathrm{QCCP}}$ can be written as

$$
\begin{equation*}
P_{\mathrm{QCCP}}:=\left\{x \in\{0,1\}^{m}: \quad \sum_{e \in \delta^{+}(i)} x_{e}=\sum_{e \in \delta^{-}(i)} x_{e}=1 \quad \forall i \in N\right\} \tag{1.18}
\end{equation*}
$$

which can be obtained from $P_{\text {QTSP }}$ by dropping the cut-set or subtour elimination constraints. A cycle cover in a directed graph is also called a directed 2 -factor. Notice that the convex hull of the elements in $P_{\mathrm{QCCP}}$ is easy to describe, since the underlying constraint matrix is totally unimodular. The corresponding integer polytope is denoted by the directed 2 -factor polytope. This polytope is equivalent to the perfect matching polytope of a related bipartite graph $B(G)$, the so-called bipartite representation of $G$, as shown by [275], see also Section 3.4.2 of this thesis. The perfect matching polytope of bipartite graphs is well-understood, see e.g., [48, 331].

The QCCP is known to be $\mathcal{N} \mathcal{P}$-hard in the strong sense [138, 274] and is mainly studied due to its close connection with the QTSP. Solving the QCCP is often regarded as a subproblem in order to obtain lower bounds for the QTSP.

Similar to the case of the QTSP, several special cases with respect to the objective function of the QCCP are considered. In the angular-metric CCP the quadratic costs represent the turning angle induced by two successive arcs. This problem has applications in robotics and is also shown to be $\mathcal{N} \mathcal{P}$-hard [6]. Galbiati et al. [152] introduce another special case of the QCCP: the minimum reload cost cycle cover problem (MinRC3). The MinRC3 problem asks for a minimum cycle cover in an arc-colored graph under the reload cost model discussed in Section 1.4.1. The MinRC3 problem is proven to be $\mathcal{N} \mathcal{P}$-hard in the strong sense [152]. A detailed overview of the MinRC3 problem and its applications can be found in [63].

Several papers have been written about solution methods for the QCCP or its related problems. Jäger and Molitor [224] introduce the QCCP in order to use the QCCP bounds as lower bounds in a branch-and-bound algorithm for the QTSP. Galbiati et al. [152] consider various integer programming formulations for the MinRC3 problem. One of these formulations is exploited in a column generation framework to compute lower bounds for the problem. Several LP-based lower bounds for the problem are considered in [274], among which lower bounds exploiting the linearization problem of the QCCP, see Section 1.4.4 and Chapter 2 of this thesis. Lower and upper bounds based on semidefinite programming are proposed in [275], see also Chapter 3 of this work. Fischer [133] derives polyhedral properties for a linearized formulation of the QCCP by proving that several inequalities are facet-defining. Büyükçolak et al. [63] study the MinRC3 problem on complete undirected graphs with a so-called equitable or nearly equitable 2 -edge coloring. These are edge-colorings containing only two colors, where the number of edges adjacent to a vertex that are colored by each of the classes differs by no more than one (respectively two) for each vertex in $G$. A polynomial time algorithm that constructs a monochromatic cycle cover is derived for these types of graphs (except for some special cases). Finally, several heuristics for the QCCP have been considered. Aggarwal et al. [6] provide a $O(\log n)$-approximation algorithm for the angularmetric CCP. The QCCP in combination with a rounding procedure is applied by Staněk et al. [347] to construct heuristics for the QTSP. Galbiati et al. [152] construct upper bounds for the MinRC3 problem by a local search algorithm based on 2-exchange and 3-exchange neighbourhoods.

### 1.4.3 The graph partition problem

Let $G=(V, E)$ be an undirected graph with vertex set $V$ and edge set $E$, where $n:=|V|$. Also, let $w: E \rightarrow \mathbb{R}$ be a weight function and let $k$ be an integer with $2 \leq k \leq n-1$. The goal of the graph partition problem (GPP) is to find a partition of the vertex set $V$ into $k$ disjoint sets, i.e., $V=S_{1} \cup \cdots \cup S_{k}$, where $S_{1}, \ldots, S_{k}$ have specified sizes $m_{1} \geq \ldots \geq m_{k} \geq 1$ with $\sum_{j=1}^{k} m_{j}=n$, such that the total weight of edges having endpoints in different sets is minimized.

Several special cases of the GPP are known: In case $m_{1}=\cdots=m_{k}=\frac{n}{k}$, the resulting GPP is known as the $k$-equipartition problem ( $k$-EP). If $k=2$, we refer to the problem as the graph bisection problem (GBP). In case $k=2$ and $m_{1}=m_{2}$, the problem is called the equicut problem (ECP).

In order to model the problem, let $P \in\{0,1\}^{n \times k}$ denote a partition matrix such that

$$
P_{i j}:=\left\{\begin{array}{ll}
1 & \text { if } i \in S_{j}, \\
0 & \text { otherwise, }
\end{array} \quad \text { for all } i \in[n], j \in[k] .\right.
$$

Moreover, define $x \in\{0,1\}^{n k}$ by $x:=\operatorname{vec}(P)$, i.e., the concatenation of the characteristic vectors of the sets $S_{j}, j \in[k]$. Let $W=\left(w_{i j}\right) \in \mathcal{S}^{n}$ denote the weight matrix of $G$, where $w_{i j}=w(\{i, j\})$ if $\{i, j\} \in E$ and $w_{i j}=0$ otherwise. Since $P_{:, j}^{\top} W P_{:, \ell}$ equals the total weight of edges having one endpoint in $S_{j}$ and one endpoint in $S_{\ell}$, the total weight of edges joining different sets of the partition equals

$$
\frac{1}{2} \sum_{j \in[k]} \sum_{\ell \in[k]} P_{:, j}^{\top} W P_{:, \ell}=\frac{1}{2} x^{\top}\left(\left(\mathbf{J}_{k}-\mathbf{I}_{k}\right) \otimes W\right) x .
$$

Let $P_{\text {GPP }}$ denote the set of all characteristic vectors $x$ of partitions of $G$ that satisfy the cardinality constraints imposed by $m_{1}, \ldots, m_{k}$. Then, the GPP can be modeled as

$$
\begin{equation*}
\min \quad \frac{1}{2} x^{\top}\left(\left(\mathbf{J}_{k}-\mathbf{I}_{k}\right) \otimes W\right) x \quad \text { s.t. } \quad x \in P_{\mathrm{GPP}} . \tag{1.19}
\end{equation*}
$$

Using matrix notation, the set $P_{\text {GPP }}$ can be written as

$$
\begin{equation*}
P_{\mathrm{GPP}}:=\left\{x \in\{0,1\}^{k n}:\binom{\mathbf{I}_{k} \otimes \mathbf{1}_{n}^{\top}}{\mathbf{1}_{k}^{\top} \otimes \mathbf{I}_{n}} x=\binom{\mathbf{m}}{\mathbf{1}_{n}}\right\} \tag{1.20}
\end{equation*}
$$

where $\mathbf{m}:=\left[\begin{array}{lll}m_{1} & \ldots & m_{k}\end{array}\right]^{\top}$ denotes the column vector of set cardinalities. Indeed, the first row of the block system in (1.20) manages that the sum of elements in each set $S_{j}$ equals $m_{j}$. The other block row manages that each vertex in $V$ is assigned to exactly one set $S_{j}$. One easily verifies that the constraint matrix in (1.20) is totally unimodular, implying that the convex hull of the elements in $P_{\text {GPP }}$ is an integer polytope. This polytope we denote by the $k$-partition polytope and is a special case of the transportation polytope, see e.g., [111].

We below present a different formulation of the GPP that we frequently use in this dissertation. The total weight of the partition equals the sum of all edge weights minus the total weight of edges that have both endpoints in the same set. Hence, the objective function can also be written as

$$
\frac{1}{2} \operatorname{tr}\left(W\left(\mathbf{J}_{n}-P P^{\top}\right)\right)=\frac{1}{2} \operatorname{tr}\left(L P P^{\top}\right)
$$

where $L:=\operatorname{Diag}\left(W \mathbf{1}_{n}\right)-W$ denotes the weighted Laplacian matrix of $G$. Now, the GPP
can be formulated as the following binary quadratic optimization problem:

$$
\begin{array}{ll}
\min & \frac{1}{2}\left\langle L, P P^{\top}\right\rangle \\
\text { s.t. } & P \mathbf{1}_{k}=\mathbf{1}_{n}  \tag{1.21}\\
& P^{\top} \mathbf{1}_{n}=\mathbf{m} \\
& P_{i j} \in\{0,1\} \quad \forall i \in[n], j \in[k] .
\end{array}
$$

Clearly, the constraints on $P$ manage that $P$ is a partition matrix, where each set has cardinality $m_{j}$.

The GPP is proven to be $\mathcal{N} \mathcal{P}$-hard by Garey et al. [165]. It has attracted the attention of many researchers due to its wide applicability, e.g., in clustering problems, floor planning, VLSI design, parallel computing, data mining, air traffic, image processing, image segmentation, quantum circuit design, see e.g., the book by Bichot and Siarry [47]. For an overview of recent advances in graph partition problems, we refer the reader to [59].

We present below a brief summary of solution approaches for the GPP or its variants that are based on semidefinite programming. Donath and Hoffman [104] derive an eigenvalue bound for the GPP, which is later improved by Rendl and Wolkowicz [321]. Wolkowicz and Zhao [371] derive SDP relaxations for the GPP including matrix variables of order $n k+1$. Sotirov [343] derives another more compact SDP relaxation for the GPP that includes matrix variables of order $n$.

For the special case of the $k$-EP, Karisch and Rendl [231], among others, show that the bounds introduced in [104] and [321] can be obtained as the solution of a semidefinite program. Moreover, Karisch and Rendl present several SDP relaxations for the $k$-EP that dominate the bounds from [104, 321]. One of the SDP relaxations in [231] includes additional polyhedral cuts and currently provides the strongest known SDP bound for the $k$-EP. This relaxation is further investigated for the special case where $G$ is a highly symmetric graph by Van Dam and Sotirov [90]. De Klerk et al. [233] utilize the model of the $k$-EP as a quadratic assignment problem in order to obtain an alternative relaxation for the $k$-EP, which is thereafter reduced by exploiting symmetries.

For the special case of the GBP, two equivalent SDP relaxations with matrix variables of order $n$ are derived in [232, 343]. Sotirov [344] derives yet another SDP relaxation of order $n$ that dominates the ones presented in [232, 343]. It turns out that the latter relaxation is equivalent to the general GPP relaxation from [371].

Although the SDP relaxations with additional cutting planes turn out to be strong for both the $k$-EP [231] and the GBP, state-of-the-art interior-point methods have difficulties solving these relaxations. An alternative solution method based on a first-order method in a cutting-plane framework is considered in [278], see also Chapter 4 in this thesis.

Several of the above-mentioned SDP bounds are exploited in exact methods for solving the GPP, the $k$-EP or the GBP. Karish et al. [232] develop a branch-and-bound algorithm for the GBP based on SDP bounds with polyhedral constraints. Alternatively, Hager et al. [193] propose a branch-and-bound algorithm for the GBP that exploits continuous quadratic programming formulations of the problem. It is reported in [193] that the use of SDP bounds leads to the best performance of this algorithm. Armbruster et al. [21] consider a branch-andcut framework on large and sparse GBP instances, using either linear or SDP relaxations. Numerical results suggest the superiority of the semidefinite programming approach.

### 1.4.4 The linearization problem of binary quadratic programs

Let us reconsider the general binary quadratic program of (1.13). If there exists a vector $p \in \mathbb{R}^{n}$ such that we have

$$
x^{\top} Q x=p^{\top} x \quad \text { for every } x \in P
$$

then we say that the binary quadratic program is linearizable. If such a vector $p$ exists, we call $p$ a linearization vector of $Q$ for the given BQP. By abuse of terminology, we also say that the matrix $Q$ is linearizable for the given optimization problem. If (1.13) is linearizable, this implies that we can equivalently solve the following binary linear program:

$$
\begin{array}{cl}
\min & p^{\top} x \\
\text { s.t. } & x \in P=\left\{x \in\{0,1\}^{n}: A x \leq b\right\} \tag{1.22}
\end{array}
$$

Indeed, the obtained binary linear program (1.22) can be easier to solve than (1.13), e.g., due to the set $\operatorname{conv}(P)$ having an explicit polyhedral description, or due to the fact that $(1.22)$ is a well-known problem for which an efficient combinatorial algorithm exists.

The linearization problem of a BQP asks whether the cost matrix $Q$ is linearizable and, if so, to provide a linearization vector $p$.

Recently, the study of linearization problems has become an active field of research for many combinatorial optimization problems. Kabadi and Punnen [229, 309] study the linearization problem of the quadratic assignment problem (QAP) and provide several polynomial time algorithms that solve it. More specifically, Kabadi and Punnen [229] (resp. Punnen and Kabadi [309]) present an $O\left(n^{4}\right)$ (resp. $O\left(n^{2}\right)$ ) algorithm for the general (resp. KoopmansBeckmann) QAP linearization problem, where $n$ is the size of the problem. Adams and Waddell [5] and Cela et al. [70] consider linearizable special cases of the QAP. The linearization problem of the bilinear assignment problem is considered by Ćustić et al. [85]. The linearization problem of the quadratic minimum spanning tree problem and the quadratic traveling salesman problem are studied by Ćustić and Punnen [84] and Punnen et al. [311], respectively. The linearization problem of the quadratic shortest path problem (QSPP) is considered by Hu and Sotirov [215], who develop a polynomial time algorithm that solves the linearization problem on directed grid graphs. The same authors also derive a polynomial time algorithm for the linearization problem of the QSPP on directed acyclic graphs in [217]. The linearization problem of the quadratic cycle cover problem is studied in [274], see also Chapter 2 in this thesis.

Applications of the linearization problem of BQPs are also considered in the literature. Punnen et al. [310] study equivalent representations of quadratic combinatorial optimization problems and exploit the corresponding linearization problem to derive such representations. The linearization problem of general BQPs is studied in [217], where a new lower bounding scheme exploiting the linearization problem is introduced. It is shown that the resulting linearization-based bounds include several well-known families of bounds, including the generalized Gilmore-Lawler type bound and bounds resulting from the first level reformulation-linearization technique (RLT-1). Linearization-based bounds for the quadratic cycle cover problem that only rely on sufficient conditions of linearizability are studied in [274], see Chapter 2 of this thesis.

### 1.5 Outline of the thesis

In this dissertation, we focus on the concepts of integrality and cutting planes in semidefinite programs: two notions that have acquired a fundamental role in (integer) linear programming theory. For semidefinite programs, however, these concepts are not yet studied to the same extent and thus numerous aspects within this domain are still to be analyzed. This thesis contributes to the understanding of these notions in the framework of SDPs, where we specifically focus on several well-known classes of combinatorial optimization problems.

The contribution of this thesis can be divided into four research themes: cutting planes in SDPs, integrality in SDPs, the QCCP/QTSP and their applications and the GPP and its applications. Below we briefly motivate these four themes and their connection.

Regarding the first research theme, we investigate the versatile role of cutting planes in (integer) semidefinite programming. We investigate several general classes of cutting planes for integer semidefinite programs and study how these can be used in a branch-and-cut framework. In particular, we perform an in-depth study on the theoretical and practical properties of the Chvátal-Gomory cuts for integer SDPs. We also investigate how cutting planes can be used in (continuous) SDPs. It is well-known that polyhedral cuts can significantly strengthen the SDP relaxations of hard combinatorial problems. The current state-of-the-art SDP solvers, however, have difficulties solving models involving lots of cutting planes. As an alternative, we introduce an advanced SDP-based cutting-plane method that is suitable for the addition of polyhedral cuts. This approach is based on a combination of the ADMM and an iterative projection technique, and can handle the addition of a large number of cutting planes.

The second research theme gathers around integrality in SDPs. Mixed-integer semidefinite programming has become an active field of study very recently, mainly driven by the realization that several problems model naturally as a mixed-integer SDP (MISDP), see e.g., [71, 376]. Using a combination of generic and problem-specific approaches, we show that in fact a large number of problems allow for a formulation in terms of a MISDP, including well-known combinatorial optimization problems. We focus on some theoretical notions regarding MISDPs, such as total dual integrality, the Chvátal-Gomory closure and Lagrangian duality theory. Besides, we improve on an existing branch-and-cut algorithm for MISDPs by the incorporation of MISDP-based cutting planes.

The study on the QCCP, the QTSP and their applications form the third and largest research theme. Both problems make a recurring appearance throughout this work, often serving as the main examples to which the above-mentioned results are applied. This leads to several novel approaches for tackling these difficult problems. On top of these approaches, we also investigate other solution methods that are treated in separate chapters of this thesis. First, we introduce several LP-based bounding approaches for the QCCP, including several linearization-based bounds that follow from a study on the linearization problem of the QCCP. Also, various novel LP- and SDP-based upper bounding techniques for the QCCP are introduced, such as randomized rounding methods and a reinforcement learning approach. Finally, we consider an application of a variation of the QTSP, namely the generalized traveling salesman problem (GTSP). Given a partitioning of the nodes of a graph into clusters, the GTSP asks for an optimal closed cycle that visits at least one node from each cluster. An interesting application of this setting can be found in quantum circuit design. The final part of this thesis is fully dedicated to this application.

The fourth research theme covers the GPP and its application. It is well-known that

SDP bounds for the GPP and its variants can be substantially improved by the addition of polyhedral cuts. As mentioned earlier, we propose a new ADMM-based cutting-plane algorithm to solve such SDPs. We study this algorithm in detail for several types of graph partition problems. Moreover, we show that the GPP and its variants allow for several different MISDP formulations.

The remaining contents of this thesis are divided into six self-contained chapters. The contributions of these chapters to the above-mentioned four research themes are depicted in Figure 1.1. Below we briefly outline the main contributions per chapter.

## Chapter 2: The quadratic cycle cover problem: special cases and efficient bounds

In this chapter we study the linearization problem of the QCCP and its related lower bounds. First, we provide an alternative proof for the $\mathcal{N} \mathcal{P}$-hardness of the QCCP. Then, we derive various sufficient conditions for the cost matrix to be linearizable and use these conditions to compute strong and efficient lower bounds. We also show how to use a sufficient condition for linearizability within an iterative bounding procedure. The introduced family of linearization-based bounds contains the classical Gilmore-Lawler type bound. When taking both quality and efficiency into account, the best among here introduced bounds outperform existing lower bounds.

## Chapter 3: SDP-based bounds for the quadratic cycle cover problem via cutting-plane augmented Lagrangian methods and reinforcement learning

In this chapter we derive several SDP relaxations for the QCCP and use facial reduction to make these strictly feasible. To solve our relaxations, we propose a generic algorithm that incorporates an augmented Lagrangian method into a cutting-plane framework by utilizing Dykstra's projection algorithm. Our algorithm is suitable for solving SDP relaxations with a large number of polyhedral cuts. Computational results show that our SDP bounds and our efficient cutting-plane algorithm outperform other QCCP bounding approaches from the literature. Finally, we provide several SDP-based upper bounding techniques, among which a sequential Q-learning method that exploits a solution of our SDP relaxation within a reinforcement learning environment.


Figure 1.1: Color chart of distribution of four research themes over the different chapters.

## Chapter 4: Partitioning through projections: strong SDP bounds for large graph partition problems

In this chapter we investigate the quality of doubly nonnegative relaxations, strengthened by additional polyhedral cuts for the GPP, in particular for the special cases of the GBP and the $k$-EP, see Section 1.4.3. After reducing the size of the relaxations by facial reduction, we solve them by the SDP-based cutting-plane approach introduced in Chapter 3. Our computational study shows the power of doubly nonnegative relaxations with additional cutting planes for the GPP on large benchmark instances. The improvements compared to the standard SDP bounds, i.e., without the use of additional cutting planes, are significant.

## Chapter 5: The Chvátal-Gomory procedure for integer SDPs with applications in combinatorial optimization

In this chapter we study the class of Chvátal-Gomory (CG) cutting planes for integer SDPs and the closure obtained from taking the intersection of all such cuts. This closure is related to the notion of total dual integrality in SDPs. We derive several conditions under which a linear matrix inequality is totally dual integral. On the practical side, we show how to exploit (strengthened) CG cuts in a branch-and-cut framework for integer SDPs. Different from existing algorithms in the literature, the separation routine in our approach exploits both the PSD and the integrality constraints. In the second part of the chapter we present a comprehensive application of our approach to the QTSP. Based on the algebraic connectivity of the directed Hamiltonian cycle, two integer SDPs that model the QTSP are introduced. We show that the CG cuts resulting from these formulations contain several well-known families of cutting planes. Numerical results illustrate the practical strength of the CG cuts in our branch-and-cut algorithm.

## Chapter 6: On integrality in semidefinite programming for combinatorial optimization

In this chapter we show that a wide variety of combinatorial optimization problems can be modeled as a mixed-integer SDP. Based on a comprehensive study on discrete positive semidefinite matrices, we follow a generic approach to derive mixed-integer semidefinite programming (MISDP) formulations for binary quadratically constrained quadratic programs and binary quadratic matrix programs. Applying a problem-specific approach, we derive more compact MISDP formulations for several problems, such as the GPP. We also show that several structured problems allow for novel compact MISDP formulations through the notion of algebraic connectivity. On the applied side, we show how these MISDP models can be exploited to obtain strong bounds for the corresponding problems via Lagrangian duality. By introducing a MISDP-based projected subgradient algorithm, we show that the resulting Lagrangian dual bounds for the max-cut problem are substantially stronger than the standard SDP bound.

## Chapter 7: Exploiting symmetries in optimal quantum circuit design

In this chapter we consider a problem in quantum circuit design that is related to the generalized traveling salesman problem (GTSP). A physical limitation in quantum circuit design is the fact that gates in a quantum system can only act on qubits that are physically adjacent in the architecture. To overcome this problem, SWAP gates need to be inserted
in the circuit to make the circuit physically realizable. The nearest neighbour compliance problem (NNCP) asks for an optimal embedding of qubits in a given architecture such that the total number of SWAP gates to be inserted is minimized. Matsuo and Yamashita [269] show that this problem reduces to a large-scale shortest path problem on a sequence of Cayley graphs. If we add to the model the constraint that the final embedding should match the embedding at the start, the problem reduces to a GTSP. In this chapter, however, we only study the NNCP without this restriction.

The goal of this chapter is to reduce the shortest path formulation by exploiting the group symmetry of the graph underlying the formulation. This leads to a symmetry-reduced NNCP algorithm that involves solving a generalized network flow problem. As a byproduct of our approach, we show that the NNCP is polynomial time solvable for several classes of highly symmetric quantum architectures. Numerical tests on several architectures indicate that the reductions in the number of variables and constraints on average is at least $90 \%$. For particular architectures, NNCP instances up to 100 qubits and more than 1000 quantum gates can be solved. These results are far beyond the computational capacity when solving the instances without the exploitation of symmetries.

### 1.6 Contributions to the literature

This thesis is based on the following seven research papers:
Chapter 2 F. de Meijer and R. Sotirov ([274]). The quadratic cycle cover problem: Special cases and efficient bounds. Journal of Combinatorial Optimization, 39:1096-1128, 2020.
Chapter $3 \quad$ F. de Meijer and R. Sotirov ([275]). SDP-based bounds for the quadratic cycle cover problem via cutting-plane augmented Lagrangian methods and reinforcement learning. INFORMS Journal on Computing, 33:1262-1276, 2021. (Rewarded a Meritorious Paper Award by INFORMS Journal on Computing)
Chapter 4 F. de Meijer, R. Sotirov, A. Wiegele and S. Zhao ([278]). Partitioning through projections: Strong SDP bounds for large graph partition problems. Computers © Operations Research, 151:106088, 2023.

Chapter $5 \quad$ F. de Meijer and R. Sotirov ([276]). The Chvátal-Gomory procedure for integer SDPs with applications in combinatorial optimization. In third review round for publication in Mathematical Programming, 2023.
Chapter $6 \quad$ F. de Meijer and R. Sotirov ([277]). On integrality in semidefinite programming for discrete optimization. In second review round for publication in SIAM Journal on Optimization, 2023.
F. de Meijer and R. Sotirov. On improving semidefinite programming bounds via integer Lagrangian duality. Working paper, 2023.

Chapter $7 \quad$ F. de Meijer, D. Gijswijt and R. Sotirov ([272]). Exploiting symmetries in optimal quantum circuit design. Working paper, 2023.


# The quadratic cycle cover problem: special cases and efficient bounds 

## Chapter summary

The quadratic cycle cover problem is the problem of finding a set of node-disjoint cycles visiting all the nodes such that the total sum of interaction costs between consecutive arcs is minimized. In this chapter we study the linearization problem of the quadratic cycle cover problem and related lower bounds.

In particular, we derive various sufficient conditions for the cost matrix to be linearizable, and use these conditions to compute bounds. We also show how to use a sufficient condition for linearizability within an iterative bounding procedure. In each step, our algorithm computes the best equivalent representation of the cost matrix and its optimal linearizable matrix with respect to the given sufficient condition for linearizability. Further, we show that the classical Gilmore-Lawler type bound belongs to the family of linearization-based bounds, and therefore apply the abovementioned iterative reformulation technique. We also prove that the linearization vectors resulting from this iterative approach satisfy the constant value property.

The best among here introduced bounds outperform existing lower bounds when taking both quality and efficiency into account.

### 2.1 Introduction

A disjoint cycle cover in a directed graph is a set of node-disjoint cycles such that every node belongs to exactly one cycle. The quadratic cycle cover problem (QCCP) is the problem of finding a disjoint cycle cover in a graph such that the total sum of interaction costs between consecutive arcs is minimized. Since we assume that all cycle covers in this chapter are disjoint, we use the term cycle cover to denote this concept throughout this chapter. The QCCP is proven to be $\mathcal{N} \mathcal{P}$-hard [138]. The corresponding linear problem is called the cycle cover problem (CCP), in which one wants to find a minimum cycle cover with respect to linear arc costs. It is well-known that the CCP is solvable in polynomial time.

The QCCP is closely related to the quadratic traveling salesman problem (QTSP) which is introduced in [224]. The QTSP is the problem of finding a Hamiltonian cycle in a graph minimizing a quadratic cost function. It has applications in bioinformatics, robotics and telecommunication [135]. When we remove the subtour elimination constraints, the QTSP boils down to the QCCP. Therefore, the QCCP is often used to provide lower bounds for the QTSP [135, 224, 347]. For this reason, the quadratic cycle cover problem is an interesting optimization problem that has received more attention in the past few years.

In the literature several special cases with respect to the objective function of the QCCP are considered. For an overview of these special cases, see Section 1.4.2. We also refer to Section 1.4.2 for an overview of solution methods for the QCCP that are proposed in the literature.

We focus here on the linearization problem of the QCCP and its applications. An instance of the quadratic cycle cover problem is called linearizable if there exists an instance of the linear cycle cover problem such that the associated costs for both problems are equal for all feasible cycle covers. The linearization problem of the quadratic cycle cover problem asks whether a given instance of the QCCP is linearizable. To the best of our knowledge, we are the first to address the linearization problem of the QCCP. For a literature overview on the linearization problem of other binary quadratic problems, we refer the reader to Section 1.4.2.

### 2.1.1 Main results and outline

In this chapter, we first provide a compact proof for the QCCP being strongly $\mathcal{N} \mathcal{P}$-hard and not approximable within any constant factor unless $\mathcal{P}=\mathcal{N} \mathcal{P}$. Then, we consider the linearization problem of the QCCP and derive various sufficient conditions for an instance of the QCCP to be linearizable. In particular, we provide three different types of weak sum conditions on the data matrix for which the corresponding instance can be solved in polynomial time. Further, we present a general framework in which each sufficient condition for linearizability can be used to construct a lower bound on the optimal objective value. Each of these bounds can be computed by solving a linear programming problem, as long as the set of linearizable matrices is a polyhedron. These types of bounds are called linearization-based bounds (LBB) and were recently introduced in [217] for general binary quadratic problems. However, our LBBs exploit sufficient conditions for linearizability suited for the QCCP.

Furthermore, we show how to use a sufficient condition for linearizability within an iterative bounding procedure. In each iteration, we search for the best equivalent representation of the objective and its optimal linearizable matrix that satisfies a particular sufficient condition for linearizability. We refer to the resulting bound as the reformulation-based bound (RBB). Our iterative bounding procedure can be seen as a generalization of similar iterative procedures, see e.g., $[66,324,325]$.

Finally, we consider the classical Gilmore-Lawler (GL) type bound [172, 246]. First, we show that the GL type bound for the QCCP can be obtained by solving a single linear programming problem instead of solving $m$ (integer) subproblems, where $m$ equals the number of arcs in the graph. Then, we prove that the GL type bound belongs to the family of linearization-based bounds by providing the appropriate sufficient condition. We implement our iterative bounding procedure to compute the RBB using the GL type bound. In the literature, iterative approaches for various problems that are based on the GL type bounds use dual variables to obtain bounds, and do not search for equivalent reformulations that provide the best bound in each iteration. By construction, our approach outperforms others in terms of strength of the bound. Another interesting result is that the linearization vectors resulting from this iterative procedure satisfy the constant value property-yet another important property for linearizability.

Our numerical results show that the introduced bounding approaches are efficient and provide strong bounds compared to several methods from the literature. In particular, our most prominent bound can be computed within 60 seconds for instances up to 15000 arcs. Interestingly, the GL type bound that is known to be one of the computationally cheapest bounds for quadratic optimization problems cannot be computed on such large instances within a time span of 1800 seconds.

This chapter is organized as follows. In Section 2.2, we formally introduce the QCCP and prove its $\mathcal{N} \mathcal{P}$-hardness. In Section 2.3, the linearization problem of the QCCP is introduced and several sufficient conditions for linearizability are derived. The general framework for the computation of the linearization-based bounds is discussed in Section 2.4. These bounds are used in Section 2.5 to construct an iterative bounding procedure for each sufficient condition. In Section 2.6, we consider the classical GL type bound and prove that it belongs to the family of linearization-based bounds. We also show how the iterative procedure for this linearization-based bound boils down to the computation of the strongest GL type bound in each step. In Section 2.7, we briefly discuss several other bounds from the literature. Numerical results are given in Section 2.8.

### 2.2 The quadratic cycle cover problem

An instance $\mathcal{I}$ of the QCCP is specified by the pair $\mathcal{I}=(G, Q)$, where $G=(N, A)$ is a directed graph with $n$ vertices and $m$ arcs and $Q=\left(q_{e f}\right) \in \mathbb{R}_{+}^{m \times m}$ is a cost matrix. The entries in $Q$ are such that $q_{\text {ef }}=0$ if $f$ is not a successor of $e$. In other words, the quadratic cost of two arcs $e$ and $f$ can be nonzero only if the starting node of $f$ equals the ending node of $e$. In case we also consider linear arc costs $p: A \rightarrow \mathbb{R}_{+}$, we can put these arc costs on the diagonal of the cost matrix $Q$.

Now, let $x \in\{0,1\}^{m}$ be a characteristic vector of a cycle cover, where $x_{e}=1$ if arc $e$ belongs to the cycle cover, and 0 otherwise. Then, the QCCP can be formulated as

$$
\begin{equation*}
O P T(Q):=\min \quad x^{\top} Q x \quad \text { s.t. } \quad x \in P, \tag{2.1}
\end{equation*}
$$

where $P$ denotes the set consisting of all disjoint cycle covers in $G$, i.e.,

$$
\begin{equation*}
P:=\left\{x \in\{0,1\}^{m}: \sum_{e \in \delta^{+}(i)} x_{e}=\sum_{e \in \delta^{-}(i)} x_{e}=1 \quad \forall i \in N\right\} . \tag{2.2}
\end{equation*}
$$

The set (2.2) equals the set of directed 2 -factors in $G$. For the existence of such a directed 2 factor in a directed graph, see e.g., Chiba and Yamashita [74].

Since $\mathbf{1}^{\top} x=n$ for all $x \in P$, the assumptions that $Q \geq \mathbf{0}$ and $p \geq \mathbf{0}$ are nonrestrictive. Indeed, we can always add to $Q$ or $p$ a positive multiple of $\mathbf{J}$ or $\mathbf{1}$, respectively, and solve an equivalent problem with a nonnegative cost structure. For that reason, we assume without loss of generality that $Q \geq \mathbf{0}$ and $p \geq \mathbf{0}$ throughout this chapter.

The quadratic cycle cover problem is $\mathcal{N} \mathcal{P}$-hard [138]. Also, the related angular-metric CCP and the MinRC3 problem are shown to be $\mathcal{N} \mathcal{P}$-hard [6] and strongly $\mathcal{N} \mathcal{P}$-hard [152], respectively. We now provide an alternative reduction that establishes strong $\mathcal{N} \mathcal{P}$-hardness which is based on a reduction from the quadratic assignment problem. We consider the Koopmans-Beckmann form of the QAP introduced in [238] with nonnegative cost structure. Let $F$ and $L$ be a set of $n$ facilities and $n$ locations, respectively, $w: F \times F \rightarrow \mathbb{R}_{+}$a weight function and $d: L \times L \rightarrow \mathbb{R}_{+}$a distance function. Without loss of generality, we assume that $d_{i i}=w_{i i}=0$ for all $i \in[n]$. Then, we search for a bijection $\pi: F \rightarrow L$ such that $\sum_{i=1}^{n} \sum_{j=1}^{n} d_{\pi(i) \pi(j)} w_{i j}$ is minimized. The QAP is $\mathcal{N} \mathcal{P}$-hard in the strong sense and not approximable within any constant factor [327].

Theorem 2.1. The $Q C C P$ is $\mathcal{N P}$-hard in the strong sense and cannot be approximated within a constant factor unless $\mathcal{P}=\mathcal{N} \mathcal{P}$.

Proof. Let $\mathcal{I}$ be an instance of the QAP, i.e., we have $F=\{1, \ldots, n\}, L=\left\{1^{\prime}, \ldots, n^{\prime}\right\}$ with $|L|=|F|=n$, functions $w: F \times F \rightarrow \mathbb{R}_{+}$and $d: L \times L \rightarrow \mathbb{R}_{+}$and a positive integer $K$. We create an instance $\mathcal{I}^{\prime}$ of the QCCP that is equivalent to the QAP-instance $\mathcal{I}$.

For the reduction we create a directed graph $G=(N, A)$ that consists of cells. A cell belongs to a single facility and consists of $n$ nodes, each of them corresponding to an assignment to one of the $n$ locations. These nodes are specified by the pairs $\left(i j^{\prime}\right)$ where $i \in F$ and $j^{\prime} \in L$. For each facility $i \in F$, we define a set of $n-1$ identical cells, which we call a group. The nodes corresponding to the same assignment within a group are placed on a directed cycle, where the arcs are oriented from cell $i$ to cell $i+1$ for $i=1, \ldots, n-2$, and from cell $n-1$ to cell 1 . In this way, we obtain $n$ cycles per group, which we call inner cycles. We set the interaction cost between each of the successive arcs within a group to zero for all groups. In Figure 2.1 the group corresponding to facility 1 is given. In a similar fashion we construct groups corresponding to the remaining facilities.


Figure 2.1: Group consisting of $n-1$ cells corresponding to facility 1.

We now specify the connections between the groups. For each group, each of its $n-1$
cells is connected to exactly one cell from the $n-1$ other groups. Hence, each cell of a group is connected to a different group, resulting in $\binom{n}{2}$ connections. Connecting the cells of two groups is done by introducing a connection node and a relink node. Starting from the first group, we draw an arc from every node of one of its cells to the connection node. Successively, we draw an arc from the connection node to all the nodes of one of the cells of the second group. The same is done for the relink node, now in the reverse direction. Figure 2.2 depicts an overview of the connection between the last cell of group $i$ and the first cell of group $j$. We denote the cycles between the groups by outer cycles. In Figure 2.2 solid arcs are used for the outer cycles, while the inner cycles are drawn using dashed arcs. A similar connection via connection and relink nodes exists for all other pairs of groups.

Observe that any arc in $G$ either belongs to an inner cycle or to several outer cycles. The quadratic cost of a pair of successive arcs $(e, f)$ where $e$ belongs to an inner cycle and $f$ to an outer cycle or vice versa, is set to $\infty$. It remains to specify the interaction cost between successive arcs on an outer cycle. We only specify the quadratic cost between the arcs entering and leaving the connection node, other costs are set to zero.

Let $i$ and $j$ be two distinct groups associated with facility $i$ and $j$, respectively. Let a node in group $i$ be given by $\left(i k^{\prime}\right)$ with $k^{\prime} \in L$. Similarly, a node in group $j$ is given by $\left(j l^{\prime}\right)$ with $l^{\prime} \in L$. Let $e_{i k^{\prime}}$ denote the arc between $\left(i k^{\prime}\right)$ and the connection node and let $f_{j l^{\prime}}$ denote the arc between the connection node and ( $j l^{\prime}$ ). Then the quadratic cost between $e_{i k^{\prime}}$ and $f_{j l^{\prime}}$ is defined as follows:

$$
q_{e_{i k^{\prime}}, f_{j l^{\prime}}}:= \begin{cases}d_{k^{\prime} l^{\prime}} w_{i j}+d_{l^{\prime} k^{\prime}} w_{j i} & \text { if } k^{\prime} \neq l^{\prime} \\ \infty & \text { otherwise } .\end{cases}
$$

We repeat this construction for any two connected cells. Figure 2.3 gives a simplified overview of $G$ for $n=4$. The circles in the center denote the connections between the cells, where the connection and relink nodes are drawn using the symbols ' $\bullet$ ' and ' $*$ ', respectively. The graph $G$ has $n^{2}(n-1)+2\binom{n}{2}=\mathcal{O}\left(n^{3}\right)$ nodes and $n^{2}(n-1)+4 n\binom{n}{2}=O\left(n^{3}\right)$ arcs.

It remains to show that there exists a cycle cover in $\mathcal{I}^{\prime}$ with cost at most $K$ if and only if there exists a feasible assignment in $\mathcal{I}$ with cost at most $K$.

First, we verify that a cycle cover with finite cost in $G$ corresponds to a feasible assignment of facilities and locations. Note that the connection and relink nodes must be covered by an outer cycle, since any other cycle would induce a cost of $\infty$. Besides the connection and relink nodes, this cycle contains two nodes that each correspond to an assignment of a different facility. Moreover, these facilities must be assigned to different locations, otherwise this implies an infinite cost. The nodes in a cell that are not covered by an outer cycle must


Figure 2.2: Connection between two cells of group $i$ and $j$.


Figure 2.3: Simplified overview of $G$ for $n=4$.
be covered by an inner cycle. Consequently, nodes on these inner cycles cannot belong to an outer cycle. Therefore, for each group exactly one location is selected to be on an outer cycle, i.e., each facility is assigned to some unique location. We conclude that a cycle cover with finite cost corresponds to a feasible assignment and vice versa.

Observe that the objective value of a feasible assignment in the QAP instance equals the total cost of the corresponding cycle cover in the QCCP instance. Namely, the latter cost equals the sum of quadratic costs incurred at the $\binom{n}{2}$ connection nodes. If facility $i$ (resp. $j$ ) where $i \neq j$ is assigned to location $k^{\prime}$ (resp. $l^{\prime}$ ) where $k^{\prime} \neq l^{\prime}$, then this cost equals $d_{k^{\prime} l^{\prime}} w_{i j}+d_{l^{\prime} k^{\prime}} w_{j i}$. Taking the sum over all connections, the total cost of the cycle cover equals $\sum_{i=1}^{n} \sum_{j=1}^{n} d_{\pi(i) \pi(j)} w_{i j}$ where $\pi: F \rightarrow L$ is the bijection corresponding to the assignment.

Since the QAP is strongly $\mathcal{N} \mathcal{P}$-hard and the numbers defined in the reduction are polynomially bounded (infinite costs can be replaced by an appropriate value which is polynomially bounded in the largest number and the size of $\mathcal{I}$ ), we conclude that the QCCP is strongly $\mathcal{N} \mathcal{P}$-hard.

Moreover, as the QAP cannot be approximated within any constant factor [327] and the reduction above is clearly gap preserving, the result follows.

### 2.3 The QCCP linearization problem

In this section, we formally introduce the linearization problem of the QCCP and derive various sufficient conditions for an instance of the QCCP to be linearizable. Several of these conditions are used later on to construct lower bounds for the optimal value of the QCCP.

Let us consider the (linear) cycle cover problem. Given a cost vector $p$, the CCP is the problem of finding a cycle cover of minimum linear cost. It can be written as follows:

$$
\begin{equation*}
\min _{x \in\{0,1\}^{m}}\left\{p^{\top} x: x \in P\right\}, \tag{2.3}
\end{equation*}
$$

where $P$ is given in (2.2). Since the constraint set of $P$ is totally unimodular, it follows that the CCP is solvable in polynomial time. We call an instance $\mathcal{I}=(G, Q)$ of the QCCP linearizable if there exists a cost vector $p \in \mathbb{R}^{m}$ such that $x^{\top} Q x=p^{\top} x$ for all cycle covers $x \in P$. If such a vector $p$ exists, we call $p$ a linearization vector of $Q$ for the QCCP.

The QCCP linearization problem can be stated as follows: Given an instance $\mathcal{I}=(G, Q)$ of the QCCP, verify whether it is linearizable and, if yes, compute a linearization vector $p$ of $Q$.

In the remaining part of this section we provide sufficient conditions for the cost matrix $Q$ to be linearizable. The first type of sufficient conditions for linearizability are related to the constant value property (CVP) for cost vectors or cost matrices. The definition associated with the CCP is stated below.
Definition 2.2. A cost vector $p$ satisfies the constant value property if $p^{\top} x=p^{\top} \bar{x}$ for all cycle covers $x, \bar{x} \in P$.

A similar definition holds for the quadratic version of the problem.
Definition 2.3. A cost matrix $Q$ satisfies the constant value property if $x^{\top} Q x=\bar{x}^{\top} Q \bar{x}$ for all cycle covers $x, \bar{x} \in P$.

When $Q$ satisfies the constant value property then $Q$ is linearizable, as stated by the following lemma.
Lemma 2.4. Assume that $Q$ satisfies the constant value property, i.e., $x^{\top} Q x=\xi$ where $\xi \in \mathbb{R}$ for all $x \in P$, then $Q$ is linearizable with cost vector $p$ defined as $p_{e}=\xi / n$ for all $e \in A$.

Proof. Follows immediately from the fact that any $x \in P$ has $n$ nonzero elements.
A more restricted version of the CVP is obtained when the interaction cost of a single arc with its successor or predecessor is constant for all cycle covers $x \in P$. We refer to these properties as the row and column constant value property, respectively, see [311]. In the following definition, $e^{+}$and $e^{-}$denote the starting and ending node of arc $e$, respectively.
Definition 2.5. A cost matrix $Q$ satisfies the row CVP if there exists some $b \in \mathbb{R}^{m}$ such that for all arcs $e \in A$ we have $q_{e f}=q_{e g}=b_{e}$ for all $f, g \in \delta^{+}\left(e^{-}\right)$and $q_{e f}=0$ otherwise. A cost matrix $Q$ satisfies the column CVP if there exists some $c \in \mathbb{R}^{m}$ such that for all arcs $e \in A$ we have $q_{f e}=q_{g e}=c_{e}$ for all $f, g \in \delta^{-}\left(e^{+}\right)$and $q_{f e}=0$ otherwise.

It is not hard to verify that an instance of the QCCP is linearizable if the cost matrix $Q$ satisfies the row or column CVP.
Proposition 2.6. If $Q$ satisfies the row $C V P$ or the column $C V P$, then $Q$ is linearizable.
Proof. We prove the case when $Q$ satisfies the row CVP. Assume that $b \in \mathbb{R}^{m}$ is such that for all arcs $e \in A, q_{e f}=q_{e g}=b_{e}$ for all $f, g \in \delta^{+}\left(e^{-}\right)$and $q_{e f}=0$ otherwise. Since $q_{e f}=0$ when $e$ and $f$ are not successors, we know that $x^{\top} Q x=\sum_{e \in A} \sum_{f \in \delta^{+}\left(e^{-}\right)} q_{e f} x_{e} x_{f}$. We have

$$
\sum_{e \in A} \sum_{f \in \delta^{+}\left(e^{-}\right)} q_{e f} x_{e} x_{f}=\sum_{e \in A} x_{e} b_{e} \sum_{f \in \delta^{+}\left(e^{-}\right)} x_{f}=\sum_{e \in A} x_{e} b_{e}=b^{\top} x .
$$

The proof for the column CVP is similar.
A matrix $Q \in \mathbb{R}^{m \times m}$ is called a sum matrix if there exist $b, c \in \mathbb{R}^{m}$ such that $q_{e f}=b_{e}+c_{f}$ for all $e, f$. A weak sum matrix is a matrix for which this property holds except for the entries on the diagonal, i.e., $q_{e f}=b_{e}+c_{f}$ for all $e \neq f$. The weak sum property is used as a condition for linearizability for several quadratic problems, see e.g., [215] and [311]. Since in this chapter we only incur a cost when two arcs are successive, we use a different form of the weak sum condition in which we only put a restriction on successive arcs. We call this condition the incident weak sum property.

Definition 2.7. A matrix $Q$ is called incident weak sum if there exist vectors $b, c \in \mathbb{R}^{m}$ such that $q_{e f}=b_{e}+c_{f}$ for all $e \in A, f \in \delta^{+}\left(e^{-}\right)$and $q_{e f}=0$ otherwise. If such vectors $b$ and $c$ exist, these vectors are called supporting vectors of $Q$.

If the cost matrix $Q$ is an incident weak sum matrix, then $Q$ is linearizable as stated by the following proposition.

Proposition 2.8. Let $Q$ be an incident weak sum matrix with supporting vectors $b, c \in \mathbb{R}^{m}$. Then, $Q$ is linearizable with cost vector $p=b+c$.

Proof. We show that for all $x \in P$ we have $x^{\top} Q x=p^{\top} x$ where $p=b+c$. Note that we have $x^{\top} Q x=\sum_{e \in A} \sum_{f \in \delta^{+}\left(e^{-}\right)} q_{e f} x_{e} x_{f}$, since $q_{e f}=0$ for all arcs that are not successors. Now,

$$
\begin{aligned}
\sum_{e \in A} \sum_{f \in \delta^{+}\left(e^{-}\right)} q_{e f} x_{e} x_{f} & =\sum_{e \in A} \sum_{f \in \delta^{+}\left(e^{-}\right)}\left(b_{e}+c_{f}\right) x_{e} x_{f} \\
& =\sum_{e \in A} b_{e} x_{e} \sum_{f \in \delta^{+}\left(e^{-}\right)} x_{f}+\sum_{f \in A} c_{f} x_{f} \sum_{e \in \delta^{-}\left(f^{+}\right)} x_{e} \\
& =\sum_{e \in A} b_{e} x_{e}+\sum_{f \in A} c_{f} x_{f}=\sum_{e \in A} p_{e} x_{e} .
\end{aligned}
$$

Here we use that $\sum_{f \in \delta^{+}\left(e^{-}\right)} x_{f}=\sum_{e \in \delta^{-}\left(f^{+}\right)} x_{e}=1$, by the structure of $x$.
From Proposition 2.8 it follows that the incident weak sum property is a sufficient condition for $Q$ to be linearizable. By including linear arc costs, this result remains valid, since we only increase the entries on the diagonal of $Q$.

Moreover, note that when $Q$ satisfies the row or column CVP, then $Q$ is an incident weak sum matrix. Next, we provide a special type of instance for which the cost matrix is not by definition linearizable, but for which we can still obtain its optimal value by solving a linear cycle cover problem.

Definition 2.9. A matrix $Q \in \mathbb{R}^{m \times m}$ is called a symmetric product matrix if $Q=a a^{\top}$ for some vector $a \in \mathbb{R}^{m}$.

Equivalently, we can say that $Q$ is a symmetric product matrix if it is a symmetric positive semidefinite matrix of rank one. An instance defined on such a cost matrix that is moreover nonnegative can be solved in polynomial time, as stated by the following proposition.

Proposition 2.10. Let $\mathcal{I}=(G, Q)$ be a $Q C C P$ instance where $Q$ is a nonnegative symmetric product matrix. Then, $\mathcal{I}$ can be solved in polynomial time.

Proof. If $Q$ is a nonnegative symmetric product matrix, then there exists a nonnegative vector $a \in \mathbb{R}_{+}^{m}$ such that $Q=a a^{\top}$. Then, $x^{\top} Q x=x^{\top} a a^{\top} x=\left(a^{\top} x\right)^{\top}\left(a^{\top} x\right)=\left(a^{\top} x\right)^{2}$ for all $x \in P$. Minimizing $x^{\top} Q x$ over all $x \in P$ is then equivalent to minimizing $a^{\top} x$ over all $x \in P$.

The conditions given in Definition 2.5 and 2.7 are such that $Q$ satisfies $q_{e f}=0$ when $f$ is not a successor of $e$. Below we derive two sufficient conditions for linearizability where $Q$ can have nonzero interaction cost between nonconsecutive arcs. Although these cost matrices do not meet the assumptions of the QCCP, we can still use them to derive strong bounds for the objective value of the original problem. This is addressed in Section 2.4.

Punnen et al. [311] introduce a generalized version of the weak sum property for the QTSP. Their approach can be applied to the QCCP. However, since Punnen et al. [311] prove the condition to hold for complete graphs, we provide a proof for general digraphs.

First, we define some new terminology. Instead of writing $q_{e f}$ for $e, f \in A$ we can also write $q_{i j, k l}$ with $(i, j),(k, l) \in A$. Let $N_{i}^{+}$denote the set of nodes $j$ for which there exists an $\operatorname{arc}(i, j) \in A$, i.e., $N_{i}^{+}:=\{j \in N:(i, j) \in A\}$. Similarly, let $N_{i}^{-}$be the set of nodes $j$ for which an $\operatorname{arc}(j, i) \in A$ exists, i.e., $N_{i}^{-}:=\{j \in N:(j, i) \in A\}$. Now, we introduce the notion of a generalized weak sum matrix.

Definition 2.11. $Q$ is called a generalized weak sum matrix if there exist $B, C \in \mathbb{R}^{m \times n}$ and $D, T \in \mathbb{R}^{n \times m}$ such that $q_{i j, k l}=b_{i j, k}+c_{i j, l}+d_{i, k l}+t_{j, k l}$ for all $i, j, k, l$ with $(i, j),(k, l) \in A$. If such $B, C, D$ and $T$ exist, these matrices are called supporting matrices of $Q$.

Now we can prove the following proposition.
Proposition 2.12. Let $Q$ be a generalized weak sum matrix supported by $B, C \in \mathbb{R}^{m \times n}$ and $D, T \in \mathbb{R}^{n \times m}$. Then, $Q$ is linearizable with cost vector $p$ given by $p_{i j}=\sum_{k=1}^{n} b_{i j, k}+$ $\sum_{k=1}^{n} c_{i j, k}+\sum_{k=1}^{n} d_{k, i j}+\sum_{k=1}^{n} t_{k, i j}$.
Proof. Let $\bar{b}_{i j}:=\sum_{k=1}^{n} b_{i j, k}, \bar{c}_{i j}:=\sum_{k=1}^{n} c_{i j, k}, \bar{d}_{i j}:=\sum_{k=1}^{n} d_{k, i j}, \bar{t}_{i j}:=\sum_{k=1}^{n} t_{k, i j}$ and $p_{i j}=$ $\bar{b}_{i j}+\bar{c}_{i j}+\bar{d}_{i j}+\bar{t}_{i j}$ for all $(i, j) \in A$. Then, for $x \in P$,

$$
\begin{aligned}
x^{\top} Q x= & \sum_{i \in N} \sum_{j \in N_{i}^{+}} \sum_{k \in N} \sum_{l \in N_{k}^{+}} q_{i j, k l} x_{i j} x_{k l} \\
= & \sum_{i \in N} \sum_{j \in N_{i}^{+}} \sum_{k \in N} \sum_{l \in N_{k}^{+}} b_{i j, k} x_{i j} x_{k l}+\sum_{i \in N} \sum_{j \in N_{i}^{+}} \sum_{l \in N^{\prime}} \sum_{k \in N_{l}^{-}} c_{i j, l} x_{i j} x_{k l} \\
& +\sum_{k \in N} \sum_{l \in N_{k}^{+}} \sum_{i \in N} \sum_{j \in N_{i}^{+}} d_{i, k l} x_{i j} x_{k l}+\sum_{k \in N} \sum_{l \in N_{k}^{+}} \sum_{j \in N} \sum_{i \in N_{j}^{-}} t_{j, k l} x_{i j} x_{k l} \\
= & \sum_{i \in N} \sum_{j \in N_{i}^{+}} x_{i j} \sum_{k \in N} b_{i j, k} \sum_{l \in N_{k}^{+}} x_{k l}+\sum_{i \in N} \sum_{j \in N_{i}^{+}} x_{i j} \sum_{l \in N} c_{i j, l} \sum_{k \in N_{l}^{-}} x_{k l} \\
& +\sum_{k \in N} \sum_{l \in N_{k}^{+}} x_{k l} \sum_{i \in N} d_{i, k l} \sum_{j \in N_{i}^{+}} x_{i j}+\sum_{k \in N} \sum_{l \in N_{k}^{+}} x_{k l} \sum_{j \in N} t_{j, k l} \sum_{i \in N_{j}^{-}} x_{i j} \\
= & \sum_{i \in N} \sum_{j \in N_{i}^{+}}\left(\bar{b}_{i j}+\bar{c}_{i j}+\bar{d}_{i j}+\bar{t}_{i j}\right) x_{i j}=\sum_{i \in N} \sum_{j \in N_{i}^{+}} p_{i j} x_{i j},
\end{aligned}
$$

where we use the fact that $\sum_{l \in N_{k}^{+}} x_{k l}=\sum_{k \in N_{l}^{-}} x_{k l}=\sum_{j \in N_{i}^{+}} x_{i j}=\sum_{i \in N_{j}^{-}} x_{i j}=1$ since $x$ is a cycle cover.

Note that an incident weak sum matrix can be seen as a special case of a generalized weak sum matrix. That is, for all $(i, j) \in A$ we set $b_{i j, j}=b_{i j}$ and $b_{i j, k}=0$ for all $k \neq j$ and for all $(k, l) \in A$ we set $t_{k, k l}=t_{k l}$ and $t_{j, k l}=0$ for all $j \neq k$. Moreover, let $C$ and $D$ be zero matrices. Then, $q_{i j, j l}=b_{i j, j}+c_{i j, l}+d_{i, j l}+t_{j, j l}=b_{i j}+t_{j l}$ for all $(i, j),(j, l) \in A$ and $q_{i j, k l}=0$ otherwise.

When $Q$ is a generalized weak sum matrix, we need $4 m n$ parameters to describe $Q$. This number can be reduced by considering a more restricted version of a generalized weak sum matrix.

Definition 2.13. A matrix $Q$ is called a restricted generalized weak sum matrix if there exist $C \in \mathbb{R}^{m \times n}, D \in \mathbb{R}^{n \times m}$ and $b, t \in \mathbb{R}^{m}$ such that $q_{i j, j l}=b_{i j}+c_{i j, l}+d_{i, j l}+t_{j l}$ for all $i, j, l$ with $(i, j),(j, l) \in A$ and $q_{i j, k l}=c_{i j, l}+d_{i, k l}$ otherwise. If such $C, D$ and $b, t$ exist, these are called supporting matrices and vectors, respectively.

We can show that restricted generalized weak sum matrices are linearizable.
Proposition 2.14. If $Q$ is a restricted generalized weak sum matrix with supporting matrices $C, D$ and supporting vectors $b, t$, then $Q$ is linearizable with vector $p$ given by $p_{i j}=b_{i j}+$ $\sum_{k=1}^{n} c_{i j, k}+\sum_{k=1}^{n} d_{k, i j}+t_{i j}$.
Proof. Define $B \in \mathbb{R}^{m \times n}$ and $T \in \mathbb{R}^{n \times m}$ as follows:

$$
\begin{aligned}
& B_{i j, k}= \begin{cases}b_{i j} & \text { if } k=j, \\
0 & \text { otherwise },\end{cases} \\
& T_{k, i j}= \begin{cases}t_{i j} & \text { if } k=i, \\
0 & \text { otherwise },\end{cases} \\
& \text { for all }(i, j) \in A, k \in N,
\end{aligned}
$$

Now the matrices $B, C, D$ and $T$ are such that they satisfy the conditions of Proposition 2.12. This implies that $Q$ is linearizable with vector $p^{\prime}$ given by $p_{i j}^{\prime}=\sum_{k=1}^{n} b_{i j, k}+\sum_{k=1}^{n} c_{i j, k}+$ $\sum_{k=1}^{n} d_{k, i j}+\sum_{k=1}^{n} t_{k, i j}$. Since $\sum_{k=1}^{n} b_{i j, k}=b_{i j}$ and $\sum_{k=1}^{n} t_{k, i j}=t_{i j}$ it follows that $Q$ is linearizable with vector $p:=b_{i j}+\sum_{k=1}^{n} b_{i j, k}+\sum_{k=1}^{n} d_{k, i j}+t_{i j}$.

### 2.4 Linearization-based bounds for the QCCP

In this section we show how the sufficient conditions for linearizability can be used to derive bounds for the optimal value of the QCCP. The construction of these bounds is provided in Section 2.4.1. Section 2.4.2 shows some preliminary numerical results of these bounding procedures.

### 2.4.1 Construction of linearization-based bounds

When an instance of the QCCP is linearizable, we can solve the problem in polynomial time by solving the corresponding linear cycle cover problem. When a cost matrix $Q$ is not linearizable, we can still use the sufficient conditions for linearizability to find lower bounds for the optimal value of the problem. This approach is introduced by Hu and Sotirov [217] for general binary quadratic problems. We here use tailor-made sufficient conditions for the QCCP, which lead to efficient lower bounds as we show later in the numerical results.

Before we proceed, let us recall the linear cycle cover problem. We introduce the matrix $U \in \mathbb{R}^{n \times m}$ with $U_{i, e}=1$ if node $i$ is the starting node of arc $e$ and 0 otherwise. Similarly,
we define $V \in \mathbb{R}^{n \times m}$ with $V_{i, e}=1$ if node $i$ is the ending node of arc $e$ and 0 otherwise. Since the matrix $\left[U^{\top} V^{\top}\right]^{\top}$ is totally unimodular, the optimal value of the CCP using cost vector $p$ equals

$$
\begin{align*}
& \operatorname{OPT}(p):=\min _{x \in \mathbb{R}^{m}}\left\{p^{\top} x:\left[\begin{array}{l}
U \\
V
\end{array}\right] x=\mathbf{1}_{2 n}, x \geq 0\right\}  \tag{2.4}\\
& =\max _{y \in \mathbb{R}^{2 n}}\left\{\mathbf{1}_{2 n}^{\top} y:\left[\begin{array}{ll}
U^{\top} & V^{\top}
\end{array}\right] y \leq p\right\} \text {, } \tag{2.5}
\end{align*}
$$

where $\mathbf{1}_{2 n} \in \mathbb{R}^{2 n}$ equals the vector of ones. Note that (2.3) and (2.4) are equivalent optimization problems. The corresponding dual problem is given in (2.5).

When $Q$ is linearizable with linearization vector $p$, we can find the optimal value of the QCCP by computing $O P T(p)$ using (2.4) or (2.5). If $Q$ is not linearizable, we can search for a linearizable matrix $\hat{Q}$ that is as close as possible to $Q$. To guarantee that $\hat{Q}$ is indeed linearizable, it should satisfy one of the sufficient conditions for linearizability derived in Section 2.3. We define the sets $S_{i}(Q)$, for $i \in[3]$, consisting of cost matrices $\hat{Q}$ such that $\hat{Q}$ is linearizable w.r.t. a sufficient condition for linearizability and $Q-\hat{Q}$ is elementwise nonnegative. We have

$$
\begin{aligned}
& S_{1}(Q):=\left\{\hat{Q} \in \mathbb{R}^{m \times m}: \hat{Q} \text { is an incident weak sum matrix and } Q-\hat{Q} \geq \mathbf{0}\right\} \\
& S_{2}(Q):=\left\{\hat{Q} \in \mathbb{R}^{m \times m}: \hat{Q}\right. \text { is a restricted generalized weak sum matrix and } \\
&Q-\hat{Q} \geq \mathbf{0}\}, \\
& S_{3}(Q):=\left\{\hat{Q} \in \mathbb{R}^{m \times m}: \hat{Q} \text { is a generalized weak sum matrix and } Q-\hat{Q} \geq \mathbf{0}\right\} .
\end{aligned}
$$

Remark 2.15. We do not consider the sets of cost matrices $Q$ satisfying the row or column CVP, since these are special types of incident weak sum matrices. These type of matrices are contained in $S_{1}$.

The set $S_{i}(Q)$ can be seen as the set of all the linearizable cost matrices of a specific type that are suitable for obtaining lower bounds for the optimal value of the problem. For this purpose, we define for $i \in[3]$ the set $\tau_{i}(Q)$ of cost vectors $\hat{p} \in \mathbb{R}^{m}$ as

$$
\tau_{i}(Q):=\left\{\hat{p} \in \mathbb{R}^{m}: \text { there exists a } \hat{Q} \in S_{i}(Q) \text { such that } \hat{p}^{\top} x=x^{\top} \hat{Q} x \text { for all } x \in P\right\} .
$$

It is clear that for all $i$ and all $\hat{p} \in \tau_{i}(Q)$ we have

$$
O P T(Q)=\min _{x \in P}\left\{x^{\top} Q x\right\} \geq \min _{x \in P}\left\{x^{\top} \hat{Q} x\right\}=\min _{x \in P}\left\{\hat{p}^{\top} x\right\}=O P T(\hat{p}) .
$$

So, indeed, $\operatorname{OPT}(\hat{p})$ is a lower bound for the optimal objective value of the QCCP for all $\hat{p} \in \tau_{i}(Q)$ and $i \in[3]$. By maximizing over all cost vectors in $\tau_{i}(Q)$, we obtain the strongest linearization-based bound with respect to the set $S_{i}(Q)$, which we denote by $v_{L B B}^{i}$, see also [217]:

$$
v_{L B B}^{i}:=\max _{\hat{p} \in \tau_{i}(Q)}\{O P T(\hat{p})\}=\max _{\substack{y \in \mathbb{R}^{n}  \tag{2.6}\\
\hat{p} \in \mathbb{R}^{m}}}\left\{\mathbf{1}_{2 n}^{\top} y:\left[\begin{array}{ll}
U^{\top} & V^{\top}
\end{array}\right] y \leq \hat{p}, \hat{p} \in \tau_{i}(Q)\right\} .
$$

The corresponding bounding approaches are denoted by $L B B 1, L B B 2$ and $L B B 3$, respectively.

Remark 2.16. Recall that the matrices in $S_{2}(Q)$ and $S_{3}(Q)$ can have nonzero interaction cost for nonconsecutive arcs, so they do not satisfy the assumptions on the cost matrix of the QCCP. Nevertheless, they can still be used to derive lower bounds for the original problem based on the above-mentioned construction. Also, we do not explicitly require that $\hat{Q} \geq \mathbf{0}$ and $\hat{p} \geq \mathbf{0}$ in the sets $S_{i}(Q)$ and $\tau_{i}(Q)$, respectively. However, by the assumption $Q \geq \mathbf{0}$, it follows that the zero matrix is in $S_{i}(Q)$ for all $i \in[3]$. Since we maximize over all linearization vectors in (2.6), the optimal linearization vector $p^{*}$ will be such that $\left(p^{*}\right)^{\top} x \geq 0$ for all $x \in P$. As a result, the bounds $v_{L B B}^{i}, i \in[3]$, are nonnegative under the assumption that $Q \geq \mathbf{0}$.

As long as the set $\tau_{i}(Q)$ is a polyhedron, the corresponding bound $v_{L B B}^{i}$ can be calculated by solving the linear programming problem (2.6). The sets $\tau_{i}(Q)$ for $i \in[3]$ are indeed nonempty polyhedra, since they can be described by a finite number of linear equalities and inequalities. These polyhedral descriptions are provided in Table 2.1.

| Set | Type of cost matrix | (In)equalities which descri | the set |
| :---: | :---: | :---: | :---: |
| $\tau_{1}(Q)$ | Incident weak sum matrix | $\begin{aligned} & b_{e}+c_{f} \leq q_{e f} \\ & \hat{p}_{e}=b_{e}+c_{e} \\ & b, c \in \mathbb{R}^{m} \end{aligned}$ | $\begin{aligned} & \forall e \in A, f \in \delta^{+}\left(e^{-}\right) \\ & \forall e \in A \end{aligned}$ |
| $\tau_{2}(Q)$ | Restricted generalized weak sum matrix | $\begin{aligned} & b_{i j}+c_{i j, l}+d_{i, j l}+t_{j l} \leq q_{i j, j l} \\ & c_{i j, l}+d_{i, k l} \leq q_{i j, k l} \\ & \hat{p}_{i j}=b_{i j}+\sum_{k=1}^{n} c_{i j, k}+\sum_{k=1}^{n} d_{k, i j}+t_{i j} \\ & b, t \in \mathbb{R}^{m}, C \in \mathbb{R}^{m \times n}, D \in \mathbb{R}^{n \times m} \end{aligned}$ | $\begin{aligned} & \forall(i, j),(j, l) \in A \\ & \forall(i, j),(k, l) \in A, j \neq k \\ & \forall(i, j) \in A \end{aligned}$ |
| $\tau_{3}(Q)$ | Generalized weak sum matrix | $\begin{gathered} b_{i j, k}+c_{i j, l}+d_{i, k l}+t_{j, k l} \leq q_{i j, k l} \\ \hat{p}_{i j}=\sum_{k=1}^{n} b_{i j, k}+\sum_{k=1}^{n} c_{i j, k} \\ \quad+\sum_{k=1}^{n} d_{k, i j}+\sum_{k=1}^{n} t_{k, i j} \\ B, C \in \mathbb{R}^{m \times n}, D, T \in \mathbb{R}^{n \times m} \end{gathered}$ | $\forall(i, j),(k, l) \in A$ $\forall(i, j) \in A$ |

Table 2.1: Polyhedral descriptions of the sets $\tau_{1}(Q), \tau_{2}(Q)$ and $\tau_{3}(Q)$.

By construction, we have $\tau_{1}(Q) \subseteq \tau_{2}(Q) \subseteq \tau_{3}(Q)$ for all cost matrices $Q$. Consequently, we can establish the following result about the quality of the corresponding linearizationbased bounds.

Theorem 2.17. For all instances of the QCCP, we have $v_{L B B}^{1} \leq v_{L B B}^{2} \leq v_{L B B}^{3}$.
Proof. Follows from construction.
Let $y, \hat{p}$ and $\hat{Q}$ be optimal to (2.6) and suppose $x^{*}$ is an optimal solution to the linear cycle cover problem $\min _{x \in P}\left\{\hat{p}^{\top} x\right\}$. Then,

$$
\begin{aligned}
O P T(Q) & =\min _{x \in P}\left\{x^{\top} Q x\right\} \leq\left(x^{*}\right)^{\top} Q x^{*}=\left(x^{*}\right)^{\top} \hat{Q} x^{*}+\left(x^{*}\right)^{\top}(Q-\hat{Q}) x^{*} \\
& =\hat{p}^{\top} x^{*}+\left(x^{*}\right)^{\top}(Q-\hat{Q}) x^{*}=v_{L B B}^{i}+\left(x^{*}\right)^{\top}(Q-\hat{Q}) x^{*} .
\end{aligned}
$$

Thus, $\left(x^{*}\right)^{\top}(Q-\hat{Q}) x^{*}$ provides an upper bound on the gap between the linearization-based bound and the optimal value of the QCCP.

Hu and Sotirov [217] argue that the linearization-based bounds can be improved by extending the sets $\tau_{i}(Q)$ using a skew-symmetric matrix $M$. That is, since each skewsymmetric matrix is linearizable, a matrix $\hat{Q}$ is linearizable if and only if $\hat{Q}+M$ is linearizable for all $M$ with $M+M^{\top}=0$. Using this, the set $\tau_{1}(Q)$ can be extended to:

Note that in $\tau_{1}^{\text {skew }}(Q)$ we only include skew-symmetric matrices whose support corresponds to the pairs of successive arcs in $G$, since adding dense skew-symmetric matrices would increase computational complexity. Since $\tau_{1}(Q) \subseteq \tau_{1}^{\text {seew }}(Q)$, it follows that we can obtain a stronger bound by maximizing over $\tau_{1}^{\text {skew }}(Q)$, see Section 2.8 . The same extension can be applied to any set $\tau_{i}(Q)$.

### 2.4.2 Preliminary results

In order to check the quality of the bounds derived above, we perform a preliminary numerical study. We create instances according to the $G(n, p)$ Erdős-Rényi model [125]. Here $n$ equals the number of nodes and $p$ equals the probability that an arc is included. We create instances for various values of $n$ and $p$. The interaction cost between any two successive arcs is drawn uniformly at random as an integer from $\{1, \ldots, 100\}$. In Table 2.2 we present the bounds $v_{L B B}^{1}, v_{L B B}^{2}$ and $v_{L B B}^{3}$ and their computation times in seconds. Recall that these bounds are computed by solving (2.6) using the polyhedral descriptions of Table 2.1, where we use the solver CPLEX 12.6. Moreover, the column OPT denotes the optimum value of the QCCP instance computed by the mixed-integer quadratic programming solver of CPLEX 12.6 (after convexifying the objective function). Of course, this value can only be obtained for small graphs. Experiments are performed using a PC with an $\operatorname{Intel}(\mathrm{R})$ Core(TM) i5-6500 CPU, 3.20 GHz and 8 GB memory. The maximum computation time is set to 3600 seconds and we put 'n.a.' in the table when this maximum is reached before a solution is obtained.

By construction, the optimal solution has always an integer objective value. Therefore, we round up all bounds to the nearest integer. The results of Table 2.2 show that the linearization-based bounds $L B B 1, L B B 2$ and $L B B 3$ do not differ significantly, especially for the larger instances. At the same time, the computation times differ significantly. It turns out that $L B B 1$ is most efficient. Therefore, this bound can be preferred when taking both quality and efficiency into account.

### 2.5 Reformulated LBB approach

In this section we discuss how a reformulation of the cost matrix can be used to obtain a nondecreasing sequence of lower bounds that are based on the linearization-based bound. It is important to note that one can construct such a bounding procedure using any sufficient condition for linearizability, not only the ones discussed in Section 2.4.

Suppose we are given a sufficient condition for linearizability. Let $S(Q)$ and $\tau(Q)$ be as in Section 2.4, but now for a general sufficient condition. That is, $S(Q)$ is the set consisting of all linearizable cost matrices $\hat{Q}$ of this type with $\hat{Q} \leq Q$ and $\tau(Q)$ consists of the corresponding

| $p$ | $n$ | $m$ | OPT | $L B B 1$ |  | $L B B 2$ |  | $L B B 3$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | bound | time | bound | time | bound | time |
| 0.1 | 20 | 44 | 923 | 923 | 0.047 | 923 | 0.264 | 923 | 0.051 |
|  | 25 | 76 | 1039 | 971 | 0.005 | 971 | 0.477 | 999 | 1.227 |
|  | 30 | 100 | 1082 | 1066 | 0.013 | 1066 | 0.899 | 1082 | 0.787 |
| 0.3 | 15 | 61 | 485 | 478 | 0.010 | 478 | 0.635 | 485 | 0.714 |
|  | 20 | 118 | 438 | 377 | 0.031 | 384 | 1.265 | 390 | 77.21 |
|  | 25 | 172 | 382 | 291 | 0.050 | 295 | 2.531 | 295 | 869.0 |
| 0.5 | 15 | 116 | 226 | 215 | 0.034 | 215 | 1.407 | 215 | 90.39 |
|  | 20 | 177 | 255 | 189 | 0.059 | 190 | 12.05 | 190 | 2105 |
|  | 25 | 306 | n.a. | 172 | 0.516 | 173 | 353.6 | n.a. | 3600 |
| 0.7 | 15 | 149 | 173 | 127 | 0.017 | 128 | 0.986 | 128 | 580.5 |
|  | 20 | 263 | n.a. | 116 | 0.094 | 116 | 7.778 | n.a. | 3600 |
|  | 25 | 396 | n.a. | 129 | 0.194 | 129 | 18.26 | n.a. | 3600 |

Table 2.2: Bounds and computation times in seconds of linearization-based bounds on ErdősRényi instances.
linearization vectors. Moreover, we assume that the set $\tau(Q)$ is a polyhedral set and that $\mathbf{0} \in \tau(Q)$ whenever $Q \geq \mathbf{0}$. This latter assumption is reasonable, since $Q \geq \mathbf{0}$ implies that the zero matrix is a valid linearizable underapproximator of $Q$, see also Remark 2.16. Let $Q_{0}$ be the initial cost matrix. If $\hat{Q}_{0} \in S\left(Q_{0}\right)$, we know there exists some $p_{1} \in \tau\left(Q_{0}\right)$ such that $x^{\top} \hat{Q}_{0} x=p_{1}^{\top} x$ for all $x \in P$. This leads to the following reformulation of the objective function:

$$
\begin{equation*}
x^{\top} Q_{0} x=x^{\top} \hat{Q}_{0} x+x^{\top}\left(Q_{0}-\hat{Q}_{0}\right) x=p_{1}^{\top} x+x^{\top}\left(Q_{0}-\hat{Q}_{0}\right) x \tag{2.8}
\end{equation*}
$$

for all cycle covers $x \in P$. By letting $Q_{1}:=Q_{0}-\hat{Q}_{0}$ be the residual matrix, we obtain $x^{\top} Q_{0} x=p_{1}^{\top} x+x^{\top} Q_{1} x$ for all $x \in P$, where $Q_{1} \geq \mathbf{0}$ by construction. The vector $p_{1}$ is taken to be the largest linearization vector of $Q_{0}$ (see (2.6)), resulting in the linearizationbased bound $v_{L B B}$.

Now we can proceed in a similar way by considering the linearization problem of the residual cost matrix $Q_{1}$. We search for the optimal $\hat{Q}_{1} \in S\left(Q_{1}\right)$ and corresponding linearization vector $p_{2} \in \tau\left(Q_{1}\right)$ according to (2.6). Let $Q_{2}=Q_{1}-\hat{Q}_{1}$ be the new residual matrix. Then the objective function can be reformulated as $x^{\top} Q_{0} x=\left(p_{1}+p_{2}\right)^{\top} x+x^{\top} Q_{2} x$, where $p_{2}^{\top} x=x^{\top} \hat{Q}_{1} x$. Since $\mathbf{0} \in \tau\left(Q_{1}\right)$ due to $Q_{1} \geq \mathbf{0}$, it follows that the optimal linearization vector $p_{2}$ satisfies $p_{2}^{\top} x \geq 0$ for all $x \in P$. This implies that obtaining the linearization-based bound with respect to $p_{1}+p_{2}$ provides a possibly stronger bound than with respect to $p_{1}$. This procedure can be repeated to obtain a sequence of non-decreasing bounds.

Unfortunately, we cannot expect to find a vector $p_{2}$ for which this bound has strictly improved. Due to the polyhedrality of $\tau\left(Q_{0}\right)$, we can show that $p_{1}+p_{2} \in \tau\left(Q_{0}\right)$, which would imply that $p_{1}$ is not the optimal solution to (2.6). Thus, applying this procedure iteratively, the resulting sequence of bounds remains constant after the first iteration. To overcome this issue, we need to reformulate the residual cost matrix in each step.

In the literature, various iterative bounding procedures are proposed [62, 310, 324, 325]. In this chapter we introduce a new approach that is different in two ways. First, the existing bounding procedures are mainly based on the classical Gilmore-Lawler type bound. Our approach is based on general sufficient conditions for linearizability and we can show that the

Gilmore-Lawler type bounding procedure is a special case of this approach, see Section 2.6. Second, the existing bounding procedures mostly use a fixed reformulation of the cost matrix in each iteration. However, using a fixed reformulation is in general not the best one can do. Here, we search for the reformulation of the cost matrix that results in the strongest bound in the next iteration. For this purpose, we define the notion of an equivalent representation of a matrix, see e.g., [310].

Definition 2.18. Let $(G, Q)$ be an instance of the QCCP. Then, $(G, W)$ is an equivalent representation of $(G, Q)$ if $x^{\top} Q x=x^{\top} W x$ for all $x \in P$.

If there is no confusion about the graph $G$ under consideration, we say that $W$ is an equivalent representation of $Q$. It is easy to verify that a matrix $W=\left(w_{e f}\right) \in \mathbb{R}^{m \times m}$ is an equivalent representation of $Q$ if for all $e, f \in A$ we have $w_{e f}+w_{f e}=q_{e f}+q_{f e}$. Here, we focus on a specific type of equivalent representation, which we call an $\eta$-equivalent representation of $Q$.

Definition 2.19. Given $\eta \in[0,1]$, an equivalent representation $Q^{\eta}:=\eta Q+(1-\eta) Q^{\top}$ of $Q$ is called an $\eta$-equivalent representation.

It follows that if $W$ and $Q$ are equivalent representations, then a linearization of $W$ is also a linearization of $Q$ and vice versa.

Instead of considering the linearization problem of the residual matrix $Q_{1}$, we can consider the linearization problem of $Q_{1}^{\eta}$ for some $\eta \in[0,1]$. Since $Q_{1}^{\eta}$ has a different structure than $Q_{1}$, it is in general possible to find a linearizable matrix $\hat{Q}_{1} \in S\left(Q_{1}^{\eta}\right)$ and a corresponding linearization vector that result in a strictly stronger bound.

As already mentioned above, many approaches in the literature are based on taking a fixed value for $\eta$, e.g., $\eta=\frac{1}{2}$ which corresponds to the case of symmetrizing. This does not give the best bound in general. Instead, we search for $\eta \in[0,1]$ that results in the strongest bound in each iteration. Suppose we are in step $k$ of the algorithm in which we consider the linearization problem of the residual matrix $Q_{k-1}$. Then the optimal equivalent representation of $Q_{k-1}$ and its corresponding vector $p_{k} \in \tau\left(Q_{k-1}^{\eta}\right)$ can be computed simultaneously by solving the following problem:

$$
r_{k}:=\max _{\substack{y \in \mathbb{R}^{2 n}  \tag{2.9}\\
p_{k} \mathbb{R}^{m} \\
\eta \in[0,1]}}\left\{\mathbf{1}_{2 n}^{\top} y:\left[\begin{array}{ll}
U^{\top} & V^{\top}
\end{array}\right] y \leq p_{k}, p_{k} \in \tau\left(Q_{k-1}^{\eta}\right)\right\},
$$

which equals the additional amount of quadratic cost that can be linearized in iteration $k$. Note that if the set $\tau\left(Q_{k-1}\right)$ is a polyhedron, then $\tau\left(Q_{k-1}^{\eta}\right)$ is also a polyhedron and the corresponding problem (2.9) can be solved in polynomial time. For the sufficient conditions mentioned in Section 2.4 this is indeed the case.

Finally, we provide a new bounding procedure that is based on iteratively finding the best $\eta$-equivalent representation of the residual cost matrix and its optimal linearizable matrix. Starting with $Q_{0}=Q$, the goal is to find the best linearizable matrix $\hat{Q}_{k-1}$ of an equivalent representation of the residual matrix $Q_{k-1}$ and its corresponding linearization vector $p_{k}$. We let $p_{0}=\mathbf{0}$ and iteratively update $d_{k}=d_{k-1}+p_{k}$, which equals the total linearization vector of an approximation of $Q$. In each iteration we compute $r_{k}$ by (2.9), which is by construction nonnegative due to $\mathbf{0} \in \tau\left(Q_{k}^{\eta}\right)$ for all $k$. The final bound is given by the sum of all $r_{k}$ 's, which we call the reformulation-based bound. The procedure is given in Algorithm 2.1.

```
Algorithm 2.1 \(L B B\) Reformulation algorithm
    \(Q_{0}=Q, d_{0}=\mathbf{0}, k=1, r_{0}=\infty\)
    while \(r_{k-1}>0\) do
        Compute \(r_{k}, p_{k}\) and \(\eta\) using (2.9).
        Construct the linearizable matrix \(\hat{Q}_{k-1}\) using the optimal solution of (2.9). \(\triangleright\) See Remark 2.20
        \(Q_{k} \leftarrow \eta Q_{k-1}+(1-\eta) Q_{k-1}^{\top}-\hat{Q}_{k-1}\)
        \(d_{k} \leftarrow d_{k-1}+p_{k}\)
        \(k \leftarrow k+1\)
    end while
    \(v_{R B B}=\sum_{i=1}^{k-1} r_{i}\)
    return \(d_{k}, v_{R B B}\)
```

Remark 2.20. Note that steps 3 and 4 of Algorithm 1 depend on the specific sufficient condition for linearizability. For instance, for the incident weak sum condition we construct in step 4 the linearizable matrix $\hat{Q}_{k-1}=\left(\left(\hat{q}_{k-1}\right)_{e f}\right)$ in the following way $\left(\hat{q}_{k-1}\right)_{e f}:=b_{e}+c_{f}$ for all $e \in A, f \in \delta^{+}\left(e^{-}\right)$and $\left(\hat{q}_{k-1}\right)_{e f}:=0$ otherwise, where $b, c \in \mathbb{R}^{m}$ are obtained from (2.9).

Hu and Sotirov [217] show that in the case that the linearizable matrix $\hat{Q}$ is of the form $\hat{Q}=\left[U^{\top} V^{\top}\right] Y+\operatorname{Diag}(z)$ for some $Y \in \mathbb{R}^{2 n \times m}$ and $z \in \mathbb{R}^{m}$, the bound $v_{R B B}$ is dominated by the solution of the first level RLT relaxation introduced by Adams and Sherali [3, 4]. Here RLT stands for reformulation-linearization technique. In [217] it is moreover shown that the first level RLT bound, denoted by $v_{R L T 1}$, can be obtained by searching for the optimal linearizable matrix $\hat{Q}$ of the form $\hat{Q}=\left[U^{\top} V^{\top}\right] Y+M+\operatorname{Diag}(z)$ where $M$ is a skew-symmetric matrix.

Our preliminary numerical results show that the above algorithm does not improve significantly the $L B B 1$ bound. However, in the next section we show that our approach outperforms known iterative approaches related to the Gilmore-Lawler type bounds.

### 2.6 The Gilmore-Lawler type bound

In this section we consider the classical Gilmore-Lawler type bound. The GL procedure is a well-known approach to construct lower bounds for quadratic $0-1$ optimization problems, see e.g., $[172,246,324,325]$. We provide a compact formulation of the GL type bound that can be used to compute the bound by a single LP-problem, instead of solving $m$ subproblems. Moreover, we show that this bound in fact belongs to the family of linearization-based bounds. Therefore, based on the results of Section 2.5, we provide a bounding procedure that computes the best GL type bound in each step of the algorithm. We conclude this section by testing this new bounding procedure on some preliminary test instances.

### 2.6.1 The classical GL type bound

In the objective function of the QCCP, see (2.1), we have the quadratic term $x_{e} x_{f}$ for each two arcs $e, f \in A$ placed in succession on a cycle. To get rid of this quadratic term, for each given arc $e \in A$ potentially in the solution, we consider the cycle cover containing $e$ with minimum interaction cost with $e$. We denote this minimum contribution of arc $e$ to a solution by $z_{e}$. In particular, for all $e \in A$ we have

$$
\begin{equation*}
z_{e}:=\min \left\{Q_{e,:} x: x \in P, x_{e}=1\right\} \tag{e}
\end{equation*}
$$

where $Q_{e, \text { : }}$ denotes the $e$ th row of the cost matrix $Q$. The feasible set of $\left(P_{e}\right)$ equals the set of all node-disjoint cycle covers containing arc $e$. If this set is empty, then we set $z_{e}$ equal
to 0 since arc $e$ cannot contribute to a cycle cover.
Let $z \in \mathbb{R}^{m}$ be the vector consisting of the elements $z_{e}$ for all $e \in A$. Then, the classical GL type bound is obtained by solving the following CCP:

$$
\begin{equation*}
v_{G L}:=\min \left\{z^{\top} x: x \in P\right\} . \tag{GL}
\end{equation*}
$$

Note that the constraint matrices of $\left(P_{e}\right)$ and $(G L)$ are totally unimodular. For this reason, we can drop the integrality constraints and solve $(G L)$ and $\left(P_{e}\right)$ for all $e \in A$ as linear programming problems.

Besides computing the GL type bound by solving $(G L)$ and $\left(P_{e}\right)$ for all $e \in A$, we can also obtain its value by solving an integer linear programming (ILP) problem. The problem ( $G L_{I L P}$ ) is defined as follows:

$$
\begin{array}{rll}
\left(G L_{I L P}\right) & \min & \sum_{e \in A} \sum_{f \in A} q_{e f} y_{e f} \\
\text { s.t. } & \sum_{f \in \delta^{+}(i)} y_{e f}=\sum_{f \in \delta^{-}(i)} y_{e f}=x_{e} & \\
& y_{e e}=x_{e} & \forall i \in N, \forall e \in A \\
& y_{e f} \in\{0,1\}, x \in P &  \tag{2.12}\\
& \forall e \in A \\
& \forall e, f \in A .
\end{array}
$$

It follows from the constraints that if $x_{e}=1$, then $y_{e,:}:=\left[\begin{array}{lll}y_{e 1} & \ldots & y_{\text {em }}\end{array}\right]$ is the characteristic vector of the cheapest cycle cover containing arc $e$ and if $x_{e}=0$, then $y_{e, \text { : }}$ equals the zero vector.

Let $\left(C G L_{I L P}\right)$ be the continuous relaxation of $\left(G L_{I L P}\right)$. In this continuous relaxation we can omit the upper bounds on $x_{e}$ and $y_{e f}$ for all $e, f \in A$, since these are implied by the other constraints and the nonnegativity of $x$ and $y$. We can compute the GL type bound by solving $\left(C G L_{I L P}\right)$ as stated by the following theorem. This theorem is based on a similar result for the quadratic minimum spanning tree problem, see [325].

Theorem 2.21. The optimal value of $\left(C G L_{I L P}\right)$ equals $v_{G L}$.
Proof. Let $\lambda_{e, i}$ and $\alpha_{e, i}$ be the dual variables corresponding to constraints (2.10), i.e. $\lambda_{e, i}$ corresponds to $\sum_{f \in \delta^{+}(i)} y_{e f}=x_{e}$ and $\alpha_{e, i}$ corresponds to $\sum_{f \in \delta^{-}(i)} y_{e f}=x_{e}$. Similarly, let $\mu_{i}$ and $\gamma_{i}$ be the dual variables corresponding to the first and second equalities of constraints (2.2), and $\theta_{e}$ the dual variable corresponding to constraints (2.11). The dual problem of $\left(C G L_{I L P}\right)$ is as follows:

$$
\begin{array}{rlr}
\left(D C G L_{I L P}\right) \max & \sum_{i \in N} \mu_{i}+\sum_{i \in N} \gamma_{i} & \\
\text { s.t. } & \lambda_{e, f^{+}}+\alpha_{e, f^{-}} \leq q_{e f} \\
& \lambda_{e, e^{+}}+\alpha_{e, e^{-}}+\theta_{e} \leq q_{e e} & \forall e, f \in A, f \neq e \\
& -\sum_{i \in N} \lambda_{e, i}-\sum_{i \in N} \alpha_{e, i}+\gamma_{e^{-}}+\mu_{e^{+}}-\theta_{e} \leq 0 \quad \forall e \in A  \tag{2.15}\\
& \forall e \in A .
\end{array}
$$

The constraints (2.15) can be rewritten as $\gamma_{e^{-}}+\mu_{e^{+}} \leq \sum_{i \in N} \lambda_{e, i}+\sum_{i \in N} \alpha_{e, i}+\theta_{e}$ for all $e \in A$. In order to maximize the objective function of ( $D C G L_{I L P}$ ), we maximize the righthand side of this inequality subject to constraints (2.13)-(2.14). This gives for each $e \in A$
the following subproblem:
$\left(D C P_{e}\right) \quad z_{e}^{\prime}:=\max \left\{\sum_{i \in N} \lambda_{e, i}+\sum_{i \in N} \alpha_{e, i}+\theta_{e}:(2.13),(2.14)\right\}$.
For each fixed $e \in A$ the subproblem given above equals the dual of the continuous relaxation of $\left(P_{e}\right)$. By strong duality we know $z_{e}^{\prime}=z_{e}$ for all $e \in A$. Substitution of this term into constraint (2.15) gives a rewritten formulation for ( $D C G L_{I L P}$ ):

$$
\max \left\{\sum_{i \in N} \mu_{i}+\sum_{i \in N} \gamma_{i}: \mu_{e^{+}}+\gamma_{e^{-}} \leq z_{e} \quad \forall e \in A\right\} .
$$

This problem equals the dual of the continuous relaxation of ( $G L$ ). Because of strong duality, it follows that the optimal objective value of $\left(C G L_{I L P}\right)$ equals $v_{G L}$.

We can show that the Gilmore-Lawler type bound for the QCCP in fact belongs to the family of linearization-based bounds introduced in Section 2.4. That is, we can obtain $v_{G L}$ by searching for a linearizable cost matrix $\hat{Q}$ of a specific type that is as close as possible to $Q$. The required linearizability condition on $\hat{Q}$ is given below, and it differs from the sufficient conditions presented in Section 2.3.

Proposition 2.22. If there exists $B, C \in \mathbb{R}^{m \times n}$ and $t \in \mathbb{R}^{m}$ such that $q_{e f}=B_{e, f+}+C_{e, f-}$ for $e \neq f$ and $q_{e e}=B_{e, e^{+}}+C_{e, e^{-}}+t_{e}$ for all $e \in A$, then $Q$ is linearizable with vector $p$ given by $p_{e}=t_{e}+\sum_{i=1}^{n} B_{e, i}+\sum_{i=1}^{n} C_{e, i}$.

Proof. Let $\tilde{Q}=\left(\tilde{q}_{e f}\right)$ be defined as $\tilde{q}_{e f}=B_{e, f+}+C_{e, f^{-}}$for all $e, f \in A$. Then $\tilde{Q}$ can be seen as a generalized weak sum matrix where $D$ and $T$ are equal to the zero matrix, see Definition 2.11. According to Proposition 2.12, $\tilde{Q}$ is linearizable with supporting vector $\tilde{p}=\sum_{i=1}^{n} B_{e, i}+\sum_{i=1}^{n} C_{e, i}$. Since $Q=\tilde{Q}+\operatorname{Diag}(t)$, it follows that $Q$ is linearizable with supporting vector $p$ given by $p_{e}=t_{e}+\sum_{i=1}^{n} B_{e, i}+\sum_{i=1}^{n} C_{e, i}$.

Similar to the notation used in Section 2.4, let $S_{G L}(Q)$ denote the set of all linearizable cost matrices $\hat{Q} \in \mathbb{R}^{m \times m}$ that satisfy $\hat{Q} \leq Q$ and the conditions of Proposition 2.22. Moreover, let $\tau_{G L}(Q)$ be the following polyhedron:

$$
\tau_{G L}(Q):=\left\{\hat{p} \in \mathbb{R}^{m}: \quad \begin{array}{c}
\text { there exists a } \hat{Q} \in S_{G L}(Q) \text { such }  \tag{2.16}\\
\text { that } \hat{p}^{\top} x=x^{\top} \hat{Q} x \text { for all } x \in P
\end{array}\right\}
$$

and

$$
v_{L B B}^{G L}:=\max _{\substack{y \in \mathbb{R}^{2 n}  \tag{2.17}\\
\hat{p} \in \mathbb{R}^{m}}}\left\{\mathbf{1}_{2 n}^{\top} y:\left[\begin{array}{ll}
U^{\top} & V^{\top}
\end{array}\right] y \leq \hat{p}, \hat{p} \in \tau_{G L}(Q)\right\} .
$$

Now we prove the main result of this section which states that the classical Gilmore-Lawler type bound can be seen as a special case of linearization-based bound.

Theorem 2.23. We have $v_{L B B}^{G L}=v_{G L}$.

Proof. By using the polyhedral description of $S_{G L}(Q)$ following from Proposition 2.22, the optimization problem in (2.17) can be written as follows:

$$
\left.\begin{array}{rll}
v_{L B B}^{G L}=\max & \sum_{i=1}^{2 n} y_{i} & \\
\text { s.t. } & {\left[U^{\top} \quad V^{\top}\right.}
\end{array}\right] y \leq \hat{p} \quad \forall e, f \in A, f \neq e
$$

We show that this optimization problem is equivalent to $\left(D C G L_{I L P}\right)$, the dual problem of the continuous relaxation of $\left(G L_{I L P}\right)$. Take $B_{e, i}=\lambda_{e, i}$ and $C_{e, i}=\alpha_{e, i}$ for all $e \in A$ and $i \in N$, where $\lambda$ and $\alpha$ denote the dual vectors belonging to constraints (2.10). Similarly, let $t=\theta$ where $\theta$ equals the dual vector to constraints (2.11). Finally, let $y=\left[\mu^{\top} \gamma^{\top}\right]^{\top}$, where $\mu$ and $\gamma$ are the dual variables belonging to constraints (2.2). By substitution of these variables and combining constraints (2.19) and (2.22), we obtain the problem ( $D C G L_{I L P}$ ), i.e., the dual of $\left(C G L_{I L P}\right)$. Thus, we have $v_{L B B}^{G L}=v_{G L}$.

Theorem 2.23 shows that the GL type bound belongs to the family of linearization-based bounds. This is also shown by Hu and Sotirov [217] and Rostami et al. [324], however our proof is very different as we exploit the fact that $v_{G L}$ can be obtained by solving an LP problem, i.e., $\left(C G L_{I L P}\right)$. Additionally, we show here that the computation of the GL type bound is equivalent to the search for the optimal linearizable cost matrix $\hat{Q}$ satisfying the properties of Proposition 2.22.

### 2.6.2 The best Gilmore-Lawler type bound

Section 2.6.1 shows that the calculation of the classical GL type bound fits in the general framework discussed in Section 2.4. In this section we apply the reformulation procedure of Section 2.5 to the GL type bound. We also show that our approach outperforms several iterative approaches from the literature.

In order to apply Algorithm 2.1 to the sufficient condition for linearizability of Proposition 2.22, we need to define how to calculate $r_{k}$ for each iteration $k$, see (2.9). We rewrite the set $\tau_{G L}(Q)$, see (2.16), as follows:

$$
\tau_{G L}(Q)=\left\{\hat{p} \in \mathbb{R}^{m}: t \in \mathbb{R}^{m}, B, C \in \mathbb{R}^{m \times n},(2.20),(2.21),(2.22)\right\}
$$

which is clearly a polyhedron. Then for all $k \geq 1$ we calculate the additional amount of quadratic cost that is linearized by solving:

$$
r_{k}:=\max _{\substack{y \in \mathbb{R}^{2 n}  \tag{2.24}\\
p_{k} \mathbb{R}^{m} \\
\eta \in[0,1]}}\left\{\mathbf{1}_{2 n}^{\top} y:\left[\begin{array}{ll}
U^{\top} & V^{\top}
\end{array}\right] y=p_{k}, p_{k} \in \tau_{G L}\left(Q_{k-1}^{\eta}\right)\right\} .
$$

Observe that, opposed to the constraints in (2.9), we replaced the constraint $\left[U^{\top} V^{\top}\right] y \leq p_{k}$
by an equality constraint. This does not change the value of $r_{k}$. To verify this, suppose we solve (2.24) using the inequality constraint $\left[U^{\top} V^{\top}\right] y \leq p_{k}$ and let $\hat{y}, \hat{p}_{k}$ and $\hat{t}$ be the corresponding optimal solutions. Let $e \in A$ be such that the inequality constraint is satisfied with strict inequality. Then, without changing $\hat{y}$, we can reduce $\hat{t}_{e}$ (and thus $\hat{p}_{e}$ ) such that we get equality for $e \in A$. Although it changes the linearization vector $\hat{p}$, the resulting bound remains equal. To verify this, notice that only the left-hand side of constraint (2.21) is decreased, so the solution is still feasible and the optimal value $r_{k}$ remains unchanged. From this, it follows that one may replace $\left[U^{\top} V^{\top}\right] y \leq p_{k}$ by an equality constraint and solve $r_{k}$ as in (2.24).

Algorithm 2.1 using (2.24) in step 3 gives a new bounding procedure for the QCCP. We call the resulting bound the reformulated GL type bound ( $R G L$ ) and denote its value by $v_{R G L}$. By construction, it iteratively computes the best Gilmore-Lawler type bound among all $\eta$-equivalent representations of the cost matrix.

The algorithm proposed in this section satisfies another interesting property, namely the vectors $d_{k}$ satisfy the constant value property for all $k \geq 0$. This is an important property for linearizability because the set of linearizable cost matrices for combinatorial optimization problems with interaction costs can be characterized by the constant value property, under certain conditions, see [248].

Theorem 2.24. All $d_{k}$ where $k \geq 0$ computed during the $R G L$ approach, satisfy the constant value property, i.e., we have $d_{k}^{\top} x=d_{k}^{\top} \bar{x}$ for all feasible cycle covers $x, \bar{x} \in P$.

Proof. We apply a proof by induction on $k$. Note that the vector $d_{0}$ equals the $m \times 1$ vector of zeros which trivially satisfies the constant value property.

Now assume that the induction hypothesis is true for iteration $k-1$, i.e., $d_{k-1}^{\top} x=d_{k-1}^{\top} \bar{x}$ for all feasible cycle covers $x, \bar{x} \in P$. In iteration $k$ we solve (2.24). Let $\hat{y} \in \mathbb{R}^{2 n}$ and $\hat{p} \in \mathbb{R}^{m}$ be the optimal variables for this problem and split $\hat{y}=\left[\mu^{\top} \lambda^{\top}\right]^{\top}$ with $\mu, \lambda \in \mathbb{R}^{n}$. It follows that $\left[U^{\top} V^{\top}\right] \hat{y}=U^{\top} \mu+V^{\top} \lambda=\hat{p}$. Now let $x \in P$ be any feasible cycle cover in $G$. Then we can sum up the rows of this system of equalities for all arcs $e \in A$ in the cycle cover implied by $x$ :

$$
\sum_{e \in A: x_{e}=1}\left(\mu_{e^{+}}+\lambda_{e^{-}}\right)=\sum_{e \in A: x_{e}=1} \hat{p}_{e}, \quad \text { or equivalently, } \quad \sum_{i \in N} \mu_{i}+\sum_{i \in N} \lambda_{i}=\hat{p}^{\top} x
$$

where we use the fact that each vertex is visited exactly once on a cycle cover. So the quantity $\hat{p}^{\top} x$ is equal for all $x \in P$. As a result, $\hat{p}$ satisfies the constant value property.

The vector $d_{k}$ is constructed as $d_{k-1}+p_{k}$ with $p_{k}=\hat{p}$. Since $d_{k-1}$ and $\hat{p}$ satisfy the constant value property, it follows that $d_{k}$ satisfies the constant value property.

Remark 2.25. Since the GL type bound can be computed both as a linearization-based bound and by solving ( $C G L_{I L P}$ ) (see Theorem 2.23), the iterative approach derived in this section can also be defined in terms of $\left(C G L_{I L P}\right)$. In that case, we iteratively compute $v_{G L}$ and reformulate the cost matrix using the dual variables of $\left(C G L_{I L P}\right)$. The details of this equivalent approach can be found in [271].

Since the linearizable matrix $\hat{Q}$ of Proposition 2.22 can be written as $\hat{Q}=\left[U^{\top} V^{\top}\right] Y+$ $\operatorname{Diag}(z)$ for some $Y \in \mathbb{R}^{2 n \times m}$ and $z \in \mathbb{R}^{m}$, it follows from [217] that we have $v_{R G L} \leq v_{R L T 1}$.

### 2.6.3 Preliminary results

For the instances considered in the preliminary results of Section 2.4, we now test our Gilmore-Lawler type bounds. First, we compute the classical GL type bound, after symmetrizing the cost matrix $Q$. This bound is denoted by $G L$. Moreover, we consider the iterative GL type bounding approach where we symmetrize the cost matrix in each iteration. That is, we apply Algorithm 2.1 using (2.24) where instead of optimizing over $\eta$, we set $\eta=\frac{1}{2}$. We denote this bound by $R G L^{s y m}$. Finally, we report the bound $R G L$ which is introduced in Section 2.6.2. The maximum computation time is set to 3600 seconds. The results are given in Table 2.3. From Table 2.3 it follows that the iterative approaches significantly im-

| $p$ | $n$ | $m$ | OPT | $G L$ |  | $R G L^{\text {sym }}$ |  | $R G L$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | bound | time | bound | time | bound | time |
| 0.1 | 20 | 44 | 923 | 923 | 0.017 | 923 | 0.015 | 923 | 0.088 |
|  | 25 | 76 | 1039 | 681 | 0.039 | 864 | 4.652 | 1018 | 95.52 |
|  | 30 | 100 | 1082 | 781 | 0.053 | 899 | 2.412 | 1082 | 15.18 |
| 0.3 | 15 | 61 | 485 | 347 | 0.061 | 368 | 1.481 | 485 | 7.140 |
|  | 20 | 118 | 438 | 223 | 0.068 | 263 | 3.482 | 418 | 3600 |
|  | 25 | 172 | 382 | 176 | 0.136 | 190 | 5.105 | 276 | 3600 |
| 0.5 | 15 | 116 | 226 | 102 | 0.336 | 110 | 2.602 | 222 | 1835 |
|  | 20 | 177 | 255 | 93 | 0.118 | 103 | 5.365 | 169 | 3600 |
|  | 25 | 306 | n.a. | 66 | 0.296 | 75 | 13.80 | 105 | 3600 |
| 0.7 | 15 | 149 | 173 | 63 | 0.080 | 67 | 3.530 | 117 | 1236 |
|  | 20 | 263 | n.a. | 52 | 0.181 | 54 | 9.624 | 63 | 269.3 |
|  | 25 | 396 | n.a. | 56 | 0.365 | 62 | 18.77 | 79 | 1085 |

Table 2.3: Bounds and computation times in seconds of GL type bounds on Erdős-Rényi instances.
prove the classical GL type bound. Among these iterative approaches, $R G L$ provides much stronger bounds than $R G L^{s y m}$. We conclude that this new approach of calculating the best GL type bound in each step provides better bounds than setting $\eta=\frac{1}{2}$ in the reformulation. However, it turns out that this improvement in the quality comes at the cost of efficiency. Clearly, we can stop our algorithm after a pre-specified number of iterations and/or time.

A comparison between the GL type approaches considered in Table 2.3 and the approaches considered in the preliminary results of Section 2.4.2 will follow in Section 2.8.

### 2.7 Other bounds for the QCCP

In this section we present several known bounding approaches from the literature that can be applied to the QCCP. In the next section, we compare those bounds with the bounds introduced earlier in this chapter. We consider a column generation approach and a bound based on a mixed integer linear programming (MILP) formulation.

Galbiati et al. [152] construct a column generation approach for the MinRC3. This approach can be extended to the QCCP. To the best of our knowledge, this is the only implemented lower bounding approach for the MinRC3 in the literature.

Let $\mathcal{C}$ be the set of all directed cycles in $G$. Moreover, let $\overline{\mathcal{C}} \subseteq \mathcal{C}$ be a subset of cycles such that it contains at least one cycle cover. Let $w_{c}$ be the cost of a cycle $c \in \mathcal{C}$. Then the
restricted master problem ( $R M P$ ) is given by:
(RMP)

$$
\begin{array}{cll}
\min _{y} & \sum_{c \in \overline{\mathcal{C}}} w_{c} y_{c} & \\
\text { s.t. } & \sum_{c \in \overline{\mathcal{C}}: i \in c} y_{c}=1 & \forall i \in N \\
& y_{c} \geq 0 & \forall c \in \overline{\mathcal{C}} . \tag{2.26}
\end{array}
$$

Let $\pi \in \mathbb{R}^{n}$ be the vector of dual variables corresponding to constraint (2.25). Then the subproblem $(S P)$ searches for the cycle in $\mathcal{C}$ with the smallest (negative) reduced costs with respect to $\pi$, i.e.

$$
\min _{x, z}\left\{\begin{array}{ll}
x^{\top} Q x-z^{\top} \pi: & \sum_{e \in \delta^{+}(i)} x_{e}=\sum_{e \in \delta^{-}(i)} x_{e}=z_{i} \quad \forall i \in N  \tag{SP}\\
\sum_{e \in A} x_{e} \geq 2, x \in\{0,1\}^{m}, z \in\{0,1\}^{n}
\end{array}\right\}
$$

where $z_{i}=1$ if vertex $i$ is on the cycle and 0 otherwise. As stated in [152], the problem ( $S P$ ) is strongly $\mathcal{N} \mathcal{P}$-hard itself. The quadratic objective function can be linearized by standard linearization techniques. A lower bound on the optimal value of the QCCP can be obtained by iteratively solving the master problem and its corresponding subproblem. If a cycle with negative reduced cost is found, we add it to the set $\overline{\mathcal{C}}$. This procedure is repeated until no more cycle with negative reduced cost is found or after some predefined stopping criteria. The obtained bound is denoted by $v_{C G}$.

Based on a procedure by [2, 174], we present the QCCP as an MILP problem. Let us first fix an equivalent representation of $(G, Q)$. Let $z_{e}$ be computed as in $\left(P_{e}\right)$ for all $e \in A$, see Section 2.6.1. Moreover, we define for all $e \in A$

$$
q_{e}^{\max }:=\max \left\{Q_{e,:} x: x \in P, x_{e}=0\right\} .
$$

Note that $q_{e}^{\max }$ can be obtained by solving a linear programming problem. Then, the QCCP can be formulated as a MILP:

$$
\begin{array}{lll}
\min _{x, y} & \sum_{e \in A} y_{e} & \\
\text { s.t. } & y_{e} \geq z_{e} x_{e} & \forall e \in A \\
& y_{e} \geq Q_{e,:} x-q_{e}^{\max }\left(1-x_{e}\right) & \forall e \in A  \tag{2.28}\\
& x \in P, y \in \mathbb{R}^{m} . &
\end{array}
$$

If we relax the binary constraint on $x$, then we obtain a lower bound for the QCCP. We call this bound the MILP-based bound and we denote its value by $v_{M I L P}$. The next result shows that $v_{M I L P}$ is at least as large as the Gilmore-Lawler type bound.

Theorem 2.26. The MILP-based bound dominates the Gilmore-Lawler type bound, i.e., $v_{G L} \leq v_{M I L P}$.

Proof. Let $\beta, \delta \in \mathbb{R}_{+}^{m}$ denote the dual variables of (2.27) and (2.28), respectively. Moreover, let $\mu, \gamma \in \mathbb{R}^{n}$ denote the dual variables of the cycle cover constraints $\sum_{e \in \delta^{+}(i)} x_{e}=1$
and $\sum_{e \in \delta^{-}(i)} x_{e}=1$ for all $i \in N$, respectively. Then, the dual of the MILP-based bound equals
$(D M I L P) \quad v_{M I L P}:=\max _{\beta, \delta, \mu, \gamma} \quad \sum_{i \in N} \mu_{i}+\sum_{i \in N} \gamma_{i}-\sum_{e \in A} \delta_{e} q_{e}^{\max }$

$$
\begin{array}{lll}
\text { s.t. } & \beta_{e}+\delta_{e}=1 & \forall e \in A \\
& \mu_{e^{+}}+\gamma_{e^{-}} \leq z_{e} \beta_{e}+\delta^{\top} Q_{:, e}+\delta_{e} q_{e}^{\max } & \forall e \in A \\
& \beta_{e}, \delta_{e} \geq 0 & \forall e \in A,
\end{array}
$$

where $Q_{:, e}$ equals the $e$ th column of $Q$. Now set $\delta_{e}=0$ for all $e \in A$. Then, $\beta_{e}=1$ for all $e \in A$ due to the first set of constraints. Then, (DMILP) reduces to

$$
\begin{array}{lll}
\max _{\mu, \gamma} & \sum_{i \in N} \mu_{i}+\sum_{i \in N} \gamma_{i} & \\
\text { s.t. } & \mu_{e^{+}}+\gamma_{e^{-}} \leq z_{e} & \forall e \in A
\end{array}
$$

This problem equals the dual of the continuous relaxation of $(G L)$. Hence, it follows that $v_{G L} \leq v_{M I L P}$.

Note that the MILP-based bound and the Gilmore-Lawler type bound are comparable if the same equivalent reformulation of $(G, Q)$ is used in their computations.

### 2.8 Computational results

In this section we test our bounding approaches on a set of test instances and compare them with several approaches from the literature. We take into account the linearization-based bound $L B B 1$ from Section 2.4.1, the classical GL type bound $G L$ from Section 2.6.1, the reformulated GL type bound $R G L$ discussed in Section 2.6.2, the column generation approach $C G$ and the MILP-based bound MILP from Section 2.7, and the first level RLT bound RLT1, see [3, 4]. The GL bound and the MILP-based bound are computed after symmetrizing $Q$. Note that we do not take into account $L B B 2$ and $L B B 3$, since our preliminary experiments from Section 2.4 .2 show that $L B B 1$ is preferred when taking both quality and efficiency into account.

All lower bounds are implemented in Matlab on a PC with an Intel(R) Core(TM) i5-6500 CPU, 3.20 GHz and 8 GB memory using CPLEX 12.6 as solver.

We consider the following types of instances:

- Erdős-Rényi instances: These instances are created via the $G(n, p)$ Erdős-Rényi model [125]. The number of nodes is fixed to $n$ and each arc is included with probability $p$ independent of the other arcs. The quadratic cost between any pair of successive arcs is chosen discrete uniformly at random out of $\{0, \ldots, 100\}$.
- Manhattan instances: The Manhattan instances are introduced in [80] and resemble modern city street patterns like the streets in Manhattan. Given a finite set of positive integers $\left(n_{1}, n_{2}, \ldots, n_{k}\right)$, the graph consists of a $n_{1} \times n_{2} \times \ldots \times n_{k}$ directed grid. Each node in the interior is adjacent to its $2 k$ neighbours. The nodes on the boundary are also adjacent to the corresponding nodes on the opposite boundary. For each dimension $k$, the arcs belonging to the same layer of the grid point in the same direction. However,
the arcs of two consecutive layers point in the opposite direction. This results in a graph containing a large number of cycles. The quadratic cost between any pair of successive arcs is chosen discrete uniformly at random out of $\{0, \ldots, 10\}$.
- Angle-distance instances: The Angle-distance instances are originally constructed for the QTSP in [133]. The number of nodes $n$ and the graph density $p$ are given. The $(x, y)$-coordinates of each node is chosen discrete uniformly at random out of $\{0, \ldots, 500\}^{2}$. Exactly $\lceil p n(n-1)\rceil$ arcs are chosen uniformly at random from the total set of arcs. For each arc $e \in A$, let $d_{e}$ denote the Euclidean distance between the endpoints of $e$. Moreover, for each two successive $\operatorname{arcs} e$ and $f$, let $\alpha_{\text {ef }}$ denote the turning angle (in radians) induced by the arcs. Given some constant $\rho \in \mathbb{R}_{+}$, the quadratic cost of two successive arcs $e$ and $f$ is calculated as:

$$
q_{e f}=\left\lceil 0.1\left(\rho \cdot \alpha_{e f}+\frac{d_{e}+d_{f}}{2}\right)\right\rceil
$$

Similar as in [133], we take $\rho=40$.
For Erdős-Rényi and Angle-distance instances, preliminary experiments show that instances up to approximately 300 arcs can be solved to optimality within one hour. For the Manhattan instances the limit is around 2000 arcs, due to the small density of these types of graphs.

In total we consider two sets of experiments: experiments on small instances and experiments on large instances. Since the optimum, $R L T 1$ and $C G$ cannot be calculated for large instances, we only test these approaches on the smaller instances. Moreover, we include the bounds introduced in this chapter, namely $L B B 1$ and $R G L$. The value and computation times (in seconds) on small Erdős-Rényi instances can be found in Table 2.4. This table contains 6 instances for $n=20,25,30$ and $p=0.3,0.5$. The results on Manhattan and Angle-distance instances are reported in Tables 2.5 and 2.6 , respectively. For the Angledistance instances we take the same values for $n$ and $p$ as for the Erdős-Rényi instances, while for the Manhattan instances we consider several two- and three-dimensional instances. The maximum computation time is set to 3600 seconds. When after this time no bound is computed, we report 'n.a.' in the tables. Since the optimal value is always integer, we round up all bounds to the nearest integer value.

For the smaller instances, we see that $R L T 1$ performs best in quality. When it can be computed, it is often close to the optimal value and it dominates the other bounds. $L B B 1$ is often very close to $R L T 1$, but can be computed much more efficiently. Namely, for the Erdős-Rényi and the Angle-distance instances the computation time of $L B B 1$ for all small instances is below 0.4 seconds, whereas $R L T 1$ cannot be computed within one hour for some of these instances. The column generation approach provides strong bounds, but in most cases it is not able to compute a bound in a time span of one hour. The reformulated GL type bound performs well on the Manhattan and Angle-distance instances, see Tables 2.5 and 2.6. Although its total computation time is large, the advantage of this approach is that it provides a bound in a short time and then iteratively improves the value. This makes it possible to stop the procedure at any given time and still obtain a bound. The bounds computed by $R G L$ are in almost all cases dominated by $L B B 1$.

When taking both efficiency and quality into account, we conclude that the linearizationbased bound $L B B 1$ outperforms the other approaches. Based on Tables 2.4, 2.5 and 2.6, the value of $L B B 1$ is at least $75 \%$ of the optimal value for the Erdős-Rényi instances. For the Angle-distance and Manhattan instances, this percentage equals $98 \%$ and $96 \%$, respectively.

| $p$ | $n$ | $m$ | OPT |  | RLT1 |  | $C G$ |  | $L B B 1$ |  | $R G L$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | value | time | value | time | value | time | value | time | value | time |
| 0.3 | 20 | 119 | 319 | 10.28 | 301 | 4.825 | 289 | 102.3 | 260 | 0.020 | 285 | 3600 |
|  | 25 | 177 | 386 | 19.04 | 331 | 20.09 | 331 | 928.9 | 305 | 0.040 | 280 | 3600 |
|  | 30 | 280 | n.a. | 3600 | 284 | 70.62 | n.a. | 3600 | 274 | 0.134 | 185 | 3600 |
| 0.5 | 20 | 195 | 236 | 211.0 | 181 | 17.00 | n.a. | 3600 | 175 | 0.121 | 129 | 3600 |
|  | 25 | 327 | n.a. | 3600 | 141 | 82.52 | n.a. | 3600 | 136 | 0.233 | 89 | 3600 |
|  | 30 | 442 | n.a. | 3600 | 168 | 385.0 | n.a. | 3600 | 162 | 0.322 | 99 | 3600 |

Table 2.4: Bounds and computation times in seconds of RLT1, CG, LBB1 and RGL on small Erdős-Rényi instances.

| Instance | $n$ | $m$ | OPT |  | RLT 1 |  | $C G$ |  | $L B B 1$ |  | $R G L$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | value | time | value | time | value | time | value | time | value | time |
| $(5,5)$ | 25 | 50 | 103 | 0.483 | 103 | 1.534 | 103 | 1.484 | 103 | 0.006 | 103 | 5.756 |
| $(10,10)$ | 100 | 200 | 418 | 2.335 | 418 | 1.974 | 418 | 1645 | 418 | 0.022 | 371 | 16.06 |
| $(4,4,4)$ | 64 | 192 | 199 | 6.312 | 193 | 9.415 | 196 | 691.4 | 193 | 0.081 | 175 | 3600 |
| $(6,6,6)$ | 216 | 648 | 700 | 23.67 | 683 | 1152 | n.a. | 3600 | 683 | 0.568 | 551 | 3600 |
| $(8,8,8)$ | 512 | 1536 | 1566 | 394.1 | n.a. | 3600 | n.a. | 3600 | 1530 | 1.213 | n.a. | 3600 |
| $(10,10,10)$ | 1000 | 3000 | n.a. | 3600 | n.a. | 3600 | n.a. | 3600 | 3113 | 3.754 | n.a. | 3600 |

Table 2.5: Bounds and computation times in seconds of RLT1, CG, LBB1 and RGL on small Manhattan instances.

| $p$ | $n$ | $m$ | OPT |  | RLT1 |  | $C G$ |  | $L B B 1$ |  | $R G L$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | value | time | value | time | value | time | value | time | value | time |
| 0.3 | 20 | 114 | 474 | 2.490 | 474 | 1.719 | 474 | 416.5 | 474 | 0.002 | 474 | 64.35 |
|  | 25 | 180 | 553 | 323.9 | 553 | 4.119 | n.a. | 3600 | 552 | 0.004 | 553 | 1559 |
|  | 30 | 261 | 512 | 2951.0 | 512 | 19.52 | n.a. | 3600 | 512 | 0.079 | 494 | 3600 |
| 0.5 | 20 | 190 | 276 | 177.8 | 276 | 6.732 | n.a. | 3600 | 276 | 0.053 | 274 | 1319 |
|  | 25 | 300 | 342 | 2163.6 | 340 | 53.43 | n.a. | 3600 | 338 | 0.142 | 320 | 3600 |
|  | 30 | 435 | n.a. | 3600 | 381 | 490.5 | n.a. | 3600 | 377 | 0.332 | 355 | 3600 |

Table 2.6: Bounds and computation times in seconds of RLT1, CG, LBB1 and RGL on small Angle-distance instances.

For the larger instances, we only compute the bounds that can be computed efficiently. That is, we do not consider the iterative approaches, but only the bounds GL, MILP and $L B B 1$. We also investigate the effect of a reformulation by adding an optimal incident skew-symmetric matrix to the cost matrix, see Section 2.4.1. We apply this reformulation to $L B B 1$, which implies that we optimize over the set $\tau_{1}^{\text {skew }}(Q)$, see $(2.7)$, instead of $\tau_{1}(Q)$. The resulting bound is denoted by $L B B 1^{\text {skew }}$. For the Manhattan instances this bound is omitted, since preliminary experiments showed that this reformulation does not improve the bounds for most Manhattan instances. This could be due to the sparsity of Manhattan instances. The bounds and computation times (in seconds) for the Erdős-Rényi, Manhattan and Angle-distance instances are reported in Tables 2.7, 2.8 and 2.9, respectively. For the Erdős-Rényi and Angle-distance instances we take for $n$ values between 30 and 100 nodes
and consider $p=0.3$ and $p=0.5$. For the Manhattan instances we consider large twodimensional instances and one large three-dimensional instance. The maximum computation time for these bounds is set to 1800 seconds. Again, we round up all bound values.

For the larger instances, we see that $L B B 1$ in all cases dominates $G L$ and MILP in both quality and efficiency. The difference in quality is most present for the Erdős-Rényi instances, see Table 2.7. For the Manhattan instances, we see that GL and MILP can be calculated efficiently for instances up to 3000 arcs. However, $L B B 1$ remains efficient even for larger instances. In particular, bounds for Manhattan instances up to 15000 arcs can be computed within 60 seconds.

| $p$ | $n$ | $m$ | $G L$ |  | MILP |  | $L B B 1$ |  | $L B B 1^{\text {skew }}$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | value | time | value | time | value | time | value | time |
| $0.3$ | 30 | 284 | 111 | 0.272 | 122 | 0.435 | 230 | 0.083 | 232 | 0.766 |
|  | 40 | 468 | 117 | 0.645 | 131 | 1.006 | 265 | 0.179 | 278 | 1.711 |
|  | 50 | 754 | 121 | 1.598 | 130 | 2.410 | 267 | 0.404 | 274 | 4.184 |
|  | 60 | 1062 | 103 | 4.068 | 118 | 5.788 | 272 | 0.726 | 272 | 8.048 |
|  | 70 | 1481 | 114 | 8.910 | 123 | 12.94 | 255 | 1.660 | 258 | 15.38 |
|  | 80 | 1842 | 113 | 14.26 | 122 | 20.82 | 263 | 2.740 | 267 | 24.67 |
|  | 90 | 2385 | 114 | 23.25 | 122 | 37.61 | 259 | 5.296 | 261 | 41.74 |
|  | 100 | 2962 | 119 | 36.03 | 126 | 63.49 | 269 | 13.32 | 270 | 69.90 |
| $0.5$ | 30 | 434 | 73 | 0.557 | 79 | 0.783 | 161 | 0.182 | 163 | 9.218 |
|  | 40 | 793 | 69 | 1.607 | 74 | 2.364 | 166 | 0.554 | 169 | 10.38 |
|  | 50 | 1197 | 72 | 4.323 | 77 | 6.682 | 165 | 1.185 | 167 | 15.74 |

Table 2.7: Bounds and computation times in seconds of $G L, M I L P, L B B 1$ and $L B B 1^{\text {skew }}$ on large Erdős-Rényi instances.

| Instance | $n$ | $m$ | $G L$ |  | MILP |  | $L B B 1$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | value | time | value | time | value | time |
| $(20,20)$ | 400 | 800 | 1237 | 5.31 | 1472 | 7.491 | 1537 | 0.100 |
| $(30,30)$ | 900 | 1800 | 2813 | 56.24 | 3343 | 86.46 | 3517 | 0.410 |
| $(40,40)$ | 1600 | 3200 | 5101 | 346.8 | 6028 | 553.5 | 6302 | 1.388 |
| $(50,50)$ | 2500 | 5000 | 7983 | 1225.3 | 9424 | 1897.8 | 9828 | 2.838 |
| $(17,17,17)$ | 4913 | 14739 | n.a. | 1800 | n.a. | n.a. | 15398 | 54.79 |

Table 2.8: Bounds and computation times in seconds of $G L, M I L P$ and $L B B 1$ on large Manhattan instances.

Moreover, we conclude from Tables 2.7 and 2.9 that the addition of an incidence skewsymmetric matrix to the set $\tau_{1}(Q)$ only improves the bounds for some of the instances. In general, it turns out that the Erdős-Rényi instances can successfully be improved by this method, whereas for the Angle-distance instances only in a few cases there is an improvement. Although the computation times of $L B B 1^{\text {skew }}$ are larger than those of $L B B 1$, bounds can still be computed in a reasonable time span.

| $p$ | $n$ | $m$ | $G L$ |  | MILP |  | $L B B 1$ |  | $L B B 1^{\text {skew }}$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | value | time | value | time | value | time | value | time |
| 0.3 | 30 | 261 | 456 | 0.238 | 467 | 0.410 | 525 | 0.054 | 525 | 6.957 |
|  | 40 | 468 | 507 | 0.693 | 516 | 0.984 | 567 | 0.463 | 567 | 7.795 |
|  | 50 | 735 | 622 | 1.567 | 631 | 2.223 | 709 | 0.317 | 709 | 9.982 |
|  | 60 | 1062 | 609 | 3.684 | 618 | 5.401 | 684 | 0.694 | 684 | 13.97 |
|  | 70 | 1449 | 656 | 7.436 | 666 | 12.37 | 746 | 1.331 | 747 | 21.63 |
|  | 80 | 1896 | 749 | 13.03 | 756 | 23.77 | 867 | 2.613 | 867 | 31.96 |
|  | 90 | 2403 | 815 | 20.33 | 826 | 39.99 | 933 | 4.838 | 933 | 48.81 |
|  | 100 | 2970 | 810 | 30.35 | 823 | 66.04 | 951 | 12.50 | 952 | 78.62 |
| 0.5 | 30 | 435 | 339 | 0.516 | 343 | 0.876 | 373 | 0.168 | 373 | 16.59 |
|  | 40 | 780 | 411 | 1.456 | 418 | 2.386 | 464 | 0.474 | 464 | 16.21 |
|  | 50 | 1225 | 466 | 4.159 | 473 | 7.550 | 534 | 1.177 | 535 | 22.34 |

Table 2.9: Bounds and computation times in seconds of $G L, M I L P, L B B 1$ and $L B B 1^{\text {skew }}$ on large Angle-distance instances.

### 2.9 Conclusions

In this chapter we consider the linearization problem of the QCCP and its applications. We provide several sufficient conditions for linearizability, and show how these conditions can be used to obtain strong lower bounds for the QCCP. The linearization-based bound $L B B 1$, resulting from the incident weak sum property, is the most efficient LBB in terms of complexity and quality, see Table 2.2. We show here that the GL type bound for the QCCP also belongs to the family of linearization-based bounds, see Theorem 2.23 , by providing the appropriate sufficient condition, see Proposition 2.22.

The first level RLT bounds and/or the GL type bounds are the only linearization-based bounds for quadratic binary optimization problems that are implemented for various binary quadratic optimization problems up to date. This chapter shows that besides these two well-known bounds, the linearization-based bounds introduced here are worth considering.

Here, we also present how each sufficient condition can be used in an iterative bounding procedure. In particular, we introduce a new reformulation technique in which we search for the best $\eta$-equivalent representation of the residual cost matrix and its optimal linearizable matrix, see Algorithm 2.1. We show how the resulting iterative procedure computes the best GL type bound in each iteration. Our approach outperforms known iterative bounding procedures that use the GL type bounds, see Table 2.3. Moreover, we prove that the resulting linearization vectors in each step satisfy the constant value property, see Theorem 2.24.

Finally, our numerical results show that our approach outperforms several other bounds from the literature if we take into account both quality and efficiency. Although the linearization-based bounds $L B B 1$ are dominated by the well-known first level RLT bounds, they can be computed extremely fast. For the Manhattan instances, $L B B 1$ bounds for instances up to 15000 arcs can be computed within 60 seconds. However, other approaches fail to provide bounds for instances of this large size.

We expect that similar bounding procedures can be successfully applied to other binary quadratic optimization problems, such as the quadratic assignment problem, the quadratic minimum spanning tree and the quadratic traveling salesman problem.

# SDP-based bounds for the quadratic cycle cover problem via cutting-plane augmented Lagrangian methods and reinforcement learning 

## Chapter summary


#### Abstract

We study the quadratic cycle cover problem (QCCP), which aims to find a nodedisjoint cycle cover in a directed graph with minimum interaction costs between successive arcs. We derive several semidefinite programming (SDP) relaxations and use facial reduction to make these strictly feasible. We investigate a nontrivial relationship between the transformation matrix used in the reduction and the structure of the graph, which is exploited in an efficient algorithm that constructs this matrix for any instance of the problem. To solve our relaxations, we propose an algorithm that incorporates an augmented Lagrangian method into a cutting-plane framework by utilizing Dykstra's projection algorithm. Our algorithm is suitable for solving SDP relaxations with a large number of cutting planes. Computational results show that our SDP bounds and our efficient cutting-plane algorithm outperform other QCCP bounding approaches from the literature. Finally, we provide several SDP-based upper bounding techniques, among which a sequential Q-learning method that exploits a solution of our SDP relaxation within a reinforcement learning environment.


### 3.1 Introduction

A disjoint cycle cover in a graph is a set of node-disjoint cycles such that every node is covered by exactly one cycle. The cycle cover problem (CCP) is the problem of finding a disjoint cycle cover such that the total arc weight is minimized. In this chapter we focus on its quadratic version, which is known as the quadratic cycle cover problem (QCCP). The QCCP is the problem of finding a disjoint cycle cover in a graph such that the total sum of interaction costs between consecutive arcs is minimized. Although the problem can be defined for both directed and undirected graphs, we focus here on the asymmetric version which is defined on directed graphs.

The QCCP is introduced by Jäger and Molitor [224]. Fischer et al. [138] show that the problem is $\mathcal{N} \mathcal{P}$-hard. This result is later on strengthened in [274], where it is shown that the QCCP is strongly $\mathcal{N} \mathcal{P}$-hard and not approximable within any constant factor, see also Section 2.2 in this thesis.

The QCCP and its variations, see Section 1.4.2, have applications in various fields, such as robotics [6], cargo and energy distribution networks [370]. Moreover, it has theoretical relevance due to its close connection with the quadratic traveling salesman problem, see Section 1.4.1. For a more comprehensive overview on the background of the QCCP and its proposed solution approaches, we refer the reader to Section 1.4.2 of this dissertation.

## Main results and outline

The aim of this chapter is to construct efficient lower and upper bounding approaches for the QCCP based on semidefinite programming. To achieve this goal we introduce several methods that can be extended to a range of other optimization problems. We combine a wide variety of different techniques including facial reduction, projection methods, randomized algorithms and reinforcement learning.

First, we derive three SDP relaxations for the QCCP with increasing complexity. Our strongest SDP relaxation contains nonnegativity constraints and an additional subset of the facet-defining inequalities of the boolean quadric polytope (BQP), which make it a powerful yet very difficult to solve relaxation. As a first step in the development of our algorithmic approaches for computing QCCP lower bounds, we study the geometry of the feasible sets of our relaxations. We prove that the relaxations are not Slater feasible, and show how to perform facial reduction to project the feasible sets onto lower dimensional spaces. The transformation matrix needed for this projection is graph-specific. Therefore we propose a polynomial time algorithm based on the bipartite representation of the underlying graph that provides a sparse transformation matrix.

To solve our SDP relaxation with nonnegativity constraints, we study the following two variants of the alternating direction augmented Lagrangian method; the (original) alternating direction method of multipliers (ADMM) and the Peaceman-Rachford splitting method (PRSM) that is also known as the symmetric ADMM. Although the ADMM is tested on SDP relaxations of various optimization problems, the PRSM with larger stepsize was not implemented up to date for SDP relaxations. Our results show that the PRSM outperforms the classical ADMM for the relaxation with nonnegativity constraints. Therefore we take the PRSM as the backbone of our new approach.

It is well-known that current SDP solvers have difficulties solving relaxations including the facet-defining inequalities of the BQP. To solve our strongest relaxation including these cuts, we present an advanced cutting-plane method that extends on the PRSM: a cutting-plane
augmented Lagrangian method (CP-ALM). The CP-ALM exploits the well-known Dykstra projection algorithm to deal with the BQP cuts. We (partially) parallelize Dykstra's cyclic algorithm by clustering the set of BQP inequalities into subsets of nonoverlapping cuts. We present several other ingredients that improve the efficiency of the algorithm. The CP-ALM also exploits warm starts each time new violated cuts are added. Although it might seem that our algorithm is problem-specific, all ingredients described in this chapter can be easily extended for solving other optimization problems.

Finally, we derive several upper bounding approaches that exploit the output matrices from the CP-ALM. Let us list the most prominent ones. In our randomized undersampling algorithm we sample a partial solution and deterministically extend it to a full cycle cover. In randomized oversampling we iteratively draw a pair of successive arcs according to a distribution related to the SDP solution, until we obtain a cycle cover. Our most sophisticated rounding approach is based on a distributed reinforcement learning technique, i.e., Q-learning. In particular, we let artificial agents learn how to find cycle covers by exploiting the SDP solution matrix such that the expected total reward is maximized. The latter approach provides the best upper bounds among all presented ones. We expect that these rounding approaches can be successfully extended to relaxations of other optimization problems. Let us emphasize that it is challenging to find good feasible solutions for the QCCP, especially for large instances, since the considered graphs are not necessarily complete.

We provide extensive numerical tests on data sets used for the QCCP as well as data sets for the QTSP. Our bounds significantly outperform other bounds from the literature.

This chapter is structured as follows. In Section 3.2, we formally introduce the QCCP and study its associated directed 2 -factor polytope. In Section 3.3, we construct several SDP relaxations for the QCCP of increasing complexity. The Slater feasibility of the SDP relaxations is the topic of Section 3.4.1. Since the transformation matrices used for the facial reduction are graph-specific, we provide a polynomial time algorithm for computing their sparse expressions in Section 3.4.2. In Section 3.5, we propose a new algorithm for solving the SDP relaxations that is based on a combination of the PRSM, Dykstra's projection algorithm and a cutting-plane method. Several upper bounding approaches are discussed in Section 3.6. Section 3.7 provides an extensive numerical study of all introduced methods.

### 3.2 The quadratic cycle cover problem

In this section we formally introduce the asymmetric version of the quadratic cycle cover problem. Moreover, we introduce the directed 2 -factor polytope and consider some of its properties.

The quadratic cycle cover problem (QCCP) is the problem of finding a set of node-disjoint cycles covering all the nodes such that the sum of interaction costs between successive arcs is minimized. Since we assume that all cycle covers in this chapter are disjoint, we use the term cycle cover to denote this concept in the sequel. An instance of the QCCP is specified by the pair $(G, Q)$, where $G=(N, A)$ is a simple directed graph with $n:=|N|$ nodes and $m:=|A|$ arcs and $Q=\left(q_{e f}\right) \in \mathbb{R}_{+}^{m \times m}$ is a nonnegative cost matrix. We assume that the entries of $Q$ are such that $q_{e f}=0$ if arc $f$ is not a successor of arc $e$, i.e., if $f \notin \delta^{+}\left(e^{-}\right)$. Moreover, recall from Chapter 2 that the nonnegativity condition on $Q$ is nonrestrictive, see Section 2.2.

Let $x \in\{0,1\}^{m}$ represent the characteristic vector of a cycle cover. That is, $x_{e}=1$ if $\operatorname{arc} e$ belongs to the cycle cover and $x_{e}=0$ otherwise. Then, the QCCP can be formulated
as:

$$
\begin{equation*}
O P T(Q):=\min \left\{x^{\top} Q x: x \in P\right\}, \tag{3.1}
\end{equation*}
$$

where $P$ denotes the set of all cycle covers in $G$, i.e.,

$$
\begin{equation*}
P:=\left\{x \in\{0,1\}^{m}: \sum_{e \in \delta^{+}(i)} x_{e}=\sum_{e \in \delta^{-}(i)} x_{e}=1 \quad \forall i \in N\right\} . \tag{3.2}
\end{equation*}
$$

The linear problem corresponding to the QCCP is called the cycle cover problem (CCP). Given a linear arc-weight function, the CCP asks for a minimum weight cycle cover in $G$. The CCP reduces to the well-known linear assignment problem, see e.g., [62], and is therefore polynomial time solvable.

Let $\operatorname{conv}(P)$ be the convex hull of all characteristic vectors corresponding to directed 2factors in $G$. We call this set the directed 2-factor polytope. Let $U \in \mathbb{R}^{n \times m}$ and $V \in \mathbb{R}^{n \times m}$ be defined as

$$
U_{i, e}:=\left\{\begin{array}{ll}
1 & \text { if arc } e \text { starts at node } i, \\
0 & \text { otherwise },
\end{array} \quad V_{i, e}:= \begin{cases}1 & \text { if arc } e \text { ends at node } i, \\
0 & \text { otherwise } .\end{cases}\right.
$$

Additionally, let $u_{i}^{\top}$ and $v_{i}^{\top}$ denote the $i$ th row of $U$ and $V$, respectively. Using matrix notation, we have $P=\left\{x \in\{0,1\}^{m}:\left[U^{\top} V^{\top}\right]^{\top} x=\mathbf{1}_{2 n}\right\}$. It follows from the total unimodularity of $\left[U^{\top} V^{\top}\right]^{\top}$ that the directed 2-factor polytope can be written explicitly as:

$$
\operatorname{conv}(P)=\left\{x \in \mathbb{R}^{m}: x \geq \mathbf{0}_{m},\left[\begin{array}{l}
U  \tag{3.3}\\
V
\end{array}\right] x=\mathbf{1}_{2 n}\right\}
$$

Observe that the arcs that are never used in a cycle cover are irrelevant for the QCCP. We define the set $\mathcal{J}$ consisting of all arcs with this property, i.e.,

$$
\mathcal{J}:=\left\{f \in A: x_{f}=0 \text { for all } x \in P\right\} .
$$

The elements in $\mathcal{J}$ can be obtained in polynomial time by solving for each $f \in A$ the following CCP:

$$
z_{f}:=\max \left\{\mathbf{e}_{f}^{\top} x: x \in P\right\}
$$

The set $\mathcal{J}$ consists of all arcs $f \in A$ for which $z_{f}=0$. Without loss of generality, we can remove the arcs that are in $\mathcal{J}$ from the given instance to simplify the problem. This leads to the following assumption that applies to the rest of this chapter.

Assumption 3.1. There exists at least one cycle cover in $G$, i.e., $P \neq \emptyset$. Moreover, the set $\mathcal{J}$ is empty.
We end this section by considering the dimension of the directed 2 -factor polytope. We define

$$
\alpha:=\operatorname{rank}\left(\left[\begin{array}{l}
U  \tag{3.4}\\
V
\end{array}\right]\right) .
$$

In Section 3.4.2, we derive the value of $\alpha$ in terms of the graph. For now, we observe that $n \leq \alpha \leq 2 n-1$, provided that Assumption 3.1 holds.

It follows from the rank-nullity theorem that $\operatorname{dim}\left(\operatorname{Nul}\left(\left[U^{\top} V^{\top}\right]^{\top}\right)\right)=m-\alpha$. Let us prove the following lemma.

Lemma 3.2. Under Assumption 3.1, the dimension of the directed 2-factor polytope is $m-\alpha$.

Proof. It follows from (3.3) that

$$
\operatorname{conv}(P)=\left\{x \in \mathbb{R}^{m}:\left[\begin{array}{ll}
U^{\top} & V^{\top}
\end{array}\right]^{\top} x=\mathbf{1}_{2 n}\right\} \cap \mathbb{R}_{+}^{m}
$$

Obviously, $\mathbb{R}_{+}^{m}$ is full-dimensional, whereas the dimension of $\left\{x \in \mathbb{R}^{m}:\left[U^{\top} V^{\top}\right]^{\top} x=\mathbf{1}_{2 n}\right\}$ equals $\operatorname{dim}\left(\operatorname{Nul}\left(\left[U^{\top} V^{\top}\right]^{\top}\right)\right)=m-\alpha$. Hence, we have $\operatorname{dim}(\operatorname{conv}(P)) \leq m-\alpha$, where strict inequality holds only if $\operatorname{conv}(P)$ is fully contained in one of the facets of $\mathbb{R}_{+}^{m}$. Now, assume that there exists some arc $e$ such that $\operatorname{conv}(P) \subseteq\left\{x \in \mathbb{R}_{+}^{m}: x_{e}=0\right\}$. Consequently, we must have $P \subseteq\left\{x \in \mathbb{R}_{+}^{m}: x_{e}=0\right\}$, which implies that $e \in \mathcal{J}$. This contradicts Assumption 3.1. We conclude that $\operatorname{dim}(\operatorname{conv}(P))=m-\alpha$.

### 3.3 SDP relaxations for the QCCP

In this section we focus on constructing several semidefinite programming relaxations for the QCCP. These relaxations are increasing in strength and complexity.

The objective function of (3.1) can be rewritten as $x^{\top} Q x=\left\langle Q, x x^{\top}\right\rangle=\langle Q, X\rangle$, where we replace $x x^{\top}$ by a matrix variable $X \in \mathcal{S}^{m}$. We now relax the equality $X-x x^{\top}=\mathbf{0}$ by replacing it by the SDP constraint $X-x x^{\top} \succeq \mathbf{0}$. It follows from the Schur complement lemma that we can equivalently write $\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right) \succeq \mathbf{0}$. Moreover, since $x \in P$ is a binary vector, we have $\operatorname{diag}(X)=x$. This leads to the following basic feasible set for an SDP relaxation of the QCCP:

$$
\mathcal{F}_{\text {basic }}:=\left\{\left(\begin{array}{cc}
u_{i}^{\top} x=v_{i}^{\top} x=1 \quad \forall i \in N  \tag{3.5}\\
x & X
\end{array}\right) \in \mathcal{S}^{m+1}: \begin{array}{c}
x=\operatorname{diag}(X),\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}
\end{array}\right\} .
$$

We show below how to strengthen the feasible set (3.5) by adding valid constraints.
Since each cycle cover consists of $n$ arcs, we know that $\mathbf{1}_{m}^{\top} x=n$ for all $x \in P$. This can be written equivalently as $\operatorname{tr}(X)=n$, which we refer to as the trace constraint. Moreover, since $x x^{\top}$ is replaced by $X$, the constraint $\langle\mathbf{J}, X\rangle=n^{2}$, which we call the all-ones constraint, is also valid.

One can also add to $\mathcal{F}_{\text {basic }}$ the so-called squared linear constraints. These constraints result from taking the product of the linear constraints $u_{i}^{\top} x=1$ and $u_{j}^{\top} x=1$ for all $i, j \in N$, which yield $1=\left(u_{i}^{\top} x\right)\left(x^{\top} u_{j}\right)=\left\langle u_{i} u_{j}^{\top}, x x^{\top}\right\rangle$. Hence, the constraint $\left\langle u_{i} u_{j}^{\top}, X\right\rangle=1$ is valid for $\mathcal{F}_{\text {basic }}$. The same can be done by taking the products of the linear constraints $v_{i}^{\top} x=1$ for all $i \in N$, etc. In total, we distinguish three types of squared linear constraints that are summarized in Table 3.1.

| Type of squared linear constraint | Constraints on $X$ |
| :---: | :---: |
| Type I | $\left\langle u_{i} u_{i}^{\top}, X\right\rangle=1$ and $\left\langle v_{i} v_{i}^{\top}, X\right\rangle=1$ for all $i \in N ;$ |
| Type II | $\left\langle u_{i} u_{j}^{\top}, X\right\rangle=1$ and $\left\langle v_{i} v_{j}^{\top}, X\right\rangle=1$ for all $i, j \in N, i \neq j ;$ |
| Type III | $\left\langle u_{i} v_{j}^{\top}, X\right\rangle=1$ for all $i, j \in N$. |

Table 3.1: Three types of valid squared linear constraints for $\mathcal{F}_{\text {basic }}$.

We show below how the above-mentioned valid constraints relate. An interesting result is that the squared linear constraints of Type II and III turn out to be redundant when the Type I constraints and some other constraints are added to (3.5).
Proposition 3.3. Let $x \in \mathbb{R}^{m}$ and $X \in \mathcal{S}^{m}$ be such that $\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right) \succeq \mathbf{0}$, $\operatorname{diag}(X)=x$, $\operatorname{tr}(X)=n$ and $\langle\mathbf{J}, X\rangle=n^{2}$. If $\left\langle u_{i} u_{i}^{\top}, X\right\rangle=\left\langle v_{i} v_{i}^{\top}, X\right\rangle=1$ for all $i \in N$, then
(i) the squared linear constraints of Type II and III are redundant;
(ii) the linear constraints $u_{i}^{\top} x=u_{j}^{\top} x=1$ for all $i, j \in N$ are redundant.

Proof. (i) Let $i, j \in N$ with $i \neq j$, then we have,

$$
\begin{aligned}
\left\langle\left(u_{i}-u_{j}\right)\left(u_{i}-u_{j}\right)^{\top}, X\right\rangle & =\left\langle u_{i} u_{i}^{\top}+u_{j} u_{j}^{\top}-2 u_{i} u_{j}^{\top}, X\right\rangle \\
& =\left\langle u_{i} u_{i}^{\top}, X\right\rangle+\left\langle u_{j} u_{j}^{\top}, X\right\rangle-2\left\langle u_{i} u_{j}^{\top}, X\right\rangle \\
& =2-2\left\langle u_{i} u_{j}^{\top}, X\right\rangle \geq 0
\end{aligned}
$$

since $\left\langle u_{i} u_{i}^{\top}, X\right\rangle=\left\langle u_{j} u_{j}^{\top}, X\right\rangle=1$ and $X \succeq \mathbf{0}$. From this it follows that $\left\langle u_{i} u_{j}^{\top}, X\right\rangle \leq 1$.
Conversely, as all arcs have exactly one starting node, we have $\mathbf{1}_{m}=\sum_{i \in N} u_{i}$. Using this, we can rewrite the matrix $\mathbf{J}$ as $\mathbf{J}=\left(\sum_{i \in N} u_{i}\right)\left(\sum_{i \in N} u_{i}\right)^{\top}$ and the constraint $\langle\mathbf{J}, X\rangle=n^{2}$ as follows:

$$
n^{2}=\langle\mathbf{J}, X\rangle=\left\langle\left(\sum_{i \in N} u_{i}\right)\left(\sum_{i \in N} u_{i}\right)^{\top}, X\right\rangle=\sum_{i \in N} \sum_{j \in N}\left\langle u_{i} u_{j}^{\top}, X\right\rangle
$$

The right-hand side expression is a sum of $n^{2}$ elements for which $\left\langle u_{i} u_{j}^{\top}, X\right\rangle \leq 1$ for all $i, j \in N$. Since the sum has to be equal to $n^{2}$, it follows that $\left\langle u_{i} u_{j}^{\top}, X\right\rangle=1$ for all $i, j \in N, i \neq j$. The other equalities can be proven in a similar fashion.
(ii) Let $Y:=X-x x^{\top}$. By the Schur complement lemma, we know that $Y \succeq \mathbf{0}$. Now,

$$
1=\left\langle u_{i} u_{i}^{\top}, X\right\rangle=\left\langle u_{i} u_{i}^{\top}, Y\right\rangle+\left\langle u_{i} u_{i}^{\top}, x x^{\top}\right\rangle=\left\langle u_{i} u_{i}^{\top}, Y\right\rangle+\left(u_{i}^{\top} x\right)^{2}
$$

Since $\left\langle u_{i} u_{i}^{\top}, Y\right\rangle \geq 0$, it follows that $\left(u_{i}^{\top} x\right)^{2} \leq 1$ and, consequently, $u_{i}^{\top} x \leq 1$ for all $i \in N$.
Now we rewrite the trace constraint $\mathbf{1}_{m}^{\top} x=n$ as

$$
\mathbf{1}_{m}^{\top} x=\left(\sum_{i \in N} u_{i}\right)^{\top} x=\sum_{i \in N} u_{i}^{\top} x=n
$$

Since each term $u_{i}^{\top} x$ is bounded by 1 , equality is established only when $u_{i}^{\top} x=1$ for all $i \in N$. In a similar way one can show that $v_{i}^{\top} x=1$ for all $i \in N$.

Observe that there exist $2 n$ constraints of Type I. We show how to merge these constraints to obtain a more compact formulation. To this end, we define the matrices $\tilde{U}, \tilde{V} \in \mathbb{R}^{m \times m}$ as follows:

$$
\tilde{U}:=\sum_{i \in N}\binom{-1}{u_{i}}\binom{-1}{u_{i}}^{\top} \quad \text { and } \quad \tilde{V}:=\sum_{i \in N}\binom{-1}{v_{i}}\binom{-1}{v_{i}}^{\top} .
$$

We establish the following result.
Proposition 3.4. Let $x \in \mathbb{R}^{m}$ and $X \in \mathcal{S}^{m}$ be such that $\left(\begin{array}{ll}1 & x^{\top} \\ x & X\end{array}\right) \succeq \mathbf{0}$ and $\operatorname{diag}(X)=x$. Then, the following statements are equivalent:
(i) $\operatorname{tr}(X)=n$ and $\left\langle u_{i} u_{i}^{\top}, X\right\rangle=\left\langle v_{i} v_{i}^{\top}, X\right\rangle=1$ for all $i \in N$;
(ii) $\left\langle\tilde{U},\left(\begin{array}{ll}1 & x^{\top} \\ x & X\end{array}\right)\right\rangle=\left\langle\tilde{V},\left(\begin{array}{ll}1 & x^{\top} \\ x & X\end{array}\right)\right\rangle=0$.

Proof. It is not difficult to see that $(i) \Longrightarrow(i i)$. We now show the converse statement. We have

$$
\left\langle\tilde{U},\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right)\right\rangle=\left\langle\sum_{i \in N}\binom{-1}{u_{i}}\binom{-1}{u_{i}}^{\top},\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right)\right\rangle=\sum_{i \in N}\left(u_{i}^{\top} X u_{i}-2 u_{i}^{\top} x+1\right)=0 .
$$

Since $\binom{-1}{u_{i}}\binom{-1}{u_{i}}^{\top} \succeq \mathbf{0}$, it follows that $u_{i}^{\top} X u_{i}-2 u_{i}^{\top} x+1 \geq 0$ for all $i \in N$. Combining this with the equality above, we conclude that $u_{i}^{\top} X u_{i}-2 u_{i}^{\top} x+1=0$ for all $i \in N$.

Now define $Y:=X-x x^{\top} \succeq \mathbf{0}$. Then $u_{i}^{\top} X u_{i}-2 u_{i}^{\top} x+1=0$ can be rewritten as

$$
u_{i}^{\top}\left(Y+x x^{\top}\right) u_{i}-2 u_{i}^{\top} x+1=0, \quad \text { or equivalently, } \quad u_{i}^{\top} Y u_{i}+\left(u_{i}^{\top} x-1\right)^{2}=0
$$

Since $u_{i}^{\top} Y u_{i} \geq 0$ and $\left(u_{i}^{\top} x-1\right)^{2} \geq 0$, it follows that $u_{i}^{\top} x=1$, which in turn implies that $u_{i}^{\top} X u_{i}=2 u_{i}^{\top} x-1=1$. Similarly, one can prove that $\left\langle v_{i} v_{i}^{\top}, X\right\rangle=1$ for all $i \in N$.

Finally, since $\mathbf{1}_{m}=\sum_{i \in N} u_{i}$, we have

$$
\operatorname{tr}(X)=\mathbf{1}_{m}^{\top} x=\sum_{i \in N} u_{i}^{\top} x=n
$$

We conclude that $(i i) \Longrightarrow(i)$.

Proposition 3.4 shows that instead of the trace constraint and the squared linear constraints, we can equivalently include the merged squared linear constraints. Let us now define the
following set:

$$
\begin{align*}
& \mathcal{F}_{1}:=\left\{\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \in \mathcal{S}^{m+1}:\right.\left\langle\tilde{U},\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right)\right\rangle=\left\langle\tilde{V},\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right)\right\rangle=0  \tag{3.6}\\
&\left.\langle\mathbf{J}, X\rangle=n^{2}, \operatorname{diag}(X)=x,\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}\right\} .
\end{align*}
$$

From the above discussion it follows that $\mathcal{F}_{1} \subseteq \mathcal{F}_{\text {basic }}$. Let us now introduce our first SDP relaxation:
$\left(S D P_{1}\right)$

$$
\min \left\{\langle Q, X\rangle:\left(\begin{array}{cc}
1 & x^{\top}  \tag{3.7}\\
x & X
\end{array}\right) \in \mathcal{F}_{1}\right\} .
$$

In the sequel we show how to improve the SDP relaxation (3.7). Let us exploit the structure of a cycle cover to identify a zero pattern in $X$. For each $i \in N$, we know that there is exactly one arc $e$ in $\delta^{+}(i)$ with $x_{e}=1$ and $x_{f}=0$ for all other arcs $f$ leaving $i$. Hence, for each pair of distinct arcs $e, f \in \delta^{+}(i)$ we have $x_{e} x_{f}=0$. This leads to the valid constraint $X_{e f}=0$ for all $e, f \in \delta^{+}(i), e \neq f$. The same holds for the incoming arcs. We call these type of equalities the zero-structure constraints. We define:

$$
\begin{equation*}
\mathcal{Z}:=\{(e, f) \in A \times A: e \text { and } f \text { start or end at the same node, } e \neq f\} . \tag{3.8}
\end{equation*}
$$

Then the zero-structure constraints read that $X_{e f}=0$ for all $(e, f) \in \mathcal{Z}$.
Note that one may also add the nonnegativity constraints on matrix variables in ( $S D P_{1}$ ). For that purpose, we define the cone of nonnegative symmetric matrices, i.e.,

$$
\mathcal{N}_{+}^{m}:=\left\{X \in \mathcal{S}^{m}: X \geq \mathbf{0}\right\} .
$$

We show next that after adding nonnegativity constraints to the feasible set of ( $S D P_{1}$ ), the zero-structure constraints turn out to be redundant.

Proposition 3.5. Let $x \in \mathbb{R}^{m}$ and $X \in \mathcal{S}^{m}$ be feasible for $\left(S D P_{1}\right)$. If $X \in \mathcal{N}_{+}^{m}$, then $X_{e f}=0$ for all $(e, f) \in \mathcal{Z}$.

Proof. We prove the statement for the outgoing arcs. The proof for the incoming arcs is similar. Using Proposition 3.4, we know that $\left\langle u_{i} u_{i}^{\top}, X\right\rangle=1$ for all $i \in N$. We rewrite this equality as:

$$
1=\left\langle u_{i} u_{i}^{\top}, X\right\rangle=\sum_{e \in A} \sum_{f \in A}\left(u_{i}\right)_{e}\left(u_{i}\right)_{f} X_{e f}=\sum_{e \in \delta^{+}(i)} X_{e e}+\sum_{\substack{e, f \in \delta^{+}(i), e \neq f}} X_{e f} .
$$

Since $\operatorname{diag}(X)=x$, we have $\sum_{e \in \delta^{+}(i)} X_{e e}=\sum_{e \in \delta^{+}(i)} x_{e}=u_{i}^{\top} x=1$, where the last equality follows from Proposition 3.3. Thus, we have $\sum_{e, f \in \delta+(i), e \neq f} X_{e f}=0$, from where it follows that $X_{e f}=0$ for all $e, f \in \delta^{+}(i)$ with $e \neq f$.

Let us now define our next, tighter SDP relaxation:
$\left(S D P_{2}\right)$

$$
\min \left\{\langle Q, X\rangle:\left(\begin{array}{cc}
1 & x^{\top}  \tag{3.9}\\
x & X
\end{array}\right) \in \mathcal{F}_{1} \cap \mathcal{N}_{+}^{m+1}\right\} .
$$

To further strengthen $\left(S D P_{2}\right)$, we consider an additional set of valid inequalities. Namely, we consider cuts that are related to the well-known boolean quadric polytope introduced by Padberg [297]. The BQP of order $m$ is defined as

$$
B Q^{m}:=\operatorname{conv}\left(\left\{(x, X) \in \mathbb{R}^{m} \times \mathbb{R}^{m(m-1) / 2}: x \in\{0,1\}^{m}, X_{i j}=x_{i} x_{j} \forall 1 \leq i<j \leq m\right\}\right) .
$$

Since the matrix $X$ in our previous relaxations is such that $X_{e f}$ represents $x_{e} x_{f}$, the inequalities that are valid for $B Q^{m}$ can be added to our SDP relaxations. In [297] it is proven that the following triangle inequalities (written in our notation) define facets of $B Q^{m}$ :

$$
X_{e f}+X_{e g} \leq x_{e}+X_{f g} \quad \text { for all } \quad e, f, g \in A, e \neq f, f \neq g, e \neq g .
$$

Although there are more facet-defining inequalities for the BQP, we consider only the abovementioned ones in this chapter. Namely, our preliminary tests show that the triangle inequalities lead to the largest improvement of the SDP bounds. Note that there are $\mathcal{O}\left(\mathrm{m}^{3}\right)$ triangle inequalities and that it is challenging to solve even medium-size SDPs that include all triangle inequalities.

Let $\mathcal{T} \subseteq A \times A \times A$ denote the set of arc triples corresponding to the triangle inequalities, and $\operatorname{let} \mathcal{C}(\mathcal{T})$ be the polyhedron induced by these cuts, i.e.,

$$
\mathcal{C}(\mathcal{T}):=\left\{\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \in \mathcal{S}^{m+1}: X_{e f}+X_{e g} \leq X_{e e}+X_{f g} \forall(e, f, g) \in \mathcal{T}\right\}
$$

where we incorporated the fact that $x_{e}=X_{e e}$ for all $e \in A$ in our relaxations. Then our strongest SDP relaxation is:

$$
\min \left\{\langle Q, X\rangle:\left(\begin{array}{ll}
1 & x^{\top}  \tag{3}\\
x & X
\end{array}\right) \in \mathcal{F}_{1} \cap \mathcal{N}_{+}^{m+1} \cap \mathcal{C}(\mathcal{T})\right\} .
$$

By abuse of notation, we will also use $\mathcal{T}$ to denote a subset of the set of arc triples corresponding to the triangle inequalities within a cutting-plane environment.

### 3.4 Graph-dependent facial reduction

In this section we investigate the Slater feasibility of the relaxations constructed in Section 3.3. We prove that the relaxations are not Slater feasible and show how to obtain facially reduced relaxations. We conclude this section by providing an algorithm that computes a sparse transformation matrix required for the facial reduction. Each transformation matrix is graph-specific, and the algorithm exploits the bipartite representation of the underlying graph.

### 3.4.1 Strict feasibility by facial reduction

Recall that Slater's constraint qualification holds for an SDP relaxation if there exists a feasible solution that is also positive definite. The following lemma shows that Slater's constraint qualification does not hold for the SDP relaxation (3.7), and consequently, neither for (3.9) and (3.10).
Lemma 3.6. Let $Y:=\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right) \in \mathcal{S}_{+}^{m+1}$ be feasible for $\left(S D P_{1}\right)$. Then

$$
\operatorname{Span}\left\{\binom{-1}{u_{i}},\binom{-1}{v_{i}}: i \in N\right\} \subseteq \operatorname{Nul}(Y) .
$$

Proof. It follows directly from the fact that $\langle\tilde{U}, Y\rangle=0$ and positive semidefinite matrices having a nonnegative trace inner product that we have

$$
\left(\begin{array}{ll}
-1 & u_{i}^{\top}
\end{array}\right)\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right)\binom{-1}{u_{i}}=0
$$

for all $i \in N$. Since $Y \succeq \mathbf{0}$, this implies that $Y\binom{-1}{u_{i}}=\mathbf{0}_{m+1}$ for all $i \in N$. Thus, $\binom{-1}{u_{i}} \in$ $\operatorname{Nul}(Y)$ for all $i \in N$. Similarly, one can prove that $\left(-\bar{v}_{i}\right) \in \operatorname{Nul}(Y)$ for all $i \in N$.

Lemma 3.6 shows that our SDP relaxations are not Slater feasible. Thus, the feasible sets of the SDP relaxations are fully contained in one of the faces of $\mathcal{S}_{+}^{m+1}$. For now, we only focus on the relaxation $\left(S D P_{1}\right)$. In order to find an equivalent relaxation for $\left(S D P_{1}\right)$ that is Slater feasible, we project the problem onto the minimal face containing the feasible set, i.e., apply facial reduction, see e.g., $[52,108,191,355]$.

To find the minimal face containing the feasible set of the SDP relaxation, one needs to find its exposing vectors, i.e., the vectors orthogonal to the feasible set of the SDP relaxation. It follows from Lemma 3.6 that the following matrices satisfy that property:

$$
\begin{equation*}
\binom{-1}{u_{i}}\binom{-1}{u_{i}}^{\top} \quad \text { and } \quad\binom{-1}{v_{i}}\binom{-1}{v_{i}}^{\top} \quad \text { for all } i \in N . \tag{3.11}
\end{equation*}
$$

Now, let $\mathcal{R}$ be defined as follows:

$$
\mathcal{R}:=\left(\operatorname{Span}\left\{\binom{-1}{u_{i}},\binom{-1}{v_{i}}: i \in N\right\}\right)^{\perp}=\operatorname{Nul}\left(\left[\begin{array}{ll}
-\mathbf{1}_{n} & U  \tag{3.12}\\
-\mathbf{1}_{n} & V
\end{array}\right]\right)
$$

Observe that under Assumption 3.1 the rank of $\left[\begin{array}{cc}-\mathbf{1}_{n} & U \\ -1_{n} & V\end{array}\right]$ equals the rank of $\left[U^{\top} V^{\top}\right]^{\top}$ which we defined to be $\alpha$, see (3.4). From this it follows that $\operatorname{dim}(\mathcal{R})=m+1-\alpha$. We now define $F_{\mathcal{R}}$ to be the subset of $\mathcal{S}_{+}^{m+1}$ that is orthogonal to the exposing vectors (3.11), i.e.,

$$
F_{\mathcal{R}}:=\left\{X \in \mathcal{S}_{+}^{m+1}: \operatorname{Col}(X) \subseteq \mathcal{R}\right\}
$$

Since faces of $\mathcal{S}_{+}^{m+1}$ are known to be in correspondence with linear subspaces of $\mathbb{R}^{m+1}$, see Section 1.3.3, $F_{\mathcal{R}}$ is a face of $\mathcal{S}_{+}^{m+1}$ containing the feasible set of $\left(S D P_{1}\right)$. Later on we show
that $F_{\mathcal{R}}$ is actually the minimal face with this property, see Theorem 3.8.
In order to derive an explicit expression of $F_{\mathcal{R}}$, let $W \in \mathbb{R}^{(m+1) \times(m+1-\alpha)}$ be a matrix whose columns form a basis for $\mathcal{R}$. Then, the face $F_{\mathcal{R}}$ can be equivalently written as:

$$
\begin{equation*}
F_{\mathcal{R}}=W \mathcal{S}_{+}^{m+1-\alpha} W^{\top} . \tag{3.13}
\end{equation*}
$$

This implies that any $Y \in \mathcal{S}_{+}^{m+1}$ that is feasible for $\left(S D P_{1}\right)$ can be written as $Y=W Z W^{\top}$ for some $Z \in \mathcal{S}_{+}^{m+1-\alpha}$. By substituting this term into $\left(S D P_{1}\right)$, we obtain an equivalent relaxation in a lower dimensional space. As a direct byproduct, some of the original constraints become redundant. The resulting relaxation is as follows:

$$
\min \begin{cases}\left\langle W^{\top} \hat{Q} W, Z\right\rangle: & \left.\begin{array}{r}
\operatorname{diag}\left(W Z W^{\top}\right)=W Z W^{\top} \mathbf{e}_{1}, \\
\mathbf{e}_{1}^{\top} W Z W^{\top} \mathbf{e}_{1}=1, Z \succeq \mathbf{0} \tag{S1}
\end{array}\right\}, ~ \text {, }, ~\end{cases}
$$

where $\hat{Q}:=\left(\begin{array}{cc}0 & \mathbf{o}_{m}^{\top} \\ \mathbf{o}_{m} & Q\end{array}\right)$. Let us define the feasible set of the above relaxation for future reference:

$$
\mathcal{F}_{S 1}:=\left\{Z \in \mathcal{S}_{+}^{m+1-\alpha}: \begin{array}{r}
\operatorname{diag}\left(W Z W^{\top}\right)=W Z W^{\top} \mathbf{e}_{1}  \tag{3.15}\\
\mathbf{e}_{1}^{\top} W Z W^{\top} \mathbf{e}_{1}=1, Z \succeq \mathbf{0}
\end{array}\right\} .
$$

We show below that the SDP relaxations (3.14) and (3.7) are equivalent.
Theorem 3.7. The $S D P$ relaxation $\left(S D P_{S 1}\right)$ is equivalent to the $S D P$ relaxation $\left(S D P_{1}\right)$.
Proof. Let $Z$ be feasible for $\left(S D P_{S 1}\right)$ and define $Y:=W Z W^{\top}, X:=\left[\begin{array}{lll}\mathbf{0}_{m} & \mathbf{I}_{m}\end{array}\right] Y\left[\mathbf{0}_{m} \mathbf{I}_{m}\right]^{\top}$ and $x:=\operatorname{diag}(X)$. Our goal is to show that $x$ and $X$ are feasible for $\left(S D P_{1}\right)$.

Note that the SDP constraint is trivially satisfied. Therefore, it remains to prove that the all-ones constraint and the merged squared linear constraints hold. Observe that

$$
\binom{n}{\mathbf{1}_{m}}^{\top}\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right)\binom{-n}{\mathbf{1}_{m}}=\binom{n}{\mathbf{1}_{m}}^{\top} W Z W^{\top}\binom{-n}{\mathbf{1}_{m}}=\sum_{i \in N}\binom{n}{\mathbf{1}_{m}}^{\top} W Z W^{\top}\binom{-1}{u_{i}}=0,
$$

where the last equality follows from the construction of $W$. Since the most left term in the expression above equals $-n^{2}+\mathbf{1}_{m}^{\top} X \mathbf{1}_{m}$, it follows that $\langle\mathbf{J}, X\rangle=n^{2}$.

Next, we have

$$
\langle\tilde{U}, Y\rangle=\left\langle\sum_{i \in N}\binom{-1}{u_{i}}\binom{-1}{u_{i}}^{\top}, W Z W^{\top}\right\rangle=\sum_{i \in N}\left\langle W^{\top}\binom{-1}{u_{i}}\binom{-1}{u_{i}}^{\top} W, Z\right\rangle=0,
$$

since the columns of $W$ are orthogonal to $\left[-1 u_{i}^{\top}\right]^{\top}$ for all $i \in N$. In a similar fashion we can show that $\langle\tilde{V}, Y\rangle=0$. We conclude that the matrix $X$ and vector $x$ obtained from ( $S D P_{S 1}$ ) are feasible for $\left(S D P_{1}\right)$.

Conversely, let $Y$ be feasible for ( $S D P_{1}$ ). Then it follows from (3.13) that there exists a matrix $Z \succeq \mathbf{0}$ such that $Y=W Z W^{\top}$. Since the objective functions of $\left(S D P_{1}\right)$ and $\left(S D P_{S 1}\right)$ coincide, we conclude that the two relaxations are equivalent.

We now prove that $\left(S D P_{S 1}\right)$ is indeed Slater feasible, see also [355].
Theorem 3.8. The relaxation $\left(S D P_{S 1}\right)$ contains a Slater feasible point.

Proof. Since $\operatorname{conv}(P)$ has dimension $m-\alpha$, see Lemma 3.2, it follows that there exists an affinely independent set of vectors $\left\{x_{1}, \ldots, x_{m+1-\alpha}\right\} \subseteq P$. Because of the affinely independence of these vectors, the set

$$
\left\{\binom{1}{x_{1}},\binom{1}{x_{2}}, \ldots,\binom{1}{x_{m+1-\alpha}}\right\}
$$

is linearly independent in $\mathbb{R}^{m+1}$. Since $\left[1 x_{i}^{\top}\right]^{\top} \in \mathcal{R}$ for all $i \in[m+1-\alpha]$ and the columns of $W$ form a basis for $\mathcal{R}$, there exist vectors $y_{1}, \ldots, y_{m+1-\alpha} \in \mathbb{R}^{m+1-\alpha}$ such that $W y_{i}=\left[1 x_{i}^{\top}\right]^{\top}$ for all $i \in[m+1-\alpha]$. Moreover, the vectors $y_{i}$ are linearly independent in $\mathbb{R}^{m+1-\alpha}$ because of the linear independence of the vectors $W y_{i}$.

We define

$$
Z_{\lambda}:=\sum_{i=1}^{m+1-\alpha} \lambda_{i} y_{i} y_{i}^{\top},
$$

where $\lambda_{i} \geq 0$ for all $i \in[m+1-\alpha]$ and $\mathbf{1}^{\top} \lambda=1$, and rewrite $W Z_{\lambda} W^{\top}$ as follows:

$$
W Z_{\lambda} W^{\top}=\sum_{i=1}^{m+1-\alpha} \lambda_{i} W y_{i}\left(W y_{i}\right)^{\top}=\sum_{i=1}^{m+1-\alpha} \lambda_{i}\binom{1}{x_{i}}\binom{1}{x_{i}}^{\top} .
$$

It is not difficult to see that $Z_{\lambda}$ is feasible for $\left(S D P_{S 1}\right)$. By taking $\lambda_{i}>0$ for all $i \in[m+1-\alpha]$, the resulting matrix $Z_{\lambda}$ is nonsingular, which implies that $Z_{\lambda} \succ \mathbf{0}$. Hence, $\left(S D P_{S 1}\right)$ contains a Slater feasible point.

Note that the key in the proof of Theorem 3.8 is the known dimension of $\operatorname{conv}(P)$.
Continuing in the same vein, one can show that the following SDP relaxation is equivalent to the SDP relaxation (3.9):
$\left(S D P_{S 2}\right)$

$$
\begin{equation*}
\min \left\{\left\langle W^{\top} \hat{Q} W, Z\right\rangle: Z \in \mathcal{F}_{S 1}, \quad W Z W^{\top} \in \mathcal{N}_{+}^{m+1}\right\}, \tag{3.16}
\end{equation*}
$$

and the following relaxation equivalent to the SDP relaxation (3.10):
$\left(S D P_{S 3}\right)$

$$
\begin{equation*}
\min \left\{\left\langle W^{\top} \hat{Q} W, Z\right\rangle: Z \in \mathcal{F}_{S 1}, W Z W^{\top} \in \mathcal{N}_{+}^{m+1} \cap \mathcal{C}(\mathcal{T})\right\} \tag{3.17}
\end{equation*}
$$

### 3.4.2 A polynomial time algorithm for the transformation matrix

Although the subspace $\mathcal{R}$ has been defined algebraically in Section 3.4.1, we now focus on its relation with the graph $G$. This leads to a polynomial time algorithm for computing a sparse transformation matrix $W$ that depends on the considered graph. Although one can compute $W$ numerically, we require its sparse expression for efficient implementation of our cutting-plane algorithm, see Section 3.5.

Recall that the columns of $W$ form a basis for the subspace $\mathcal{R}$, see (3.12). A natural way to construct $W$ is as follows: let $\bar{x} \in P$ be the characteristic vector of any cycle cover in $G$. Moreover, let $\bar{W} \in \mathbb{R}^{m \times(m-\alpha)}$ be a matrix whose columns form a basis for $\operatorname{Nul}\left(\left[U^{\top} V^{\top}\right]^{\top}\right)$.

Then, the matrix

$$
W:=\left(\begin{array}{cc}
1 & \mathbf{0}_{m-\alpha}^{\top}  \tag{3.18}\\
\bar{x} & \bar{W}
\end{array}\right)
$$

forms a basis for the subspace $\mathcal{R}$. Finding a sparse expression for $W$ now boils down to finding a sparse expression for $\bar{W}$. For that purpose, we focus on a graph $B(G)$ that is induced by $G$, the so-called bipartite representation of $G$, which is introduced by BangJensen and Gutin [30]. The graph $B(G)=\left(V_{1} \sqcup V_{2}, E\right)$ is an undirected bipartite graph where $V_{1}$ and $V_{2}$ are copies of the set $N$ and the edge set $E$ is defined as:

$$
E=\left\{\{i, j\} \in V_{1} \times V_{2}:(i, j) \in A\right\} .
$$

By construction, each arc in $G$ corresponds to exactly one edge in $B(G)$, where the orientation in $G$ determines the configuration of the edges in $B(G)$. Figure 3.1 shows an example of $G$ and its corresponding bipartite representation $B(G)$. Observe that a cycle cover in $G$ corresponds to a perfect matching in $B(G)$ and vice versa.


Figure 3.1: Example of graph $G$ and its bipartite representation $B(G)$.
The matrix $\left[U^{\top} V^{\top}\right]^{\top}$ equals the incidence matrix of $B(G)$. Suppose we orient all edges of $B(G)$ from $V_{1}$ to $V_{2}$. The incidence matrix with respect to this orientation is $\left[U^{\top}-V^{\top}\right]^{\top}$. Clearly, we have $\operatorname{Nul}\left(\left[U^{\top} V^{\top}\right]^{\top}\right)=\operatorname{Nul}\left(\left[U^{\top}-V^{\top}\right]^{\top}\right)$. The null space of the incidence matrix of a directed graph is in the literature known as the flow space of a graph. Hence, it follows that the columns of $\bar{W}$ form a basis for the flow space of the bipartite representation of $G$ (with respect to the orientation from $V_{1}$ to $V_{2}$ ).

Let $C$ be a cycle in $B(G)$. Since $B(G)$ is a bipartite graph, $C$ consists of an even number of edges. Let $z \in \mathbb{R}^{m}$ denote its signed characteristic vector, i.e., we alternately assign values +1 and -1 to the edges on $C$ and assign value 0 otherwise. It is well-known that the flow space of a graph is spanned by the signed characteristic vectors of all its cycles. For more information about the flow space of a graph, we refer to e.g., [175].

Hence, $\mathcal{R}$ is related to the cycles of the bipartite representation of $G$. A natural question is how do the cycles of $B(G)$ relate to the original graph $G$ ? To answer this question, we exploit the notion of a closed antidirected trail, which is introduced in [29]. Recall that a trail is a walk in a graph that does not contain repeated arcs, but is allowed to contain
repeated nodes. A closed trail is a trail that has the same start and ending node.
Definition 3.9. A closed antidirected trail (CAT) in a directed graph $G$ is a closed trail of even length with arcs oriented alternately.

A cycle in $B(G)$ corresponds to a CAT in $G$. To verify this, let $\phi: E \rightarrow A$ be the bijection between the edges of $B(G)$ and the arcs of $G$ in the natural way. Then, $C$ equals a cycle in $B(G)$ if and only if $\phi(C)$ equals a CAT in $G$. Obviously, since $C$ starts and ends at the same vertex in $B(G), \phi(C)$ also starts and ends at the same node in $G$. Moreover, since $B(G)$ is bipartite, $C$ and thus $\phi(C)$ must be of even length. Finally, each two consecutive edges of $C$ have one common vertex in $V_{1}$ (resp. $V_{2}$ ) and the other vertices in $V_{2}$ (resp. $V_{1}$ ). By construction of $B(G)$, it follows that two consecutive edges of $C$ correspond to alternately oriented arcs in $G$. Thus $\phi(C)$ is a CAT. The converse statement can be shown in the same fashion. This leads to the following proposition.

Proposition 3.10. The flow space of $B(G)$ equals the subspace spanned by the closed antidirected trails in $G$.

We now have two interpretations of the column space of $\bar{W}$, one with respect to $G$ and the other with respect to $B(G)$. The latter one is more suitable for finding a sparse expression for $\bar{W}$.

From the fact that $\bar{W}$ has $m-\alpha$ columns and the flow space of $B(G)$ has dimension $|E|-\left|V_{1} \sqcup V_{2}\right|+c_{B(G)}$, where $c_{B(G)}$ is the number of connected components in $B(G)$, it follows that:

$$
\begin{equation*}
\alpha=\left|V_{1} \sqcup V_{2}\right|-c_{B(G)}=2 n-c_{B(G)} . \tag{3.19}
\end{equation*}
$$

Observe that the extreme cases are established by the directed cycle and the complete digraph on $n$ nodes, which yield $\alpha=n$ and $\alpha=2 n-1$, respectively.

There exist several natural bases for the flow space of a graph, see e.g., [175]. We use the following construction: Let $T$ be a spanning forest of $B(G)$ and let $E(T) \subseteq E$ denote its corresponding edge set. Then, for all $e \in E \backslash E(T)$, we know that $T \cup\{e\}$ contains a cycle. By alternately assigning values +1 and -1 to the edges of the cycle and assigning value 0 to all remaining edges, we obtain a signed characteristic vector of the cycle. By repeating this construction for all edges in $E \backslash E(T)$, we obtain $m-\alpha$ linearly independent vectors in $\operatorname{Nul}\left(\left[U^{\top} V^{\top}\right]^{\top}\right)$, which form a basis for this subspace. Finding a spanning forest $T$ can be done by a breadth first search. By making the trees in the forest rooted, we can efficiently detect cycles in $T \cup\{e\}, e \in E \backslash E(T)$, based on the levels of the vertices.

The pseudo-code for the computation of a sparse $W$ is given in Algorithm 3.1. This algorithm applies to all QCCP instances under Assumption 3.1.

Remark 3.11. Although Algorithm 3.1 uses $B(G)$ to compute $W$, it is possible to perform the same construction using the original graph $G$. This follows from the fact that the CATs of $G$ form the circuits of a matroid $(A, \mathcal{F})$ where

$$
\mathcal{F}:=\{F \subseteq A: \text { subgraph }(N(F), F) \text { does not contain a CAT }\},
$$

see [29]. Step 2 of Algorithm 3.1 then reduces to finding a maximal basis of $(A, \mathcal{F})$ using a greedy algorithm, while step 4 boils down to finding the unique CAT in $T \cup\{e\}$ using a breadth first search.

```
Algorithm 3.1 Computation of transformation matrix \(W\)
Input: \(G=(N, A)\)
    Construct the bipartite representation \(B(G)=\left(V_{1} \sqcup V_{2}, E\right)\) of \(G\).
    Find a spanning forest \(T\) of \(B(G)\).
    for \(e \in E \backslash E(T)\) do
        Find the unique cycle \(C\) in \(T \cup\{e\}\).
        Alternately assign values +1 and -1 to edges on \(C\).
        Construct vector \(w^{e} \in \mathbb{R}^{m}\) by \(w_{f}^{e}= \begin{cases} \pm 1 & \text { if } f \in C \text { (according to step 5), } \\ 0 & \text { otherwise. }\end{cases}\)
    end for
    Find a cycle cover \(\bar{x} \in P\).
    Let \(W \in \mathbb{R}^{(m+1) \times(m+1-\alpha)}\) be the matrix whose columns are \(\binom{1}{\bar{x}} \cup\left\{\binom{0}{w^{e}}: e \in E \backslash E(T)\right\}\).
```

Output: $W$

### 3.5 A cutting-plane augmented Lagrangian approach

It is known that SDP solvers based on interior-point methods exhibit problems in terms of both time and memory for solving even medium-size SDPs. Moreover, interior-point methods have difficulties with handling additional cutting planes such as nonnegativity constraints and triangle inequalities. Therefore, solving strong SDP models remains a challenging task.

Recently, a promising alternative for solving large-scale SDP relaxations based on alternating direction augmented Lagrangian methods has been investigated, see [61, 308, 350, 363, 382]. There exist several variants of alternating direction augmented Lagrangian methods for solving SDPs, see e.g., [195, 196, 216, 218, 292, 308, 382]. Here, we first consider two variants known as the (original) alternating direction method of multipliers (ADMM) and the Peaceman-Rachford splitting method (PRSM), also called the symmetric ADMM. Then, we present a novel approach that puts these alternating direction augmented Lagrangian methods into a cutting-plane framework. In particular, we show how to efficiently combine the PRSM with Dykstra's projection algorithm [115] within a cutting-plane approach.

### 3.5.1 The alternating direction method of multipliers and the PeacemanRachford splitting method

The ADMM is a first-order method that is introduced in the 1970s to solve large-scale convex optimization problems, see Section 1.3.4. Starting from the augmented Lagrangian function, it decomposes the problem into various subproblems that are relatively easy to solve. In [292], the authors use the ADMM to solve an SDP relaxation for the quadratic assignment problem and in [216] a similar approach is used to compute strong SDP bounds for the quadratic shortest path problem. Their approaches allow for inexpensive iterations and cheap ways for obtaining lower and upper bounds. In this section we first show how to exploit the approach from $[216,292]$ to solve $\left(S D P_{S 2}\right)$ by the ADMM. Then, we present the PRSM for this relaxation.

Let us rewrite $\left(S D P_{S 2}\right)$ by introducing the constraint $Y=W Z W^{\top}$. The purpose of adding this equality is to split the remaining set of constraints into the PSD constraint on $Z$ and the linear constraints on $Y$. To deal with the latter type, we introduce the following set:

$$
\mathcal{Y}:=\left\{Y \in \mathcal{S}^{m+1}: \begin{array}{c}
Y_{11}=1, \quad \operatorname{diag}(Y)=Y \mathbf{e}_{1}, \quad Y_{e f} \leq 1 \quad \forall e \neq f  \tag{3.20}\\
Y \geq \mathbf{0}, \quad \operatorname{tr}(Y)=n+1, \quad Y_{e f}=0 \quad \forall(e, f) \in \mathcal{Z}
\end{array}\right\},
$$

where $\mathcal{Z}$ is given in (3.8). Observe that $\mathcal{Y}$ also contains constraints that are redundant for $\left(S D P_{S 2}\right)$, see Section 3.3. However, these constraints are not redundant in the subproblems after splitting, see (3.23) below. By including them in $\mathcal{Y}$ we therefore speed up the convergence of the ADMM as observed in [216, 218, 292]. Indeed, these constraints make the alternating projections more accurate.

Remark 3.12. Observe that $\mathcal{Y}$ does not contain the redundant constraint $\langle\mathbf{J}, Y\rangle=(n+1)^{2}$. Namely, our preliminary experiments show that the gain in convergence after adding that constraint is not worth the additional computational effort caused by adding it to $\mathcal{Y}$, as further explained in Remark 3.13 below.

Now, the starting point of the algorithm is the following relaxation:

$$
\begin{equation*}
\min \left\{\langle\hat{Q}, Y\rangle: Y=W Z W^{\top}, Y \in \mathcal{Y}, Z \succeq \mathbf{0}\right\} \tag{3.21}
\end{equation*}
$$

that is equivalent to $\left(S D P_{S 2}\right)$. We assume that the transformation matrix $W$ is normalized such that $W^{\top} W=\mathbf{I}$. Observe that the sparse $W$ resulting from Algorithm 3.1 does not have orthogonal columns. Therefore, we apply a QR-decomposition on the matrix obtained from Algorithm 3.1. It has to be noted that by doing so, some of the sparsity of $W$ is lost.

Let $S \in \mathcal{S}^{m+1}$ denote the Lagrange multiplier for the linear constraint $Y=W Z W^{\top}$. We consider the augmented Lagrangian function of (3.21) w.r.t. this constraint for a fixed penalty parameter $\beta>0$ :

$$
L_{\beta}(Z, Y, S):=\langle\hat{Q}, Y\rangle+\left\langle S, Y-W Z W^{\top}\right\rangle+\frac{\beta}{2}\left\|Y-W Z W^{\top}\right\|_{F}^{2}
$$

The ADMM aims to minimize $L_{\beta}(Z, Y, S)$ subject to $Y \in \mathcal{Y}$ and $Z \succeq \mathbf{0}$, while iteratively updating $S$. This problem can be decomposed into subproblems, where we only minimize with respect to one of the matrix variables, while keeping the others fixed.

Suppose that ( $Z^{p}, Y^{p}, S^{p}$ ) denotes the $p$ th iterate of the ADMM. Then the new iterate ( $Z^{p+1}, Y^{p+1}, S^{p+1}$ ) can be obtained by the following updates:

$$
(A D M M)\left\{\begin{align*}
Z^{p+1} & :=\arg \min _{Z \succeq 0} L_{\beta}\left(Z, Y^{p}, S^{p}\right)  \tag{3.22}\\
Y^{p+1} & :=\arg \min _{Y \in \mathcal{Y}} L_{\beta}\left(Z^{p+1}, Y, S^{p}\right) \\
S^{p+1} & :=S^{p}+\gamma \cdot \beta \cdot\left(Y^{p+1}-W Z^{p+1} W^{\top}\right)
\end{align*}\right.
$$

Here $\gamma \in\left(0, \frac{1+\sqrt{5}}{2}\right)$ is the stepsize parameter for updating the Lagrange multiplier $S$, see e.g., [363]. The efficiency of the ADMM depends on the difficulty of solving the subproblems (3.22) and (3.23).

The $Z$-subproblem can be solved as follows, see also [216, 292]:

$$
\begin{aligned}
Z^{p+1} & =\arg \min _{Z \succeq 0}\left(\left\langle\hat{Q}, Y^{p}\right\rangle-\frac{1}{2 \beta}\left\|S^{p}\right\|_{F}^{2}+\frac{\beta}{2}\left\|W Z W^{\top}-\left(Y^{p}+\frac{1}{\beta} S^{p}\right)\right\|_{F}^{2}\right) \\
& =\arg \min _{Z \succeq 0}\left\|W Z W^{\top}-\left(Y^{p}+\frac{1}{\beta} S^{p}\right)\right\|_{F}^{2}=\mathcal{P}_{\succeq 0}\left(W^{\top}\left(Y^{p}+\frac{1}{\beta} S^{p}\right) W\right),
\end{aligned}
$$

where $\mathcal{P}_{\succeq 0}(\cdot)$ denotes the orthogonal projection onto the cone of positive semidefinite matrices, which can be performed explicitly, see e.g., [205].

The $Y$-subproblem can be rewritten as follows:

$$
\begin{aligned}
Y^{p+1} & =\arg \min _{Y \in \mathcal{Y}}\left(\left\langle\hat{Q}, W Z^{p+1} W^{\top}\right\rangle-\frac{\beta}{2}\left\|\frac{\hat{Q}+S^{p}}{\beta}\right\|_{F}^{2}+\frac{\beta}{2}\left\|Y-W Z^{p+1} W^{\top}+\frac{\hat{Q}+S^{p}}{\beta}\right\|_{F}^{2}\right) \\
& =\arg \min _{Y \in \mathcal{Y}}\left\|Y-\left(W Z^{p+1} W^{\top}-\frac{\hat{Q}+S^{p}}{\beta}\right)\right\|_{F}^{2}=\mathcal{P} \mathcal{Y}\left(W Z^{p+1} W^{\top}-\frac{\hat{Q}+S^{p}}{\beta}\right),
\end{aligned}
$$

where $\mathcal{P} \mathcal{Y}(\cdot)$ denotes the orthogonal projection onto the polyhedral set $\mathcal{Y}$.
We now show how to project a matrix $M \in \mathcal{S}^{m+1}$ onto $\mathcal{Y}$. For that purpose, we define several operators, see Table 3.2.

| Operator | Description |  |  |
| :--- | :--- | :--- | :--- |
| $T_{\text {arrow }}$ | $:$ | $\mathcal{S}^{m+1} \rightarrow \mathbb{R}^{m}$ | $T_{\text {arrow }}\left(\left(\begin{array}{cc}x_{0} & x^{\top} \\ x & X\end{array}\right)\right)=\frac{1}{3}(\operatorname{diag}(X)+2 x)$. |
| $T_{\text {arrow }}^{*}$ | $:$ | $\mathbb{R}^{m} \rightarrow \mathcal{S}^{m+1}$ | $T_{\text {arrow }}^{*}(x)=\left(\begin{array}{cc}0 & \frac{1}{3} x^{\top} \\ \frac{1}{3} x & \operatorname{Diag}\left(\frac{1}{3} x\right)\end{array}\right)$. |
|  |  |  |  |
| $T_{\text {inner }}$ | $:$ | $\mathcal{S}^{m+1} \rightarrow \mathcal{S}^{m+1}$ | $T_{\text {inner }}\left(\left(\begin{array}{cc}x_{0} & x^{\top} \\ x & X\end{array}\right)\right)=\left(\begin{array}{cc}0 & \mathbf{0}_{m}^{\top} \\ \mathbf{0}_{m} & \tilde{X}-\operatorname{Diag}(\tilde{X})\end{array}\right)$ where $\tilde{X} \in \mathcal{S}^{m}$ is s.t. |
|  |  | $\tilde{X}_{e f}=0$ if $(e, f) \in \mathcal{Z}$ and $\tilde{X}_{e f}=X_{e f}$ otherwise. |  |
| $T_{\text {box }}$ | $:$ | $\mathcal{S}^{m+1} \rightarrow \mathcal{S}^{m+1}$ | $T_{\text {box }}(X)_{e f}=\min \left(\max \left(X_{e f}, 0\right), 1\right)$ for all $(e, f)$. |

Table 3.2: Overview of operators and their definitions.
Let $\hat{M}$ denote the projection of a matrix $M$ onto $\mathcal{Y}$. The projection can be split into two parts: the projection of the so-called arrow of $M$, i.e., the first row, first column and diagonal of $M$, and the projection of the remaining entries. We specify details below.

We clearly have $\hat{M}_{11}=1$. The remaining entries of the arrow of $\hat{M}$ are obtained as the solution to the following minimization problem:

$$
\min _{y \in \mathbb{R}^{m}}\left\{\left\|y-T_{\text {arrow }}(M)\right\|_{2}^{2}: \mathbf{1}^{\top} y=n, y \geq \mathbf{0}\right\} .
$$

Observe that the problem above boils down to a projection of a vector onto the simplex $\Delta(n)$, where $\Delta(a):=\left\{x \in \mathbb{R}^{m}: \mathbf{1}^{\top} x=a, x \geq \mathbf{0}\right\}$ for all nonnegative $a \in \mathbb{R}$. The projection onto $\Delta(a)$, denoted by $\mathcal{P}_{\Delta(a)}(\cdot)$, can be performed explicitly in $O(m \log m)$, see [199]. The projection of the remaining entries of $M$ is trivial. We conclude that the explicit projection of $M$ onto $\mathcal{Y}$ equals:

$$
\mathcal{P}_{\mathcal{Y}}(M)=\mathbf{E}_{11}+T_{\mathrm{box}}\left(T_{\text {inner }}(M)\right)+T_{\text {arrow }}^{*}\left(3 \cdot \mathcal{P}_{\Delta(n)}\left(T_{\text {arrow }}(M)\right)\right),
$$

where $\mathbf{E}_{11}:=\mathbf{e}_{1} \mathbf{e}_{1}^{\top} \in \mathcal{S}^{m+1}$. The fact that our SDP relaxations satisfy the constant trace property, i.e., $\operatorname{tr}(Y)=n+1$, is exploited in the $Y$-subproblem. The presence of the constant trace property in SDPs has been exploited recently in conditional gradient-based augmented Lagrangian methods. These methods iteratively solve a minimization problem with respect to the set of positive semidefinite matrices having fixed trace, see e.g., [260, 261, 377]. In contrast, our method exploits the constant trace property in the polyhedral projections.

Remark 3.13. As indicated in Remark 3.12, the constraint $\langle\mathbf{J}, Y\rangle=(n+1)^{2}$ is not added to $\mathcal{Y}$ for efficiency reasons. If this constraint would be included, we require the uppertriangular elements $Y_{e f}$ with $e \neq f$ and $(e, f) \notin \mathcal{Z}$ to be between 0 and 1 and together add up to $\frac{1}{2} n(n-1)$. Hence, this asks for a projection onto an $\mathcal{O}\left(m^{2}\right)$-dimensional capped simplex [13], resulting in a worst-case complexity of $\mathcal{O}\left(m^{4}\right)$. Since the ADMM requires this projection to be done in every iteration, the gain in convergence is not worth the additional computation overhead of including it.

In the ADMM the Lagrange multiplier is only updated after both primal variables have been updated. We present below the Peaceman-Rachford splitting method (PRSM) or the symmetric ADMM with larger stepsize [196]. This method consists of two dual updates per iteration. Let ( $Z^{p}, Y^{p}, S^{p}$ ) denote the $p$ th iterate of the PRSM. Then, the following iterative scheme is applied:

$$
(P R S M)\left\{\begin{align*}
Z^{p+1} & :=\arg \min _{Z \succeq \mathbf{0}} L_{\beta}\left(Z, Y^{p}, S^{p}\right),  \tag{3.25}\\
S^{p+\frac{1}{2}} & :=S^{p}+\gamma_{1} \cdot \beta \cdot\left(Y^{p}-W Z^{p+1} W^{\top}\right), \\
Y^{p+1} & :=\arg \min _{Y \in \mathcal{Y}} L_{\beta}\left(Z^{p+1}, Y, S^{p+\frac{1}{2}}\right), \\
S^{p+1} & :=S^{p+\frac{1}{2}}+\gamma_{2} \cdot \beta \cdot\left(Y^{p+1}-W Z^{p+1} W^{\top}\right) .
\end{align*}\right.
$$

Here $\gamma_{1}$ and $\gamma_{2}$ are parameters that must be carefully chosen in order to guarantee convergence. The PRSM is known for accelerated speed of convergence in comparison with other ADMM-like algorithms, see [196].

### 3.5.2 ADMM versus PRSM: preliminary results

In Section 3.5.1 we present two methods for solving ( $S D P_{S 2}$ ): the ADMM and the PRSM. Both approaches can be incorporated within the cutting-plane augmented Lagrangian method that we present later. We here provide some preliminary experiments to present the behaviour of both methods in terms of convergence.

We consider a test set of 10 Erdős-Rényi instances with $m$ ranging from 250 to 750 , see Section 3.7 for a specification of these instances. For each instance, we use the ADMM and the PRSM to compute $\left(S D P_{S 2}\right)$ under the same parameter settings as will be explained in Section 3.7. We compute lower bounds obtained from the methods, see Section 3.5.5.3, and scale them such that the final bound is indexed to 100 . Figure 3.2 a shows these scaled bounds for all instances, while Figure 3.2b shows their average over all instances with respect to the number of iterations performed.

Figure 3.2 shows that although both methods converge, the PRSM in general produces strong lower bounds faster than the ADMM. This is in line with the accelerated numerical performance of the PRSM in contrast to the ADMM presented in [196]. Because we desire a fast convergence when iteratively adding cuts, we incorporate the PRSM in the cutting-plane augmented Lagrangian approach introduced in Section 3.5.5.

### 3.5.3 Projection onto a single BQP Cut

The implementation of the ADMM and the PRSM discussed in the previous section can be used to solve $\left(S D P_{S 1}\right)$ and $\left(S D P_{S 2}\right)$. In order to solve $\left(S D P_{S 3}\right)$, the constraints $Y \in \mathcal{C}(\mathcal{T})$ are added to the set of polyhedral constraints, which significantly increases the complexity of


Figure 3.2a: Lower bounds for the ADMM (dashed) and the PRSM (solid) for full test set.


Figure 3.2b: Lower bounds for the ADMM (dashed) and the PRSM (solid) on average.
the $Y$-subproblem (3.23). To project onto $\mathcal{Y} \cap \mathcal{C}(\mathcal{T})$, we use an iterative projection framework, see Section 3.5.4. In this section, we first show how to project onto the polyhedron induced by a single triangle inequality.

Let $(e, f, g)$ denote an arc triple with $e \neq f, f \neq g, e \neq g$, corresponding to a (possibly violated) triangle inequality, e.g., resulting from a cutting-plane framework. Now, let $\mathcal{H}_{\text {efg }}$ be the following polyhedron:

$$
\mathcal{H}_{e f g}:=\left\{Y \in \mathcal{S}^{m+1}: Y_{e f}+Y_{e g} \leq Y_{e e}+Y_{f g}, \operatorname{diag}(Y)=Y \mathbf{e}_{1}\right\}
$$

Let $\mathcal{P}_{\mathcal{H}_{e f g}}(M)$ denote the projection of a matrix $M \in \mathcal{S}^{m+1}$ onto $\mathcal{H}_{\text {efg }}$. This projection can be obtained explicitly as stated by the following lemma, where we restrict ourselves to the matrices $M$ that make their appearance in the algorithm that is presented in Section 3.5.4.

Lemma 3.14. Let $M \in \mathcal{S}^{m+1}$ be such that $\operatorname{diag}(M)=M \mathbf{e}_{1}$ and let $\hat{M}:=\mathcal{P}_{\mathcal{H}_{e f g}}(M)$. If $M_{e f}+M_{e g} \leq \frac{M_{e e}+2 M_{1 e}}{3}+M_{f g}$, then

$$
\hat{M}_{s t}= \begin{cases}\frac{1}{3} M_{e e}+\frac{2}{3} M_{1 e} & \text { if }(s, t) \in\{(1, e),(e, 1),(e, e)\} \\ M_{s t} & \text { otherwise }\end{cases}
$$

If $M_{e f}+M_{e g}>\frac{M_{e e}+2 M_{1 e}}{3}+M_{f g}$, then the projection $\hat{M}$ can be written explicitly as:

$$
\hat{M}_{s t}= \begin{cases}\frac{1}{11} M_{e e}+\frac{2}{11} M_{1 e}+\frac{3}{11} M_{f g}+\frac{8}{11} M_{e f}-\frac{3}{11} M_{e g} & \text { if }(s, t) \in\{(e, f),(f, e)\}, \\ \frac{1}{11} M_{e e}+\frac{2}{11} M_{1 e}+\frac{3}{11} M_{f g}-\frac{3}{11} M_{e f}+\frac{8}{11} M_{e g} & \text { if }(s, t) \in\{(e, g),(g, e)\}, \\ -\frac{1}{11} M_{e e}-\frac{2}{11} M_{1 e}+\frac{8}{11} M_{f g}+\frac{3}{11} M_{e f}+\frac{3}{11} M_{e g} & \text { if }(s, t) \in\{(f, g),(g, f)\}, \\ \frac{3}{11} M_{e e}+\frac{6}{11} M_{1 e}-\frac{2}{11} M_{f g}+\frac{2}{11} M_{e f}+\frac{2}{11} M_{e g} & \text { if }(s, t) \in\{(1, e),(e, 1),(e, e)\}, \\ M_{s t} & \text { otherwise. }\end{cases}
$$

Proof. The matrix $\hat{M}$ equals the solution of the following convex optimization problem:

$$
\min _{\hat{M} \in \mathcal{S}^{m+1}}\left\{\|\hat{M}-M\|_{F}^{2}: \hat{M} \in \mathcal{H}_{e f g}\right\} .
$$

Since $\hat{M}_{s t}=M_{s t}$ for all entries $(s, t)$ that are not involved in the constraints, this optimization problem boils down to:

$$
\begin{aligned}
\min _{\delta, \theta, \mu, \pi} & 2\left(\delta-M_{e f}\right)^{2}+2\left(\theta-M_{e g}\right)^{2}+2\left(\mu-M_{f g}\right)^{2}+\left(\pi-M_{e e}\right)^{2}+2\left(\pi-M_{1 e}\right)^{2} \\
\text { s.t. } & \delta+\theta \leq \pi+\mu .
\end{aligned}
$$

The explicit expression of $\hat{M}$ follows from the KKT-conditions of the problem above. Let $\lambda \geq 0$ be the Lagrange multiplier of the inequality $\delta+\theta \leq \pi+\mu$. Then, the KKT conditions lead to the following system:

$$
\begin{cases}4\left(\delta-M_{e f}\right)+\lambda=0, & \lambda \geq 0 \\ 4\left(\theta-M_{e g}\right)+\lambda=0, & \lambda(\delta+\theta-\mu-\pi)=0 \\ 4\left(\mu-M_{f g}\right)-\lambda=0, & \delta+\theta \leq \pi+\mu \\ 2\left(\pi-M_{e e}\right)+4\left(\pi-M_{1 e}\right)-\lambda=0 . & \end{cases}
$$

Complementarity implies that either $\mu=\delta+\theta-\pi$ or $\lambda=0$. The latter case leads to the KKT-point $(\delta, \theta, \mu, \pi)=\left(M_{e f}, M_{e g}, M_{f g}, \frac{M_{e e}+2 M_{1 e}}{3}\right)$, which is optimal if and only if $M_{e f}+M_{e g} \leq \frac{M_{e e}+2 M_{1 e}}{3}+M_{f g}$. Otherwise, the substitution $\mu=\delta+\theta-\pi$ leads to the system

$$
\left\{\begin{array} { l } 
{ 4 ( \delta - M _ { e f } ) + \lambda = 0 } \\
{ 4 ( \theta - M _ { e g } ) + \lambda = 0 } \\
{ 4 ( \delta + \theta - \pi - M _ { f g } ) - \lambda = 0 } \\
{ 6 \pi - 2 M _ { e e } - 4 M _ { 1 e } - \lambda = 0 }
\end{array} \quad \text { or equivalently, } \quad \left\{\begin{array}{l}
\delta=-\frac{1}{4} \lambda+M_{e f} \\
\theta=-\frac{1}{4} \lambda+M_{e g} \\
4\left(\delta+\theta-\pi-M_{f g}\right)-\lambda=0 \\
\lambda=6 \pi-2 M_{e e}-4 M_{1 e}
\end{array}\right.\right.
$$

Substitution into the third equation yields

$$
\begin{array}{rlrl} 
& & 4 \pi & =4\left(-\frac{1}{4} \lambda+M_{e f}-\frac{1}{4} \lambda+M_{e g}-M_{f g}\right)-\lambda \\
& \Longleftrightarrow & 4 \pi & =-3 \lambda+4 M_{e f}+4 M_{e g}-4 M_{f g} \\
& \Longleftrightarrow \quad 4 \pi & =-3\left(6 \pi-2 M_{e e}-4 M_{1 e}\right)+4 M_{e f}+4 M_{e g}-4 M_{f g} \\
& \Longleftrightarrow & \pi & =\frac{3}{11} M_{e e}+\frac{6}{11} M_{1 e}-\frac{2}{11} M_{f g}+\frac{2}{11} M_{e f}+\frac{2}{11} M_{e g}
\end{array}
$$

By substitution of this expression into the remaining three equations, we obtain:

$$
\begin{aligned}
& \lambda=-\frac{4}{11} M_{e e}-\frac{8}{11} M_{1 e}-\frac{12}{11} M_{f g}+\frac{12}{11} M_{e f}+\frac{12}{11} M_{e g}, \\
& \delta=\frac{1}{11} M_{e e}+\frac{2}{11} M_{1 e}+\frac{3}{11} M_{f g}+\frac{8}{11} M_{e f}-\frac{3}{11} M_{e g}, \\
& \theta=\frac{1}{11} M_{e e}+\frac{2}{11} M_{1 e}+\frac{3}{11} M_{f g}-\frac{3}{11} M_{e f}+\frac{8}{11} M_{e g}, \\
& \mu=-\frac{1}{11} M_{e e}-\frac{2}{11} M_{1 e}+\frac{8}{11} M_{f g}+\frac{3}{11} M_{e f}+\frac{3}{11} M_{e g} .
\end{aligned}
$$

By setting $\hat{M}_{s t}=\delta$ for $(s, t) \in\{(e, f),(f, e)\}, \hat{M}_{s t}=\theta$ for $(s, t) \in\{(e, g),(g, e)\}, \hat{M}_{s t}=\mu$ for $(s, t) \in\{(f, g),(g, f)\}$ and $\hat{M}_{s t}=\pi$ for $(s, t) \in\{(1, e),(e, 1),(e, e)\}$, the claim follows.

### 3.5.4 Semi-parallel Dykstra's projection algorithm

A reasonable argument for the fact that a cutting-plane technique in an alternating direction augmented Lagrangian approach has never been considered before, is the increasing complexity of the involved projections. In our case, it requires a projection onto the intersection of $\mathcal{Y}$, see (3.20), and a finite collection of polyhedra $\mathcal{H}_{\text {efg }}$. This can be performed in an iterative approach based on Dykstra's projection algorithm [55, 115]. Although there exist some similarities between the ADMM and Dykstra's algorithm, see [351], we are the first that combine both methods to compute SDP bounds.

Finding the projection onto the intersection of polyhedra or general convex sets is a well-known problem for which multiple algorithms have been proposed. For a detailed background on projection methods, we refer the reader to [35, 69]. Bauschke and Koch [35] compare several projection algorithms for problems motivated by road design and conclude that Dykstra's cyclic algorithm performs best for projections onto the intersection of convex sets. The idea behind Dykstra's algorithm is to iteratively project a deflected version of the previous iterate onto the individual sets. This method was first proposed by Dykstra [115] for closed convex cones in finite-dimensional Euclidean spaces and later generalized to closed convex sets in Hilbert spaces by Boyle and Dykstra [55].

From now on, we assume the set $\mathcal{T}$ to be ordered. That is, $\mathcal{T}=\left(\left(e_{i}, f_{i}, g_{i}\right)\right)_{i=1}^{T}$ is an ordered set of $T$ arc triples corresponding to violated inequalities. We are interested in the following best approximation problem:

$$
\begin{equation*}
\min \|\hat{M}-M\|_{F}^{2} \quad \text { s.t. } \quad \hat{M} \in \mathcal{Y}_{\mathcal{T}}:=\mathcal{Y} \cap\left(\bigcap_{\left(e_{i}, f_{i}, g_{i}\right) \in \mathcal{T}} \mathcal{H}_{e_{i} f_{i} g_{i}}\right) \text {, } \tag{3.29}
\end{equation*}
$$

where $M$ is the matrix that we project onto $\mathcal{Y}_{\mathcal{T}}$. Observe that $\mathcal{Y}_{\mathcal{T}}=\mathcal{Y} \cap \mathcal{C}(\mathcal{T})$.
Dykstra's algorithm initializes the so-called normal matrices $R_{\mathcal{Y}}^{0}=\mathbf{0}$ and $R_{e_{i} f_{i} g_{i}}^{0}=\mathbf{0}$ for
all $\left(e_{i}, f_{i}, g_{i}\right) \in \mathcal{T}$. Now, we set $X^{0}=M$ and iterate for $k \geq 1$ :

$$
\left.\begin{array}{rl}
X^{k} & :=\mathcal{P}_{\mathcal{Y}}\left(X^{k-1}+R_{\mathcal{Y}}^{k-1}\right) \\
R_{\mathcal{Y}}^{k} & :=X^{k-1}+R_{\mathcal{Y}}^{k-1}-X^{k} \\
L_{e_{i}} f_{i} g_{i} & :=X^{k}+R_{e_{i} f_{i} g_{i}}^{k-1} \\
X^{k} & :=\mathcal{P}_{\mathcal{H}_{i} f_{i} g_{i}}\left(L_{e_{i} f_{i} g_{i}}\right) \\
R_{e_{i} f_{i} g_{i}}^{k} & :=L_{e_{i} f_{i} g_{i}}-X^{k}
\end{array}\right\} \text { for } i=1, \ldots, T
$$

Several authors have shown that the sequence $\left(X^{k}\right)_{k \geq 1}$ strongly converges to the solution of the best approximation problem (3.29), see [55, 150, 194]. Since we project onto the polyhedra $\mathcal{Y}, \mathcal{H}_{e_{1} f_{1} g_{1}}, \ldots, \mathcal{H}_{e_{T} f_{T} g_{T}}$ in a cyclic order, the iterates (CycDyk) are refered to as Dykstra's cyclic algorithm. It was shown in [96] that (CycDyk) has a linear rate of convergence in case the sets to be projected on are polyhedral. Observe that if $\mathcal{T}=\emptyset$, then (CycDyk) boils down to a single projection onto $\mathcal{Y}$.

Instead of projecting on each polyhedron one after another, it is also possible to project on all polyhedra simultaneously. This method is referred to as Dykstra's parallel algorithm. We refer the interested reader to Appendix A. 1 for an implementation and some details of this parallel version. Although the parallel version takes longer to converge in our case, the projections can be done simultaneously, which might be beneficial if used on parallel machines. Preliminary experiments show that in our cutting-plane setting the parallel version, not implemented on parallel machines, is not able to improve on the cyclic version.

To increase the efficiency of (CycDyk), we can, however, partly parallelize the algorithm. Note that a projection onto $\mathcal{H}_{e_{i} f_{i} g_{i}}$ only concerns the entries $\left(e_{i}, f_{i}\right),\left(e_{i}, g_{i}\right),\left(f_{i}, g_{i}\right),\left(e_{i}, e_{i}\right)$ and $\left(1, e_{i}\right)$. Hence, if two projections onto $\mathcal{H}_{e_{1} f_{1} g_{1}}$ and $\mathcal{H}_{e_{2} f_{2} g_{2}}$ take place one after another and $\left\{e_{1}, f_{1}, g_{1}\right\} \cap\left\{e_{2}, f_{2}, g_{2}\right\}=\emptyset$, they can in fact be performed simultaneously. We partition the triples in $\mathcal{T}$ into $r$ clusters $C_{i}, i=1, \ldots, r$, such that $C_{1} \cup \cdots \cup C_{r}=\mathcal{T}$ and $C_{i} \cap C_{j}=\emptyset$ for all $i, j$. By doing so, an iterate of (CycDyk) is performed in $r+1$ consecutive steps, instead of $T+1$ consecutive steps. This requires the set of clusters $\left(C_{i}\right)_{i=1}^{r}$ to be ordered again, e.g., by fixing some arbitrary ordering. More details about this clustering step are given in Section 3.5.5.2. This provides a semi-parallel implementation of (CycDyk).

We take the following actions to further accelerate the algorithm:

- All matrices in (CycDyk) are symmetric, hence we save memory by only working with the upper triangular part of the matrices;
- The normal matrices $R_{e_{i} f_{i} g_{i}}^{k}$ for all $\left(e_{i}, f_{i}, g_{i}\right) \in \mathcal{T}$ are very sparse, i.e., the only nonzero entries correspond to indices $\left(e_{i}, f_{i}\right),\left(e_{i}, g_{i}\right),\left(f_{i}, g_{i}\right),\left(e_{i}, e_{i}\right)$ and (1, $\left.e_{i}\right)$. Therefore, we work with normal vectors corresponding to the nonzero elements in $R_{e_{i} f_{i} g_{i}}^{k}$ instead of using full $(m+1) \times(m+1)$ matrices. This has the additional advantage that the memory needed does not increase with the size of the instance;
- The projection onto $\mathcal{Y}$ is considerably more costly than the projection onto the triangle inequalities in terms of computation time. Instead of performing all separate projections exactly once and iterate, numerical tests show that the convergence is accelerated if we perform the projection onto $\mathcal{Y}$ only occasionally. That is, after the projection onto $\mathcal{Y}$ we perform the $T$ triangle inequality projections $K$ times in a cyclic order before we again project onto $\mathcal{Y}$.


### 3.5.5 A cutting-plane augmented Lagrangian method

In this section we combine the PRSM discussed in Sections 3.5.1 and 3.5.2 with the projection method discussed in Sections 3.5.3 and 3.5.4. This leads to a cutting-plane augmented Lagrangian method (CP-ALM). To the best of our knowledge, no such algorithm exists for solving SDP problems.

In the CP-ALM, we iteratively solve $\left(S D P_{S 3}\right)$ for a set of ordered cuts $\mathcal{T}$ using the PRSM. Each time the PRSM has converged up to some precision, we evaluate the solution for violated cuts and add the numCuts most violated ones to $\mathcal{T}$ according to some (arbitrary) ordering, where numCuts is a predefined parameter, and repeat. An advantage of using the PRSM in a cutting-plane approach, as opposed to an interior-point method, is that after the addition of new cuts we can start the new PRSM loop from the last obtained triple $\left(Z^{p}, Y^{p}, S^{p}\right)$. In other words, we exploit the use of warm starts, which speeds up the convergence. Overall convergence of the CP-ALM follows from that of the PRSM [196] and the fact that it suffices to solve the subproblems to a nearly optimal solution [118].

The CP-ALM is provided in Algorithm 3.2. In the sequel, we explain several ingredients of the algorithm in more detail.

```
Algorithm 3.2 CP-ALM
Input: \(\varepsilon_{P R S M}, \varepsilon_{\text {stag }}, \varepsilon_{\text {proj }}\), maxIter, maxTotalIter, maxStagIter
    Compute \(\widetilde{W}\) by Algorithm 3.1 and perform a QR-decomposition on \(\widetilde{W}\) to obtain \(W\).
    Set \(Y^{0}=\mathbf{0}, Z^{0}=\mathbf{0}, S^{0}=\mathbf{0}, p=0\) and \(\mathcal{T}=\emptyset\).
    while stopping criteria not met do \(\quad\) See Section 3.5.5.1
        while stopping criteria not met do \(\quad \square\) See Section 3.5.5.1
            \(Z^{p+1}:=\mathcal{P}_{\succeq \mathbf{0}}\left(W^{\top}\left(Y^{p}+\frac{1}{\beta} S^{p}\right) W\right)\).
            \(S^{p+\frac{1}{2}}:=S^{p}+\gamma_{1} \cdot \beta \cdot\left(Y^{p}-W Z^{p+1} W^{\top}\right)\).
            \(Y^{p+1}:=\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}\left(W Z^{p+1} W^{\top}-\frac{\hat{Q}+S^{p+\frac{1}{2}}}{\beta}\right)\) by solving (3.29) using semi-parallel (CycDyk).
            \(S^{p+1}:=S^{p+\frac{1}{2}}+\gamma_{2} \cdot \beta \cdot\left(Y^{p+1}-W Z^{p+1} W^{\top}\right)\).
            \(p \leftarrow p+1\).
        end while
        Identify violated inequalities and add the numCuts most violated cuts to \(\mathcal{T}\) w.r.t. some ordering.
        Cluster the cuts in \(\mathcal{T}\) into sets \(C_{1}, \ldots, C_{r}\). \(\triangleright\) See Section 3.5.5.2
    end while
    Compute \(L B\left(S^{p}\right)\) using the final dual variable \(S^{p}\). \(\triangleright\) See Section 3.5.5.3
Output: \(L B\left(S^{p}\right)\)
```


### 3.5.5.1 Stopping criteria

The inner while-loop of Algorithm 3.2 constructs a PRSM sequence for a fixed $\mathcal{T}$. Experiments show that the algorithm is stabilized if, as opposed to adding many cuts at once, we add cuts smoothly in order to keep the residuals small. Hence, we want the inner PRSM sequence to converge before adding new cuts to $\mathcal{T}$. We consider three types of stopping criteria for the inner while-loop:

1. Let $\varepsilon_{P R S M}>0$ be a predefined tolerance parameter. The inner while-loop is terminated after iteration $p$ if

$$
\min \left(\left\|Y^{p+1}-W Z^{p+1} W^{\top}\right\|_{F}, \beta\left\|W^{\top}\left(Y^{p+1}-Y^{p}\right) W\right\|_{F}\right)<\varepsilon_{P R S M}
$$

The first term on the left-hand side measures primal feasibility, while the second term measures dual feasibility.
2. We stop when a fixed number of iterations maxIter is reached.
3. We add a stagnation criterion. Let $\varepsilon_{\text {stag }}>0$ be a parameter. We introduce a variable stagIter that is increased by one each time we have $\left|\left\langle Y^{p+1}, \hat{Q}\right\rangle-\left\langle Y^{p}, \hat{Q}\right\rangle\right|<\varepsilon_{\text {stag }}$. We stop the inner while-loop whenever stagIter $>$ maxStagIter for some predefined integer maxStagIter.
The Dykstra algorithm in line 7 of Algorithm 3.2 is stopped when $\left\|X^{p+1}-X^{p}\right\|_{F}<\varepsilon_{\text {proj }}$ for some predefined $\varepsilon_{\text {proj }}>0$.

Finally, the outer while-loop, i.e., the cutting-plane part, is stopped whenever no more violated cuts can be found or after a predefined number of iterations maxTotalIter $>$ maxIter has been reached.

### 3.5.5.2 Clustering

As explained in Section 3.5.4, the cyclic Dykstra algorithm can be partially parallelized by partitioning the set $\mathcal{T}$ into $r$ clusters of nonoverlapping cuts. We explain here how this clustering is done.

Let $H=(V, E)$ denote a graph where each node $i \in V$ represents a cut in $\mathcal{T}$ and two nodes are connected by an edge whenever the corresponding cuts are overlapping. Clustering $\mathcal{T}$ into the smallest number of nonoverlapping sets is then equivalent to finding a minimum coloring in $H$. This problem is known to be $\mathcal{N} \mathcal{P}$-hard. Galinier and Hertz [154] provide an overview of graph coloring heuristics, where it is concluded that the Tabucol algorithm of Hertz and De Werra [203] is overall very successful. We implement here the improved Tabucol algorithm provided in [153].

### 3.5.5.3 Lower bound

After each CP-ALM iterate $p$, we obtain a triple ( $Z^{p}, Y^{p}, S^{p}$ ) which allows us to compute $\left\langle\hat{Q}, Y^{p}\right\rangle$. Although this value converges to the optimal solution of the SDP relaxation $\left(S D P_{S 3}\right)$, the convergence is typically not monotonic, which implies that this value does not necessarily provide a lower bound for the QCCP instance. We can still use the output of the CP-ALM to obtain a lower bound. Various methods for obtaining lower bounds from approximate solutions have been proposed in the literature [117, 225, 292]. We adopt here the method introduced by Oliveira et al. [292].

Let $\mathcal{W}^{\top} \mathcal{S W}:=\left\{S: W^{\top} S W \preceq \mathbf{0}\right\}$. Then, a lower bound is obtained by solving:

$$
\begin{equation*}
L B\left(S^{p}\right):=\min _{Y \in \mathcal{Y}_{\mathcal{T}}}\left\langle\hat{Q}+\mathcal{P}_{\mathcal{W}^{\top} \mathcal{S W}}\left(S^{p}\right), Y\right\rangle, \tag{3.30}
\end{equation*}
$$

where $\mathcal{P}_{\mathcal{W}^{\top} \mathcal{S} \mathcal{W}}\left(S^{p}\right)$ is the projection of $S^{p}$ onto the set $\mathcal{W}^{\top} \mathcal{S} \mathcal{W}$. This projection can be performed efficiently, see [292]. Moreover, note that (3.30) is a linear programming problem.

### 3.6 SDP-based upper bounds for the QCCP

The matrices resulting from the CP-ALM can be used to construct upper bounds for the QCCP. In this section we derive several upper bounding approaches, among which a deterministic method, two randomized algorithms and a Q-learning algorithm that is based
on reinforcement learning. We are not aware of other SDP-based rounding algorithms that make use of reinforcement learning. We end the section by providing a hybrid approach that combines all aforementioned heuristics.

### 3.6.1 Best Euclidean approximation

Let ( $Z^{\text {out }}, Y^{\text {out }}, S^{\text {out }}$ ) be the outcome of the CP-ALM. Throughout the entire section we assume that the CP-ALM is solved up to high precision in order for the utilized results to be valid. Let $x^{\text {out }}$ be the vector consisting of the diagonal elements of $Y^{\text {out }}$ excluding the first entry. As $x^{\text {out }}$ is an approximation of the optimal cycle cover, one can search for the vector $x \in P$ that is closest to $x^{\text {out }}$ in Euclidean norm. This vector can be obtained as follows:

$$
\begin{equation*}
x^{*}:=\arg \max \left\{x^{\top} x^{\text {out }}: x \in P\right\} . \tag{3.31}
\end{equation*}
$$

The problem (3.31) can be solved as a linear cycle cover problem, e.g., via a linear programming solver. The corresponding upper bound is $U B_{E B}:=\left(x^{*}\right)^{\top} Q x^{*}$.

### 3.6.2 Randomized undersampling

Randomized SDP-based heuristics have proven to be successful for various optimization problems, mainly sparked by the seminal work of Goemans and Williamson [176]. A widely used procedure in the design of approximation algorithms is randomized rounding [313], which rounds a relaxed solution to a solution for the original problem that is close to optimal in expectation. We present an SDP-based randomized rounding algorithm that we refer to as randomized undersampling.

Let $x^{\text {out }} \in \mathbb{R}^{m}$ be as discussed in Section 3.6.1. Observe that since all entries of $x^{\text {out }}$ are nonnegative and $\sum_{e \in \delta^{+}(i)} x_{e}^{\text {out }}=1$, see (3.2), we can view $\left\{x_{e}\right\}_{e \in \delta^{+}(i)}$ as a probability distribution on all arcs leaving node $i$. Similarly, $\left\{x_{e}\right\}_{e \in \delta^{-}(i)}$ represents a probability distribution on the set of arcs entering node $i$. Hence, for each node $i$ we can draw exactly one arc from $\delta^{+}(i)$ according to the distribution $\left\{x_{e}\right\}_{e \in \delta^{+}(i)}$. Let $y_{1} \in\{0,1\}^{m}$ denote the characteristic vector of the outcome of these $n$ trials. We do the same for the incoming arcs, yielding a vector $y_{2} \in\{0,1\}^{m}$. By construction we have $U y_{1}=V y_{2}=\mathbf{1}_{n}$, but $V y_{1}=U y_{2}=\mathbf{1}_{n}$ are not necessarily satisfied.

The vector $y=y_{1} \circ y_{2}$ denotes a partial cycle cover that satisfies $U y \leq \mathbf{1}_{n}$ and $V y \leq \mathbf{1}_{n}$. Observe that the probability of including arc $e$ in $y$ equals $x_{e}^{2}$. To extend $y$ to a feasible cycle cover, we define:

$$
\begin{equation*}
N^{+}:=\left\{i \in N: y_{e}=0 \quad \forall e \in \delta^{+}(i)\right\} \quad \text { and } \quad N^{-}:=\left\{i \in N: y_{e}=0 \quad \forall e \in \delta^{-}(i)\right\} . \tag{3.32}
\end{equation*}
$$

We still have to select exactly one arc from $\delta^{+}(i)$ for all $i \in N^{+}$and one arc from $\delta^{-}(i)$ for all $i \in N^{-}$to extend $y$ to a feasible cycle cover. We can do this by solving a modified version of (3.31). Let $U_{N^{+}} \in \mathbb{R}^{\left|N^{+}\right| \times m}$ (resp. $V_{N^{-}} \in \mathbb{R}^{\left|N^{-}\right| \times m}$ ) denote the submatrix of $U$ (resp. $V$ ) induced by the rows corresponding to $N^{+}$(resp. $N^{-}$). Let us define the following vector:

$$
\bar{x}_{e}^{\text {out }}:= \begin{cases}-\infty & \text { if } e^{-} \in N \backslash N^{-} \text {or } e^{+} \in N \backslash N^{+},  \tag{3.33}\\ x_{e}^{\text {out }} & \text { otherwise },\end{cases}
$$

where some values are set to $-\infty$ in order to avoid in- or outflows larger than one. We now
solve

$$
\begin{equation*}
z^{*}:=\underset{z \in \mathbb{R}^{m}}{\arg \max }\left\{z^{\top} \bar{x}^{\text {out }}: U_{N^{+}} z=\mathbf{1}_{\left|N^{+}\right|}, \quad V_{N^{-}} z=\mathbf{1}_{\left|N^{-}\right|}, \mathbf{0}_{m} \leq z \leq \mathbf{1}_{m}\right\} . \tag{3.34}
\end{equation*}
$$

A partial solution $y$ can be extended to a feasible cycle cover if and only if the optimal value to (3.34) is finite. Indeed, in that case we have $y+z^{*} \in P$, which yields the bound $U B_{U S}=\left(y+z^{*}\right)^{\top} Q\left(y+z^{*}\right)$. We now repeat this procedure and store the smallest obtained bound.

As we select at most $n$ arcs at random and extend the solution to a full cycle cover, we call this method randomized undersampling. The steps of this method are summarized in Algorithm 3.3.

```
Algorithm 3.3 Randomized undersampling for the QCCP
Input: \(G, Q, x^{\text {out }}\)
    Initialize \(y_{1}=\mathbf{0}_{m}\) and \(y_{2}=\mathbf{0}_{m}\).
    for \(i \in N\) do
        Draw \(f_{1}\) from \(\delta^{+}(i)\) with respect to \(\left\{x_{e}^{\text {out }}\right\}_{e \in \delta^{+}(i)}\) and \(f_{2}\) from \(\delta^{-}(i)\) with respect to \(\left\{x_{e}^{\text {out }}\right\}_{e \in \delta^{-}(i)}\).
        Set \(y_{1}\left(f_{1}\right)=1\) and \(y_{2}\left(f_{2}\right)=1\).
    end for
    \(y \leftarrow y_{1} \circ y_{2}\).
    Obtain the sets \(N^{+}\)and \(N^{-}\)and the vector \(\bar{x}^{\text {out }} \in \mathbb{R}^{m}\) as in (3.32) and (3.33), respectively.
    if problem (3.34) has a finite objective value then
        \(U B_{U S} \leftarrow\left(y+z^{*}\right)^{\top} Q\left(y+z^{*}\right)\) where \(z^{*}\) is computed by (3.34).
    else
            Go back to Step 2
    end if
Output: \(U B_{U S}\)
```


### 3.6.3 Randomized oversampling

Instead of sampling a partial solution and deterministically extend it to a full cycle cover, we can also randomly add arcs to a subgraph $H$ of $G$ until it contains a cycle cover visiting all nodes. We call this method randomized oversampling.

We initialize $H=(N, \emptyset)$ and iteratively add pairs of successive arcs to $H$. This is done randomly using a probability distribution on the set $\delta^{-}(i) \times \delta^{+}(i)$ for all $i \in N$. We use a rank-one approximation of $Y^{\text {out }}$ for the sake of finite convergence, see Lemma 3.15 below.

The best rank-one approximation of $Y^{\text {out }}$ is given by $\lambda_{\max } w w^{\top}$, where $\lambda_{\max }$ and $w \in \mathbb{R}^{m+1}$ are the corresponding Perron-Frobenius eigenvalue and eigenvector, respectively. Let $w_{0}$ denote the zeroth entry of $w$ and let $\bar{w} \in \mathbb{R}^{m}$ be the vector obtained by excluding $w_{0}$ from $w$. It follows from the Perron-Frobenius theorem that $w$ can be chosen such that it has nonnegative entries. Since the vectors $\left[-1 u_{i}^{\top}\right]^{\top}$ and $\left[-1 v_{i}^{\top}\right]^{\top}$ are eigenvectors of $Y^{\text {out }}$ associated with the eigenvalue zero, see Lemma 3.6, it follows that

$$
u_{i}^{\top} \bar{w}=w_{0} \quad \text { and } \quad v_{i}^{\top} \bar{w}=w_{0} \quad \text { for all } i \in N .
$$

Suppose that $w_{0}=0$. Then $u_{i}^{\top} \bar{w}=v_{i}^{\top} \bar{w}=0$ for all $i \in N$, which implies that $w$ only contains zeros. Since this contradicts with the fact that $\|w\|>0$, we have $w_{0}>0$.

Now, let $r \in \mathbb{R}^{m}$ be defined as $r:=\frac{1}{w_{0}} \bar{w}$. Since $u_{i}^{\top} r=v_{i}^{\top} r=1$ for all $i \in N$ and $r \geq \mathbf{0}$, we conclude that $r$ is contained in the directed 2 -factor polytope. Hence, we can view $\left\{r_{e} \cdot r_{f}\right\}_{(e, f) \in \delta^{-}(i) \times \delta^{+}(i)}$ as a probability distribution on the pairs of successive arcs for all $i \in N$.

The oversampling algorithm, see Algorithm 3.4, iteratively draws a pair of successive arcs $(e, f)$ around $i \in N$ according to the distribution implied by $r$ and adds this pair to $H$. We repeat this until $H$ contains a cycle cover. The best among possibly multiple cycle covers in $H$ is obtained by solving problem (3.31) with respect to $x^{\text {out }}$ restricted to the arcs in $H$.

```
Algorithm 3.4 Randomized oversampling for the QCCP
Input: \(G, Q, Y^{\text {out }}, x^{\text {out }}\)
    Obtain Perron-Frobenius eigenpair \(\left(w, \lambda_{\max }\right)\) of \(Y^{\text {out }}\) and let \(r=\frac{1}{w_{0}} \bar{w}\).
    Let \(H=(N, \emptyset)\) be the empty subgraph of \(G\).
    while \(H\) contains no directed 2 -factor do
        for \(i \in N\) do
            Select \((e, f)\) according to probability distribution \(\left\{r_{e} \cdot r_{f}\right\}_{(e, f) \in \delta^{-}(i) \times \delta+(i)}\). Set \(H \leftarrow H \cup\{e, f\}\)
        end for
    end while
    Solve (3.31) with respect to \(x^{\text {out }}\) restricted to \(H\), and compute the corresponding upper bound \(U B_{O S}\).
Output: \(U B_{O S}\)
```

We can prove the following result with respect to the termination of Algorithm 3.4.
Lemma 3.15. Algorithm 3.4 terminates in a finite number of steps with high probability.
Proof. Observe that a pair of successive arcs $(e, f) \in \delta^{-}(i) \times \delta^{+}(i)$ can be added to $H$ either by sampling $e$ and $f$ simultaneously in step 5 of Algorithm 3.4 or since both arcs are added to $H$ in combination with some other arc. In the former case, we say that the pair $(e, f)$ is drawn around $i$. For all $i \in N,(e, f) \in \delta^{-}(i) \times \delta^{+}(i)$ and $k \geq 0$, let $Y_{i,(e, f)}^{k}$ denote the following random variable:

$$
Y_{i,(e, f)}^{k}:= \begin{cases}0 & \text { if }(e, f) \text { is not drawn around } i \text { during the first } k \text { iterations, } \\ 1 & \text { otherwise }\end{cases}
$$

Observe that $Y_{i,(e, f)}^{k}$ is independent over $i$, as step 5 is performed independently over $N$. Since the probability that a pair $(e, f)$ is added to $H$ in a single iteration equals $r_{e} \cdot r_{f}$, we have

$$
\operatorname{Pr}\left(Y_{i,(e, f)}^{k}=1\right)=1-\operatorname{Pr}\left(Y_{i,(e, f)}^{k}=0\right)=1-\left(1-r_{e} \cdot r_{f}\right)^{k}
$$

Since $r \in \operatorname{conv}(P)$, there must be at least one cycle cover, say $\bar{x} \in P$, that has full support in $r$. We define the functions $p: N \rightarrow A$ and $s: N \rightarrow A$ that map each node $i$ to its incoming and outgoing arc in $\bar{x}$, respectively. We show that the probability that the support of $\bar{x}$ is in $H$ converges to 1 if $k$ increases. Since the probability that a pair of successive $\operatorname{arcs}(e, f) \in \delta^{-}(i) \times \delta^{+}(i)$ is present in $H$ after $k$ iterations is at least $\operatorname{Pr}\left(Y_{i,(e, f)}^{k}=1\right)$, we have:

$$
\begin{aligned}
\operatorname{Pr}(\operatorname{supp}(\bar{x}) \subseteq H \text { after } k \text { iterations }) & \geq \operatorname{Pr}\left(\prod_{i \in N} Y_{i,(p(i), s(i))}^{k}=1\right) \\
& =\prod_{i \in N}\left(1-\left(1-r_{p(i)} \cdot r_{s(i)}\right)^{k}\right) .
\end{aligned}
$$

Since there exists an $\xi>0$ such that $r_{p(i)} \cdot r_{s(i)}>\xi$ for all $i \in N$, we have

$$
\operatorname{Pr}(\operatorname{supp}(\bar{x}) \subseteq H \text { after } k \text { iterations }) \geq\left(1-(1-\xi)^{k}\right)^{n}
$$

Now, for any $q<1$, take $k^{*}=\left\lceil\frac{\log (1-\sqrt[n]{\bar{q}})}{\log (1-\xi)}\right\rceil$. Then $\operatorname{Pr}\left(\operatorname{supp}(\bar{x}) \subseteq H\right.$ after $k^{*}$ iterations $) \geq q$. Thus, the cycle cover $\bar{x}$ is included in $H$ after a finite number of iterations with arbitrary high probability.

### 3.6.4 Sequential Q-learning

The final rounding approach we propose is based on a distributed reinforcement learning (RL) technique, namely Q-learning [361]. Q-learning is a branch of machine learning in which artificial agents learn how to take actions in order to maximize an expected total reward. Recently, RL techniques have shown to be successful in deriving good feasible solutions for combinatorial optimization problems, see e.g., [33]. We propose here an algorithm in which a set of agents learn how to find (near-)optimal cycles in $G$ by exploiting our SDP relaxation. Our sequential Q-learning algorithm (SQ-algorithm) is inspired by the work of Gambardella and Dorigo [158] and exploits the solution of the CP-ALM within the learning process.

In the sequential Q-learning algorithm we introduce $n$ agents each having the independent task to construct a set of node-disjoint cycles. This is done iteratively by adding nodes to the agent's current path until the path contains a directed cycle or no more nodes can be added. For each agent $k \in[n]$, let $P_{k}$ denote its current path and let $c_{k}$ and $p_{k}$ denote the current node and its predecessor on the agent's search, respectively. Besides, let $J_{k}$ be the set of nodes that is not placed on a cycle by agent $k$. In each iteration, the successor $s_{k}$ of $c_{k}$ is selected among one of the nodes in $N^{+}\left(c_{k}\right) \cap J_{k}$, where $N^{+}\left(c_{k}\right)$ is the set of nodes reachable from $c_{k}$ via a single arc, based on a matrix $S Q \in \mathbb{R}^{m \times m}$. This matrix indicates on position $(e, f)$ how useful it is to traverse an arc $f$ after an arc $e$. We select the successor $s_{k}$ that leads to a high $S Q\left(\left(p_{k}, c_{k}\right),\left(c_{k}, s_{k}\right)\right)$-value and add it to $P_{k}$. If the addition of $s_{k}$ to $P_{k}$ does not result in a cycle, we set the current node $c_{k}$ to be $s_{k}$. If the addition of $s_{k}$ does lead to a cycle $C_{k} \subseteq P_{k}$, we memorize this cycle into the agent's partial solution vector $y_{k} \in\{0,1\}^{m}$ and set $c_{k}$ to one of the nodes not yet on a cycle. An agent's search terminates whenever no new successor can be found, i.e., $N^{+}\left(c_{k}\right) \cap J_{k}=\emptyset$, or when $y_{k}$ is a full cycle cover. If one of these events occurs, we deacivitate the agent. We repeat the steps above for all active agents, until all agents have been deactivated. The decision tree of a single agent in the cycle-building phase is depicted in Figure 3.3, where the state $\Delta$ represents the situation in which an agent is deactivated.

When all agents are in state $\Delta$, we obtained $n$ vectors $y_{k}$ that represent sets of nodedisjoint cycles, not necessarily full cycle covers. At the end of the cycle-building phase, the (partial) solution $y_{k}$ that has relative minimum cost is used to update the $S Q$-matrix via delayed reinforcement learning. Now all agents are again activited and a new cycle-building trial starts using the new $S Q$-matrix until certain stopping criteria are satisfied, e.g., after a fixed number of trials.

To decide which successor $s_{k}$ to select for a given $p_{k}, c_{k}$ and $J_{k}$, we define a fit function $f$ that depends on the $S Q$-values and the quadratic costs $Q=\left(q_{e f}\right)$. The fit of visit$\operatorname{ing} u \in N^{+}\left(c_{k}\right) \cap J_{k}$ after $c_{k}$ is:

$$
f\left(u \mid p_{k}, c_{k}, J_{k}\right):= \begin{cases}{\left[\sum_{e \in \delta^{+}\left(J_{k}, c_{k}\right)} S Q\left(e,\left(c_{k}, u\right)\right)\right]^{\delta} \cdot\left[\sum_{e \in \delta^{+}\left(J_{k}, c_{k}\right)} \frac{1}{q_{e,\left(c_{k}, u\right)}+\epsilon}\right]^{\beta}} & \text { if } p_{k}=\emptyset \\ {\left[S Q\left(\left(p_{k}, c_{k}\right),\left(c_{k}, u\right)\right)\right]^{\delta} \cdot\left[\frac{1}{q_{\left(p_{k}, c_{k}\right),\left(c_{k}, u\right)+\epsilon}}\right]^{\beta}} & \text { otherwise }\end{cases}
$$

where $\delta, \beta>0$ are parameters which represent the relative importance between the learned


Figure 3.3: Schematic decision tree corresponding to cycle-building phase of a single agent.
$S Q$-values and the quadratic costs and $\epsilon>0$ is a small value to deal with quadratic costs that are zero. After computing the fit for all potential successors, we deterministically select the one with the highest fit value or select randomly proportional to their fit values, i.e.,

$$
s_{k}= \begin{cases}\underset{u \in N^{+}\left(c_{k}\right) \cap J_{k}}{\arg \max } f\left(u \mid p_{k}, c_{k}, J_{k}\right) & \text { if } q \leq q_{0}  \tag{3.35}\\ S & \text { otherwise }\end{cases}
$$

where $S$ is a random variable over the set $N^{+}\left(c_{k}\right) \cap J_{k}$, where each node is chosen with probability proportional to its fit value. The parameter $q_{0} \in[0,1]$ from (3.35) is the probability of selecting the successor node deterministically.

The $S Q$-values measure the usefulness of traversing two successive arcs. Recall that $Y^{\text {out }}$ is the output of the CP-ALM. As $Y_{e f}^{o u t}$ is likely to be larger when two arcs $e$ and $f$ are in an optimal solution, we initialize the $S Q$-matrix by setting $S Q(e, f)=Y_{e f}^{o u t}$ for all pairs of successive arcs $(e, f)$. The $S Q$-update is based on a mixture between local memory and a reinforcement learning, similar to [158]:

$$
\begin{equation*}
S Q(e, f) \leftarrow(1-\alpha) S Q(e, f)+\alpha\left(\Delta S Q(e, f)+\gamma \max _{g \in \delta^{+}\left(f^{-}, J_{k}\right)} S Q(f, g)\right) \tag{3.36}
\end{equation*}
$$

where $\alpha, \gamma \in(0,1)$ represent the learning rate and discount factor, respectively. The learning update consists of a discounted reward of the next state and a reinforcement term $\Delta S Q(e, f)$. Similar to the algorithm of [158], we assume that this reinforcement term is zero throughout the cycle-building phase and update it only at the end of a trial. Hence, we only incur a delayed reinforcement term $\Delta S Q(e, f)$. The discounted reward, however, is incorporated during the cycle-building phase.

The delayed reinforcement of a pair of successive arcs $(e, f)$ can be seen as a reward for cost minimal cycles that is obtained at the end of each trial. After all agents are deactivated, each vector $y_{k}$ is the characteristic vector of a set of node-disjoint cycles. For each agent $k$ that constructed at least one cycle, we compute $L_{k}:=\left(y_{k}^{\top} Q y_{k}\right) /\left(\mathbf{1}^{\top} y_{k}\right)$, i.e., the relative cost per arc in $y_{k}$. Let $k_{\text {best }}$ denote the agent that constructed the solution with the smallest value $L_{k}$, and let $L_{b e s t}$ denote its relative cost per arc. Then $\Delta S Q(e, f)$ is computed as:

$$
\Delta S Q(e, f)= \begin{cases}\frac{\Omega}{L_{\text {best }}} & \text { if }(e, f) \text { is a pair of successive arcs in } y_{k_{b e s t}},  \tag{3.37}\\ 0 & \text { otherwise },\end{cases}
$$

where $\Omega$ is a constant.

We let the SQ-algorithm run until some fixed number of trials has passed. All cycles that have been constructed throughout the entire algorithm are stored in memory. Let $\Gamma$ denote the number of distinct cycles that are constructed and define the matrix $B \in \mathbb{R}^{n \times \Gamma}$ as follows:

$$
B_{i, k}= \begin{cases}1 & \text { if node } i \text { is on cycle } k \\ 0 & \text { otherwise }\end{cases}
$$

Let $b \in \mathbb{R}^{\Gamma}$ be the vector containing the quadratic cost of each cycle. Then the best upper bound based on our SQ-algorithm is obtained by solving the following set partitioning problem (SPP):

$$
\begin{equation*}
\min \left\{b^{\top} x: B x=\mathbf{1}_{n}, x \in\{0,1\}^{\Gamma}\right\} . \tag{3.38}
\end{equation*}
$$

As the SPP is $\mathcal{N} \mathcal{P}$-hard, computing an optimal solution to (3.38) might be too much to ask for. Instead, an approximate solution to (3.38) can be obtained efficiently, e.g., by using the Lagrangian heuristic of Atamtürk et al. [25] which is able to compute near-optimal or even optimal solutions to (3.38) most of the time. For moderate values of $\Gamma$ and $n$, however, current ILP solvers are able to solve (3.38) to optimality in a very short time.

A pseudocode of the SQ-algorithm is provided in Algorithm 3.5.

### 3.6.5 Hybrid upper bounding algorithm

The design of the SQ-algorithm discussed in the previous section gives rise to a straightforward hybrid implementation of all above-mentioned upper bounding approaches. Indeed, by adding all cycles that have been created by the best Euclidean approximation, the undersampling and the oversampling algorithm to the matrix $B$ and solve or approximate the corresponding SPP, a hybrid upper bound $U B_{H Y}$ is obtained which provably outperforms any independent implementation of the mentioned upper bounds.

### 3.7 Computational results

We now test the introduced SDP-based lower and upper bounds on several sets of instances and compare them to various bounds from the literature.

This section is organized as follows: we start by introducing the test sets and the parameter settings that we consider. After that, the performance of the lower and upper bounds are discussed in Section 3.7.2 and 3.7.3, respectively.

### 3.7.1 Design of computational experiments

The SDP bounds that we take into account are $\left(S D P_{S 2}\right)$ and $\left(S D P_{S 3}\right)$, which we obtain via the PRSM and the CP-ALM, respectively. The CP-ALM is implemented as presented in Algorithm 3.2, i.e., using the PRSM and Dykstra's semi-parallel projection algorithm in the subproblem. We present results for different number of added cuts. To compare our SDP bounds, we use the first level RLT bound (RLT1), see Adams and Sherali [3, 4], the MILPbased bound (MILP) and the linearization based bound (LBB1) from [274], see Chapter 2. Since upper bounds for the QCCP are never considered before, we present and compare the upper bounds introduced in Section 3.6.

```
Algorithm 3.5 Sequential Q-learning for QCCP
nput: \(G, Q, Y^{\text {out }}\)
    For all pairs of successive \(\operatorname{arcs}(e, f)\), initialize \(S Q(e, f)=Y_{e f}^{o u t}\).
    For all agents \(k \in[n]\), initialize the starting node \(c_{k}=k \in N\), starting edge \(e_{k}=\emptyset\) and \(J_{k}=N\). Set
    \(P_{k}=\left\{c_{k}\right\}\) and \(y_{k}=\mathbf{0}_{m}\) and activate all agents.
    while there is at least one active agent do
        for all active agents \(k\) do
            Update the fit function \(f\left(u \mid p_{k}, c_{k}, J_{k}\right)\) for all \(u \in N^{+}\left(c_{k}\right) \cap J_{k}\) and obtain \(s_{k}\) by (3.35).
            Add \(s_{k}\) to \(P_{k}\).
        end for
        for all active agents \(k\) with \(p_{k} \neq \emptyset\) do
            \(S Q\left(\left(p_{k}, c_{k}\right),\left(c_{k}, s_{k}\right)\right) \leftarrow(1-\alpha) S Q\left(\left(p_{k}, c_{k}\right),\left(c_{k}, s_{k}\right)\right)+\alpha \gamma \max _{e \in \delta^{+}\left(s_{k}, J_{k}\right)} S Q\left(\left(c_{k}, s_{k}\right), e\right)\).
        end for
        for all active agents \(k\) do
            if \(P_{k}\) contains a cycle \(C_{k}\) then
                Set \(J_{k} \leftarrow J_{k} \backslash C_{k}\).
                Set \(\left(y_{k}\right)_{e}=1\) for all \(\operatorname{arcs} e\) in \(C_{k}\).
                if \(J_{k}=\emptyset\) then
                    Deactivate agent \(k\).
                else
                    Set \(p_{k} \leftarrow \emptyset\) and choose \(c_{k}\) uniformly at random out of \(J_{k}\). Set \(P_{k} \leftarrow\left\{c_{k}\right\}\).
                end if
            else
                    Set \(p_{k} \leftarrow c_{k}\) and \(c_{k} \leftarrow s_{k}\).
                if \(N^{+}\left(c_{k}\right) \cap J_{k}=\emptyset\) then Deactivate agent \(k\) end if
            end if
        end for
    end while
    for all pairs of successive \(\operatorname{arcs}(e, f)\) do
        Compute the delayed reinforcement \(\Delta S Q(e, f)\) according to (3.37).
        \(S Q(e, f) \leftarrow(1-\alpha) S Q(e, f)+\alpha \Delta S Q(e, f)\)
    end for
    If stopping criteria are met, obtain \(U B_{S Q}\) using (3.38). Otherwise, go to Step 2.
Output: \(U B_{S Q}\)
```

All lower and upper bounds are implemented in Matlab on a PC with an $\operatorname{Intel}(\mathrm{R})$ Core(TM) i7-8700 CPU, 3.20 GHz and 8 GB RAM. The linear programming problems appearing in our approaches and in the computation of MILP, LBB1 and RLT1 are solved using CPLEX 12.7.1. As all costs are integer, we round up all bounds. All computation times reported in this section concern wall-clock times.

We test our bounds on three sets of instances:

- Reload instances: The reload instances are the same as the ones used in Rostami et al. [326] for the QTSP and are based on a similar setting from Fischer et al. [135]. The underlying graph is the complete directed graph on $n$ nodes. The quadratic costs are based on the reload model [370], where each arc is randomly assigned a color from a color set $L$. The quadratic costs between two successive arcs with the same color is zero. If successive arcs $e$ and $f$ are assigned distinct colors $s$ and $t$, respectively, the costs equal $r(s, t)$, where $r: L \times L \rightarrow\{1, \ldots, D\}$ is a reload cost function. The function $r$ is constructed uniformly at random. We consider 60 instances with $n \in\{10,15,20\}, D \in\{1,10\}$ and $|L|=20$. Preliminary experiments have shown that the bound ( $S D P_{S 2}$ ) is very often equal to the optimum value (see also Table 3.11). For that reason, we do not consider the addition of cutting planes for these instances.
- Erdős-Rényi instances: These instances are based on the $G(n, p)$ model by Erdős
and Rényi [125]. A graph is constructed by fixing $n$ nodes and including each arc independently with probability $p$. We present two types of cost structures on these instances:
- Uniform Erdös-Rényi instances: the quadratic cost between any pair of successive arcs is chosen discrete uniformly at random from $\{0, \ldots, 100\}$;
- Reload Erdős-Rényi instances: the quadratic cost between any pair of successive arcs is based on a reload cost model using 20 colors and reload costs drawn uniformly from $\{1, \ldots, 100\}$.

We consider 15 instances of each type for $n$ between 20 and 80 and $p$ between 0.3 and 0.5.

- Manhattan instances: Comellas et al. [80] introduced multidimensional directed grid instances that resemble the street pattern of cities like New York and Barcelona. Given a set of positive integers $\left(n_{1}, \ldots, n_{k}\right)$, the Manhattan instances are constructed as explained in [274], see also Section 2.8. The quadratic costs between any pair of successive arcs is chosen discrete uniformly at random out of $\{0, \ldots, 10\}$. We consider a set of 32 Manhattan instances ranging from type $(5,5)$ to type $(9,10,10)$.

For the computation of the bounds we need to specify various parameters. The PRSM is implemented using $\beta=\lceil m / n\rceil, \gamma_{1}=0.9$ and $\gamma_{2}=1.09$, see (3.25)-(3.28), as preliminary experiments show that this setting gives the most stable performance. The CP-ALM uses the same PRSM parameters in the subproblem, where $K=5$ is used in the semi-parallel implementation of (CycDyk), see the third bullet point on page 80. The stopping criteria of the PRSM and the CP-ALM are as explained in Section 3.5.5.1, where we use $\varepsilon_{\text {stag }}=10^{-5}$ and $\varepsilon_{\text {proj }}=10^{-8}$. The parameter $\varepsilon_{P R S M}$ is initially set to $10^{-6}$, but after the addition of cuts increased to $10^{-4}$, since solving the $Y$-subproblem using Dykstra's algorithm is significantly slower than the initial $Y$-subproblem without cuts. Hence, we allow for a lower precision. For the same reason, the maximum number of iterations of the inner while-loop (i.e., maxIter, see Section 3.5.5.1) of the CP-ALM is initialized to some value and decreased when the first cuts are added, after which we do not change it anymore. For the Erdős-Rényi and Manhattan instances, we initialize maxIter to 1000 and 1500, respectively, and decrease it to 500 after the addition of cuts. The initial iteration limit for the Manhattan instances is larger, as the CP-ALM needs more iterations to converge for these type of instances.

It turns out that the number of cuts added per main loop, i.e., the value of numCuts, see Section 3.5.5, is of major importance for the quality of the final bound. To demonstrate this behaviour, the lower bounds against the iteration number for a moderate-size ErdősRényi instance (ER_4 with $n=35$ and $m=361$ ) is plotted in Figure 3.4 for various values of numCuts using an iteration limit of 2500 . The base line shows the behaviour of the PRSM, i.e., the CP-ALM without the addition of cuts. It is clear that the addition of cuts after 1000 iterations immediately starts improving the bounds. Moreover, as one might expect, the addition of more cuts leads to a higher lower bound, although the largest improvement is due to the addition of the first few cuts. As the addition of more cuts also leads to higher computation times, a trade-off between quality and time has to be made. Based on preliminary experiments, we report results for numCuts $=50,150,300$ and 500 for the ErdősRényi instances. For the Manhattan instances, we only show results for numCuts $=300$ and 500 , as the addition of a small number of cuts does not significantly improve the bounds.


Figure 3.4: Behaviour of the PRSM and the CP-ALM for different values of numCuts for instance ER_4.

Finally, we need to specify the maximum total number of iterations maxTotalIter, see Section 3.5.5.1. For the reload instances we set this value to 2500 iterations, although the algorithm in most cases terminates earlier for these instances due to the other stopping criteria. The value of maxTotalIter for the other two instance types is based on preliminary tests. Similar to the PRSM and the ADMM, the CP-ALM can suffer from tailing off. Since the addition of more cuts makes later iterations more expensive, one has to decide carefully when to stop. This threshold mainly depends on the value of $m$. Figure 3.5 shows the behaviour of the lower bounds averaged over numCuts $=150,300$ and 500 on three instances: a small, a moderate-size and a large instance. We normalize the bounds in order to make them comparable, i.e., the plots show the fraction of the final lower bound that is obtained after each iteration. Although at first sight there seems not much difference, one can see from the zoomed image on the right-hand side that the CP-ALM converges relatively faster for smaller instances.


Figure 3.5: Normalized lower bounds (averaged over different number of cuts) for three instances with different numbers of arcs. Right figure shows zoomed plot including a threshold at 0.996 .

Based on these preliminary results, the parameter maxTotalIter is set to 2500,3000
or 3500 if $m<500,500 \leq m<1000$ and $m \geq 1000$, respectively, for the Erdős-Rényi instances. For the Manhattan instances these values are 3000, 3500 and 4000, respectively, using the same distinction on $m$.

For the computation of upper bounds, we compute the randomized undersampling and oversampling bounds 500 times for each instance and return the best value. For the SQalgorithm, we use different parameter settings for each instance type based on preliminary tests. It turns out that the algorithm performs best if the value of $\delta$ is significantly larger than the value of $\beta$, see page 86 , i.e., we put more emphasis on the SDP-based SQ-values than on the original quadratic costs. This difference seems more beneficial for larger $n$, since more agents provide more reliable information on useful cycles. Hence, we use $(\delta, \beta)=(20,1)$ for the Erdős-Rényi and Manhattan instances, while we use $(\delta, \beta)=(5,1)$ for the reload instances. Furthermore, we use $q_{0}=0.4, \gamma=0.6$ and $\Omega=3(m / n)$, see (3.35), (3.36) and (3.37), respectively, for all instance types. Finally, as it is not clear from our tests which value of the learning rate parameter $\alpha$, see (3.36), provides the best results, we run the SQ-algorithm three times using $\alpha=0.3,0.5$ and 0.7 and solve the final SPP, see (3.38), using all generated cycles. The number of iterations of the SQ-algorithm is set to 500 for the Erdős-Rényi and the reload instances, while it is set to 100 for the Manhattan instances, due to the large number of nodes. As the final SPPs can be solved efficiently by CPLEX for all our instances, we report the optimal SPP bounds.

### 3.7.2 Results on lower bounds

With respect to the efficiency of our algorithm, we observe that if we run the CP-ALM without any cutting planes, the bottleneck of the code is the projection onto the positive semidefinite cone. This is not surprising, as the computational complexity of performing an (approximate) eigenvalue decomposition is worse than that of the cheap polyhedral projection, whose bottleneck is the projection onto a simplex. When we start adding cuts, Dykstra's algorithm starts taking over the major part of the computation time. Although each separate projection can be done efficiently, the total computation time of Dykstra's algorithm suffers from its linear convergence and the dependence on the number of cutting planes.

We now discuss our findings with respect to the strength of the lower bounds on all test instances. For the reload instances we compare the performance of $\left(S D P_{S 2}\right)$ to the performance of $(M I L P),(L B B 1)$ and $(R L T 1)$. We omit the brackets from now on to indicate the bound values. Table 3.3 shows for each of the 60 reload instances the bound value resulting from each of the approaches. Table 3.4 shows all computation times for the reload instances, including the number of iterations and the average of the primal and dual residual, see Section 3.5.5.1, for the PRSM. To visualize the quality of the bounds over the entire reload test set, Figure 3.6 shows a boxplot of the test data in Table 3.3. On the $y$-axis the deviation from the average bound is presented, i.e., for each instance we compute the ratio of each single bound over the average value of the four bounds and these ratios are visualized per bound type.

| Instance | $n$ | D | MILP | $L B B 1$ | RLT 1 | $S D P_{S 2}$ | Instance | $n$ | D | MILP | $L B B 1$ | RLTT1 | $S D P_{S 2}$ |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| REL1 | 10 | 1 | 3 | 4 | 4 | 4 | REL31 | 15 | 1 | 1 | 4 | 4 | 5 |
| REL2 |  | 10 | 3 | 9 | 9 | 9 | REL32 |  | 10 | 1 | 7 | 8 | 11 |
| REL3 |  | 1 | 3 | 4 | 5 | 5 | REL33 |  | 1 | 1 | 4 | 4 | 4 |
| REL4 |  | 10 | 3 | 8 | 9 | 12 | REL34 |  | 10 | 1 | 5 | 5 | 8 |
| REL5 |  | 1 | 3 | 4 | 4 | 4 | REL35 |  | 1 | 1 | 4 | 4 | 4 |
| REL6 |  | 10 | 5 | 12 | 13 | 14 | REL36 |  | 10 | 1 | 4 | 5 | 8 |
| REL7 |  | 1 | 3 | 4 | 5 | 5 | REL37 |  | 1 | 2 | 6 | 6 | 6 |
| REL8 |  | 10 | 4 | 9 | 11 | 11 | REL38 |  | 10 | 1 | 9 | 9 | 11 |
| REL9 |  | 1 | 2 | 2 | 2 | 2 | REL39 |  | 1 | 1 | 3 | 4 | 3 |
| REL10 |  | 10 | 4 | 9 | 11 | 12 | REL40 |  | 10 | 1 | 6 | 7 | 7 |
| REL11 |  | 1 | 2 | 3 | 3 | 3 | REL41 | 20 | 1 | 0 | 3 | 3 | 4 |
| REL12 |  | 10 | 5 | 9 | 9 | 9 | REL42 |  | 10 | 1 | 4 | 4 | 7 |
| REL13 |  | 1 | 2 | 4 | 4 | 4 | REL43 |  | 1 | 0 | 2 | 2 | 3 |
| REL14 |  | 10 | 3 | 9 | 11 | 11 | REL44 |  | 10 | 0 | 5 | 5 | 7 |
| REL15 |  | 1 | 3 | 4 | 4 | 4 | REL45 |  | 1 | 0 | 2 | 2 | 3 |
| REL16 |  | 10 | 3 | 8 | 9 | 11 | REL46 |  | 10 | 0 | 5 | 5 | 6 |
| REL17 |  | 1 | 4 | 4 | 4 | 4 | REL47 |  | 1 | 0 | 2 | 2 | 3 |
| REL18 |  | 10 | 3 | 8 | 9 | 10 | REL48 |  | 10 | 0 | 3 | 3 | 5 |
| REL19 |  | 1 | 3 | 5 | 5 | 5 | REL49 |  | 1 | 0 | 3 | 3 | 4 |
| REL20 |  | 10 | 3 | 10 | 11 | 11 | REL50 |  | 10 | 0 | 5 | 6 | 8 |
| REL21 | 15 | 1 | 2 | 4 | 4 | 5 | REL51 |  | 1 | 0 | 3 | 3 | 3 |
| REL22 |  | 10 | 2 | 9 | 9 | 12 | REL52 |  | 10 | 0 | 3 | 4 | 6 |
| REL23 |  | 1 | 1 | 3 | 3 | 4 | REL53 |  | 1 | 0 | 3 | 3 | 4 |
| REL24 |  | 10 | 1 | 7 | 8 | 11 | REL54 |  | 10 | 0 | 6 | 6 | 9 |
| REL25 |  | 1 | 1 | 4 | 5 | 5 | REL55 |  | 1 | 0 | 2 | 2 | 3 |
| REL26 |  | 10 | 1 | 6 | 6 | 9 | REL56 |  | 10 | 1 | 6 | 6 | 8 |
| REL27 |  | 1 | 1 | 4 | 4 | 4 | REL57 |  | 1 | 0 | 3 | 3 | 4 |
| REL28 |  | 10 | 1 | 7 | 7 | 9 | REL58 |  | 10 | 0 | 3 | 4 | 7 |
| REL29 |  | 1 | 1 | 5 | 5 | 6 | REL59 |  | 1 | 0 | 2 | 2 | 3 |
| REL30 |  | 10 | 0 | 6 | 7 | 10 | REL60 |  | 10 | 0 | 5 | 5 | 8 |

Table 3.3: Comparison of different bounds for reload instances.


Figure 3.6: Boxplot showing the quality of lower bounds $M I L P, L B B 1, R L T 1$ and $S D P_{S 2}$ on reload instances.

| Instance | $\frac{M I L P}{\text { time }}$ | $\frac{L B B 1}{\text { time }}$ | $\frac{R L T 1}{\text { time }}$ | $S D P_{S 2}$ |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | time | iter | res |
| REL1 | 0.112 | 0.007 | 1.004 | 0.278 | 437 | 0.003 |
| REL2 | 0.113 | 0.007 | 0.398 | 0.215 | 300 | 0.032 |
| REL3 | 0.111 | 0.006 | 0.350 | 0.751 | 1192 | 0.005 |
| REL4 | 0.106 | 0.006 | 0.380 | 0.624 | 978 | $<0.001$ |
| REL5 | 0.107 | 0.006 | 0.321 | 0.332 | 529 | 0.001 |
| REL6 | 0.106 | 0.006 | 0.358 | 0.230 | 377 | 0.045 |
| REL7 | 0.108 | 0.006 | 0.364 | 0.665 | 1044 | 0.005 |
| REL8 | 0.107 | 0.006 | 0.366 | 0.182 | 294 | 0.086 |
| REL9 | 0.107 | 0.006 | 0.217 | 0.370 | 594 | 0.042 |
| REL10 | 0.106 | 0.006 | 0.377 | 0.430 | 681 | 0.053 |
| REL11 | 0.108 | 0.006 | 0.320 | 0.358 | 586 | 0.027 |
| REL12 | 0.108 | 0.006 | 0.334 | 0.147 | 232 | 0.078 |
| REL13 | 0.108 | 0.005 | 0.384 | 0.273 | 441 | 0.004 |
| REL14 | 0.106 | 0.006 | 0.380 | 0.287 | 327 | 0.038 |
| REL15 | 0.108 | 0.005 | 0.284 | 0.396 | 589 | 0.039 |
| REL16 | 0.107 | 0.006 | 0.374 | 0.494 | 790 | 0.048 |
| REL17 | 0.109 | 0.005 | 0.218 | 0.425 | 695 | 0.040 |
| REL18 | 0.107 | 0.006 | 0.393 | 0.442 | 695 | 0.041 |
| REL19 | 0.108 | 0.005 | 0.425 | 0.659 | 1034 | 0.005 |
| REL20 | 0.106 | 0.006 | 0.408 | 0.170 | 278 | 0.170 |
| REL21 | 0.421 | 0.050 | 9.542 | 3.146 | 1371 | 0.007 |
| REL22 | 0.412 | 0.048 | 8.475 | 3.950 | 1684 | <0.001 |
| REL23 | 0.415 | 0.044 | 8.617 | 3.066 | 1277 | 0.006 |
| REL24 | 0.411 | 0.048 | 8.693 | 2.695 | 1124 | 0.001 |
| REL25 | 0.415 | 0.044 | 8.565 | 5.069 | 2149 | 0.006 |
| REL26 | 0.413 | 0.046 | 8.697 | 2.701 | 1117 | $<0.001$ |
| REL27 | 0.414 | 0.044 | 9.208 | 6.097 | 2500 | 0.012 |
| REL28 | 0.411 | 0.051 | 8.229 | 1.667 | 689 | 0.017 |
| REL29 | 0.414 | 0.045 | 8.249 | 1.972 | 818 | 0.006 |
| REL30 | 0.421 | 0.050 | 8.676 | 3.979 | 1626 | <0.001 |
| REL31 | 0.415 | 0.043 | 9.720 | 3.044 | 1279 | 0.006 |
| REL32 | 0.412 | 0.047 | 8.450 | 3.002 | 1183 | 0.005 |
| REL33 | 0.415 | 0.044 | 8.626 | 5.998 | 2500 | 0.022 |
| REL34 | 0.410 | 0.046 | 9.133 | 0.913 | 380 | 0.040 |
| REL35 | 0.419 | 0.048 | 8.277 | 5.998 | 2500 | 0.017 |
| REL36 | 0.413 | 0.048 | 8.711 | 1.925 | 775 | 0.010 |
| REL37 | 0.419 | 0.045 | 8.143 | 3.439 | 1417 | 0.006 |
| REL38 | 0.412 | 0.048 | 8.004 | 6.119 | 2500 | 0.032 |
| REL39 | 0.414 | 0.045 | 7.099 | 5.864 | 2433 | 0.028 |
| REL40 | 0.415 | 0.044 | 8.357 | 3.131 | 1291 | 0.066 |
| REL41 | 1.062 | 0.137 | 120.2 | 28.54 | 2500 | 0.008 |
| REL42 | 1.088 | 0.133 | 142.1 | 17.93 | 1501 | 0.001 |
| REL43 | 1.075 | 0.145 | 127.2 | 29.58 | 2500 | 0.007 |
| REL44 | 1.083 | 0.137 | 120.2 | 24.44 | 1990 | 0.001 |
| REL45 | 1.067 | 0.127 | 105.8 | 21.15 | 1781 | 0.007 |
| REL46 | 1.095 | 0.139 | 118.4 | 6.955 | 570 | 0.035 |
| REL47 | 1.067 | 0.155 | 158.5 | 19.05 | 1626 | 0.007 |
| REL48 | 1.091 | 0.138 | 184.2 | 4.340 | 357 | 0.059 |
| REL49 | 1.073 | 0.132 | 143.0 | 24.12 | 2067 | 0.007 |
| REL50 | 1.091 | 0.133 | 118.1 | 4.428 | 355 | 0.061 |
| REL51 | 1.067 | 0.136 | 97.63 | 19.34 | 1651 | 0.007 |
| REL52 | 1.093 | 0.130 | 162.1 | 23.48 | 1907 | 0.001 |
| REL53 | 1.076 | 0.138 | 128.9 | 29.56 | 2500 | 0.007 |
| REL54 | 1.080 | 0.129 | 107.1 | 21.26 | 1725 | 0.001 |
| REL55 | 1.064 | 0.146 | 128.8 | 30.50 | 2500 | 0.009 |
| REL56 | 1.086 | 0.132 | 127.0 | 19.09 | 1548 | 0.001 |
| REL57 | 1.071 | 0.152 | 122.1 | 19.68 | 1578 | 0.007 |
| REL58 | 1.095 | 0.136 | 145.5 | 5.723 | 430 | 0.041 |
| REL59 | 1.069 | 0.136 | 119.0 | 17.12 | 1439 | 0.007 |
| REL60 | 1.091 | 0.126 | 129.4 | 6.227 | 502 | 0.035 |

Table 3.4: Computation times in seconds, average residuals and number of iterations for reload instances.

It follows from Table 3.3 and Figure 3.6 that $S D P_{S 2}$ clearly provides the strongest bounds, followed by $R L T 1, L B B 1$ and finally by $M I L P$, which behaves poorly for most of the instances. In fact, the hierarchy $L B B 1 \leq R L T 1 \leq S D P_{S 2}$ can be proven easily and holds with strict inequality for the majority of the instances. It can be seen that $S D P_{S 2}$ performs generally about 1.5 times better than the average of the four bounds. The bound $S D P_{S 2}$ even turns out to be optimal for $88 \%$ of the instances.

When considering Table 3.4, it follows that although the computation times are larger than those of MILP and LBB1, the SDP-bound can be computed efficiently for most of the instances. The computation times are always within 30 seconds and for $75 \%$ of the instances within 10 seconds, while the computation time of RLT1 is above 90 seconds for $67 \%$ of the instances. Moreover, although the optimum can be computed for all tested reload instances, the computation time is in some cases as large as 2000 seconds. Hence, for the reload instances we conclude that $S D P_{S 2}$ can be favoured above other bounds in both quality and time.

Next, we consider the Erdős-Rényi instances. Table 3.5 shows the bound values for the Erdős-Rényi test set, among which the bounds $S D P_{S 2}$ and $S D P_{S 3}$ for various number of cuts. We do not consider the first level RLT bound, as it cannot be efficiently computed for the majority of the instances. The column $O P T$ reports the optimal solution if this solution could be computed in 3 hours and '-' otherwise. The computation times are reported in Table 3.6 and the average of primal and dual residual and the number of final cuts in the CP-ALM for the SDP bounds are reported in Table 3.7.

For the Erdős-Rényi instances we also see that $S D P_{S 2}$ significantly outperforms MILP and $L B B 1$ in terms of quality of the bound. Moreover, it is clear that we can successfully improve the bounds by adding cuts using the new CP-ALM. Except for the instances where $S D P_{S 2}$ is already optimal, we see that $S D P_{S 3}$ provides a strictly higher bound already after adding 50 cuts at a time. For most instances, this improvement of $S D P_{S 3}$ compared to $S D P_{S 2}$ is about $3 \%-6 \%$. Interestingly, this improvement seems to be independent of the problem size. As we already observed in Figure 3.4, we see that a higher value of numCuts leads to a higher lower bound. This higher value comes, however, at the cost of computation time as can be seen from Table 3.6. When taking both quality and efficiency into account, it seems beneficial to add only a small number of cuts, as this often leads to a significant increase of the bound at a relatively low computational cost. For instances up to $1000 \operatorname{arcs}$ the CP-ALM terminates often within 30 minutes, while SDP bounds for instances up to 1850 arcs (!) can be computed within 2 hours. Hence, the CP-ALM is able to provide strong lower bounds for very large-scale SDPs in a reasonable time span, whereas the interior-point method of Mosek [284] can solve ( $S D P_{S 2}$ ) for instances up to only 300 arcs without running out of memory.

Finally, we consider the performance of the lower bounds on the Manhattan instances, which can be found in Table 3.8 and 3.9. With respect to the quality of the bounds we can draw the same conclusions as before. Namely, the SDP bound $S D P_{S 3}$ performs best on all instances, followed by $S D P_{S 2}$. Since the optimal values for many of these instances can be computed, we moreover see that our SDP bounds are very close to optimal. Although we again see that the cuts can successfully improve the lower bounds, the relative improvement is smaller than for the Erdős-Rényi test set. An explanation can be found by looking at the residuals in Table 3.9b, which are significantly larger than the residuals for the first two types of instances. Apparently, the Manhattan instances need more iterations to converge, probably due to the inner structure of these instances. Stopping the CP-ALM when it has only partly converged, leads to weaker and less stable lower bounds. Namely, the reported
lower bound is obtained by a projection of the current dual matrix, and further experiments show that in particular the dual residual converges slowly. The residuals increase with the size of the instance. Hence, we expect that even better bounds for the Manhattan instances can be obtained by letting the CP-ALM run for more iterations. However, we conclude from the current tables that the SDP bounds for the Manhattan instances significantly outperform the bounds from the literature in a reasonable time span.

| Instance | $p$ | $n$ | $m$ | $O P T$ | MILP | $L B B 1$ | $S D P_{S 2}$ | $S D P_{S 3}$ |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  |  |  |  |  | $\begin{gathered} \text { numCuts } \\ 50 \end{gathered}$ | $\begin{gathered} \text { numCuts } \\ 150 \end{gathered}$ | $\begin{gathered} \text { numCuts } \\ 300 \end{gathered}$ | $\begin{gathered} \text { numCuts } \\ 500 \end{gathered}$ |
| ER1 | 0.3 | 20 | 119 | 319 | 165 | 260 | 319 | 319 | 319 | 319 | 319 |
| RER1 |  |  | 113 | 293 | 154 | 274 | 293 | 293 | 293 | 293 | 293 |
| ER2 |  | 25 | 177 | 386 | 167 | 305 | 386 | 386 | 386 | 386 | 386 |
| RER2 |  |  | 169 | 391 | 151 | 303 | 391 | 391 | 391 | 391 | 391 |
| ER3 |  | 30 | 284 | - | 122 | 230 | 287 | 292 | 294 | 295 | 296 |
| RER3 |  |  | 256 | 281 | 69 | 208 | 258 | 262 | 264 | 265 | 266 |
| ER4 |  | 35 | 361 | - | 138 | 273 | 328 | 331 | 333 | 335 | 336 |
| RER4 |  |  | 347 | - | 61 | 189 | 233 | 236 | 238 | 239 | 240 |
| ER5 |  | 40 | 468 | - | 131 | 265 | 318 | 321 | 322 | 323 | 324 |
| RER5 |  |  | 495 | - | 17 | 177 | 215 | 217 | 219 | 219 | 220 |
| ER6 |  | 45 | 592 | - | 138 | 287 | 330 | 333 | 336 | 337 | 338 |
| RER6 |  |  | 623 | - | 9 | 110 | 146 | 148 | 149 | 150 | 151 |
| ER7 |  | 50 | 754 | - | 130 | 267 | 313 | 316 | 318 | 319 | 319 |
| RER7 |  |  | 746 | - | 3 | 91 | 116 | 117 | 118 | 119 | 119 |
| ER8 |  | 60 | 1062 | - | 118 | 272 | 301 | 303 | 304 | 305 | 305 |
| RER8 |  |  | 995 | - | 1 | 74 | 93 | 94 | 95 | 95 | 95 |
| ER9 |  | 70 | 1481 | - | 123 | 255 | 286 | 287 | 288 | 289 | 289 |
| RER9 |  |  | 1512 | - | 0 | 99 | 131 | 132 | 132 | 133 | 133 |
| ER10 |  | 80 | 1842 | - | 122 | 263 | 291 | 292 | 293 | 293 | 293 |
| RER10 |  |  | 1859 | - | 0 | 33 | 52 | 53 | 53 | 53 | 54 |
| ER11 | 0.5 | 20 | 195 | 236 | 95 | 175 | 227 | 232 | 233 | 234 | 234 |
| RER11 |  |  | 194 | 172 | 34 | 136 | 172 | 172 | 172 | 172 | 172 |
| ER12 |  | 25 | 327 | - | 67 | 136 | 169 | 171 | 172 | 173 | 173 |
| RER12 |  |  | 308 | 99 | 7 | 57 | 84 | 85 | 86 | 87 | 87 |
| ER13 |  | 30 | 434 | - | 79 | 161 | 197 | 200 | 201 | 202 | 202 |
| RER13 |  |  | 435 | - | 9 | 106 | 139 | 141 | 142 | 143 | 143 |
| ER14 |  | 40 | 793 | - | 74 | 166 | 196 | 198 | 199 | 199 | 200 |
| RER14 |  |  | 770 | - | 0 | 50 | 72 | 73 | 74 | 73 | 74 |
| ER15 |  | 50 | 1197 | - | 77 | 165 | 188 | 189 | 190 | 191 | 191 |
| RER15 |  |  | 1235 | - | 0 | 18 | 35 | 36 | 37 | 37 | 37 |

Table 3.5: Comparison of different bounds for Erdős-Rényi instances.

| Instance | MILP | $L B B 1$ | $S D P_{S 2}$ | $S D P_{S 3}$ |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { numCuts } \\ 50 \end{gathered}$ | $\begin{gathered} \text { numCuts } \\ 150 \end{gathered}$ | $\begin{gathered} \text { numCuts } \\ 300 \end{gathered}$ | $\begin{gathered} \text { numCuts } \\ 500 \end{gathered}$ |
| ER1 | 0.201 | 0.016 | 0.390 | 0.330 | 0.470 | 0.400 | 0.330 |
| RER1 | 0.194 | 0.008 | 0.150 | 0.140 | 0.130 | 0.130 | 0.150 |
| ER2 | 0.333 | 0.019 | 2.193 | 1.700 | 1.790 | 1.660 | 1.710 |
| RER2 | 0.319 | 0.016 | 3.520 | 18.37 | 155.6 | 55.27 | 163.8 |
| ER3 | 0.827 | 0.068 | 11.89 | 35.15 | 128.6 | 333.2 | 924.8 |
| RER3 | 0.673 | 0.042 | 8.120 | 29.55 | 67.15 | 110.0 | 205.0 |
| ER4 | 1.151 | 0.106 | 28.17 | 54.61 | 86.61 | 137.7 | 237.3 |
| RER4 | 1.166 | 0.107 | 24.83 | 60.40 | 93.71 | 161.4 | 257.9 |
| ER5 | 1.914 | 0.139 | 48.57 | 86.64 | 121.2 | 230.8 | 526.3 |
| RER5 | 2.088 | 0.153 | 53.83 | 89.76 | 112.7 | 148.2 | 234.4 |
| ER6 | 2.856 | 0.201 | 99.59 | 203.4 | 253.4 | 350.3 | 499.2 |
| RER6 | 3.048 | 0.220 | 113.4 | 200.4 | 247.1 | 345.7 | 494.7 |
| ER7 | 4.489 | 0.327 | 168.4 | 291.3 | 356.6 | 607.0 | 1173 |
| RER7 | 4.207 | 0.297 | 164.6 | 285.2 | 358.1 | 447.9 | 601.8 |
| ER8 | 10.824 | 0.625 | 463.0 | 870.8 | 1000 | 1306 | 2198 |
| RER8 | 7.969 | 0.529 | 340.5 | 624.0 | 652.6 | 748.8 | 950.6 |
| ER9 | 24.184 | 1.346 | 1305 | 2160 | 2293 | 2838 | 4517 |
| RER9 | 25.232 | 1.420 | 1381 | 2308 | 2371 | 2555 | 2961 |
| ER10 | 42.034 | 2.273 | 2446 | 4110 | 4088 | 4548 | 7178 |
| RER10 | 41.74 | 2.305 | 2516 | 4035 | 4130 | 4451 | 4868 |
| ER11 | 0.397 | 0.031 | 6.071 | 22.95 | 67.97 | 178.4 | 512.1 |
| RER11 | 0.415 | 0.027 | 4.640 | 13.53 | 32.20 | 51.59 | 82.89 |
| ER12 | 0.967 | 0.107 | 17.99 | 37.11 | 70.64 | 200.9 | 524.0 |
| RER12 | 0.862 | 0.088 | 14.07 | 31.89 | 51.67 | 97.69 | 170.2 |
| ER13 | 1.554 | 0.155 | 42.23 | 77.36 | 108.5 | 196.2 | 472.9 |
| RER13 | 1.579 | 0.150 | 42.59 | 77.82 | 98.80 | 142.1 | 208.3 |
| ER14 | 4.539 | 0.431 | 201.7 | 348.8 | 493.6 | 868.5 | 2840 |
| RER14 | 4.283 | 0.389 | 187.8 | 319.2 | 373.9 | 457.3 | 627.8 |
| ER15 | 12.663 | 1.068 | 721.6 | 1243 | 1426 | 1854 | 3773 |
| RER15 | 12.99 | 1.141 | 795.2 | 1438 | 1427 | 1559 | 1775 |

Table 3.6: Comparison of computation times (in seconds) for Erdős-Rényi instances.

| Instance | $S D P_{S 2}$ <br> res | $S D P_{S 3}$ |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  | $\begin{gathered} \text { numCuts } \\ 50 \end{gathered}$ |  | $\begin{gathered} \text { numCuts } \\ 150 \end{gathered}$ |  | $\begin{gathered} \text { numCuts } \\ 300 \end{gathered}$ |  | $\begin{gathered} \text { numCuts } \\ 500 \end{gathered}$ |  |
|  |  | res | cuts | res | cuts | res | cuts | res | cuts |
| ER1 | $<0.001$ | <0.001 | 0 | <0.001 | 0 | $<0.001$ | 0 | $<0.001$ | 0 |
| RER1 | $<0.001$ | <0.001 | 0 | <0.001 | 0 | <0.001 | 0 | <0.001 | 0 |
| ER2 | 0.003 | 0.002 | 0 | 0.002 | 0 | 0.003 | 0 | 0.003 | 0 |
| RER2 | <0.001 | <0.001 | 100 | <0.001 | 294 | <0.001 | 303 | <0.001 | 532 |
| ER3 | 0.002 | 0.002 | 148 | 0.003 | 446 | 0.003 | 878 | 0.003 | 1455 |
| RER3 | 0.002 | 0.003 | 150 | 0.003 | 444 | 0.004 | 882 | 0.005 | 1461 |
| ER4 | 0.002 | 0.002 | 148 | 0.003 | 450 | 0.003 | 899 | 0.003 | 1494 |
| RER4 | 0.003 | 0.002 | 150 | 0.002 | 446 | 0.002 | 887 | 0.003 | 1479 |
| ER5 | 0.002 | 0.002 | 150 | 0.002 | 449 | 0.007 | 898 | 0.003 | 1494 |
| RER5 | 0.003 | 0.003 | 150 | 0.003 | 450 | 0.003 | 899 | 0.004 | 1497 |
| ER6 | 0.001 | 0.003 | 200 | 0.004 | 598 | 0.005 | 1195 | 0.004 | 1992 |
| RER6 | 0.003 | 0.003 | 200 | 0.003 | 598 | 0.003 | 1199 | 0.003 | 1995 |
| ER7 | 0.001 | 0.002 | 200 | 0.002 | 597 | 0.002 | 1195 | 0.008 | 1996 |
| RER7 | 0.002 | 0.002 | 199 | 0.002 | 598 | 0.002 | 1191 | 0.002 | 1987 |
| ER8 | 0.001 | 0.002 | 250 | 0.002 | 750 | 0.002 | 1500 | 0.002 | 2500 |
| RER8 | 0.002 | 0.002 | 200 | 0.002 | 599 | 0.003 | 1198 | 0.003 | 1998 |
| ER9 | 0.001 | 0.001 | 250 | 0.002 | 749 | 0.002 | 1497 | 0.002 | 2497 |
| RER9 | 0.003 | 0.003 | 248 | 0.007 | 745 | 0.003 | 1497 | 0.003 | 2500 |
| ER10 | 0.002 | 0.002 | 250 | 0.002 | 750 | 0.002 | 1500 | 0.003 | 2500 |
| RER10 | 0.003 | 0.003 | 250 | 0.003 | 750 | 0.003 | 1499 | 0.003 | 2500 |
| ER11 | 0.001 | 0.003 | 148 | 0.003 | 440 | 0.003 | 877 | 0.006 | 1458 |
| RER11 | 0.055 | <0.001 | 50 | <0.001 | 150 | <0.001 | 300 | <0.001 | 500 |
| ER12 | 0.001 | 0.002 | 150 | 0.002 | 450 | 0.003 | 898 | 0.003 | 1498 |
| RER12 | 0.002 | 0.002 | 133 | 0.003 | 415 | 0.003 | 851 | 0.003 | 1431 |
| ER13 | 0.001 | 0.002 | 149 | 0.002 | 445 | 0.002 | 892 | 0.002 | 1481 |
| RER13 | 0.003 | 0.003 | 150 | 0.003 | 448 | 0.003 | 891 | 0.004 | 1481 |
| ER14 | 0.001 | 0.001 | 200 | 0.002 | 597 | 0.002 | 1196 | 0.004 | 1992 |
| RER14 | 0.002 | 0.003 | 200 | 0.002 | 600 | 0.019 | 1200 | 0.003 | 2000 |
| ER15 | 0.001 | 0.001 | 250 | 0.001 | 750 | 0.002 | 1500 | 0.002 | 2500 |
| RER15 | 0.003 | 0.003 | 250 | 0.003 | 750 | 0.003 | 1500 | 0.003 | 2500 |

Table 3.7: Comparison of residuals and total number of cuts for Erdős-Rényi instances.

| Instance | Type | $n$ | m | OPT | MILP | $L B B 1$ | $S D P_{S 2}$ | $S D P_{S 3}$ |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  |  |  |  |  | $\begin{gathered} \text { numCuts } \\ 300 \end{gathered}$ | $\begin{gathered} \text { numCuts } \\ 500 \end{gathered}$ |
| MH1 | $(5,5)$ | 25 | 50 | 103 | 102 | 103 | 103 | 103 | 103 |
| MH2 | $(10,10)$ | 100 | 200 | 418 | 394 | 418 | 418 | 418 | 418 |
| MH3 | $(15,15)$ | 225 | 450 | 892 | 847 | 892 | 892 | 892 | 892 |
| MH4 | $(16,16)$ | 256 | 512 | 1030 | 985 | 1030 | 1030 | 1030 | 1030 |
| MH5 | $(17,17)$ | 289 | 578 | 1226 | 1162 | 1214 | 1226 | 1226 | 1226 |
| MH6 | $(18,18)$ | 324 | 648 | 1283 | 1230 | 1282 | 1282 | 1283 | 1283 |
| MH7 | $(19,19)$ | 361 | 722 | 1448 | 1378 | 1446 | 1446 | 1446 | 1446 |
| MH8 | $(20,20)$ | 400 | 800 | 1539 | 1472 | 1537 | 1536 | 1537 | 1537 |
| MH9 | $(25,25)$ | 625 | 1250 | 2572 | 2439 | 2559 | 2568 | 2568 | 2568 |
| MH10 | $(4,4,4)$ | 64 | 192 | 199 | 156 | 193 | 199 | 199 | 199 |
| MH11 | $(4,4,5)$ | 80 | 240 | 258 | 203 | 249 | 258 | 258 | 258 |
| MH12 | $(4,5,5)$ | 100 | 300 | 343 | 260 | 324 | 342 | 342 | 342 |
| MH13 | $(4,5,6)$ | 120 | 360 | 400 | 312 | 384 | 398 | 400 | 400 |
| MH14 | $(5,5,5)$ | 125 | 375 | 391 | 304 | 376 | 391 | 391 | 391 |
| MH15 | $(5,5,6)$ | 150 | 450 | 528 | 422 | 513 | 528 | 528 | 528 |
| MH16 | $(5,6,6)$ | 180 | 540 | 607 | 479 | 586 | 607 | 607 | 607 |
| MH17 | $(5,6,7)$ | 210 | 630 | 698 | 539 | 668 | 696 | 697 | 697 |
| MH18 | $(6,6,6)$ | 216 | 648 | 700 | 561 | 683 | 697 | 698 | 699 |
| MH19 | $(6,6,7)$ | 252 | 756 | 834 | 663 | 808 | 830 | 832 | 832 |
| MH20 | $(6,7,7)$ | 294 | 882 | 994 | 779 | 958 | 990 | 992 | 992 |
| MH21 | $(6,7,8)$ | 336 | 1008 | 1087 | 847 | 1047 | 1079 | 1083 | 1083 |
| MH22 | $(7,7,7)$ | 343 | 1029 | 1162 | 907 | 1107 | 1155 | 1158 | 1159 |
| MH23 | $(7,7,8)$ | 392 | 1176 | 1246 | 975 | 1201 | 1238 | 1241 | 1242 |
| MH24 | $(7,8,8)$ | 448 | 1344 | 1449 | 1135 | 1393 | 1439 | 1442 | 1442 |
| MH25 | $(7,8,9)$ | 504 | 1512 | 1645 | 1281 | 1576 | 1626 | 1631 | 1631 |
| MH26 | $(8,8,8)$ | 512 | 1536 | 1566 | 1247 | 1530 | 1555 | 1557 | 1557 |
| MH27 | $(8,8,9)$ | 576 | 1728 | 1883 | 1485 | 1817 | 1861 | 1866 | 1867 |
| MH28 | $(8,9,9)$ | 648 | 1944 | 2075 | 1643 | 2003 | 2057 | 2060 | 2060 |
| MH29 | $(8,9,10)$ | 720 | 2160 | 2339 | 1850 | 2259 | 2309 | 2313 | 2314 |
| MH30 | $(9,9,9)$ | 729 | 2187 | - | 1894 | 2329 | 2416 | 2421 | 2422 |
| MH31 | $(9,9,10)$ | 810 | 2430 | - | 2081 | 2535 | 2603 | 2608 | 2608 |
| MH32 | $(9,10,10)$ | 900 | 2700 | - | 2304 | 2817 | 2886 | 2888 | 2889 |

Table 3.8: Comparison of different bounds for Manhattan instances.

| Instance | MILP <br> time | $L B B 1$ <br> time | $S D P_{S 2}$ <br> time | $S D P_{S 3}$ |  | Instance | $S D P_{S 2}$ <br> res | $S D P_{S 3}$ |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { numCuts } \\ 300 \\ \hline \text { time } \\ \hline \end{gathered}$ | numCuts500time |  |  | $\begin{gathered} \text { numCuts } \\ 300 \end{gathered}$ |  | $\begin{gathered} \text { numCuts } \\ 500 \end{gathered}$ |  |
|  |  |  |  |  |  |  |  | res | cuts | res | cuts |
| MH1 | 0.103 | 0.016 | 0.047 | 0.030 | 0.040 | MH1 | 0.001 | 0.002 | 0 | <0.001 | 0 |
| MH2 | 0.309 | 0.011 | 0.916 | 0.839 | 0.882 | MH2 | $<0.001$ | <0.001 | 0 | 0.006 | 0 |
| MH3 | 2.665 | 0.036 | 20.78 | 57.07 | 103.5 | MH3 | <0.001 | <0.001 | 300 | <0.001 | 500 |
| MH4 | 3.805 | 0.044 | 39.27 | 221.2 | 414.6 | MH4 | 0.013 | 0.013 | 600 | 0.013 | 1000 |
| MH5 | 5.406 | 0.057 | 52.62 | 380.2 | 834.5 | MH5 | 0.017 | 0.011 | 1200 | 0.011 | 2000 |
| MH6 | 7.684 | 0.063 | 69.39 | 558.2 | 1478.1 | MH6 | 0.011 | 0.010 | 916 | 0.01 | 1500 |
| MH7 | 10.29 | 0.079 | 86.52 | 336.9 | 588.5 | MH7 | 0.024 | 0.024 | 1200 | 0.024 | 2000 |
| MH8 | 13.44 | 0.118 | 112.3 | 489.5 | 898.9 | MH8 | 0.024 | 0.023 | 1200 | 0.023 | 2000 |
| MH9 | 49.72 | 0.233 | 343.6 | 2226 | 3210 | MH9 | 0.029 | 0.035 | 1500 | 0.034 | 2500 |
| MH10 | 0.329 | 0.013 | 3.138 | 40.71 | 101.6 | MH10 | <0.001 | <0.001 | 311 | <0.001 | 569 |
| MH11 | 0.491 | 0.022 | 4.557 | 119.9 | 251.0 | MH11 | 0.012 | <0.001 | 574 | <0.001 | 969 |
| MH12 | 0.722 | 0.033 | 3.847 | 3.717 | 3.628 | MH12 | 0.002 | 0.004 | 0 | 0.004 | 0 |
| MH13 | 0.963 | 0.047 | 10.76 | 117.2 | 202.6 | MH13 | 0.004 | 0.018 | 827 | 0.022 | 1322 |
| MH14 | 1.103 | 0.052 | 18.83 | 133.8 | 267.6 | MH14 | 0.022 | 0.041 | 759 | 0.040 | 1284 |
| MH15 | 2.305 | 0.073 | 28.47 | 177.8 | 377.5 | MH15 | 0.021 | 0.036 | 858 | 0.044 | 1219 |
| MH16 | 3.520 | 0.092 | 54.29 | 252.1 | 422.5 | MH16 | 0.005 | 0.031 | 1072 | 0.037 | 1842 |
| MH17 | 5.265 | 0.135 | 78.98 | 291.3 | 429.1 | MH17 | 0.009 | 0.031 | 1075 | 0.035 | 1781 |
| MH18 | 5.548 | 0.138 | 84.35 | 364.7 | 470.6 | MH18 | 0.005 | 0.013 | 1117 | 0.016 | 1773 |
| MH19 | 8.333 | 0.159 | 115.8 | 397.0 | 527.1 | MH19 | 0.012 | 0.027 | 1032 | 0.030 | 1707 |
| MH20 | 12.98 | 0.191 | 162.0 | 512.9 | 829.1 | MH20 | 0.013 | 0.026 | 1121 | 0.030 | 1821 |
| MH21 | 18.76 | 0.234 | 261.3 | 888.5 | 1201 | MH21 | 0.008 | 0.019 | 1500 | 0.022 | 2500 |
| MH22 | 20.29 | 0.234 | 272.8 | 881.1 | 1177 | MH22 | 0.009 | 0.020 | 1500 | 0.022 | 2500 |
| MH23 | 29.29 | 0.290 | 382.2 | 967.7 | 1457 | MH23 | 0.010 | 0.020 | 1500 | 0.019 | 2500 |
| MH24 | 43.01 | 0.338 | 525.9 | 1482 | 1697 | MH24 | 0.011 | 0.023 | 1500 | 0.026 | 2500 |
| MH25 | 61.54 | 0.425 | 670.5 | 1695 | 1880 | MH25 | 0.013 | 0.024 | 1500 | 0.026 | 2500 |
| MH26 | 63.89 | 0.444 | 732.8 | 1726 | 2265 | MH26 | 0.030 | 0.043 | 1500 | 0.047 | 2500 |
| MH27 | 90.82 | 0.512 | 939.7 | 2354 | 2752 | MH27 | 0.025 | 0.037 | 1500 | 0.040 | 2500 |
| MH28 | 132.2 | 0.634 | 1227 | 2869 | 3222 | MH28 | 0.036 | 0.050 | 1500 | 0.054 | 2500 |
| MH29 | 177.0 | 0.772 | 1597 | 3414 | 4264 | MH29 | 0.034 | 0.044 | 1500 | 0.048 | 2500 |
| MH30 | 181.4 | 0.815 | 1600 | 3358 | 3643 | MH30 | 0.028 | 0.038 | 1500 | 0.040 | 2500 |
| MH31 | 249.3 | 0.948 | 2096 | 4242 | 4710 | MH31 | 0.039 | 0.048 | 1500 | 0.051 | 2500 |
| MH32 | 344.2 | 1.128 | 2773 | 5530 | 5851 | MH32 | 0.031 | 0.045 | 1500 | 0.047 | 2500 |

(a) Computation times (in seconds)
(b) Average residuals and number of cuts

Table 3.9: Comparison of computation times, average residuals and total number of added cuts for Manhattan instances.

### 3.7.3 Upper bounds and overall results

We discuss here the results on the upper bounds and provide an overview of the relative gap between best lower and upper bounds for all instances. Table 3.10 shows several statistics related to the performance of the hybrid and nonhybrid upper bounds on the full test set. Besides, it provides the average percentage gap between best lower and upper bound per instance type. Table 3.11 provides an overview of the best lower bound, best upper bound and their relative gap for the full set of instances. For each instance and upper bound type, we compute the upper bound based on the SDP solution resulting from the CP-ALM, and select the best among all to report in Table 3.11. Since, by construction, the hybrid algorithm always provides the best among all upper bounds, the last column of Table 3.11 indicates which of the nonhybrid heuristics performs best when applied independently. Since all upper bounds can be computed relatively fast, we omit computation times here.

It follows from the tables that our bounds are very strong for the Manhattan and the
reload instances, as the average gap between the best lower and best upper bound using the hybrid heuristic is $1.25 \%$ and $3.90 \%$, respectively. For the Erdős-Rényi instances this gap is much larger. It is known that the quality of a lower bound, and thus also of a related upper bound, deteriorate when the size of the problem increases. Also, the results indicate that the reload and Manhattan instances are easier to solve than the Erdős-Rényi instances for all here tested QCCP approaches. Nevertheless, the average gap on the Erdős-Rényi instances with up to 1000 arcs is only $10 \%$.

When comparing the different upper bounds, we conclude that the SQ-algorithm overall outperforms the other methods, followed by oversampling and undersampling rounding. We however observe a clear relationship with the instance type. For the Erdős-Rényi instances the SQ-algorithm is convincingly the best heuristic, while for the reload instances the other methods perform reliable as well, probably due to the smaller instance size. For the Manhattan instances, however, the sequential Q-learning heuristic performs well on the smaller instances, but is outperformed by oversampling rounding for larger $m$. This can be explained by the smaller number of iterations of the SQ-algorithm for these type of instances. Since the number of agents in the SQ-algorithm for the Manhattan instances is significantly larger than for the other instance types, we needed to decrease the number of iterations in order to be able to solve the resulting SPP efficiently. Hence, the learning effect of the SQalgorithm is decreased, while it is in particular that part that makes the algorithm powerful. Nevertheless, we observe for almost all Manhattan instances that the hybrid algorithm obtains a strictly stronger upper bound than $U B_{E B}, U B_{U S}$ or $U B_{O S}$. This means that the SQ-algorithm, although not always the favoured heuristic when implemented independently, creates cycles that can lead to an improvement of the best upper bound.

### 3.8 Conclusions

This chapter provides an in-depth theoretical as well as practical study on the QCCP. We provide various lower and upper bounds for the QCCP based on semidefinite programming. Moreover, we introduce efficient methods to compute these bounds and give an analysis of their theoretical properties.

We first introduce three SDP relaxations with increasing complexity. Our strongest SDP relaxation, $\left(S D P_{S 3}\right)$, see (3.17), contains a large number of constraints which makes it a strong but very difficult to solve relaxation. Since there are no efficient solvers that can solve SDP relaxations including BQP cuts, we derive a cutting-plane augmented Lagrangian method that is designed to solve such relaxations, see Algorithm 3.2. Our algorithm starts from the Peaceman-Rachford splitting method where the involved polyhedral set is strengthened throughout the algorithm by adding valid cuts. To project onto the polyhedral

| Statistics on upper bounds and average gaps |  |  |  |  |
| :--- | ---: | :--- | :--- | :--- |
| Average gap on all instances | $20.02 \%$ | Percentage of instances $U B_{E B}$ performs best | $36.89 \%$ |  |
| Average gap on Erdős-Rényi instances | $72.30 \%$ | Percentage of instances $U B_{U S}$ performs best | $53.28 \%$ |  |
| Average gap on Manhattan instances | $1.25 \%$ | Percentage of instances $U B_{O S}$ performs best | $68.85 \%$ |  |
| Average gap on Reload instances | $3.90 \%$ | Percentage of instances $U B_{S Q}$ performs best | $77.87 \%$ |  |
| Average gap on instances with $m \leq 1000$ | $10.58 \%$ | Percentage of instances $U B_{H Y}$ strictly lower | $25.41 \%$ |  |
|  |  | than others |  |  |

Table 3.10: Statistics on performance of upper bounds and average gaps on total test set.
set, we implement a semi-parallelized version of Dykstra's cyclic projection algorithm, see Section 3.5.4 for details. Parallelization here refers to clustering the set of BQP inequalities into subsets of nonoverlapping cuts. Besides the parallelization step we implement several other efficiency improving steps that contribute to the effectiveness of the CP-ALM. Our algorithm also benefits from warm starts when adding new cuts. The CP-ALM is able to compute lower bounds for large instances up to 2700 arcs, thus having a semidefinite constraint of order 2700, including 7290000 nonnegative constrained variables, and up to 2500 BQP cuts within two hours.

We also introduce several upper bounding approaches that exploit matrices resulting from the CP-ALM, including randomized undersampling (see Algorithm 3.3) and randomized oversampling (see Algorithm 3.4). Additionally, we propose an SDP-based distributed reinforcement learning algorithm, which we call sequential Q-learning, see Algorithm 3.5. Starting from the SDP solution matrix, we let artificial agents learn how to find near-optimal cycles in the graph. We are not aware of other approaches in the literature that combine SDP and reinforcement learning.

We perform extensive numerical experiments. Our numerical results show that both semidefinite programming bounds $S D P_{S 2}$ and $S D P_{S 3}$ outperform the current strongest QCCP bounds. The results show that $S D P_{S 3}$ bounds are significantly better than $S D P_{S 2}$ bounds, provided that there exist violated triangle inequalities. Among the upper bounding approaches, our sequential Q-learning algorithm is the winner. The average gap between the best lower and upper bounds on test instances with up to 1000 arcs is about $10 \%$, while for certain instances this average gap can be as low as $1.25 \%$, see Table 3.10 and Table 3.11 for details.

Several of the newly introduced approaches can be extended to other problems. The various components of the CP-ALM are rather general, which make it possible to adopt it for solving other SDP models that involve a large number of cutting planes, such as for the quadratic traveling salesman problem. In Chapter 4 we show how a version of the CPALM is applied to compute bounds for several graph partition problems. Our SDP-based reinforcement learning approach can also be extended for finding feasible solutions to other optimization problems. We expect that the sequential Q-learning approach should perform well for problems on complete graphs.

| Instance | Best lower bound | Hybrid upper bound | Gap <br> (\%) | Best nonhybrid heuristic |
| :---: | :---: | :---: | :---: | :---: |
| ER1 | 319 | 319 | 0 | EB, US, OS, SQ |
| RER1 | 293 | 293 | 0 | EB, US, OS, SQ |
| ER2 | 386 | 386 | 0 | EB, US, OS, SQ |
| RER2 | 391 | 391 | 0 | EB, US, OS, SQ |
| ER3 | 296 | 311 | 5 | OS, SQ |
| RER3 | 266 | 288 | 8 | OS |
| ER4 | 336 | 447 | 33 | SQ |
| RER4 | 240 | 294 | 23 | SQ |
| ER5 | 324 | 404 | 25 | SQ |
| RER5 | 220 | 321 | 46 | SQ |
| ER6 | 338 | 451 | 33 | SQ |
| RER6 | 151 | 253 | 68 | SQ |
| ER7 | 319 | 493 | 55 | SQ |
| RER7 | 119 | 236 | 98 | SQ |
| ER8 | 305 | 525 | 72 | SQ |
| RER8 | 95 | 283 | 198 | SQ |
| ER9 | 289 | 520 | 80 | SQ |
| RER9 | 133 | 399 | 200 | SQ |
| ER10 | 293 | 455 | 55 | SQ |
| RER10 | 54 | 312 | 478 | SQ |
| ER11 | 234 | 236 | 1 | SQ |
| RER11 | 172 | 172 | 0 | US, OS, SQ |
| ER12 | 173 | 187 | 8 | SQ |
| RER12 | 87 | 113 | 30 | SQ |
| ER13 | 202 | 245 | 21 | SQ |
| RER13 | 143 | 169 | 18 | SQ |
| ER14 | 200 | 280 | 40 | SQ |
| RER14 | 74 | 170 | 130 | SQ |
| ER15 | 191 | 326 | 71 | SQ |
| RER15 | 37 | 175 | 373 | SQ |
| MH1 | 103 | 103 | 0 | EB, US, OS, SQ |
| MH2 | 418 | 418 | 0 | EB, US, OS, SQ |
| MH3 | 892 | 892 | 0 | EB, US, OS, SQ |
| MH4 | 1030 | 1030 | 0 | EB, US, OS, SQ |
| MH5 | 1226 | 1226 | 0 | EB, US, OS |
| MH6 | 1283 | 1283 | 0 | EB, US, OS, SQ |
| MH7 | 1446 | 1448 | 0 | EB, US, OS, SQ |
| MH8 | 1537 | 1539 | 0 | EB |
| MH9 | 2568 | 2572 | 0 | EB |
| MH10 | 199 | 199 | 0 | EB, US, OS, SQ |
| MH11 | 258 | 258 | 0 | EB, US, OS, SQ |
| MH12 | 342 | 348 | 2 | US |
| MH13 | 400 | 400 | 0 | EB, US, OS |
| MH14 | 391 | 391 | 0 | EB, US, OS |
| MH15 | 528 | 528 | 0 | EB, US, OS |
| MH16 | 607 | 607 | 0 | EB, US, OS |
| MH17 | 697 | 698 | 0 | OS |
| MH18 | 699 | 706 | 1 | OS |
| MH19 | 832 | 839 | 1 | US, OS |
| MH20 | 992 | 999 | 1 | OS |
| MH21 | 1083 | 1093 | 1 | OS |
| MH22 | 1159 | 1171 | 1 | OS |
| MH23 | 1242 | 1272 | 2 | OS |
| MH24 | 1442 | 1498 | 4 | OS |
| MH25 | 1631 | 1702 | 4 | OS |
| MH26 | 1557 | 1576 | 1 | OS |
| MH27 | 1867 | 1940 | 4 | OS |
| MH28 | 2060 | 2141 | 4 | OS |
| MH29 | 2314 | 2426 | 6 | OS |
| MH30 | 2422 | 2552 | 5 | OS |
| MH31 | 2608 | 2775 | 6 | OS |
| MH32 | 2889 | 3077 | 7 | OS |


| Instance |  | Hybrid <br> upper <br> bound | Gap <br> (\%) | Best nonhybrid heuristic |
| :---: | :---: | :---: | :---: | :---: |
| REL1 | 4 | 4 | 0 | EB, US, OS, SQ |
| REL2 | 9 | 9 | 0 | EB, US, OS, SQ |
| REL3 | 5 | 5 | 0 | EB, US, OS, SQ |
| REL4 | 12 | 12 | 0 | US |
| REL5 | 4 | 4 | 0 | EB, US, OS, SQ |
| REL6 | 14 | 14 | 0 | US, OS, SQ |
| REL7 | 5 | 5 | 0 | US, OS, SQ |
| REL8 | 11 | 11 | 0 | EB, US, OS, SQ |
| REL9 | 2 | 2 | 0 | EB, US, OS, SQ |
| REL10 | 12 | 12 | 0 | EB, US, OS, SQ |
| REL11 | 3 | 3 | 0 | EB, US, OS, SQ |
| REL12 | 9 | 9 | 0 | EB, US, OS, SQ |
| REL13 | 4 | 4 | 0 | US, OS, SQ |
| REL14 | 11 | 11 | 0 | EB, US, OS, SQ |
| REL15 | 4 | 4 | 0 | EB, US, OS, SQ |
| REL16 | 11 | 11 | 0 | EB, US, OS, SQ |
| REL17 | 4 | 4 | 0 | EB, US, OS, SQ |
| REL18 | 10 | 10 | 0 | EB, US, OS, SQ |
| REL19 | 5 | 5 | 0 | EB, US, OS, SQ |
| REL20 | 11 | 11 | 0 | EB, US, OS, SQ |
| REL21 | 5 | 5 | 0 | US, OS, SQ |
| REL22 | 12 | 12 | 0 | SQ |
| REL23 | 4 | 4 | 0 | US, OS, SQ |
| REL24 | 11 | 11 | 0 | US, OS, SQ |
| REL25 | 5 | 5 | 0 | EB, US, OS, SQ |
| REL26 | 9 | 10 | 11 | SQ |
| REL27 | 4 | 4 | 0 | US, OS, SQ |
| REL28 | 9 | 9 | 0 | US, OS, SQ |
| REL29 | 6 | 6 | 0 | US, OS, SQ |
| REL30 | 10 | 10 | 0 | US, OS, SQ |
| REL31 | 5 | 5 | 0 | US, OS, SQ |
| REL32 | 11 | 11 | 0 | US, OS, SQ |
| REL33 | 4 | 4 | 0 | US, OS, SQ |
| REL34 | 8 | 8 | 0 | OS, SQ |
| REL35 | 4 | 4 | 0 | EB, US, OS, SQ |
| REL36 | 8 | 8 | 0 | US, OS, SQ |
| REL37 | 6 | 6 | 0 | US, OS, SQ |
| REL38 | 11 | 11 | 0 | EB, US, OS, SQ |
| REL39 | 3 | 3 | 0 | EB, US, OS, SQ |
| REL40 | 7 | 7 | 0 | EB, US, OS, SQ |
| REL41 | 4 | 4 | 0 | EB, US, OS, SQ |
| REL42 | 7 | 11 | 57 | SQ |
| REL43 | 3 | 3 | 0 | EB, US, OS, SQ |
| REL44 | 7 | 10 | 43 | SQ |
| REL45 | 3 | 3 | 0 | SQ |
| REL46 | 6 | 6 | 0 | EB, US, OS, SQ |
| REL47 | 3 | 3 | 0 | SQ |
| REL48 | 5 | 5 | 0 | OS, SQ |
| REL49 | 4 | 4 | 0 | US, OS, SQ |
| REL50 | 8 | 8 | 0 | OS |
| REL51 | 3 | 3 | 0 | SQ |
| REL52 | 6 | 9 | 50 | SQ |
| REL53 | 4 | 4 | 0 | EB, US, OS, SQ |
| REL54 | 9 | 11 | 22 | SQ |
| REL55 | 3 | 3 | 0 | EB, US, OS, SQ |
| REL56 | 8 | 11 | 38 | SQ |
| REL57 | 4 | 4 | 0 | US, OS, SQ |
| REL58 | 7 | 7 | 0 | US, OS, SQ |
| REL59 | 3 | 3 | 0 | SQ |
| REL60 | 8 | 9 | 13 | OS, SQ |

Table 3.11: Overview of best lower bounds, best hybrid and nonhybrid upper bounds and their relative gaps for all instances.

# Partitioning through projections: strong SDP bounds for large graph partition problems 

## Chapter summary

The graph partition problem (GPP) aims at clustering the vertex set of a graph into a fixed number of disjoint subsets of given sizes such that the sum of weights of edges joining different sets is minimized. This chapter investigates the quality of doubly nonnegative (DNN) relaxations, i.e., relaxations having matrix variables that are both positive semidefinite and nonnegative, strengthened by additional polyhedral cuts for two variations of the GPP: the $k$-equipartition and the graph bisection problem. After reducing the size of the relaxations by facial reduction, we solve them by a cutting-plane algorithm that combines an augmented Lagrangian method with Dykstra's projection algorithm. Since many components of our algorithm are general, the algorithm is suitable for solving various DNN relaxations with a large number of cutting planes.

We are the first to show the power of DNN relaxations with additional cutting planes for the GPP on large benchmark instances up to 1024 vertices. Computational results show impressive improvements in strengthened DNN bounds.

### 4.1 Introduction

The graph partition problem (GPP) is the problem of partitioning the vertex set of a graph into a fixed number of subsets, say $k$, of given sizes such that the sum of weights of edges joining different sets is minimized. In the case that all sets are of equal sizes we refer to the resulting GPP as the $k$-equipartition problem $(k$-EP). The case of the graph partition problem with $k=2$ is known as the graph bisection problem (GBP). In the GBP the sizes of two subsets might differ. The special case of the GBP where both subsets have the same size is known in the literature as the equicut problem, see e.g. [232].

The graph partition problem is known to be $\mathcal{N} \mathcal{P}$-hard [165]. It is a fundamental problem that is extensively studied, mostly due to its applications in numerous fields, including VLSI design, social networks, floor planning, data mining, air traffic, image processing, image segmentation, parallel computing and telecommunication, see e.g., the book [47] and the references therein. Recent studies in quantum circuit design [214] also relate to the general graph partition problem. Furthermore, the GPP is used to compute bounds for the bandwidth problem [320]. For an overview of recent advances in graph partitioning, we refer the reader to [59].

There exist bounding approaches for the GPP that are valid for all variations of the GPP. We list only some of them in this paragraph. Donath and Hoffman [104] derive an eigenvalue bound for the graph partition problem. That bound is improved by Rendl and Wolkowicz [321]. Wolkowicz and Zhao [371] derive semidefinite programming (SDP) relaxations for the graph partition problem that are based on the so-called vector lifting approach. That is, relaxations in [371] have matrix variables of order $n k+1$, where $n$ is the number of vertices in the graph. An SDP relaxation based on the so-called matrix lifting approach is derived in [343], resulting in a compact relaxation for the GPP having matrix variables of order $n$. The relaxation from [343] is a doubly nonnegative (DNN) relaxation, which is an SDP relaxation over the set of nonnegative matrices. In general, vector lifting relaxations provide stronger bounds than matrix lifting relaxations, see e.g., [241, 343].

For the $k$-equipartition problem, Karisch and Rendl [231], among others, show how to reformulate the Donath-Hoffman and the Rendl-Wolkowicz relaxations as semidefinite programming problems. Karisch and Rendl also present several SDP relaxations with increasing complexity for the $k$-EP with matrix variables of order $n$ that dominate relaxations from [104, 321]. The strongest SDP relaxation from [231] is a DNN relaxation with additional triangle and independent set inequalities. That relaxation provides the best known SDP bounds for the $k$-EP. However, it is difficult to compute those bounds for general graphs with more than 300 vertices when using interior-point methods.

In this chapter we focus on the relaxations derived in [371] and [231] for large instances. For a more comprehensive overview of SDP-based solution approaches for the graph partition problem and its variants, we refer to Section 1.4.3.

## Main results and outline

Doubly nonnegative relaxations are known to provide superior bounds for various optimization problems. Although additional cutting planes further improve DNN relaxations, it is extremely difficult to compute the resulting bounds already for relaxations with matrix variables of order 300 via interior-point methods. We design an efficient algorithm for computing DNN bounds with a huge number of additional cutting planes and show the power of the resulting bounds for two variations of the graph partition problem.

We conduct a study for the $k$-equipartition problem and the graph bisection problem. Although there exists a DNN relaxation for the GPP [371] that is suitable for both problems, we study the problems separately. Namely, the $k$-EP allows various equivalent SDP relaxations having different sizes of matrix variables due to the problem's invariance under permutations of the subsets. Since one can solve DNN relaxations with smaller matrix variables more efficiently than those with larger matrix variables, we consider the matrix lifting DNN relaxation for the $k$-EP from [231] that is strengthened by the triangle and independent set inequalities. On the other hand, the vector lifting DNN relaxation for the GBP from [371] is known to dominate matrix lifting DNN relaxations for the same problem. Therefore, we consider the vector lifting DNN relaxation for the GBP and strengthen it by adding boolean quadric polytope (BQP) inequalities. Since the relaxation from [371] that we rely on was originally proposed for the general GPP, our approach for the GBP can be generalized in a straightforward way.

Prior to solving the DNN relaxations, we use facial reduction to obtain equivalent smaller dimensional relaxations that are strictly feasible. The approach we use for the GBP is based on the dimension of the underlying polytope. Although strict feasibility of an SDP is not required for our solver, it makes the procedure more efficient.

To solve the DNN relaxations with additional polyhedral inequalities, we design a cuttingplane algorithm called the cutting-plane ADMM (CP-ADMM). Our algorithm combines the Alternating Direction Method of Multipliers (ADMM) with Dykstra's projection algorithm. The ADMM exploits the natural splitting of the relaxations that arises from the facial reduction. Dykstra's cyclic projection algorithm finds projections onto polyhedra induced by the violated cuts. Since facial reduction eliminates redundant constraints and projects a relaxation onto a smaller dimensional space, the projections in the CP-ADMM are easier and faster. To further improve efficiency of the CP-ADMM, we cluster nonoverlapping cuts, which allows us to perform the projections in each cluster simultaneously. Efficiency of the algorithm is also due to the exploitation of warm starts, as well as the use efficient separation routines. Since we present the various components of the CP-ADMM in a general way, the algorithm is suitable for solving various DNN relaxations incorporating additional cutting planes.

Our numerical results show that the CP-ADMM computes strong GPP bounds for graphs with up to 1024 vertices by adding at most 50000 cuts in less than two hours. Since our algorithm does not require lots of memory, we are able to compute strong bounds for even larger graphs than presented here. Numerical results also show that the additional cutting planes significantly improve the DNN relaxations, and that the resulting bounds can close gaps for instances with up to 500 vertices.

This chapter is structured as follows. In Section 4.2 we introduce the graph partition problem. Section 4.2.1 presents DNN relaxations for the $k$-EP. In Section 4.2.2 we present DNN relaxations for the GBP and show how to apply facial reduction by exploiting the dimension of the bisection polytope. Our cutting-plane augmented Lagrangian algorithm is introduced in Section 4.3. The main ingredients of the algorithm are given in Section 4.3.1 and Section 4.3.2. In particular, Section 4.3.1 explains steps of the ADMM and Section 4.3.2 introduces Dykstra's projection algorithm and its semi-parallelized version. The CP-ADMM is outlined in Section 4.3.3. Section 4.4 considers various families of cutting planes that are used to strengthen DNN relaxations for the $k$-EP and GBP. Numerical results are given in Section 4.5.

### 4.2 The graph partition problem

Let $G=(V, E)$ be an undirected graph with vertex set $V, n:=|V|$, and edge set $E$. Let $w: E \rightarrow \mathbb{R}$ be an edge weight function and let $k$ be an integer such that $2 \leq k \leq n-1$. The graph partition problem is to partition the vertex set of $G$ into $k$ disjoint sets $S_{1}, \ldots, S_{k}$ of specified sizes $m_{1} \geq \cdots \geq m_{k} \geq 1, \sum_{j=1}^{k} m_{j}=n$ such that the total weight of edges joining different sets $S_{j}$ is minimized. If $k=2$, then we refer to the corresponding graph partition problem as the graph bisection problem. If $m_{1}=\cdots=m_{k}=n / k$, then the resulting GPP is known as the $k$-equipartition problem.

Let $W=\left(w_{i j}\right) \in \mathcal{S}^{n}$ denote the weight matrix of $G$, where $w_{i j}=w(\{i, j\})$ if $\{i, j\} \in E$ and $w_{i j}=0$ otherwise, and let $\mathbf{m}:=\left[m_{1} \ldots m_{k}\right]^{\top}$. For a given partition of $G$ into $k$ subsets, let $P=\left(P_{i j}\right) \in\{0,1\}^{n \times k}$ be the partition matrix defined as follows:

$$
P_{i j}:=\left\{\begin{array}{ll}
1 & \text { if } i \in S_{j},  \tag{4.1}\\
0 & \text { otherwise, }
\end{array} \quad \text { for all } i \in[n], j \in[k]\right.
$$

Thus, the $j$ th column of $P$ is the characteristic vector of $S_{j}$. The total weight of the partition, i.e., the sum of weights of edges that join different sets equals

$$
\frac{1}{2} \operatorname{tr}\left(W\left(\mathbf{J}_{n}-P P^{\top}\right)\right)=\frac{1}{2} \operatorname{tr}\left(L P P^{\top}\right)
$$

where $L:=\operatorname{Diag}\left(W \mathbf{1}_{n}\right)-W$ is the weighted Laplacian matrix of $G$. The GPP can be formulated as the following binary optimization problem:
(GPP)

$$
\begin{array}{ll}
\min _{P} & \frac{1}{2}\left\langle L, P P^{\top}\right\rangle \\
\text { s.t. } & P \mathbf{1}_{k}=\mathbf{1}_{n} \\
& P^{\top} \mathbf{1}_{n}=\mathbf{m} \\
& P_{i j} \in\{0,1\} \quad \forall i \in[n], j \in[k], \tag{4.2~d}
\end{array}
$$

where $P \in \mathbb{R}^{n \times k}$. Note that the objective function (4.2a) is quadratic. The constraints (4.2b) ensure that each vertex must be in exactly one subset. The cardinality constraints (4.2c) take care that the number of vertices in subset $S_{j}$ is $m_{j}$ for $j \in[k]$.

Let us briefly consider the polytope induced by all feasible partitions of $G$. Let $F_{n}^{k}(\mathbf{m})$ be the set of all characteristic vectors representing a partition of $n$ vertices into $k$ disjoint sets corresponding to the cardinalities in $\mathbf{m}$. In other words, $F_{n}^{k}(\mathbf{m})$ contains binary vectors of the form $\operatorname{vec}(P)$, where $P$ is a partition matrix, see (4.1):

$$
\begin{equation*}
F_{n}^{k}(\mathbf{m})=\left\{x \in\{0,1\}^{k n}:\binom{\mathbf{I}_{k} \otimes \mathbf{1}_{n}^{\top}}{\mathbf{1}_{k}^{\top} \otimes \mathbf{I}_{n}} x=\binom{\mathbf{m}}{\mathbf{1}_{n}}\right\} . \tag{4.3}
\end{equation*}
$$

We now define $\operatorname{conv}\left(F_{n}^{k}(\mathbf{m})\right)$ as the $k$-partition polytope. Since the constraint matrix defining $F_{n}^{k}(\mathbf{m})$ is totally unimodular, this polytope can be explicitly written as follows:

$$
\begin{equation*}
\operatorname{conv}\left(F_{n}^{k}(\mathbf{m})\right)=\left\{x \in \mathbb{R}^{k n}:\binom{\mathbf{I}_{k} \otimes \mathbf{1}_{n}^{\top}}{\mathbf{1}_{k}^{\top} \otimes \mathbf{I}_{n}} x=\binom{\mathbf{m}}{\mathbf{1}_{n}}, x \geq \mathbf{0}\right\} \tag{4.4}
\end{equation*}
$$

The $k$-partition polytope can be seen as a special case of a transportation polytope, see e.g., [111]. We now derive the dimension of the $k$-partition polytope, which will be exploited in Section 4.2.2. The following result is implied by the dimension of the transportation polytope [111]. However, we add a proof for completeness.
Theorem 4.1. The dimension of $\operatorname{conv}\left(F_{n}^{k}(\mathbf{m})\right)$ equals $(k-1)(n-1)$.
Proof. Let $B:=\binom{\mathbf{I}_{k} \otimes \mathbf{1}_{n}^{\top}}{\mathbf{1}_{k}^{\top} \otimes \mathbf{I}_{n}}$. Since for all $i \in[k n]$ there exists a partition such that $x_{i}=1$, we know that $\operatorname{dim}\left(\operatorname{conv}\left(F_{n}^{k}(\mathbf{m})\right)\right)=\operatorname{dim}(\operatorname{Nul}(B))$.

Let $b_{1}, \ldots, b_{k n}$ denote the columns of $B$. Since $\left\{b_{1}, \ldots, b_{n}, b_{n+1}, b_{2 n+1}, \ldots, b_{(k-1) n+1}\right\}$ is linearly independent, it follows that $\operatorname{rank}(B) \geq n+k-1$. Next, we define for all $l=1, \ldots, k-1$ and $i=2, \ldots, n$ a vector $w^{l, i} \in \mathbb{R}^{k n}$ as follows:

$$
\left(w^{l, i}\right)_{j}= \begin{cases}+1 & \text { if } j=1 \text { or } j=l \cdot n+i \\ -1 & \text { if } j=i \text { or } j=l \cdot n+1 \\ 0 & \text { otherwise }\end{cases}
$$

One can verify that $B w^{l, i}=\mathbf{0}$ for all $l=1, \ldots, k-1$ and $i=2, \ldots, n$. Moreover, since $w^{l, i}$ is the only vector that has a nonzero entry on position $l n+i$ among all defined vectors, the set $\left\{w^{l, i}: l=1, \ldots, k-1, i=2, \ldots, n\right\}$ is a linearly independent set. This proves that $\operatorname{dim}(\operatorname{Nul}(B)) \geq(k-1)(n-1)$.

Since $\operatorname{rank}(B)+\operatorname{dim}(\operatorname{Nul}(B))=k n$, we conclude that $\operatorname{dim}(\operatorname{Nul}(B))=(k-1)(n-1)$.

### 4.2.1 DNN relaxations for the $\boldsymbol{k}$-equipartition problem

Throughout this section we focus on the special case of the $k$-EP, i.e., $m_{j}=n / k$ for all $j \in[k]$. There exist several ways to obtain semidefinite programming relaxations for the $k$-EP. Namely, to obtain an SDP relaxation for the $k$-EP one can linearize the objective function of $(G P P)$ by introducing a matrix variable of order $n$, which results in a matrix lifting relaxation, see e.g., [231, 343]. Another approach is to linearize the objective function by lifting the problem in the space of $(n k+1) \times(n k+1)$ matrices, which results in a vector lifting relaxation, see [371]. We call a DNN relaxation basic if it does not contain additional cutting planes such as triangle inequalities, etc. It is proven in [342] that the basic matrix and vector lifting DNN relaxations for the $k$-EP are equivalent. A more elegant proof of the same result can be found in Kuryatnikova et al. [241]. Since one can solve the basic matrix lifting relaxation from [231] more efficiently than the equivalent vector lifting relaxation from [371], we develop our algorithm for the matrix lifting relaxation for the $k$-EP.

To linearize the objective from $(G P P)$ we replace $P P^{\top}$ by a matrix variable $Y \in \mathcal{S}^{n}$. From (4.2b) it follows that $Y_{i i}=\sum_{j=1}^{k} P_{i j}^{2}=\sum_{j=1}^{k} P_{i j}=1$ for all $i \in[n]$. From (4.2b) (4.2c) we have $Y \mathbf{1}_{n}=P P^{\top} \mathbf{1}_{n}=\frac{n}{k} P \mathbf{1}_{k}=\frac{n}{k} \mathbf{1}_{n}$. After putting those constraints together, adding $Y \geq \mathbf{0}$ and $Y \succeq \mathbf{0}$, we arrive at the following DNN relaxation introduced by Karisch and Rendl [231]:
$\left(D N N_{E P}\right)$

$$
\begin{array}{cl}
\min _{Y} & \frac{1}{2}\langle L, Y\rangle \\
\text { s.t. } & \operatorname{diag}(Y)=\mathbf{1}_{n}  \tag{4.5}\\
& Y \mathbf{1}_{n}=\frac{n}{k} \mathbf{1}_{n} \\
& Y \succeq \mathbf{0}, \quad Y \geq \mathbf{0} .
\end{array}
$$

We refer to $\left(D N N_{E P}\right)$ as the basic matrix lifting relaxation. We show below that the nonnegativity constraints in $\left(D N N_{E P}\right)$ are redundant for the equicut problem.

Lemma 4.2. Let $k=2$ and $Y \in \mathcal{S}_{+}^{n}$ be such that $\operatorname{diag}(Y)=\mathbf{1}_{n}$ and $Y \mathbf{1}_{n}=\frac{n}{2} \mathbf{1}_{n}$. Then, $Y \geq \mathbf{0}$.

Proof. From $Y \mathbf{1}_{n}=\frac{n}{2} \mathbf{1}_{n}$ it follows that $\mathbf{1}_{n}$ is an eigenvector of $Y$ corresponding to the eigenvalue $n / 2$. Then, the eigenvalue decomposition of $Y$ is

$$
Y=\frac{1}{2} \mathbf{J}_{n}+\sum_{i=2}^{n} \lambda_{i} v_{i} v_{i}^{\top}
$$

where $v_{i}$ is the eigenvector of $Y$ corresponding to the eigenvalue $\lambda_{i}$ for $i=2, \ldots, n$. Moreover, eigenvectors $v_{i}$ are orthogonal to $\mathbf{1}_{n}$. Thus $2 Y-\mathbf{J}_{n}=2 \sum_{i=2}^{n} \lambda_{i} v_{i} v_{i}^{\top} \succeq \mathbf{0}$.

Now, let $Z:=2 Y-\mathbf{J}_{n}$. From $\operatorname{diag}(Y)=\mathbf{1}_{n}$ it follows that $\operatorname{diag}(Z)=\mathbf{1}_{n}$. Since $Z \succeq \mathbf{0}$ we have that $-1 \leq Z_{i j} \leq 1$ for all $i, j \in[n]$, which implies that $Y_{i j} \geq 0$ for all $i, j \in[n]$.

For a different proof of Lemma 4.2 see e.g., Theorem 4.3 in [231]. The relaxation ( $D N N_{E P}$ ) can be further strengthened by adding triangle and independent set inequalities, see Section 4.4.1 and 4.4.2, respectively. This strengthened relaxation is proposed in [231] and provides currently the strongest SDP bounds for the $k$-EP.

As proposed in [231], one can eliminate $Y \mathbf{1}_{n}=\frac{n}{k} \mathbf{1}_{n}$ in (4.5) and project the relaxation onto a smaller dimensional space, by exploiting the following result.

Lemma 4.3 ([231]). Let $V \in \mathbb{R}^{n \times(n-1)}$ such that $V^{\top} \mathbf{1}_{n}=\mathbf{0}$ and $\operatorname{rank}(V)=n-1$. Then,

$$
\left\{Y \in \mathcal{S}^{n}: \begin{array}{c}
\operatorname{diag}(Y)=\mathbf{1}_{n} \\
Y \mathbf{1}_{n}=\frac{n}{k} \mathbf{1}_{n}
\end{array}\right\}=\left\{\frac{1}{k} \mathbf{J}_{n}+V R V^{\top}: R \in \mathcal{S}^{n-1}, \operatorname{diag}\left(V R V^{\top}\right)=\frac{k-1}{k} \mathbf{1}_{n}\right\}
$$

The matrix $V$ in Lemma 4.3 can be any matrix which columns form a basis for $\mathbf{1}_{n}^{\perp}$, e.g.,

$$
\begin{equation*}
V=\binom{\mathbf{I}_{n-1}}{-\mathbf{1}_{n-1}^{\top}} \tag{4.6}
\end{equation*}
$$

We use the result of Lemma 4.3 and replace $Y$ by $\frac{1}{k} \mathbf{J}_{n}+V R V^{\top}$ in $\left(D N N_{E P}\right)$, which leads to the following equivalent relaxation:

$$
\begin{array}{ll}
\min _{R} & \left\langle L_{E P}, V R V^{\top}\right\rangle \\
\text { s.t. } & \operatorname{diag}\left(V R V^{\top}\right)=\frac{k-1}{k} \mathbf{1}_{n}  \tag{4.7}\\
& V R V^{\top} \geq-\frac{1}{k} \mathbf{J}_{n}, \quad R \succeq \mathbf{0}
\end{array}
$$

where $R \in \mathcal{S}_{+}^{n-1}$. Here, we exploit $\left\langle L, \mathbf{J}_{n}\right\rangle=0$ to rewrite the objective, and define

$$
\begin{equation*}
L_{E P}:=\frac{1}{2} L . \tag{4.8}
\end{equation*}
$$

It is not difficult to verify that the matrix

$$
\hat{R}=\frac{n(k-1)}{k(n-1)} \mathbf{I}_{n-1}-\frac{(k-1)}{k(n-1)} \mathbf{J}_{n-1}
$$

is feasible for (4.7), see also [231]. The matrix $\hat{R}$ has two distinct eigenvalues, namely $\frac{n(k-1)}{k(n-1)}$ with multiplicity $n-2$ and $\frac{(k-1)}{k(n-1)}$ with multiplicity one. This implies that $\hat{R} \succ 0$. Also,

$$
\frac{1}{k} \mathbf{J}_{n}+V \hat{R} V^{\top}=\frac{n(k-1)}{k(n-1)} \mathbf{I}_{n}+\frac{(n-k)}{k(n-1)} \mathbf{J}_{n}>\mathbf{0}
$$

and thus $V \hat{R} V^{\top}>-\frac{1}{k} \mathbf{J}_{n}$. This shows that $\hat{R}$ is a Slater feasible point of (4.7).
For future reference, we define the following sets:

$$
\begin{align*}
\mathcal{R}_{E P} & :=\left\{R \in \mathcal{S}^{n-1}: R \succeq \mathbf{0}\right\}  \tag{4.9}\\
\mathcal{X}_{E P} & :=\left\{X \in \mathcal{S}^{n}: \operatorname{diag}(X)=\frac{k-1}{k} \mathbf{1}_{n},-\frac{1}{k} \mathbf{J}_{n} \leq X \leq \frac{k-1}{k} \mathbf{J}_{n}\right\} \tag{4.10}
\end{align*}
$$

Now, we rewrite the DNN relaxation (4.7) as follows:

$$
\begin{equation*}
\min \left\{\left\langle L_{E P}, X\right\rangle: X=V R V^{\top}, R \in \mathcal{R}_{E P}, X \in \mathcal{X}_{E P}\right\} \tag{4.11}
\end{equation*}
$$

Note that $\mathcal{X}_{E P}$ also contains upper bound constraints on $X$, which are redundant for (4.7). These constraints speed up the convergence of our algorithm, as explained in Section 4.3.1. In the same section, it becomes clear that the inclusion of redundant constraints should not complicate the structure of $\mathcal{X}_{E P}$ too much. Whether or not to include a redundant constraint, is determined by an empirical trade-off between these measures.

### 4.2.2 DNN relaxations for the bisection problem

For the graph bisection problem there exist both vector and matrix lifting SDP relaxations. The matrix lifting relaxations derived in $[232,343]$ are equivalent and have matrix variables of order $n$. A vector lifting SDP relaxation for the GBP is derived by Wolkowicz and Zhao [371] and has a matrix variable of order $2 n+1$. The DNN relaxation from [371] dominates the basic matrix lifting DNN relaxations, i.e., DNN relaxations without additional cutting planes, see [343] for a proof. In [344] a matrix lifting DNN relaxation with additional cutting planes is derived for the GBP that is equivalent to the DNN relaxation from [371]. Although the relaxation from [344] has a matrix variable of order $n$, we work with the vector lifting DNN relaxation because it has a more appropriate structure for our ADMM approach. Due to the relaxation from [371] being defined for the general graph partition problem, the approach that we follow can be easily extended to partitioning into more than two classes.

In this section we present the vector lifting DNN relaxation from [371] and show how to obtain its facially reduced equivalent version by using properties of the bisection polytope. As a byproduct, we also study properties of the feasible set of the DNN relaxation, see Theorem 4.4.

Let $\mathbf{m}:=\left[\begin{array}{ll}m_{1} & m_{2}\end{array}\right]^{\top}$ such that $m_{1}+m_{2}=n$ be given. To derive a vector lifting SDP relaxation for the GBP we linearize the objective from $(G P P)$ by lifting variables into $\mathcal{S}^{2 n+1}$. In particular, let $P \in\{0,1\}^{n \times 2}$ be a partition matrix and $x=\operatorname{vec}(P)$. We use the properties
of the Kronecker product and the trace to rewrite the objective as follows:

$$
\operatorname{tr}\left(L P P^{\top}\right)=\operatorname{vec}(P)^{\top}\left(\mathbf{I}_{2} \otimes L\right) \operatorname{vec}(P)=x^{\top}\left(\mathbf{I}_{2} \otimes L\right) x=\left\langle\mathbf{I}_{2} \otimes L, x x^{\top}\right\rangle
$$

Now, we replace $x x^{\top}$ by a matrix variable $\hat{X} \in \mathcal{S}^{2 n}$. The constraint $\hat{X}=x x^{\top}$ can be weakened to $\hat{X}-x x^{\top} \succeq \mathbf{0}$, which is equivalent to $X:=\left(\begin{array}{cc}1 & x^{\top} \\ x & \hat{X}\end{array}\right) \succeq \mathbf{0}$ by the well-known Schur complement lemma.

In the sequel, we use the following block notation for matrices in $\mathcal{S}^{2 n+1}$ :

$$
X=\left(\begin{array}{ccc}
1 & \left(x^{1}\right)^{\top} & \left(x^{2}\right)^{\top} \\
x^{1} & X^{11} & X^{12} \\
x^{2} & X^{21} & X^{22}
\end{array}\right)
$$

where $x^{1}$ (resp., $x^{2}$ ) corresponds to the first (resp., second) column in $P$, and $X^{i j}$ corresponds to $x^{i}\left(x^{j}\right)^{\top}$ for $i, j=1,2$.

Now, from $\mathbf{1}_{n}^{\top} x^{i}=m_{i}$ for $i=1,2$, it follows that $\operatorname{tr}\left(X^{i i}\right)=m_{i}, \operatorname{tr}\left(\mathbf{J}_{n} X^{i i}\right)=m_{i}^{2}$ and $\operatorname{tr}\left(\mathbf{J}_{n}\left(X^{12}+X^{21}\right)\right)=2 m_{1} m_{2}$. From $x^{1} \circ x^{2}=\mathbf{0}$ it follows that $\operatorname{diag}\left(X^{12}\right)=\mathbf{0}$.

The above derivation results in the following vector lifting SDP relaxation for the GBP [371]:

$$
\begin{align*}
& \min _{X} \frac{1}{2}\left\langle L, X^{11}+X^{22}\right\rangle \\
& \text { s.t. } \operatorname{tr}\left(X^{i i}\right)=m_{i}, \operatorname{tr}\left(\mathbf{J}_{n} X^{i i}\right)=m_{i}^{2} \quad \forall i \in[2] \\
& \quad \operatorname{diag}\left(X^{12}\right)=\mathbf{0}, \operatorname{tr}\left(\mathbf{J}_{n}\left(X^{12}+X^{21}\right)\right)=2 m_{1} m_{2}  \tag{4.12}\\
& \quad X=\left(\begin{array}{ccc}
1 & \left(x^{1}\right)^{\top} & \left(x^{2}\right)^{\top} \\
x^{1} & X^{11} & X^{12} \\
x^{2} & X^{21} & X^{22}
\end{array}\right) \succeq \mathbf{0}, \quad x^{i}=\operatorname{diag}\left(X^{i i}\right) \quad \forall i \in[2],
\end{align*}
$$

where $X \in \mathcal{S}^{2 n+1}$. By imposing nonnegativity constraints on the matrix variable in $\left(S D P_{B P}\right)$, we obtain the following DNN relaxation:
$\left(D N N_{B P}\right) \quad\left(S D P_{B P}\right) \& X \geq \mathbf{0}$.
The relaxation ( $D N N_{B P}$ ) can be further strengthened by additional cutting planes. We propose adding the boolean quadric polytope inequalities, see Section 4.4.3.

The zero pattern on off-diagonal blocks in (4.12) can be written using a linear operator $\mathcal{G}_{\mathcal{J}}(\cdot)$, known as the Gangster operator, see [371]. The operator $\mathcal{G}_{\mathcal{J}}: \mathcal{S}^{2 n+1} \rightarrow \mathcal{S}^{2 n+1}$ is defined as

$$
\mathcal{G}_{\mathcal{J}}(X)= \begin{cases}X_{i j} & \text { if }(i, j) \in \mathcal{J} \\ 0 & \text { otherwise }\end{cases}
$$

where

$$
\mathcal{J}:=\left\{(i, j): \begin{array}{r}
i=1+(p-1) n+q, j=1+(r-1) n+q  \tag{4.14}\\
q \in[n], p, r \in\{1,2\}, p \neq r
\end{array}\right\} .
$$

The constraints $\operatorname{diag}\left(X^{12}\right)=\operatorname{diag}\left(X^{21}\right)=\mathbf{0}$ are given by $\mathcal{G}_{\mathcal{J}}(X)=\mathbf{0}$.
We now show how to project the SDP relaxation (4.12) onto a smaller dimensional space in order to obtain an equivalent strictly feasible relaxation by facial reduction. Although such reduction is performed for the general graph partitioning problem in [371], our approach differs by relying on the polytope of all bisections. We first apply facial reduction to the relaxation $\left(S D P_{B P}\right)$, after which we derive the facially reduced equivalent of $\left(D N N_{B P}\right)$.

We start by deriving two properties that hold for all feasible solutions of $\left(S D P_{B P}\right)$.
Theorem 4.4. Let $X=\left(\begin{array}{ccc}1 & \left(x^{1}\right)^{\top} & \left(x^{2}\right)^{\top} \\ x^{1} & X^{11} & X^{12} \\ x^{2} & X^{21} & X^{22}\end{array}\right)$ with $\hat{X}=\left(\begin{array}{ll}X^{11} & X^{12} \\ X^{21} & X^{22}\end{array}\right)$ and $x=\binom{x^{1}}{x^{2}}$ be feasible for $\left(S D P_{B P}\right)$. Then,
(i) $a_{i}^{\top}\left(\hat{X}-x x^{\top}\right) a_{i}=0$ where $a_{i}=\mathbf{e}_{i} \otimes \mathbf{1}_{n}, \mathbf{e}_{i} \in \mathbb{R}^{2}, i \in[2]$;
(ii) $b_{i}^{\top}\left(\hat{X}-x x^{\top}\right) b_{i}=0$ where $b_{i}=\mathbf{1}_{2} \otimes \mathbf{e}_{i}, \mathbf{e}_{i} \in \mathbb{R}^{n}, i \in[n]$.

Proof. ( $i$ ) Without loss of generality we take $i=1$. Then $a_{1}=\mathbf{e}_{1} \otimes \mathbf{1}_{n}$, which yields

$$
a_{1}^{\top}\left(\hat{X}-x x^{\top}\right) a_{1}=\mathbf{1}_{n}^{\top} X^{11} \mathbf{1}_{n}-\mathbf{1}_{n}^{\top} x^{1}\left(x^{1}\right)^{\top} \mathbf{1}_{n}=\operatorname{tr}\left(\mathbf{J}_{n} X^{11}\right)-\operatorname{tr}\left(X^{11}\right)^{2}=m_{1}^{2}-m_{1}^{2}=0
$$

using the constraints of (4.12). The proof for $i=2$ is similar.
(ii) We first show that any feasible solution to (4.12) satisfies $\operatorname{diag}\left(X^{11}\right)+\operatorname{diag}\left(X^{22}\right)=\mathbf{1}_{n}$. For all $i \in[n]$ we define $v^{i} \in \mathbb{R}^{2 n}$ as

$$
\left(v^{i}\right)_{j}:= \begin{cases}-1 & \text { if } j=i \text { or } j=n+i \\ 0 & \text { otherwise }\end{cases}
$$

From $\hat{X}-x x^{\top} \succeq \mathbf{0}$, we have

$$
\binom{1}{v^{i}}^{\top}\left(\begin{array}{cc}
1 & x^{\top} \\
x & \hat{X}
\end{array}\right)\binom{1}{v^{i}} \geq 0 \quad \text { implying } \quad X_{i i}^{11}+X_{i i}^{22} \leq 1
$$

where we used the fact that $\operatorname{diag}(\hat{X})=x$. Since

$$
n=m_{1}+m_{2}=\operatorname{tr}\left(X^{11}\right)+\operatorname{tr}\left(X^{22}\right)=\sum_{i=1}^{n}\left(X_{i i}^{11}+X_{i i}^{22}\right)
$$

and the latter summation consists of $n$ elements bounded above by one, it follows that for all $i \in[n]$ we have $X_{i i}^{11}+X_{i i}^{22}=1$.

Now, for $b_{i}=\mathbf{1}_{2} \otimes \mathbf{e}_{i}, i \in[n]$, we have

$$
b_{i}^{\top}\left(\hat{X}-x x^{\top}\right) b_{i}=X_{i i}^{11}+X_{i i}^{12}+X_{i i}^{21}+X_{i i}^{22}-\left(X_{i i}^{11}+X_{i i}^{22}\right)^{2}
$$

Since $\operatorname{diag}\left(X^{12}\right)=\operatorname{diag}\left(X^{21}\right)=\mathbf{0}$ and $\operatorname{diag}\left(X^{11}\right)+\operatorname{diag}\left(X^{22}\right)=\mathbf{1}_{n}$, we conclude that for all $i \in[n]$ we have $b_{i}^{\top}\left(\hat{X}-x x^{\top}\right) b_{i}=1-1^{2}=0$.

We can exploit the properties stated in Theorem 4.4 to identify vectors in the null space of all feasible solutions of $\left(S D P_{B P}\right)$. To do so, we use the following basic result, see e.g., [319].

Lemma 4.5. Let $X \in \mathcal{S}^{l}, x \in \mathbb{R}^{l}$ and $a \in \mathbb{R}^{l}$ be such that $X-x x^{\top} \succeq \mathbf{0}, a^{\top} x=t$ for some $t \in \mathbb{R}$, and $a^{\top}\left(X-x x^{\top}\right) a=0$. Then $\left[-t a^{\top}\right]^{\top}$ is an eigenvector of $\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right)$ with respect to eigenvalue 0.

It follows from the constraints of (4.12) that $a_{i}^{\top} x=m_{i}$ for $i \in[2]$ and $b_{i}^{\top} x=1$ for $i \in[n]$, where $a_{i}$ and $b_{i}$ are defined as in Theorem 4.4. As a result, Theorem 4.4 and Lemma 4.5 imply that

$$
\begin{equation*}
\binom{-m_{i}}{\mathbf{e}_{i} \otimes \mathbf{1}_{n}}, i \in[2], \quad \text { and } \quad\binom{-1}{\mathbf{1}_{2} \otimes \mathbf{e}_{i}}, i \in[n] \tag{4.15}
\end{equation*}
$$

are eigenvectors of $\left(\begin{array}{cc}1 & x^{\top} \\ x & \hat{X}\end{array}\right)$ with respect to eigenvalue 0. Now, let the vectors in (4.15) define the rows of a matrix $T \in \mathbb{R}^{(n+2) \times(2 n+1)}$, i.e.,

$$
T:=\left(\begin{array}{cc}
-\mathbf{m} & \mathbf{I}_{2} \otimes \mathbf{1}_{n}^{\top} \\
-\mathbf{1}_{n} & \mathbf{1}_{2}^{\top} \otimes \mathbf{I}_{n}
\end{array}\right)
$$

Moreover, let $\mathcal{V}=\operatorname{Nul}(T)$. Any $a \in \mathcal{V}$ defines an element $a a^{\top}$ exposing the feasible set of $\left(S D P_{B P}\right)$. It follows from the facial geometry of the cone of positive semidefinite matrices that the feasible set of $\left(S D P_{B P}\right)$ is contained in

$$
S_{\mathcal{V}}:=\left\{X \in \mathcal{S}_{+}^{2 n+1}: \operatorname{Col}(X) \subseteq \mathcal{V}\right\}
$$

which is a face of $\mathcal{S}_{+}^{2 n+1}$. It remains to prove that this is actually the minimal face of $\mathcal{S}_{+}^{2 n+1}$ containing the feasible set of $\left(S D P_{B P}\right)$. For that purpose, we consider the underlying bisection polytope $\operatorname{conv}\left(F_{n}^{2}(\mathbf{m})\right)$, see (4.4). Theorem 4.1 implies $\operatorname{dim}\left(\operatorname{conv}\left(F_{n}^{2}(\mathbf{m})\right)\right)=n-1$. Besides, observe that $T$ is constructed as the constraint matrix defining conv $\left(F_{n}^{2}(\mathbf{m})\right)$ augmented with an additional column. Since this additional column does not increase its rank, we have $\operatorname{rank}(T)=n+1$, which implies that $\operatorname{dim}(\mathcal{V})=n$. Let $V \in \mathbb{R}^{(2 n+1) \times n}$ be a matrix whose columns form a basis for $\mathcal{V}$. Then the face $S_{\mathcal{V}}$ can be equivalently written as

$$
\begin{equation*}
S_{\mathcal{V}}=V \mathcal{S}_{+}^{n} V^{\top} \tag{4.16}
\end{equation*}
$$

To show that $S_{\mathcal{V}}$ is the minimal face containing the feasible set of $\left(S D P_{B P}\right)$ we apply a result by Tunçel [355].
Theorem $4.6([355])$. Given $F \subseteq \mathbb{R}^{l}$, let $\mathcal{F}:=\left\{\left(\begin{array}{cc}1 & x^{\top} \\ x & \hat{X}\end{array}\right) \in \mathcal{S}_{+}^{l+1}: \mathcal{A}\left(\left(\begin{array}{cc}1 & x^{\top} \\ x & \hat{X}\end{array}\right)\right)=\mathbf{0}\right\}$, with $\mathcal{A}: \mathcal{S}^{l+1} \rightarrow \mathbb{R}^{p}$ a linear transformation, be a relaxation of the lifted polyhedron

$$
\operatorname{conv}\left(\left\{\binom{1}{x}\binom{1}{x}^{\top}: x \in F\right\}\right)
$$

Suppose that $\mathcal{F} \subseteq V \mathcal{S}_{+}^{d} V^{\top}$ for some full-rank matrix $V \in \mathbb{R}^{(l+1) \times d}$. If $\operatorname{dim}(\operatorname{conv}(F))=d-1$, then there exists some $R \succ \mathbf{0}$ such that $V R V^{\top} \in \mathcal{F}$.

Chapter 4. Partitioning through projections: strong SDP bounds for large graph partition problems

Based on Theorem 4.6, we can now show the minimality of the face $S_{\mathcal{V}}$ for both $\left(S D P_{B P}\right)$ and $\left(D N N_{B P}\right)$.

Theorem 4.7. The set $S_{\mathcal{V}}$ is the minimal face of $\mathcal{S}_{+}^{2 n+1}$ containing the feasible set of $\left(S D P_{B P}\right)$. If $m_{1}, m_{2} \geq 2$, then $S_{\mathcal{V}}$ is also the minimal face of $\mathcal{S}_{+}^{2 n+1}$ containing the feasible set of $\left(D N N_{B P}\right)$.

Proof. The feasible region of $\left(S D P_{B P}\right)$ can be written in the form of $\mathcal{F}$ in the statement of Theorem 4.6. To show minimality for $\left(S D P_{B P}\right)$, it suffices to show that there exists a matrix $R \in \mathcal{S}_{+}^{n}, R \succ \mathbf{0}$ such that $V R V^{\top}$ is feasible for $\left(S D P_{B P}\right)$. As $\operatorname{dim}\left(\operatorname{conv}\left(F_{n}^{2}(\mathbf{m})\right)\right)=n-1$, it immediately follows from Theorem 4.6 that such matrix, say $R_{1}$, exists.

To prove the second statement, it suffices to show that there exists an $R \in \mathcal{S}_{+}^{n}$ such that $R \succ \mathbf{0}$ and $\left(V R V^{\top}\right)_{i j}>0$ for all $(i, j) \in \mathcal{J}_{C}$, where $\mathcal{J}_{C}=([2 n+1] \times[2 n+1]) \backslash \mathcal{J}$, see (4.14). Since $m_{1}, m_{2} \geq 2$, it follows that for any $(i, j) \in \mathcal{J}_{C}$ there exists a bisection $x^{i j}$ such that

$$
\left(\binom{1}{x^{i j}}\binom{1}{x^{i j}}^{\top}\right)_{i j}>0
$$

Let $R^{i j} \in \mathcal{S}_{+}^{n}$ denote the matrix such that $V R^{i j} V^{\top}=\left(\begin{array}{c}{ }_{x}^{i j}\end{array}\right)\left({ }_{x^{i j}}\right)^{\top}$, which consists by construction of $S_{\mathcal{V}}$. Now, let $R_{2}$ be any positive convex combination of the elements in $\left\{R^{i j}:(i, j) \in \mathcal{J}_{C}\right\}$. By construction, $R_{2} \succeq \mathbf{0}$, while $\left(V R_{2} V^{\top}\right)_{i j}>0$ for all $(i, j) \in \mathcal{J}_{C}$. Finally, any positive convex combination of $R_{1}$ and $R_{2}$ provides a matrix $R$ with the desired properties.

The result of Theorem 4.7 can be exploited to derive strictly feasible equivalent versions of $\left(S D P_{B P}\right)$ and $\left(D N N_{B P}\right)$. We focus here only on the DNN relaxation $\left(D N N_{B P}\right)$. Theorem 4.7 allows us to replace $X$ by $V R V^{\top}$ in $\left(D N N_{B P}\right)$, where we can take

$$
V:=\left(\begin{array}{c|c}
1 & \mathbf{0}_{n-1}^{\top}  \tag{4.17}\\
\hline m_{1} / n & \\
\vdots & \mathbf{I}_{n-1} \\
m_{1} / n & -\mathbf{1}_{n-1}^{\top} \\
\hline m_{2} / n & \\
\vdots & -\mathbf{I}_{n-1} \\
m_{2} / n & \mathbf{1}_{n-1}^{\top}
\end{array}\right) .
$$

Because of the structure of $V$, most of the constraints in $\left(D N N_{B P}\right)$ become redundant. One can easily verify that the resulting relaxation in lower dimensional space is as follows, see e.g., [371]:

$$
\begin{array}{cl}
\min _{R} & \operatorname{tr}\left(L_{B P} V R V^{\top}\right) \\
\text { s.t. } & \mathcal{G}_{\mathcal{J}}\left(V R V^{\top}\right)=\mathbf{0}  \tag{4.18}\\
& \left(V R V^{\top}\right)_{1,1}=1 \\
& V R V^{\top} \geq \mathbf{0}, R \in \mathcal{S}_{+}^{n},
\end{array}
$$

where

$$
L_{B P}:=\frac{1}{2}\left(\begin{array}{cc}
0 & \mathbf{0}^{\top}  \tag{4.19}\\
\mathbf{0} & \mathbf{I}_{2} \otimes L
\end{array}\right)
$$

and $L$ is the weighted Laplacian matrix of $G$. Let us now define the following sets:

$$
\begin{align*}
\mathcal{R}_{B P} & :=\left\{R \in \mathcal{S}^{n}: R \succeq \mathbf{0}\right\}  \tag{4.20}\\
\mathcal{X}_{B P} & :=\left\{\begin{array}{ll} 
& \mathcal{G}_{\mathcal{J}}(X)=\mathbf{0}, X_{1,1}=1, \operatorname{tr}\left(X^{i i}\right)=m_{i} \quad \forall i \in[2] \\
X \in \mathcal{S}^{2 n+1}: \begin{array}{l}
\operatorname{diag}\left(X^{11}\right)+\operatorname{diag}\left(X^{22}\right)=\mathbf{1}_{n}, X \mathbf{e}_{1}=\operatorname{diag}(X) \\
\mathbf{0} \leq X \leq \mathbf{J}
\end{array}
\end{array} .\right. \tag{4.21}
\end{align*}
$$

Now, we are ready to rewrite the facially reduced DNN relaxation (4.18) as follows:

$$
\begin{equation*}
\min \left\{\left\langle L_{B P}, X\right\rangle: X=V R V^{\top}, R \in \mathcal{R}_{B P}, X \in \mathcal{X}_{B P}\right\} \tag{4.22}
\end{equation*}
$$

Note that $\mathcal{X}_{B P}$ also contains constraints that are redundant for (4.18).

### 4.3 A cutting-plane augmented Lagrangian algorithm

SDP has proven effective for modeling optimization problems and providing strong bounds. It is well-known that SDP solvers based on interior-point methods might have considerable memory demands already for medium-scale problems. Recently, promising alternatives for solving large-scale SDP relaxations have been investigated. We refer the interested reader to [61, 260, 308, 350, 363, 377] for algorithms based on alternating direction augmented Lagrangian methods for solving SDPs. For efficient approaches to solving DNN relaxations, see also e.g., [216, 218, 252, 292, 365, 380]. To the best of our knowledge only [275] incorporates an augmented Lagrangian method into a cutting-plane framework. The authors of [275] consider only one type of cutting planes. Here, we incorporate various types of cutting planes into one framework and use a more efficient version of the ADMM than the one used in [275].

In Section 4.3 .1 we describe variants of the ADMM that are used within our cuttingplane algorithm. Section 4.3 .2 presents Dykstra's cyclic projection algorithm that is used for projections onto polyhedra induced by the violated cuts. Section 4.3 .3 presents our cutting-plane augmented Lagrangian algorithm.

### 4.3.1 The alternating direction method of multipliers

The ADMM is a first-order method from the 1970s that is developed for solving convex optimization problems. This method decomposes an optimization problem into several subproblems that are easier to solve than the original problem. There exist several variants of the ADMM for solving SDPs. We consider here a variant of the ADMM that resembles variants from [216, 292], where we additionally consider an adaptive stepsize term proposed by Lorenz and Tran-Dinh [258] when solving the $k$-EP.

In order to describe the ADMM scheme for solving SDP relaxations for both problems, the $k$-equipartition problem (4.11) and the graph bisection problem (4.22), we introduce the following unified notation: For the $k$-EP, define $\bar{L}:=L_{E P}, \mathcal{R}:=\mathcal{R}_{E P}$ and $\mathcal{X}:=\mathcal{X}_{E P}$ (see

Chapter 4. Partitioning through projections: strong SDP bounds for large graph partition problems
resp., (4.8), (4.9), (4.10)), and for the GBP define $\bar{L}:=L_{B P}, \mathcal{R}:=\mathcal{R}_{B P}$ and $\mathcal{X}:=\mathcal{X}_{B P}$ (see resp., (4.19), (4.20), (4.21)).

Let $S$ denote the Lagrange multiplier for the constraint $X=V R V^{\top}$. Then, the augmented Lagrangian function of (4.11) and (4.22) w.r.t. the constraint $X=V R V^{\top}$ for a penalty parameter $\sigma$ is as follows:

$$
\begin{equation*}
\mathcal{L}_{\sigma}(X, R, S)=\langle\bar{L}, X\rangle+\left\langle S, X-V R V^{\top}\right\rangle+\frac{\sigma}{2}\left\|X-V R V^{\top}\right\|_{F}^{2} \tag{4.23}
\end{equation*}
$$

In each iteration, the ADMM minimizes $\mathcal{L}_{\sigma}(X, R, S)$ subject to $X \in \mathcal{X}$ and $R \in \mathcal{R}$ and updates $S$ via a stepsize update. The ADMM update scheme requires a matrix $V$ that has orthonormal columns that can be obtained by applying a QR-decomposition to (4.6) for the $k$-EP and to (4.17) for the GBP. Thus, from now on we assume that $V^{\top} V=\mathbf{I}$.

Let $\left(R^{p}, X^{p}, S^{p}\right)$ denote the $p$ th iterate of the ADMM. The next iterate $\left(R^{p+1}, X^{p+1}, S^{p+1}\right)$ is obtained as follows:

$$
\begin{align*}
R^{p+1} & :=\underset{R \in \mathcal{R}}{\arg \min } \mathcal{L}_{\sigma^{p}}\left(R, X^{p}, S^{p}\right)  \tag{4.24a}\\
X^{p+1} & :=\underset{X \in \mathcal{X}}{\arg \min } \mathcal{L}_{\sigma^{p}}\left(R^{p+1}, X, S^{p}\right)  \tag{4.24b}\\
S^{p+1} & :=S^{p}+\gamma \cdot \sigma^{p} \cdot\left(X^{p+1}-V R^{p+1} V^{\top}\right) \tag{4.24c}
\end{align*}
$$

where $\gamma \in\left(0, \frac{1+\sqrt{5}}{2}\right)$ is a parameter for updating the dual multiplier $S^{p}$, see e.g., [363].
There exist different ways for dealing with the stepsize term $\gamma \cdot \sigma^{p}$. One possibility is to keep $\sigma^{p}$ and $\gamma$ fixed during the algorithm. In this approach, $\sigma^{p}$ depends on the problem data and $\gamma$ has a value larger than one. This is known in the literature as the ADMM with larger stepsize, as originally proposed by [141]. An alternative is the ADMM with adaptive stepsize term as introduced in [258]. In that case $\gamma=1$ and the parameter $\sigma^{p}$ is updated as follows:

$$
\begin{equation*}
\sigma^{p+1}:=\left(1-\omega^{p+1}\right) \sigma^{p}+\omega^{p+1} \mathcal{P}_{\left[\sigma_{\min }, \sigma_{\max }\right]} \frac{\left\|S^{p+1}\right\|_{F}}{\left\|X^{p+1}\right\|_{F}} \tag{4.25}
\end{equation*}
$$

where $\omega^{p+1}:=2^{-p / 100}$ is the weight, $\sigma_{\min }$ and $\sigma_{\max }$ are the box bounds for $\sigma^{p}$, and $\mathcal{P}_{\left[\sigma_{\min }, \sigma_{\max }\right]}$ is the projection onto $\left[\sigma_{\min }, \sigma_{\max }\right.$ ].

Recall that we added redundant constraints for the SDP relaxations (4.11) and (4.22) to the set $\mathcal{X}$. Those constraints are, though, not redundant in the subproblem (4.24b). They are included to speed up the convergence of the ADMM in practice, see e.g., [216, 275, 292].

One can solve the $R$-subproblem (4.24a) as follows:

$$
\begin{aligned}
R^{p+1} & =\underset{R \in \mathcal{R}}{\arg \min } \mathcal{L}_{\sigma^{p}}\left(R, X^{p}, S^{p}\right)=\underset{R \in \mathcal{R}}{\arg \min }\left\langle S^{p},-V R V^{\top}\right\rangle+\frac{\sigma^{p}}{2}\left\|X^{p}-V R V^{\top}\right\|_{F}^{2} \\
& =\underset{R \in \mathcal{R}}{\arg \min }\left\|V^{\top}\left(X^{p}+\frac{1}{\sigma^{p}} S^{p}\right) V-R\right\|_{F}^{2}=\mathcal{P} \succeq \mathbf{0}\left(V^{\top}\left(X^{p}+\frac{1}{\sigma^{p}} S^{p}\right) V\right)
\end{aligned}
$$

where $\mathcal{P}_{\succeq \mathbf{0}}(\cdot)$ denotes the orthogonal projection onto the cone of positive semidefinite matrices.
The $X$-subproblem (4.24b) can be solved as follows:

$$
X^{p+1}=\underset{X \in \mathcal{X}}{\arg \min } \mathcal{L}_{\sigma^{p}}\left(R^{p+1}, X, S^{p}\right)=\underset{X \in \mathcal{X}}{\arg \min }\left\langle\bar{L}+S^{p}, X\right\rangle+\frac{\sigma^{p}}{2}\left\|X-V R^{p+1} V^{\top}\right\|_{F}^{2},
$$

$$
=\underset{X \in \mathcal{X}}{\arg \min }\left\|X-\left(V R^{p+1} V^{\top}-\frac{1}{\sigma^{p}}\left(\bar{L}+S^{p}\right)\right)\right\|_{F}^{2}=\mathcal{P}_{\mathcal{X}}\left(V R^{p+1} V^{\top}-\frac{1}{\sigma^{p}}\left(\bar{L}+S^{p}\right)\right)
$$

where $\mathcal{P} \mathcal{X}(\cdot)$ denotes the orthogonal projection onto the polyhedral set $\mathcal{X}$, where $\mathcal{X}$ is given in (4.10) and (4.21) for the $k$-EP and GBP, respectively. We below show how this projection can be performed for the GBP. The projector for the $k$-EP, which has a simpler structure, can be obtained similarly.

Recall from (4.21) that the polyhedral set $\mathcal{X}_{B P}$ looks as follows:

$$
\mathcal{X}_{B P}=\left\{X=\left(\begin{array}{ccc}
1 & \left(x^{1}\right)^{\top} & \left(x^{2}\right)^{\top} \\
x^{1} & X^{11} & X^{12} \\
x^{2} & X^{21} & X^{22}
\end{array}\right) \in \mathcal{S}^{2 n+1}: \begin{array}{l}
\mathcal{G}_{\mathcal{J}}(X)=\mathbf{0}, \operatorname{tr}\left(X^{i i}\right)=m_{i} \quad \operatorname{diag}\left(X^{11}\right)+\operatorname{diag}\left(X^{22}\right)=\mathbf{1}_{n} \\
\mathbf{0} \leq X \leq \mathbf{J}, \quad X \mathbf{e}_{1}=\operatorname{diag}(X)
\end{array}\right\}
$$

Let $\mathcal{P}_{\mathcal{X}_{B P}}: \mathcal{S}^{2 n+1} \rightarrow \mathcal{S}^{2 n+1}$ denote the projection onto $\mathcal{X}_{B P}$.

Observe that each constraint that defines $\mathcal{X}_{B P}$ either acts on the diagonal, first row, and first column of the matrix, or on the remaining entries. In the latter case, an entry $X_{i j}$ is either bounded by 0 and 1 or equals 0 if $(i, j) \in \mathcal{J}$. These projections are very simple and are given by the operators $T_{\text {inner }}$ and $T_{\text {box }}$ in Table 4.1.

Next, we focus on the entries on the diagonal, first row, and first column of the orthogonal projection. Suppose $Y=\mathcal{P}_{\mathcal{X}_{B P}}(X)$ and let $y_{1}:=\operatorname{diag}\left(Y^{11}\right)$ and $y_{2}:=\operatorname{diag}\left(Y^{22}\right)$. Then $y_{1}$ and $y_{2}$ can be obtained via the following optimization problem:

$$
\begin{align*}
\min _{y_{1}, y_{2} \in \mathbb{R}^{n}} & \left(y_{1}-\operatorname{diag}\left(X^{11}\right)\right)^{\top}\left(y_{1}-\operatorname{diag}\left(X^{11}\right)\right)+2\left(y_{1}-x^{1}\right)^{\top}\left(y_{1}-x^{1}\right) \\
& +\left(y_{2}-\operatorname{diag}\left(X^{22}\right)\right)^{\top}\left(y_{2}-\operatorname{diag}\left(X^{22}\right)\right)+2\left(y_{2}-x^{2}\right)^{\top}\left(y_{2}-x^{2}\right)  \tag{4.26}\\
\text { s.t. } & \mathbf{1}_{n}^{\top} y_{1}=m_{1}, \mathbf{1}_{n}^{\top} y_{2}=m_{2}, y_{1}+y_{2}=\mathbf{1}_{n}, y_{1} \geq \mathbf{0}_{n}, y_{2} \geq \mathbf{0}_{n}
\end{align*}
$$

Using basic algebra, one can show that the optimal $y_{1}$ to $(4.26)$ is attained by the minimizer of the following optimization problem:

$$
\begin{align*}
\min _{y_{1} \in \mathbb{R}^{n}} & \left\|y_{1}-\left(\frac{1}{6}\left(\operatorname{diag}\left(X^{11}\right)-\operatorname{diag}\left(X^{22}\right)\right)+\frac{1}{3}\left(x^{1}-x^{2}\right)+\frac{1}{2} \mathbf{1}_{n}\right)\right\|_{2}^{2}  \tag{4.27}\\
\text { s.t. } & \mathbf{1}_{n}^{\top} y_{1}=m_{1}, \quad \mathbf{0}_{n} \leq y_{1} \leq \mathbf{1}_{n}
\end{align*}
$$

while the corresponding optimal $y_{2}$ to (4.26) is $y_{2}=\mathbf{1}_{n}-y_{1}$. Observe that (4.27) is equivalent to a projection onto the capped simplex $\bar{\Delta}\left(m_{1}\right)=\left\{y \in \mathbb{R}^{n}: \mathbf{1}_{n}^{\top} y=m_{1}, \mathbf{0}_{n} \leq y \leq \mathbf{1}_{n}\right\}$. The projection onto $\bar{\Delta}\left(m_{1}\right)$ we denote by $\mathcal{P}_{\bar{\Delta}\left(m_{1}\right)}: \mathbb{R}^{n} \rightarrow \mathbb{R}^{n}$, which can be performed efficiently, see [13]. We define the operator $T_{\text {arrow }}$, see Table 4.1, to embed the optimal $y_{1}$ and $y_{2}$ in the space $\mathcal{S}^{2 n+1}$.

Chapter 4. Partitioning through projections: strong SDP bounds for large graph partition problems

|  | Operator | Description |
| :---: | :---: | :---: |
| $T_{\text {inner }}$ | $\mathcal{S}^{2 n+1} \rightarrow \mathcal{S}^{2 n+1}$ | $T_{\text {inner }}(X)_{i j}=0$ if $i=1$ or $j=1$ or $i=j$ or $(i, j) \in \mathcal{J}$ and $T_{\text {inner }}(X)_{i j}=X_{i j}$ otherwise. |
| $T_{\text {box }}$ | $\mathcal{S}^{2 n+1} \rightarrow \mathcal{S}^{2 n+1}$ | $T_{\text {box }}(X)_{i j}=\min \left(\max \left(X_{i j}, 0\right), 1\right)$ for all $(i, j)$. |
| $T_{\text {arrow }}$ | $: \quad \mathbb{R}^{n} \rightarrow \mathcal{S}^{2 n+1}$ | $T_{\text {arrow }}(x)=\left(\begin{array}{ccc}1 & x^{\top} & \left(\mathbf{1}_{n}-x\right)^{\top} \\ x & \operatorname{Diag}(x) & \mathbf{0} \\ \mathbf{1}_{n}-x & \mathbf{0} & \operatorname{Diag}\left(\mathbf{1}_{n}-x\right)\end{array}\right)$ |

Table 4.1: Overview of operators and their definitions.

Now, the projector $\mathcal{P}_{\mathcal{X}_{B P}}$ can be written out explicitly as follows:

$$
\begin{aligned}
\mathcal{P}_{\mathcal{X}_{B P}}\left(\left(\begin{array}{ccc}
x^{0} & \left(x^{1}\right)^{\top} & \left(x^{2}\right)^{\top} \\
x^{1} & X^{11} & X^{12} \\
x^{2} & X^{21} & X^{22}
\end{array}\right)\right)=T_{\text {box }}\left(T_{\text {inner }}\left(\left(\begin{array}{ccc}
x^{0} & \left(x^{1}\right)^{\top} & \left(x^{2}\right)^{\top} \\
x^{1} & X^{11} & X^{12} \\
x^{2} & X^{21} & X^{22}
\end{array}\right)\right)\right) \\
+T_{\text {arrow }}\left(\mathcal{P}_{\bar{\Delta}\left(m_{1}\right)}\left(\frac{1}{6}\left(\operatorname{diag}\left(X^{11}\right)-\operatorname{diag}\left(X^{22}\right)\right)+\frac{1}{3}\left(x^{1}-x^{2}\right)+\frac{1}{2} \mathbf{1}_{n}\right)\right) .
\end{aligned}
$$

The performance of the ADMM greatly depends on the stepsize term. Our preliminary tests show that for the $k$-EP the updating scheme (4.24)-(4.25) with adaptive stepsize term outperforms the ADMM with larger stepsize. That is, our adaptive ADMM performs better than the ADMM variants from [216, 218, 292]. Moreover, preliminary results show that it outperforms the algorithm from [252]. For the GBP, however, our preliminary tests show that it is more beneficial to keep $\sigma^{p}$ fixed and use larger $\gamma$. The resulting version of the ADMM resembles versions from [216, 218, 292]. Consequently, we initialize the ADMM (4.24) for the $k$-EP by

$$
\begin{equation*}
R^{0}=\mathbf{0}, \quad X^{0}=\frac{k-1}{k} \mathbf{I}_{n}, \quad S^{0}=\mathbf{0}, \quad \sigma^{0}=\left\lceil\frac{n}{k}\right\rceil, \quad \gamma=1 \tag{4.28}
\end{equation*}
$$

and for the GBP we set

$$
\begin{equation*}
R^{0}=\mathbf{0}, \quad X^{0}=\mathbf{e}_{1} \mathbf{e}_{1}^{\top}, \quad S^{0}=\mathbf{0}, \quad \sigma^{0}=\left\lceil\left(\frac{2 n}{m_{1}}\right)^{2}\right\rceil, \quad \gamma=1.608 \tag{4.29}
\end{equation*}
$$

### 4.3.2 Clustered Dykstra's projection algorithm

Both DNN problems, (4.11) and (4.22), can be strengthened by adding valid cutting planes, see Section 4.4. Since these cutting planes are polyhedral, it is natural to include additional cuts to the set $\mathcal{X}$. This addition will, however, spoil the easy structure of $\mathcal{X}$. As a result, finding the explicit projection onto this new polyhedral set becomes a difficult task, even after the addition of a single cut. In [275] this issue is resolved by splitting the polyhedral set into subsets and using iterative projections based on Dykstra's algorithm [55, 115]. This algorithm finds the projection onto the intersection of a finite number of polyhedral sets,
assuming that the projection onto each of the separate sets is known. Although there exist many algorithms for finding such projection in the literature, the recent study of [35] shows superior behaviour of Dykstra's cyclic projection algorithm.

In this section we briefly present Dykstra's algorithm and show how to implement it efficiently by clustering nonoverlapping cuts. Similar to the previous section, we present a generic version of the algorithm that can be applied to both the $k$-EP and the GBP.

Let us assume that $\mathcal{T}=\left(t_{i}\right)_{i=1}^{T}$ is an ordered index set of $T$ cutting planes on the primal variable $X$ in the ADMM scheme, i.e., every $t_{i} \in \mathcal{T}$ corresponds to a single cut. Also, for each $t_{i} \in \mathcal{T}$ let $\mathcal{H}_{t_{i}}$ be a polyhedron that is induced by the cut $t_{i}$. One can think of $\mathcal{H}_{t_{i}}$ as the halfspace induced by the cut $t_{i}$, where additional constraints are added as long as the projection onto $\mathcal{H}_{t_{i}}$ remains efficient. In Section 4.4 we show how the set $\mathcal{T}$ and the polyhedra $\mathcal{H}_{t_{i}}$ look like for cuts related to both the $k$-EP and the GBP, and present the projectors onto the sets $\mathcal{H}_{t_{i}}$.

When adding the cuts in $\mathcal{T}$ to the relaxation, the polyhedral set $\mathcal{X}$ has to be replaced by

$$
\begin{equation*}
\mathcal{X}_{\mathcal{T}}:=\mathcal{X} \cap\left(\bigcap_{t_{i} \in \mathcal{T}} \mathcal{H}_{t_{i}}\right) \tag{4.30}
\end{equation*}
$$

The $X$-subproblem of the ADMM scheme $(4.24 \mathrm{~b})$ asks for the projection onto $\mathcal{X}_{\mathcal{T}}$. That is, for a given matrix $M$, one wants to solve the following best approximation problem:

$$
\begin{equation*}
\min _{\hat{M}}\|\hat{M}-M\|_{F}^{2} \quad \text { s.t. } \quad \hat{M} \in \mathcal{X}_{\mathcal{T}} \tag{4.31}
\end{equation*}
$$

Since the structure of $\mathcal{X}_{\mathcal{T}}$ is too complex to perform the projection in one step, the idea behind Dykstra's algorithm is to use iterative projections. Let $\mathcal{P}_{\mathcal{H}_{t_{i}}}(\cdot)$ denote the projection onto $\mathcal{H}_{t_{i}}$ for each $t_{i} \in \mathcal{T}$. Also, we assume that $\mathcal{P}_{\mathcal{X}}(\cdot)$ is known.

In Dykstra's algorithm we initialize the so-called normal matrices $N_{\mathcal{X}}^{0}=\mathbf{0}$ and $N_{t_{i}}^{0}=\mathbf{0}$ for all $t_{i} \in \mathcal{T}$. These normal matrices have the same size as the primal variable $X$ in the ADMM scheme. Moreover, we initialize $X^{0}=M$. For $q \geq 1$, the algorithm iteratively updates:

$$
\begin{aligned}
X^{q} & :=\mathcal{P}_{\mathcal{X}}\left(X^{q-1}+N_{\mathcal{X}}^{q-1}\right) \\
N_{\mathcal{X}}^{q} & :=X^{q-1}+N_{\mathcal{X}}^{q-1}-X^{q} \\
L_{t_{i}} & :=X^{q}+N_{t_{i}}^{q-1} \\
X^{q} & :=\mathcal{P}_{\mathcal{H}_{t_{i}}}\left(L_{t_{i}}\right) \\
N^{q} & :-V^{q} \quad \text { for } i=1, \ldots, T .
\end{aligned}
$$

Observe that the polyhedra $\mathcal{X}, \mathcal{H}_{t_{1}}, \ldots, \mathcal{H}_{t_{T}}$ are considered in a cyclic order. Therefore, the iterative scheme (CycDyk) is also known in the literature as Dykstra's cyclic projection algorithm. The sequence $\left(X^{q}\right)_{q \geq 1}$ strongly converges to the solution of the best approximation problem (4.31), see e.g., [47, 55, 150], and the convergence rate is known to be linear [96] in case the sets to be projected on are polyhedral.

We perform several actions to implement the algorithm (CycDyk) as efficiently as possible. First, we can reduce the number of iterations needed to converge by adding some of the constraints of $\mathcal{X}$ also to the sets $\mathcal{H}_{t_{i}}$. This brings the sets $\mathcal{H}_{t_{i}}$ closer to the intersection $\mathcal{X}_{\mathcal{T}}$, leading to faster convergence. A restriction on this addition is that we should still be
able to find the explicit projection onto $\mathcal{H}_{t_{i}}$. In Section 4.4.3 we show how some of the constraints from the DNN relaxation of the bisection problem are added to the polyhedra $\mathcal{H}_{t_{i}}$, while keeping the structure of the polyhedra sufficiently simple.

Second, as observed in [275], it is possible to partly parallelize the algorithm (CycDyk). The cuts in $\mathcal{T}$ are often very sparse. This implies that the projection onto $\mathcal{H}_{t_{i}}$ only involves a small number of entries, while the other entries are kept fixed. This property can be exploited by clustering nonoverlapping cuts. Two cuts $\left\langle A_{t_{1}}, X\right\rangle \leq b_{t_{1}}$ and $\left\langle A_{t_{2}}, X\right\rangle \leq b_{t_{2}}$ are called nonoverlapping if the matrices $A_{t_{1}}$ and $A_{t_{2}}$ have disjoint supports. When the cuts indexed by $t_{1}$ and $t_{2}$ are nonoverlapping, the projections onto $\mathcal{H}_{t_{1}}$ and $\mathcal{H}_{t_{2}}$ are independent and, hence, can be performed simultaneously. This idea can be generalized by creating clusters of nonoverlapping cuts. Suppose we cluster the set $\mathcal{T}$ into $r$ clusters $C_{i}, i \in[r]$ such that $C_{1} \cup \cdots \cup C_{r}=\mathcal{T}, C_{i} \cap C_{j}=\emptyset$ for $i \neq j, i, j \in[r]$, and all cuts in $C_{i}, i \in[r]$ are nonoverlapping. Then, an iterate of (CycDyk) is performed in $r+1$ consecutive steps, instead of $T+1$. Observe that this requires the set of clusters $\left(C_{i}\right)_{i=1}^{r}$ to be ordered.

To cluster the cuts, we proceed as follows. We denote by $H$ an undirected graph in which each vertex represents a cutting plane indexed by an element from $\mathcal{T}$. Two vertices in $H$ are connected by an edge if and only if two cuts are overlapping. Clustering $\mathcal{T}$ into nonoverlapping sets corresponds to clustering vertices of $H$ into independent sets. Therefore, clustering $\mathcal{T}$ into the smallest number of nonoverlapping sets reduces to finding a minimum coloring in $H$. Since the graph coloring problem is $\mathcal{N} \mathcal{P}$-hard, we use an efficient heuristic algorithm from [153] to find a near-optimal coloring.

### 4.3.3 The cutting-plane ADMM

In this section we put all elements of our cutting-plane algorithm together. In particular, we combine the ADMM from Section 4.3.1 and the clustered implementation of Dykstra's projection algorithm from Section 4.3.2 into a cutting-plane ADMM-based algorithm. We refer to this algorithm as the cutting-plane ADMM. Algorithm 4.1 provides a pseudo-code of our algorithm. Since the CP-ADMM solves a two-block separable convex problem, it is guaranteed to converge, see e.g., [53, 363]. Convergence of the (generalized) ADMM in case the subproblems are solved approximately is established in [118, Theorem 8]. The stopping criteria and input parameters are specified in Section 4.3.3.2 and Section 4.5, respectively.

The CP-ADMM is designed to solve DNN relaxations for the GPP with additional cutting planes. In particular, Algorithm 4.1 can solve the DNN relaxation for the $k$-EP, see (4.11), that is strengthened by the triangle inequalities (4.35) and independent set inequalities (4.38). Similarly, Algorithm 4.1 also solves the DNN relaxation for the GBP, see (4.22), with additional BQP inequalities (4.42).

Let us outline the main steps of the CP-ADMM. Initially, the ordered set $\mathcal{T}$ is empty and the algorithm solves the basic DNN relaxation, i.e., the DNN relaxation without additional cutting planes, using the ADMM as described in Section 4.3.1. After one of the stopping criteria from the inner while-loop is satisfied, see Section 4.3.3.2, a valid lower bound is computed based on the current approximate solution, see Section 4.3.3.1. Then, the algorithm identifies violated cuts and adds the numCuts most violated ones to $\mathcal{T}$ according to some (arbitrary) ordering. We also remove existing cuts from $\mathcal{T}$ in case they are no longer active. To increase performance, the cuts induced by tuples in $\mathcal{T}$ are clustered by using a heuristic for the graph coloring problem from [153]. The procedure is repeated, where the projection onto $\mathcal{X}_{\mathcal{T}}$, see (4.30), is performed by the semi-parallelized version of Dykstra's projection algorithm, see Section 4.3.2. The outer while-loop stops whenever one of the global stopping
criteria is met.
The CP-ADMM can be extended to solve various DNN relaxations with a large number of additional cutting planes. We remark that computing such strong bounds was not possible until now even for medium-sized problems and limited number of cutting planes.

```
Algorithm 4.1 CP-ADMM for the GPP
Input: weighted Laplacian matrix \(\bar{L}, m_{1} \geq \ldots \geq m_{k}, U B, \varepsilon_{\mathrm{ADMM}}, \varepsilon_{\mathrm{proj}}\), numCuts, maxOuterLoops,
    maxTime
    Initialization: Set \(\left(R^{0}, X^{0}, S^{0}\right), \sigma^{0}\) and \(\gamma\) by using (4.28) or (4.29). Set \(p=0, \mathcal{T}=\emptyset\).
    Obtain \(V\) by applying a QR-decomposition to (4.6) for the \(k\)-EP and to (4.17) for the GBP.
    while stopping criteria not met do
        while stopping criteria not met do
            \(R^{p+1}:=\mathcal{P}_{\succeq 0}\left(V^{\top}\left(X^{p}+\frac{1}{\sigma^{p}} S^{p}\right) V\right)\).
            \(X^{p+1}:=\mathcal{P}_{\mathcal{X}_{\mathcal{T}}}\left(V R^{p+1} V^{\top}-\frac{1}{\sigma^{p}}\left(\bar{L}+S^{p}\right)\right)\) by solving (4.31) using (CycDyk).
            \(S^{p+1}:=S^{p}+\gamma \cdot \sigma^{p} \cdot\left(X^{p+1}-V R^{p+1} V^{\top}\right)\).
            If adaptive stepsize term is used, update \(\sigma^{p+1}\) by using (4.25).
            \(p \leftarrow p+1\).
        end while
        Compute a valid lower bound \(L B\left(S^{p}\right)\) by using (4.33).
        Remove inactive cuts from \(\mathcal{T}\). Identify violated inequalities and add the numCuts most violated
        cuts to \(\mathcal{T}\) w.r.t. some ordering.
        Cluster the cuts in \(\mathcal{T}\).
    end while
Output: valid lower bound \(L B\left(S^{p}\right)\)
```


### 4.3.3.1 Valid lower bounds

There are several existing ways to obtain valid lower bounds when stopping iterative algorithms earlier, see e.g., [252, 292]. We compute valid lower bounds by exploiting the approach from [252].

We use our uniform notation for the $k$-EP and the GBP to derive the Lagrangian dual problem for both problems. Let $\mathcal{L}(X, R, S):=\mathcal{L}_{0}(X, R, S)$, see (4.23), denote the Lagrangian function of (4.11) and (4.22) with respect to dualizing $X=V R V^{\top}$. Moreover, we define the restricted set $\overline{\mathcal{R}}$ for both problems as follows:

$$
\overline{\mathcal{R}}:=\mathcal{R} \cap\{R: \operatorname{tr}(R)=\xi\}
$$

where $\xi=\frac{(k-1) n}{k}$ for the $k$-EP and $\xi=n+1$ for the GBP. Indeed, the trace constraints added to $\overline{\mathcal{R}}$ follow from (4.10) and (4.21), respectively. Replacing the constraint $R \in \mathcal{R}$ by $R \in \overline{\mathcal{R}}$ in (4.11) and (4.22) does not change the corresponding problems. We can, however, exploit it in the corresponding Lagrangian dual problem:

$$
\begin{equation*}
\max _{S \in \mathcal{S}^{q}} \min _{X \in \mathcal{X}_{\mathcal{T}}, R \in \overline{\mathcal{R}}} \mathcal{L}(X, R, S)=\max _{S \in \mathcal{S}^{q}}\left\{\min _{X \in \mathcal{X}_{\mathcal{T}}}\langle\bar{L}+S, X\rangle-\xi \lambda_{\max }\left(V^{\top} S V\right)\right\} \tag{4.32}
\end{equation*}
$$

where $\lambda_{\max }\left(V^{\top} S V\right)$ is the largest eigenvalue of $V^{\top} S V$, and $q$ is the appropriate order of the cone of symmetric matrices. In (4.32) we exploit the well-known Rayleigh principle. It follows from (4.32) that for any $S \in \mathcal{S}^{q}$ one can obtain a valid lower bound by computing:

$$
\begin{equation*}
L B(S)=\min _{X \in \mathcal{X}_{\mathcal{T}}}\langle\bar{L}+S, X\rangle-\xi \lambda_{\max }\left(V^{\top} S V\right) \tag{4.33}
\end{equation*}
$$

where $\xi$ differs for both problems. Since the minimization problem in (4.33) is a linear programming problem, the computation of valid lower bounds is efficient.

### 4.3.3.2 Stopping criteria for the CP-ADMM

We use different stopping criteria for the inner and outer while-loops in Algorithm 4.1. The following measure is used as one of the stopping criteria for the inner while-loop:

$$
\max \left\{\frac{\left\|X^{p}-V R^{p} V^{\top}\right\|_{F}}{1+\left\|X^{p}\right\|_{F}}, \sigma \frac{\left\|X^{p+1}-X^{p}\right\|_{F}}{1+\left\|S^{p}\right\|_{F}}\right\}<\varepsilon_{\mathrm{ADMM}},
$$

where $\varepsilon_{\text {ADMM }}$ is the prescribed tolerance precision. We also stop the inner while-loop when maxTime is reached.

The Dykstra's projection algorithm (CycDyk) stops when $\left\|X^{q+1}-X^{q}\right\|_{F}<\varepsilon_{\text {proj }}$ for a given input parameter $\varepsilon_{\text {proj }}$.

We consider the following types of stopping criteria for the outer while-loop:

- The algorithm stops if the gap between a valid lower bound, that is rounded up to the closest integer, and a given upper bound $U B$ is closed.
- The algorithm stops if an improvement in lower bounds between two consecutive outer loops is less than the prescribed threshold, i.e., 0.001 .
- The algorithm stops if the number of new cuts to be added in the next outer loop is small, i.e., $<0.25 n$.
- The algorithm stops if the maximum number of outer loops maxOuter Loops is reached.
- The algorithms stops immediately if the maximum computation time maxTime is reached.

We specify the values of the input parameters in Section 4.5.

### 4.3.3.3 Efficient ingredients of the CP-ADMM

Algorithm 4.1 is efficient due to the following ingredients:

1. Warm starts. After identifying new cuts we start the new ADMM iterate from the last obtained triple ( $R^{p}, X^{p}, S^{p}$ ). Observe that there is no warm start strategy for an interior-point method.
2. Scaling of data. Since the course of the algorithm depends on the magnitude of the objective value coefficients, we can improve the robustness of the proposed algorithm by an appropriate scaling of the data. Therefore, we scale the objective by a scalar $\rho \in \mathbb{R}$ that depends on the problem and its size. Namely, for the $k$-EP we set $\rho=1 /\|L\|_{F}$ for $n \leq 400, \rho=k /\|L\|_{F}$ for $400<n \leq 800$, and $\rho=n /\left(k\|L\|_{F}\right)$ otherwise. For the GBP we use $\rho=1$. The values for $\rho$ are obtained by extensive numerical tests.
3. Clustering. A crucial ingredient for improving the performance of Dykstra's projection algorithm is clustering cuts, see Section 4.3.2.
4. Separation. We introduce a probabilistic independent set separation method to separate independent set inequalities, see Algorithm 4.2 in Section 4.4.

Similar to what has been indicated in Section 3.7.2, the bottleneck of the CP-ADMM is the projection onto the positive semidefinite cone when $\mathcal{T}=\emptyset$. When we start adding cuts, Dykstra's algorithm starts taking over the major part of the computation time.

### 4.4 Valid cutting planes, their projectors and separators

In this section we consider various families of cutting planes that strengthen the DNN relaxations for the $k$-EP and GBP. In the light of adding them in the cutting-plane augmented Lagrangian algorithm of Section 4.3, we present for each cut type a polyhedral set $\mathcal{H}_{t_{i}}$ induced by the cut (and, possibly, a subset of the constraints from the corresponding DNN relaxation). We show how to explicitly project a matrix onto these polyhedral sets. The efficient separation of these cut types is also considered.

In total we consider three types of cutting planes: two for the $k$-EP and one for the GBP.

### 4.4.1 Triangle inequalities for the $k$-EP

Let us consider the relaxation $\left(D N N_{E P}\right)$, see (4.5) for the $k$-equipartition problem. Marcotorchino [262] as well as Grötschel and Wakabayashi [185] observe that the linear relaxation of the $k$-equipartition problem can be strengthened by adding the triangle inequalities:

$$
\begin{equation*}
Y_{i j}+Y_{i l} \leq 1+Y_{j l} \quad \text { for all triples }(i, j, l), i \neq j, j \neq l, i \neq l . \tag{4.34}
\end{equation*}
$$

For a given triple $(i, j, l)$ of distinct vertices, the triangle constraint (4.34) ensures that if $i$ and $j$ are in the same set of the partition and so are $i$ and $l$, then also $j$ and $l$ have to belong to the same set of the partition. Karisch and Rendl [231] use these inequalities to strengthen $\left(D N N_{E P}\right)$.

To obtain the equivalent facially reduced relaxation (4.7), we apply the linear transformation $X=Y-\frac{1}{k} \mathbf{J}_{n}$, see Section 4.2.1. As we apply our cutting-plane algorithm on this latter relaxation, we also perform this transformation on the triangle inequalities. The transformed cuts are as follows:

$$
\begin{equation*}
X_{i j}+X_{i l} \leq \frac{k-1}{k}+X_{j l} \quad \text { for all triples }(i, j, l), i \neq j, j \neq l, i \neq l . \tag{4.35}
\end{equation*}
$$

Observe that there exist $3\binom{n}{3}$ triangle inequalities.
To incorporate the cutting planes (4.35) into our cutting-plane augmented Lagrangian algorithm, we define for each cut a polyhedral set that is induced by the cut. For each triple ( $i, j, l$ ) we define the polyhedron $\mathcal{H}_{i j l}^{\Delta} \subseteq \mathcal{S}^{n}$ as follows:

$$
\begin{equation*}
\mathcal{H}_{i j l}^{\Delta}:=\left\{X \in \mathcal{S}^{n}: X_{i j}+X_{i l} \leq \frac{k-1}{k}+X_{j l}\right\} . \tag{4.36}
\end{equation*}
$$

Let $\mathcal{P}_{\mathcal{H}_{i j l}}: \mathcal{S}^{n} \rightarrow \mathcal{S}^{n}$ denote the operator that projects a matrix in $\mathcal{S}^{n}$ onto $\mathcal{H}_{i j l}^{\Delta}$. As $\mathcal{H}_{i j l}^{\Delta}$ is a halfspace, this projection has an easy closed-form expression.

Identifying the most violated inequalities of the form (4.35) can be done by a complete enumeration. This separation can be done in $O\left(n^{3}\right)$.

### 4.4.2 Independent set inequalities for the $k$-EP

Chopra and Rao [76] introduced a further type of inequalities that are valid for the linear relaxation of the $k$-equipartition problem, namely

$$
\begin{equation*}
\sum_{i, j \in I, i<j} Y_{i j} \geq 1 \quad \text { for all } I \subseteq V \text { with }|I|=k+1 \tag{4.37}
\end{equation*}
$$

which are known as the independent set inequalities. These inequalities have also been used for the SDP relaxation in the work of Karisch and Rendl [231]. The intuition behind these constraints is that for all subsets of $k+1$ nodes, there must always be two nodes that are in the same set of the partition. Thus, the graph with adjacency matrix $Y$ has no independent set of size $k+1$.

Using the linear transformation $X=Y-\frac{1}{k} \mathbf{J}_{n}$, we obtain the following equivalent inequalities that are valid for the facially reduced relaxation (4.7):

$$
\begin{equation*}
\sum_{i, j \in I, i<j} X_{i j} \geq \frac{1-k}{2} \quad \text { for all } I \text { with }|I|=k+1 \tag{4.38}
\end{equation*}
$$

Observe that there are $\binom{n}{k+1}$ independent set inequalities.
Let us define for each set $I \subseteq V$ with $|I|=k+1$ a polyhedral set $\mathcal{H}_{I}^{I S} \subseteq \mathcal{S}^{n}$ that is induced by the cut, i.e.,

$$
\begin{equation*}
\mathcal{H}_{I}^{I S}:=\left\{X \in \mathcal{S}^{n}: \sum_{i, j \in I, i<j} X_{i j} \geq \frac{1-k}{2}\right\} \tag{4.39}
\end{equation*}
$$

We let $\mathcal{P}_{\mathcal{H}_{I}^{I S}}: \mathcal{S}^{n} \rightarrow \mathcal{S}^{n}$ denote the projector onto the halfspace $\mathcal{H}_{I}^{I S}$, which can be performed by a closed-form expression.

In order to find the most violated inequalities of type (4.38), we need a separator for independent set inequalities. Exact separation of these inequalities for general $k$ is known to be $\mathcal{N} \mathcal{P}$-hard [122, 345]. Complete enumeration leads to a running time of $O\left(n^{k+1}\right)$, which is reasonable for $k=2$ and $k=3$, but becomes practically very expensive for $k>3$. For these larger $k$, we apply a combination of two separation heuristics to identify violated inequalities. First, we apply the deterministic separation heuristic from [14]. This method efficiently generates at most $n$ inequalities, which turn out to be effective as numerical experiments in [14] suggest.

On top of the heuristic from [14], we also introduce a probabilistic independent set inequality separation heuristic. Although this algorithm relies on the same idea as the deterministic heuristic from [14], the greedy selection of a new vertex to add in the set $C$ is randomized with probabilities inversely proportional to their values in the current solution matrix $X$. A pseudo-code of this heuristic is given in Algorithm 4.2. The parameter $N_{R}$ corresponds to the number of repetitions, while $\varepsilon>0$ is a sensitivity parameter. Low values of $\varepsilon$ lead to very sensitive behaviour with respect to differences in the current solution $X$, while the selection eventually resembles a uniform distribution when $\varepsilon$ is increased. The advantage of this randomization is that the combination of both heuristics can yield more than $n$ violated independent set inequalities.

```
Algorithm 4.2 Probabilistic separation method for independent set inequalities
Input: the number of partitions \(k\), the size of graph \(n\), output matrix \(X\) from the ADMM, number of
    repetitions \(N_{R}\), sensitivity parameter \(\varepsilon>0\).
    Initialization: \(Y=X+\frac{1}{k} \mathbf{J}_{n}, \mathcal{C}=\emptyset\).
    for \(r \in\left[N_{R}\right]\) do
        Choose vertex \(v\) uniformly at random from \([n]\).
        \(C \leftarrow\{v\}\)
        \(S \leftarrow[n] \backslash\{v\}\)
        for \(l \in[k]\) do
            Define \(p_{i}:=\sum_{j \in C} y_{i j}+\varepsilon\) for all \(i \in S\)
            Define \(q_{i}:=\frac{\left(1 / p_{i}\right)}{\sum_{i \in S}\left(1 / p_{i}\right)}\) for all \(i \in S\)
            Randomly select vertex \(i \in S\) according to probability mass function \(\left\{q_{i}\right\}_{i \in S}\)
            \(C \leftarrow C \cup\{i\}, S \leftarrow S \backslash\{i\}\)
        end for
        if \(C \notin \mathcal{C}\) then
            \(\mathcal{C} \leftarrow \mathcal{C} \cup\{C\}\)
        end if
    end for
Output: a collection of violated distinct independent set inequalities \(\mathcal{C}\), its violation vector \(v\).
```


### 4.4.3 BQP inqualities for the GBP

We now consider the relaxation $\left(D N N_{B P}\right)$, see (4.13). The relaxation ( $D N N_{B P}$ ) can be further strengthened by adding the following inequalities:

$$
\begin{array}{r}
0 \leq X_{i j} \leq X_{i i} \\
X_{i i}+X_{j j} \leq 1+X_{i j} \\
X_{i l}+X_{j l} \leq X_{l l}+X_{i j} \\
X_{i i}+X_{j j}+X_{l l} \leq X_{i j}+X_{i l}+X_{j l}+1, \tag{4.43}
\end{array}
$$

where $X=\left(X_{i j}\right) \in \mathcal{S}^{2 n+1}$ and $1 \leq i, j, l \leq 2 n, i \neq j, i \neq l, j \neq l$. The inequalities (4.40)(4.43) are facet defining inequalities of the boolean quadric polytope [297]. Wolkowicz and Zhao [371] prove that the inequalities (4.40) and (4.41) are already implied by the constraints in (4.12). Moreover, preliminary numerical results show that the inequalities (4.42) make larger improvements in the bounds when added to the DNN relaxation than the inequalities (4.43). Therefore, we consider only the constraints (4.42) within our algorithm.

Different from the SDP relaxation of the $k$-EP, the polyhedral set $\mathcal{X}_{B P}$ is a subset of the lifted space $\mathcal{S}^{2 n+1}$. As a result, the polyhedral part induced by a BQP cut of the form (4.42) is also a subset of $\mathcal{S}^{2 n+1}$. For each triple $(i, j, l)$ with $2 \leq i, j, l \leq 2 n+1, i \neq j, i \neq l, j \neq l$, we define the following polyhedron:

$$
\left.\mathcal{H}_{i j l}^{B Q P}:=\left\{\left(\begin{array}{rrr}
1 & \left(x^{1}\right)^{\top} & \left(x^{2}\right)^{\top}  \tag{4.44}\\
x^{1} & X^{11} & X^{12} \\
x^{2} & X^{21} & X^{22}
\end{array}\right) \in \mathcal{S}^{2 n+1}: \quad X=\left(\begin{array}{cc}
X^{11} & X^{12} \\
X^{21} & X^{22}
\end{array}\right), x^{1}+x^{2}=\mathbf{1}_{n}\right\} \begin{array}{l} 
\\
X_{i l}+X_{j l} \leq X_{l l}+X_{i j} \\
\operatorname{diag}\left(X^{i i}\right)=x^{i} \quad \forall i \in[2]
\end{array}\right\} .
$$

The polyhedron $\mathcal{H}_{i j l}^{B Q P}$ is not only induced by the BQP cut, it also contains a subset of the constraints of the relaxation (4.13). This idea is inspired by the approach in [275], where the inclusion of additional constraints in each polyhedron in Dykstra's algorithm speeds up the

Chapter 4. Partitioning through projections: strong SDP bounds for large graph partition problems
convergence. Since the structure of $\mathcal{H}_{i j l}^{B Q P}$ must remain simple enough to project onto it via a closed form expression, it is impractical to add all constraints from (4.13). The set $\mathcal{H}_{i j l}^{B Q P}$ is chosen such that we are still able to project onto it explicitly.

Let $\mathcal{P}_{\mathcal{H}_{i j l}^{B Q P}}: \mathcal{S}^{2 n+1} \rightarrow \mathcal{S}^{2 n+1}$ denote the projector onto $\mathcal{H}_{i j l}^{B Q P}$. Given that the matrix that is projected already satisfies $\operatorname{diag}\left(X^{11}\right)=x^{1}, \operatorname{diag}\left(X^{22}\right)=x^{2}$ and $x^{1}+x^{2}=\mathbf{1}_{n}$, which is always the case in our implementation, this projector is specified by the result below.

Lemma 4.8. Let $M=\left(\begin{array}{ccc}1 & \operatorname{diag}\left(M^{11}\right)^{\top} & \operatorname{diag}\left(M^{22}\right)^{\top} \\ \operatorname{diag}\left(M^{11}\right) & M^{11} & M^{12} \\ \operatorname{diag}\left(M^{22}\right) & M^{21} & M^{22}\end{array}\right) \in \mathcal{S}^{2 n+1}$ be such that $\operatorname{diag}\left(M^{11}\right)+\operatorname{diag}\left(M^{22}\right)=\mathbf{1}_{n}$ and let $\hat{M}:=\mathcal{P}_{\mathcal{H}_{i j l}^{B Q P}}(M)$. If $M_{i l}+M_{j l} \leq M_{i j}+\frac{1}{6} M_{l l}+$ $\frac{1}{3} M_{1 l}-\frac{1}{6} M_{l^{*} l^{*}}-\frac{1}{3} M_{1 l^{*}}+\frac{1}{2}$, then

$$
\hat{M}_{p q}= \begin{cases}\frac{1}{6} M_{l l}+\frac{1}{3} M_{1 l}-\frac{1}{6} M_{l^{*} l^{*}}-\frac{1}{3} M_{1 l^{*}}+\frac{1}{2} & \text { if }(p, q) \in\{(l, l),(1, l),(l, 1)\} \\ -\frac{1}{6} M_{l l}-\frac{1}{3} M_{1 l}+\frac{1}{6} M_{l^{*} l^{*}}+\frac{1}{3} M_{1 l^{*}}+\frac{1}{2} & \text { if }(p, q) \in\left\{\left(l^{*}, l^{*}\right),\left(1, l^{*}\right),\left(l^{*}, 1\right)\right\} \\ M_{p q} & \text { otherwise }\end{cases}
$$

Otherwise, $\hat{M}$ is such that

where $l^{*}$ is obtained from $l$ by $l^{*}:=2+(l+n-2) \bmod 2 n$.

Proof. The matrix $\hat{M}$ is the solution to $\min _{\hat{M} \in \mathcal{S}^{2 n+1}}\left\{\|\hat{M}-M\|_{F}^{2}: \hat{M} \in \mathcal{H}_{i j l}\right\}$. The inequality describing $\mathcal{H}_{i j l}^{B Q P}$ only involves the pairs $(i, l),(j, l),(i, j)$ and $(l, l)$. Since $\operatorname{diag}\left(X^{11}\right)=x^{1}$, $\operatorname{diag}\left(X^{22}\right)=x^{2}$ and $x^{1}+x^{2}=\mathbf{1}_{n}$ should be satisfied for $\hat{M}$, any change in $(l, l)$ also has an effect on the pairs $(1, l),\left(l^{*}, l^{*}\right)$ and $\left(1, l^{*}\right)$, where $l^{*}$ is the index corresponding to $l$ in the diagonal block not containing $l$. Taking these pairs into account, we can restrict ourselves to the following convex optimization problem:

$$
\begin{aligned}
\min _{\alpha, \beta, \gamma, \mu} & 2\left(\alpha-M_{i l}\right)^{2}+2\left(\beta-M_{j k}\right)^{2}+2\left(\gamma-M_{i j}\right)^{2}+\left(\mu-M_{l l}\right)^{2}+2\left(\mu-M_{1 l}\right)^{2} \\
& +\left(1-\mu-M_{l^{*} l^{*}}\right)^{2}+2\left(1-\mu-M_{1 l^{*}}\right)^{2} \\
\text { s.t. } & \alpha+\beta \leq \gamma+\mu .
\end{aligned}
$$

Let $\lambda \geq 0$ denote the Lagrange multiplier for the inequality, then the KKT conditions imply
the following system:

$$
\left\{\begin{array}{l}
4\left(\alpha-M_{i l}\right)+\lambda=0 \\
4\left(\beta-M_{j l}\right)+\lambda=0 \\
4\left(\gamma-M_{i j}\right)-\lambda=0 \\
2\left(\mu-M_{l l}\right)+4\left(\mu-M_{1 l}\right)+2\left(\mu-1+M_{l^{*} l^{*}}\right)+4\left(\mu-1+M_{1 l^{*}}\right)-\lambda=0 \\
\lambda(\alpha+\beta-\gamma-\mu)=0 \\
\alpha+\beta \leq \gamma+\mu \\
\lambda \geq 0
\end{array}\right.
$$

Complementarity implies that either $\alpha+\beta=\gamma+\mu$ or $\lambda=0$. The latter case leads to the KKT-point $(\alpha, \beta, \gamma, \mu)=\left(M_{i l}, M_{j l}, M_{i j}, \frac{1}{6} M_{l l}+\frac{1}{3} M_{1 l}-\frac{1}{6} M_{l^{*} l^{*}}-\frac{1}{3} M_{1 l^{*}}+\frac{1}{2}\right)$, which is optimal if $M_{i l}+M_{j l} \leq M_{i j}+\frac{1}{6} M_{l l}+\frac{1}{3} M_{1 l}-\frac{1}{6} M_{l^{*} l^{*}}-\frac{1}{3} M_{1 l^{*}}+\frac{1}{2}$.

Now assume that $\lambda \neq 0$. Then $\alpha+\beta=\gamma+\mu$. The first four equalities of the KKT system can be rewritten as:

$$
\begin{aligned}
& \alpha=M_{i l}-\frac{1}{4} \lambda, \quad \beta=M_{j l}-\frac{1}{4} \lambda, \quad \gamma=M_{i j}+\frac{1}{4} \lambda \\
& \mu=\frac{1}{6} M_{l l}+\frac{1}{3} M_{1 l}-\frac{1}{6} M_{l^{*} l^{*}}-\frac{1}{3} M_{1 l^{*}}+\frac{1}{2}+\frac{1}{12} \lambda
\end{aligned}
$$

Substitution into $\alpha+\beta=\gamma+\mu$ yields

$$
\begin{aligned}
& M_{i l}-\frac{1}{4} \lambda+M_{j l}-\frac{1}{4} \lambda=M_{i j}+\frac{1}{4} \lambda+\frac{1}{6} M_{l l}+\frac{1}{3} M_{1 l}-\frac{1}{6} M_{l^{*} l^{*}}-\frac{1}{3} M_{1 l^{*}}+\frac{1}{2}+\frac{1}{12} \lambda \\
\Longleftrightarrow \quad \lambda & =\frac{12}{10}\left(M_{i l}+M_{j l}-M_{i j}-\frac{1}{6} M_{l l}-\frac{1}{3} M_{1 l}+\frac{1}{6} M_{l^{*} l^{*}}+\frac{1}{3} M_{1 l^{*}}-\frac{1}{2}\right) .
\end{aligned}
$$

Substitution of this expression for the Lagrange multiplier into the remaining equalities provides the optimal values for $\alpha, \beta, \gamma$ and $\mu$ :

$$
\begin{aligned}
\alpha & =\frac{7}{10} M_{i l}-\frac{3}{10} M_{j l}+\frac{3}{10} M_{i j}+\frac{1}{20} M_{l l}+\frac{1}{10} M_{1 l}-\frac{1}{20} M_{l^{*} l^{*}}-\frac{1}{10} M_{1 l^{*}}+\frac{3}{20} \\
\beta & =-\frac{3}{10} M_{i l}+\frac{7}{10} M_{j l}+\frac{3}{10} M_{i j}+\frac{1}{20} M_{l l}+\frac{1}{10} M_{1 l}-\frac{1}{20} M_{l^{*} l^{*}}-\frac{1}{10} M_{1 l^{*}}+\frac{3}{20} \\
\gamma & =\frac{3}{10} M_{i l}+\frac{3}{10} M_{j l}+\frac{7}{10} M_{i j}-\frac{1}{20} M_{l l}-\frac{1}{10} M_{1 l}+\frac{1}{20} M_{l^{*} l^{*}}+\frac{1}{10} M_{1 l^{*}}-\frac{3}{20} \\
\mu & =\frac{1}{10} M_{i l}+\frac{1}{10} M_{j l}-\frac{1}{10} M_{i j}+\frac{3}{20} M_{l l}+\frac{3}{10} M_{1 l}-\frac{3}{20} M_{l^{*} l^{*}}-\frac{3}{10} M_{1 l^{*}}+\frac{9}{20}
\end{aligned}
$$

Setting $\hat{M}_{i l}=\hat{M}_{l i}=\alpha, \hat{M}_{j l}=\hat{M}_{l j}=\beta, \hat{M}_{i j}=\hat{M}_{j i}=\gamma, \hat{M}_{l l}=\hat{M}_{1 l}=\hat{M}_{l 1}=\mu$ and $\hat{M}_{l^{*} l^{*}}=\hat{M}_{1 l^{*}}=\hat{M}_{l^{*} 1}=1-\mu$ gives the final result.

Separating the BQP inequalities (4.42) can be done in $O\left(n^{3}\right)$ by complete enumeration.

### 4.5 Computational results

We implemented our algorithm CP-ADMM in Matlab. For efficiency some separation routines have been coded in C. In order to evaluate the quality of the bounds and the run times to compute these bounds, we test our algorithms on various instances from the literat-
ure. All experiments were run on an Intel Xeon, E5-1620, 3.70 GHz with 32 GB memory. To compute valid lower bounds after each run of the inner while-loops we use Mosek [284]. Note that the computation of a valid bound after the inner while-loops is necessary for checking the stopping criteria.

We now describe the data sets used in our evaluation. Most of these instances were also considered in [20] and [21].

- $G_{|V|,|V| p}$ and $U_{|V|,|V| \pi d^{2}}$ : randomly generated graphs by Johnson et al. [228].
- $G_{|V|,|V| p}$ : graphs $G=(V, E)$, with $|V| \in\{124,250,500,1000\}$ and four individual edge probabilities $p$. These probabilities were chosen depending on $|V|$, so that the average expected degree of each node was approximately $|V| p \in\{2.5,5,10,20\}$.
- $U_{|V|,|V| \pi d^{2}}$ : graphs $G=(V, E)$, with $|V| \in\{500,1000\}$ with distance value $d$ such that $|V| \pi d^{2} \in\{5,10,20,40\}$. To form such a graph $G=(V, E)$, one chooses $2|V|$ independent numbers uniformly from the interval $(0,1)$ and views them as coordinates of $|V|$ nodes on the unit square. An edge is inserted between two vertices if and only if their Euclidean distance is at most $d$.
- Mesh graphs from [131, 346]: Instances from finite element meshes; all nonzero edge weights are equal to one. Graph names begin with 'mesh', followed by the number of vertices and the number of edges.
- KKT graphs: These instances originate from nested bisection approaches for solving sparse symmetric linear systems. Each instance consists of a graph that represents the support structure of a sparse symmetric linear system, for details see [202].
- Toroidal 2D- and 3D-grid graphs arise in physics when computing ground states for Ising spinglasses, see e.g., [202]. They are generated using the rudy graph generator [322]:
- spinglass2pm_ $n_{r}$ : A toroidal 2D-grid for a spinglass model with weights $\{+1,-1\}$. The grid has size $n_{r} \times n_{r}$, i.e., $|V|=n_{r}^{2}$. The percentage of edges with negative weights is $50 \%$.
- spinglass3pm_ $n_{r}$ : A toroidal 3D-grid for a spinglass model with weights $\{+1,-1\}$. The grid has size $n_{r} \times n_{r} \times n_{r}$, i.e., $|V|=n_{r}^{3}$. The percentage of edges with negative weights is $50 \%$.


### 4.5.1 Computational results for the $k$-EP

In the CP-ADMM, see Algorithm 4.1, we input an upper bound $U B$ and the parameters maxTime, numCuts, maxOuterloops, $\varepsilon_{\text {proj }}$, and $\varepsilon_{\text {ADMM }}$. We also require the bounds $\sigma_{\min }$, and $\sigma_{\max }$ for the adaptive stepsize term. The setting of these parameters is as follows:

- As an upper bound we input the values we obtained by heuristics or the optimal solution given in the literature.
- The maximal number of cuts added in each outer while-loop, numCuts, is $3 n$ for graphs with $n \leq 300$ and $5 n$ when $n>300$. These values are determined by preliminary tests, see also Appendix A.2.
- The maximal number of outer while-loops is 30 for instances with $n \leq 300$, and 10 when $n>300$.
- The precision for Dykstra's projection algorithm $\varepsilon_{\text {proj }}$ is set to $10^{-4}$.
- The inner precision $\varepsilon_{\mathrm{ADMM}}$ is $10^{-4}$ in the last iteration and $10^{-3}$ in all previous loops.
- The maximum computation time maxTime is set to 2 hours.
- The bounds $\sigma_{\min }$ and $\sigma_{\max }$ for $\sigma^{p}$ are $10^{-5}$ and $10^{3}$, respectively.

In each outer while-loop we separate MaxIneq violated inequalities. We experimented with two strategies: One strategy is to first add violated triangle inequalities and, in case less than MaxIneq violated triangle inequalities are found, we add violated independent set inequalities. The other strategy is to mix the two kinds of cuts and search for violated triangle and independent set inequalities together. The experiments showed that the latter strategy obtains better results, i.e., better bounds within the same time. Therefore, in the final setting we search for the most violated inequalities from both, the triangle and independent set inequalities.

The separation of triangle inequalities is done by complete enumeration. Searching for independent set inequalities is also done by complete enumeration if $k \in\{2,3\}$. For $k \in\{4,5\}$ we apply the heuristic from [14] and Algorithm 4.2, as explained in Section 4.4.2.

In Table 4.2 we compare the eigenvalue lower bound by Donath and Hoffman [104] (denoted by DH) with the lower bound when computing the DNN relaxation as well as the DNN relaxation with additional cuts on selected graphs from our testbed. We do not include other bounds from the literature, such as bounds from [193], as they are weaker than the DH bound. The DH bound is obtained by solving the corresponding SDP relaxation using Mosek [284]. The numbers in the table show that the DNN bound significantly improves over the DH bound. Moreover, our ADMM algorithm requires less time and memory to compute the DNN bound than Mosek to compute the DH bound. Adding cuts to the DNN gives an even more substantial improvement. Hence, including triangle and independent set inequalities is much stronger than including nonnegativity constraints only. As the DH bound is not competitive with the DNN bound and far from the DNN+cuts bound, we will not include it in the subsequent presentation of the numerical results.

As shown by [342, 343], the bound (4.11) is equivalent to the SDP bounds derived in [233, 343 ] and the bound [371] with nonnegativity constraints. Also, it dominates the bounds [104, 321] as shown by [231]. For that reason, we do not include these SDP approaches in the numerical results.

Computing an equipartition using commercial solvers is out of reach unless the graphs are extremely sparse. For instances in Table 4.2, Gurobi (with the default settings) solves the small and very sparse instances within a few seconds, but obtains a gap of more than $40 \%$ after 2 hours for larger, but still reasonably sparse graphs. E.g., for $G_{250,5}$, a graph on 250 vertices with density $2 \%$, the gap after two hours is more than $40 \%$. For $G_{250,20}$, a graph on 250 vertices and density $8 \%$, the gap is even more than $80 \%$ after two hours. This limited power of commercial solvers to tackle the $k$-EP is also observed in [59]. Hence, we omit the comparison to Gurobi or other LP-based solvers in the tables.

In Tables 4.3 to 4.9 we give the details of our numerical results. In the first 4 columns we list the name of the instance, the number of vertices $n$, the partition number $k$ and an

Chapter 4. Partitioning through projections: strong SDP bounds for large graph partition problems

| Graph | $\boldsymbol{n}$ | $\boldsymbol{k}$ | UB | DH | DNN | DNN + Cuts |
| ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| mesh.70.120 | 70 | 2 | 7 | 1.93 | 2.91 | 6.02 |
| KKT.lowt01 | 82 | 2 | 13 | 2.47 | 4.88 | 12.43 |
| mesh.148.265 | 148 | 4 | 22 | 5.46 | 8.13 | 21.23 |
| $G_{124,2.5}$ | 124 | 2 | 13 | 4.59 | 7.29 | 12.01 |
| $G_{124,10}$ | 124 | 2 | 178 | 138.24 | 152.86 | 170.88 |
| $G_{124,20}$ | 124 | 2 | 449 | 403.08 | 418.67 | 439.96 |
| $G_{250,25}$ | 250 | 2 | 29 | 10.99 | 15.16 | 28.30 |
| $G_{250,5}$ | 250 | 2 | 114 | 70.21 | 81.52 | 105.00 |
| $G_{250,10}$ | 250 | 2 | 357 | 280.25 | 303.02 | 330.40 |

Table 4.2: Comparison of different lower bounds.
upper bound. The upper bounds are obtained by heuristics or the optimal solution from the literature.

In columns 5 and 6 the lower bound (rounded up) and the time when solving the DNN relaxation are given. Finally, in the remaining columns we display the results when adding cuts to the DNN relaxation: we report the rounded up lower bounds, the computation time and the improvement (in \%) on the rounded up lower bounds with respect to the DNN relaxation without cuts. We decided to report the improvement with respect to the rounded values, since otherwise for small numbers the percentages are incredibly huge. E.g., for instance $U_{1000,5}$ and $k=4$, the value of the DNN bound is 0.17 , the DNN+cuts bound is 2.45 , giving an improvement of $1341.2 \%$. The rounded up values are 1 and 3 , respectively, giving a $200 \%$ improvement, which reflects the situation much better. Moreover, we present the percentage of the gap between the upper bound and the rounded DNN relaxation without cuts that is closed by the rounded DNN relaxation with cutting planes.

In columns 10 and 11 we list the number of triangle cuts and independent set cuts present when stopping the algorithm. In the last two columns, the number of iterations of the ADMM and the number of outer while-loops is reported.

As can be observed in all tables, the bounds improve drastically when adding triangle and independent set inequalities to the DNN relaxation while the time for computing these bounds is still reasonable. We remark here that we can stop the CP-ADMM at any time and provide a valid lower bound.

The results show that the $\mathrm{CP}-\mathrm{ADMM}$ takes much more of the triangle inequalities when computing bounds for the $k$-equipartition problem. Remember that we search for triangle and independent set inequalities together. Hence, there are more triangle inequalities with a large violation which means that the triangle inequalities contribute more to the strength of the bound than the independent set inequalities.

We discuss the results in more detail in the subsequent sections.

### 4.5.1.1 Detailed results for $k=2$

In Tables 4.3 and 4.4 we report results for $k=2$. Table 4.3 includes graphs with up to 274 vertices, the DNN bound for these graphs can be computed within a few seconds. After adding in total some 500 up to 16000 triangle inequalities and additional 300 up to roughly 7000 independent set inequalities, the bound improves between $4.55 \%$ and $300 \%$. In several cases, the bound closes the gap to the best known upper bound, as can be seen from
the large percentage of the gap that is closed by the addition of cutting planes. Otherwise, the algorithm stops due to the little improvement of the bounds in consecutive outer whileloops. The time for computing these bounds ranges from a few seconds to 17 minutes.

In Table 4.4 we consider larger graphs with 500 and 1000 vertices. The DNN bound can be computed for these graphs within 12 minutes. After adding triangle and independent set inequalities, the bound improves up to $200 \%$ in a running of 40 seconds up to 2 hours. On the $G$ graphs one observes that the improvement of the bound when adding cuts gets more significant as the graphs get sparser. Note that for all these instances we stop because the number of outer while-loops is reached or the improvement of the bounds in consecutive outer while-loops is too small.

We give further results for graphs from the literature in Table A. 3 in Appendix A.2. For most of these we prove optimality of the best found $k$-equipartition, confirming the high quality of our lower bounds.

### 4.5.1.2 Detailed results for $k>2$

In Tables 4.5 to 4.9 we report results for $k>2$. As in the case for $k=2$, the bounds can be significantly improved while the time for obtaining the bounds is still reasonable. However, we close the gap for fewer instances as for $k=2$, although the percentage of the gap closed by the addition of cutting planes is in almost all cases substantial. The improvement for the larger graphs after adding cuts to the DNN relaxation is up to $200 \%$ for $k=4$ and up to $300 \%$ for $k=5$.

It has been observed by $[231,256,365]$ that when $k$ increases, the nonnegativity constraints become more important for the strength of the bound, while the number of violated triangle inequalities becomes smaller. This claim is supported by our numerical results, which show that the effect of the additional cutting planes is more significant for $k=2$ than for $k>2$. Since the largest $k$ we consider, i.e., $k=6$, is rather small, we still observe an improvement by adding triangle and independent set constraints.

For smaller graphs the CP-ADMM stops because the improvement of the lower bound compared to the previous iteration is below the threshold 0.001, see Section 4.3.3.2. The largest improvement in the DNN+cuts bound w.r.t. the DNN bound is $500 \%$. Again, we observe that for graphs with more than 250 vertices, the algorithm typically stops because the maximum number of outer loops is reached.

### 4.5.2 Computational results for the GBP

In Algorithm 4.1, we input an upper bound $U B$ and parameters numCuts, maxOuterloops, $\varepsilon_{\text {proj }}, \varepsilon_{\text {ADMM }}$, and maxTime. The setting of these parameters is as follows.

- As an upper bound we input the numbers we obtained by heuristics.
- The maximal number of cuts added in each outer while-loop, numCuts, is $3 n$ for graphs with $n \leq 300$ and $5 n$ when $n>300$.
- The maximal number of outer while-loops is 30 for instances with $n \leq 300$, and 10 when $n>300$.
- The precision for Dykstra's projection algorithm $\varepsilon_{\text {proj }}$ is set to $10^{-6}$.
- The inner precision $\varepsilon_{\text {ADMM }}$ is $10^{-5}$ in the first and last inner while-loop, and $10^{-4}$ in all other inner while-loops.
- The maximum computation time maxTime is set to 2 hours.

The parameters differ from the experiments for the $k$-equipartition problem since for the bisection problem the size of the matrix in the SDP is of order $2 n+1$, i.e., more than double the size than for the $k$-equipartition problem. Violated BQP inequalities (4.42) are found by an enumeration search.

We take $m_{1}=\lceil n p\rceil$ where $n$ is the number of vertices in the graph and for $p$ we choose a number out of $\{0.6,0.65,0.7\}$.

It is shown in $[343,344]$ that the DNN bound (4.22) is equivalent to the bounds [233, 344], while it dominates the bounds derived in [104, 232, 321, 343]. Since the DNN relaxation (4.22) is currently the strongest known bound for the GBP, we do not include alternative SDP bounds in the numerical results.

The results are given in Table 4.10 for smaller graphs and in Table 4.11 for larger graphs. Similar as for the $k$-equipartition problem, we observe a significant improvement of the bound after adding inequalities. For graphs with up to 274 vertices, see Table 4.10, the improvement of the DNN+cuts bound over the DNN bound ranges between $2.23 \%$ and $200 \%$ after adding up to 16000 BQP inequalities. For five of the graphs in Table 4.10 the algorithm stopped because the gap was closed, for four graphs the improvement of the lower bound was only minor and four times the number of violated cuts found was too small. Only for one graph the algorithm stopped due to the time limit. For the larger graphs we typically stop because of the time limit of 2 hours, see Table 4.11. For those graphs the DNN+cuts bound improves over the DNN bound between $0.44 \%$ and $17.32 \%$ after adding between 8000 and 24200 BQP inequalities. The percentage of the gap closed by the addition of cuts ranges from $3.23 \%$ to $78.26 \%$, and is especially large for the spinglass instances.

One can observe that the results are somewhat weaker than for the $k$-equipartition problem. Note that the nature of the cuts added to the DNN relaxations for the $k$-EP and GBP differs, and the size of matrix variables in the GBP relaxations is more than twice larger than in the case of the $k$-EP for the same graph. For large graphs, we are able to add roughly up to 50000 cuts for the $k$-EP and 24000 cuts for the GBP within a time span of 2 hours.

### 4.6 Conclusions

This study aims to investigate and expand the boundary of obtaining strong DNN bounds with additional cutting planes for large graph partition problems. Due to memory requirements, state-of-the-art interior-point methods are only capable of solving medium-size SDPs and are not suitable for handling lots of polyhedral cuts. We overcome both difficulties by utilizing a first-order method in a cutting-plane framework. Our approach focuses on two variations of the graph partition problem: the $k$-equipartition problem and the graph bisection problem.

We first derive DNN relaxations for both problems, see (4.5) and (4.13), then we apply facial reduction to obtain strictly feasible equivalent relaxations, see (4.7) and (4.18), respectively. To prove the minimality of the face of the DNN cone containing the feasible set of (4.13), we exploit the dimension of the bisection polytope, see Theorem 4.1. After facial reduction, both relaxations enclose a natural splitting of the feasible set into polyhedral and positive semidefinite constraints. Moreover, both relaxations can be further strengthened by several types of cutting planes.

To solve both relaxations, we use an ADMM update scheme, see (4.24), that is incorporated in a cutting-plane framework, leading to the so-called CP-ADMM, see Al-
gorithm 4.1. The cutting planes are handled in the polyhedral subproblem by exploiting a semi-parallelized version of Dykstra's cyclic projection algorithm, see Section 4.3.2. The CP-ADMM benefits from warm starts whenever new cuts are added, provides valid lower bounds even after solving with low precision, and can be implemented efficiently. A particular ingredient of the CP-ADMM are the projectors onto polyhedra induced by the cutting planes. Projection operators for three types of cutting planes that are effective for the graph partition problem, i.e., the triangle, independent set and BQP inequalities, are derived.

Numerical experiments show that using our CP-ADMM algorithm we are able to produce high-quality bounds for graphs up to 1024 vertices. We experimented with several graph types from the literature. For structured graphs of medium size and the 2-EP we often close the gap in a few seconds or at most a couple of minutes. For bisection problems on those graphs, we also close the gaps in many cases. For larger graphs, we are able to add polyhedral cuts roughly up to 50000 for $k$-EP and 24000 for the GBP within 2 hours, which results in strong lower bounds. Our results provide benchmarks for solving medium and large scale graph partition problems.

This research can be extended in several directions. As indicated in Section 4.2.2, our approach for the GBP has a straightforward extention to the general GPP strengthened by BQP cuts. Motivated by the optimistic results for the $k$-EP and the GBP, we expect that strong bounds from DNN relaxations with additional cutting planes can be obtained for other partition problems, such as the vertex separator problem and the max-cut problem. Moreover, since the major ingredients of our algorithm are presented generally, establishing an approach for solving general DNN relaxations with additional cutting planes is an interesting future research direction. Our results also provide new perspectives on solving large-scale optimization problems to optimality by using SDPs.

| graph | $n$ | $k$ | UB | DNN |  | DNN+cuts |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | imp. <br> \% | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | \#tri. cuts | \#ind. <br> set cuts | \#iter. | \#outer <br> loops |
| $G_{124,2.5}$ | 124 | 2 | 13 | 8 | 1.25 | 13 | 31.56 | 62.50 | 100.00 | 3298 | 1166 | 848 | 12 |
| $G_{124,5}$ | 124 | 2 | 63 | 47 | 0.65 | 62 | 72.30 | 31.91 | 93.75 | 4354 | 482 | 1325 | 13 |
| $G_{124,10}$ | 124 | 2 | 178 | 153 | 0.40 | 171 | 9.86 | 11.76 | 72.00 | 2663 | 432 | 570 | 9 |
| $G_{124,20}$ | 124 | 2 | 449 | 419 | 0.39 | 440 | 6.67 | 5.01 | 70.00 | 2266 | 509 | 485 | 8 |
| $G_{250,2.5}$ | 250 | 2 | 29 | 16 | 4.83 | 29 | 968.05 | 81.25 | 71.88 | 15865 | 2135 | 2068 | 24 |
| $G_{250,5}$ | 250 | 2 | 114 | 82 | 2.67 | 105 | 174.11 | 28.05 | 50.94 | 10063 | 374 | 1184 | 14 |
| $G_{250,10}$ | 250 | 2 | 357 | 304 | 2.10 | 331 | 27.74 | 8.88 | 41.98 | 5352 | 415 | 575 | 9 |
| $G_{250,20}$ | 250 | 2 | 828 | 747 | 1.77 | 781 | 10.59 | 4.55 | 100.00 | 4516 | 617 | 484 | 7 |
| mesh. 70.120 | 70 | 2 | 7 | 3 | 0.19 | 7 | 2.42 | 133.33 | 100.00 | 474 | 366 | 351 | 4 |
| mesh. 74.129 | 74 | 2 | 8 | 4 | 0.20 | 8 | 3.74 | 100.00 | 100.00 | 575 | 313 | 518 | 4 |
| mesh.138.232 | 138 | 2 | 8 | 2 | 1.42 | 8 | 124.38 | 300.00 | 100.00 | 5298 | 3810 | 1135 | 22 |
| mesh.148.265 | 148 | 2 | 7 | 3 | 2.25 | 7 | 6.15 | 133.33 | 100.00 | 1064 | 1156 | 481 | 5 |
| mesh.274.469 | 274 | 2 | 7 | 2 | 13.69 | 7 | 1000.60 | 250.00 | 100.00 | 14038 | 7334 | 1926 | 26 |
| KKT.lowt01 | 82 | 2 | 13 | 5 | 0.22 | 13 | 6.23 | 160.00 | 100.00 | 515 | 715 | 357 | 5 |
| Table 4.3: Instances considered in [193, Table 9], $k=2$. |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | DNN |  | DNN+Cuts |  |  |  |  |  |  |  |
| graph | $n$ | $k$ | UB | LB <br> (rounded) | clocktime <br> (s) | LB (rounded) | clocktime <br> (s) | $\begin{gathered} \text { imp. } \\ \% \end{gathered}$ | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | \#tri. cuts | \#ind. set cuts | \#iter. | \#outer loops |
| $G_{500,2.5}$ | 500 | 2 | 49 | 25 | 45.83 | 45 | 1425.01 | 80.00 | 83.33 | 24549 | 451 | 1444 | 10 |
| $G_{500,5}$ | 500 | 2 | 218 | 156 | 22.60 | 197 | 592.33 | 26.28 | 66.13 | 24356 | 644 | 1188 | 10 |
| $G_{500,10}$ | 500 | 2 | 626 | 513 | 15.90 | 554 | 125.15 | 7.99 | 36.28 | 13069 | 713 | 678 | 8 |
| $G_{500,20}$ | 500 | 2 | 1744 | 1566 | 14.78 | 1613 | 42.91 | 3.00 | 26.40 | 9705 | 1076 | 520 | 6 |
| $G_{1000,2.5}$ | 1000 | 2 | 102 | 45 | 421.80 | 74 | 7206.86 | 64.44 | 50.88 | 44815 | 185 | 1731 | 9 |
| $G_{1000,5}$ | 1000 | 2 | 451 | 307 | 204.65 | 379 | 2183.88 | 23.45 | 50.00 | 49054 | 946 | 1822 | 10 |
| $G_{1000,10}$ | 1000 | 2 | 1367 | 1113 | 152.35 | 1179 | 523.82 | 5.93 | 25.98 | 25413 | 1272 | 1251 | 7 |
| $G_{1000,20}$ | 1000 | 2 | 3389 | 3007 | 140.69 | 3079 | 350.84 | 2.39 | 18.85 | 19195 | 1813 | 1078 | 6 |
| $U_{500,5}$ | 500 | 2 | 2 | 1 | 74.43 | 2 | 2528.19 | 100.00 | 100.00 | 25000 | 0 | 3062 | 10 |


| $U_{500,10}$ | 500 | 2 | 26 | 8 | 39.52 | 23 | 708.11 | 187.50 | 83.33 | 17824 | 7176 | 1742 |
| ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| $U_{500,20}$ | 500 | 2 | 178 | 56 | 66.28 | 153 | 838.74 | 173.21 | 79.51 | 14817 | 10183 | 2638 |
| $U_{500,40}$ | 500 | 2 | 412 | 163 | 97.47 | 379 | 1562.31 | 132.52 | 86.75 | 17444 | 7556 | 3024 |
| $U_{1000,5}$ | 1000 | 2 | 1 | 0 | 725.58 | 0 | 7200.31 | 0.00 | 0.00 | 50000 | 0 | 6237 |
| $U_{1000,10}$ | 1000 | 2 | 39 | 8 | 674.90 | 24 | 3086.56 | 200.00 | 51.61 | 43836 | 6164 | 3357 |
| $U_{1000,20}$ | 1000 | 2 | 222 | 51 | 213.67 | 136 | 3877.13 | 166.67 | 49.71 | 42215 | 7785 | 3016 |
| $U_{1000,40}$ | 1000 | 2 | 737 | 240 | 561.08 | 640 | 4665.77 | 166.67 | 80.48 | 40190 | 9810 | 3199 |

Table 4.4: Large $G$ and $U$ graphs from [228], $k=2$.

| graph | $n$ | $k$ | UB | DNN |  | DNN+cuts |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { imp. } \\ \% \end{gathered}$ | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | \#tri. <br> cuts | \#ind. <br> set cuts | \#iter. | \#outer loops |
| mesh.69.212 | 69 | 3 | 10 | 4 | 0.04 | 9 | 1093.39 | 125.00 | 83.33 | 1375 | 4421 | 1266 | 28 |
| mesh.138.232 | 138 | 3 | 13 | 2 | 0.30 | 12 | 1019.33 | 500.00 | 90.91 | 3862 | 8558 | 1877 | 30 |
| $G_{124,2.5}$ | 124 | 4 | 23 | 13 | 0.18 | 21 | 215.18 | 61.54 | 80.00 | 5124 | 456 | 1741 | 15 |
| $G_{124,5}$ | 124 | 4 | 104 | 80 | 0.09 | 94 | 53.86 | 17.50 | 58.33 | 3160 | 188 | 1003 | 9 |
| $G_{124,10}$ | 124 | 4 | 290 | 251 | 0.09 | 261 | 5.54 | 3.98 | 25.64 | 1519 | 130 | 325 | 6 |
| $G_{124,20}$ | 124 | 4 | 720 | 660 | 0.07 | 670 | 3.04 | 1.52 | 16.67 | 949 | 221 | 234 | 4 |
| mesh.148.265 | 148 | 4 | 22 | 9 | 0.20 | 22 | 101.94 | 144.44 | 100.00 | 3382 | 458 | 1197 | 10 |
| mesh.70.120 | 70 | 5 | 20 | 12 | 0.04 | 18 | 97.04 | 50.00 | 75.00 | 1401 | 279 | 1570 | 8 |
| KKT.putt01 | 115 | 5 | 104 | 82 | 0.62 | 101 | 208.23 | 23.17 | 86.36 | 2255 | 310 | 2193 | 19 |
| $G_{250,2.5}$ | 250 | 5 | 58 | 27 | 0.67 | 50 | 839.04 | 85.19 | 74.19 | 12682 | 68 | 2047 | 17 |
| $G_{250,5}$ | 250 | 5 | 209 | 147 | 0.37 | 172 | 97.98 | 17.01 | 40.32 | 6747 | 3 | 726 | 9 |
| $G_{250,10}$ | 250 | 5 | 629 | 523 | 0.28 | 540 | 11.74 | 3.25 | 16.04 | 2605 | 5 | 257 | 5 |
| $G_{250,25}$ | 250 | 5 | 1418 | 1263 | 0.25 | 1277 | 4.90 | 1.11 | 9.03 | 1545 | 8 | 176 | 3 |
| mesh.138.232 | 138 | 6 | 24 | 9 | 0.12 | 21 | 332.90 | 133.33 | 80.00 | 5974 | 650 | 1605 | 16 |


| graph | $n$ | $k$ | UB | DNN |  | DNN+cuts |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | imp. <br> \% | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | \#tri. <br> cuts | \#ind. set cuts | \#iter. | \#outer loops |
| spinglass2pm_18 | 324 | 3 | -256 | -291 | 11.40 | -270 | 1095.10 | 7.22 | 60.00 | 5274 | 7686 | 1797 | 8 |
| spinglass2pm_21 | 441 | 3 | -352 | -395 | 34.63 | -370 | 1204.90 | 6.33 | 58.14 | 3280 | 5540 | 2786 | 4 |
| spinglass2pm_24 | 576 | 3 | -453 | -512 | 67.53 | -474 | 1962.94 | 7.42 | 64.41 | 3662 | 7858 | 2178 | 4 |
| spinglass2pm_27 | 729 | 3 | -580 | -656 | 114.62 | -607 | 3169.04 | 7.47 | 64.47 | 7289 | 7291 | 2587 | 4 |
| spinglass2pm_30 | 900 | 3 | -723 | -809 | 219.84 | -748 | 6713.41 | 7.54 | 70.93 | 7922 | 10078 | 3442 | 4 |
| spinglass2pm_18 | 324 | 4 | -263 | -293 | 9.63 | -271 | 297.85 | 7.51 | 73.33 | 6417 | 63 | 1003 | 4 |
| spinglass2pm_20 | 400 | 4 | -326 | -361 | 14.45 | -333 | 164.25 | 7.76 | 80.00 | 7821 | 179 | 487 | 4 |
| spinglass2pm_22 | 484 | 4 | -399 | -441 | 29.47 | -408 | 408.03 | 7.48 | 78.57 | 16686 | 254 | 535 | 7 |
| spinglass2pm_24 | 576 | 4 | -465 | -516 | 47.83 | -475 | 668.25 | 7.95 | 80.39 | 16769 | 511 | 740 | 6 |
| spinglass2pm_26 | 676 | 4 | -551 | -611 | 55.27 | -562 | 897.17 | 8.02 | 81.67 | 19628 | 652 | 687 | 6 |
| spinglass2pm_28 | 784 | 4 | -635 | -709 | 97.88 | -656 | 3102.63 | 7.48 | 71.62 | 19350 | 250 | 1500 | 5 |
| spinglass2pm_32 | 1024 | 4 | -823 | -922 | 170.91 | -842 | 3978.22 | 8.68 | 80.81 | 15198 | 162 | 1572 | 3 |
| spinglass2pm_20 | 400 | 5 | -327 | -362 | 14.31 | -333 | 225.31 | 8.01 | 82.86 | 8000 | 0 | 695 | 4 |
| spinglass2pm_25 | 625 | 5 | -507 | -566 | 64.71 | -515 | 1221.12 | 9.01 | 86.44 | 12500 | 0 | 1150 | 4 |
| spinglass2pm_30 | 900 | 5 | -736 | -818 | 151.53 | -751 | 2110.83 | 8.19 | 81.71 | 22500 | 0 | 1116 | 5 |

Table 4.7: Three-dimensional spinglass graphs, $k \in\{3,4,5\}$.

| graph | $n$ | $k$ | UB | DNN |  | DNN+cuts |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { imp. } \\ \% \end{gathered}$ | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | \#tri. <br> cuts | $\begin{aligned} & \text { \#ind. } \\ & \text { set cuts } \end{aligned}$ | \#iter. | \#outer <br> loops |
| $G_{500,2.5}$ | 500 | 4 | 96 | 43 | 19.91 | 70 | 1491.29 | 62.79 | 50.94 | 24966 | 34 | 1419 | 10 |
| $G_{500,5}$ | 500 | 4 | 375 | 253 | 10.78 | 297 | 867.97 | 17.39 | 36.07 | 23646 | 23 | 1063 | 10 |
| $G_{500,10}$ | 500 | 4 | 1016 | 806 | 6.81 | 834 | 70.81 | 3.47 | 13.33 | 8737 | 36 | 380 | 5 |
| $G_{500,20}$ | 500 | 4 | 2753 | 2418 | 5.61 | 2434 | 27.93 | 0.66 | 4.78 | 5232 | 44 | 254 | 4 |
| $G_{1000,2.5}$ | 1000 | 4 | 200 | 74 | 174.11 | 117 | 7204.31 | 58.11 | 34.13 | 49998 | 2 | 1519 | 10 |
| $G_{1000,5}$ | 1000 | 4 | 767 | 485 | 81.82 | 570 | 3446.47 | 17.53 | 30.14 | 50000 | 0 | 1340 | 10 |
| $G_{1000,10}$ | 1000 | 4 | 2219 | 1732 | 52.46 | 1774 | 295.29 | 2.42 | 8.62 | 16558 | 5 | 636 | 4 |
| $G_{1000,20}$ | 1000 | 4 | 5422 | 4620 | 45.23 | 4640 | 132.93 | 0.43 | 2.49 | 8229 | 12 | 414 | 3 |
| $U_{500,5}$ | 500 | 4 | 22 | 2 | 50.02 | 6 | 1596.20 | 200.00 | 20.00 | 23854 | 1146 | 2966 | 10 |
| $U_{500,10}$ | 500 | 4 | 115 | 25 | 46.02 | 54 | 1208.51 | 116.00 | 32.22 | 24672 | 328 | 1560 | 10 |
| $U_{500,20}$ | 500 | 4 | 358 | 162 | 68.67 | 304 | 1502.73 | 87.65 | 72.45 | 24228 | 772 | 2061 | 10 |
| $U_{500,40}$ | 500 | 4 | 1020 | 549 | 94.56 | 927 | 1644.27 | 68.85 | 80.25 | 23946 | 1054 | 2546 | 10 |
| $U_{1000,5}$ | 1000 | 4 | 11 | 1 | 649.01 | 3 | 5195.14 | 200.00 | 20.00 | 50000 | 0 | 4565 | 10 |
| $U_{1000,10}$ | 1000 | 4 | 182 | 23 | 218.23 | 51 | 2929.24 | 121.74 | 17.61 | 49820 | 180 | 2081 | 10 |
| $U_{1000,20}$ | 1000 | 4 | 735 | 160 | 433.61 | 362 | 4847.54 | 126.25 | 35.13 | 49447 | 553 | 2832 | 10 |
| $U_{1000,40}$ | 1000 | 4 | 1596 | 671 | 534.88 | 1342 | 5502.61 | 100.00 | 72.54 | 49100 | 900 | 3147 | 10 |
| Table 4.8: Large $G$ and $U$ graphs from [228], $k=4$. |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | DNN |  | DNN+cuts |  |  |  |  |  |  |  |
| graph | $n$ | $k$ | UB | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | imp. $\%$ | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | \#tri. <br> cuts | $\begin{aligned} & \text { \#ind. } \\ & \text { set cuts } \end{aligned}$ | \#iter. | \#outer <br> loops |
| $G_{500,2.5}$ | 500 | 5 | 108 | 49 | 17.47 | 78 | 1366.47 | 59.18 | 49.15 | 24973 | 27 | 1416 | 10 |
| $G_{500,5}$ | 500 | 5 | 403 | 279 | 8.77 | 319 | 625.09 | 14.34 | 32.26 | 19998 | 2 | 924 | 8 |
| $G_{500,10}$ | 500 | 5 | 1113 | 880 | 5.56 | 899 | 57.74 | 2.16 | 8.15 | 6387 | 0 | 304 | 4 |
| $G_{500,20}$ | 500 | 5 | 2987 | 2618 | 4.47 | 2624 | 22.93 | 0.23 | 1.63 | 2868 | 1 | 207 | 3 |
| $G_{1000,2.5}$ | 1000 | 5 | 228 | 83 | 157.92 | 133 | 7207.64 | 60.24 | 34.48 | 50000 | 0 | 1514 | 10 |


| $G_{1000,5}$ | 1000 | 5 | 854 |  | 532 |  | 71.13 |  | 612 |  | 2758.98 | 15.04 |  | 24.84 | 45000 |  | 0 |  | 1140 |  | 9 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $G_{1000,10}$ | 1000 | 5 | 2449 |  | 1881 |  | 40.91 |  | 1908 |  | 205.35 | 1.44 |  | 4.75 | 11499 |  | 0 | 0 | 510 |  | 3 |
| $G_{1000,20}$ | 1000 | 5 | 5904 |  | 4990 |  | 35.05 |  | 4997 |  | 66.87 | 0.14 |  | 0.77 | 4215 |  | 0 | 0 | 285 |  | 2 |
| $U_{500,5}$ | 500 | 5 | 21 |  | 3 |  | 34.36 |  | 8 |  | 1391.01 | 166.67 |  | 27.78 | 24772 |  | 228 |  | 1770 |  | 10 |
| $U_{500,10}$ | 500 | 5 | 98 |  | 35 |  | 48.88 |  | 72 |  | 1310.76 | 105.71 |  | 58.73 | 24744 |  | 256 |  | 1812 |  | 10 |
| $U_{500,20}$ | 500 | 5 | 439 |  | 216 |  | 67.61 |  | 373 |  | 1302.06 | 72.69 |  | 70.40 | 24638 |  | 362 |  | 1911 |  | 10 |
| $U_{500,40}$ | 500 | 5 | 1230 |  | 771 |  | 85.37 |  | 1165 |  | 1807.18 | 51.10 |  | 85.84 | 24090 |  | 910 |  | 2499 |  | 10 |
| $U_{1000,5}$ | 1000 | 5 | 29 |  | 1 |  | 529.28 |  | 4 |  | 5104.28 | 300.00 |  | 10.71 | 50000 |  | 0 | ) 3 | 3908 |  | 10 |
| $U_{1000,10}$ | 1000 | 5 | 220 |  | 34 |  | 175.33 |  | 76 |  | 3285.83 | 123.53 |  | 22.58 | - 49879 |  | 121 |  | 2058 |  | 10 |
| $U_{1000,20}$ | 1000 | 5 | 716 |  | 216 |  | 411.07 |  | 469 |  | 4581.52 | 117.13 |  | 50.60 | 49336 |  | 664 |  | 2706 |  | 10 |
| $U_{1000,40}$ | 1000 | 5 | 1836 |  | 887 |  | 497.69 |  | 1609 |  | 4864.62 | 81.40 |  | 76.08 | 49010 |  | 990 |  | 2625 |  | 10 |
| Table 4.9: Large $G$ and $U$ graphs from [228], $k=5$. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| graph |  | $n$ | $m_{1}$ | UB | DNN |  |  |  | DNN+cuts |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  | cloc | cktime (s) | $\begin{gathered} \hline \mathbf{L B} \\ \text { (rounded) } \end{gathered}$ |  | clocktime (s) |  | $\begin{gathered} \text { imp. } \\ \% \end{gathered}$ | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ |  | $\begin{gathered} \text { \#BQP } \\ \text { cuts } \end{gathered}$ |  | \#iter. | r. | \#outer loops |  |
|  | 4,2.5 |  | 124 | 75 | 12 |  | 7 |  | 11.29 |  | 10 | 0 | 7077.11 | 42.86 |  | 60.00 | 7399 |  | 29486 |  | 21 |  |
|  | 124,5 | 124 | 87 | 49 |  | 42 |  | 14.23 |  | 49 | 9 | 616.35 | 16.67 |  | 100.00 | 4054 |  | 15861 |  | 13 |  |
|  | 24,10 | 124 | 81 | 159 |  | 141 |  | 12.26 |  | 150 |  | 154.13 | 6.38 |  | 50.00 | 2429 |  | 12633 |  | 8 |  |
|  | 24,20 | 124 | 75 | 414 |  | 399 |  | 12.93 |  | 412 |  | 373.60 | 3.26 |  | 86.67 | 2716 |  | 19067 |  | 8 |  |
| $G_{25}$ | 0,2.5 | 250 | 175 | 20 |  | 12 |  | 202.63 |  | 16 | 6 | 7200.52 | 33.33 |  | 50.00 | 15750 |  | 38639 |  | 21 |  |
|  | 250,5 | 250 | 163 | 101 |  | 74 |  | 129.11 |  | 90 | 0 | 1710.97 | 21.62 |  | 59.26 | 9702 |  | 23906 |  | 15 |  |
|  | 50,10 | 250 | 150 | 343 |  | 289 |  | 81.51 |  | 310 |  | 441.65 | 7.27 |  | 38.89 | 5454 |  | 16145 |  | 9 |  |
|  | 5,20 | 250 | 175 | 673 |  | 627 |  | 94.47 |  | 641 |  | 473.62 | 2.23 |  | 30.43 | 4416 |  | 17655 |  | 6 |  |
| mesh. 13 | 8. 232 | 138 | 90 | 6 |  | 2 |  | 21.51 |  |  | 6 | 2647.73 | 200.00 |  | 100.00 | 7452 |  | 24424 |  | 17 |  |
| mesh. 14 | 8.265 | 148 | 89 | 9 |  | 4 |  | 36.96 |  |  | 8 | 3074.68 | 100.00 |  | 80.00 | 7104 |  | 31656 |  | 16 |  |
| mesh. 27 | 4.469 | 274 | 192 | 10 |  | 2 |  | 259.55 |  |  | 5 | 7200.52 | 150.00 |  | 37.50 | 15618 |  | 43682 |  | 19 |  |
| mesh. 7 | . 120 | 70 | 46 | 6 |  | 4 |  | 4.54 |  |  | 6 | 374.20 | 50.00 |  | 100.00 | 1470 |  | 12431 |  | 6 |  |
| mesh. 7 | 4.129 | 74 | 45 | 7 |  | 5 |  | 4.02 |  |  | 7 | 28.01 | 40.00 |  | 100.00 | 1110 |  | 7695 |  | 4 |  |
| KKT_lo | wt01 | 82 | 58 | 13 |  | 10 |  | 3.57 |  | 13 | 3 | 371.49 | 30.00 |  | 100.00 | 984 |  | 1216 |  | 3 |  |

Table 4.10: Instances considered in [193, Table 9], $m=\left(m_{1}, n-m_{1}\right)$.

| graph | $n$ | $m_{1}$ | UB | DNN |  | DNN+cuts |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | imp. <br> \% | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | $\begin{gathered} \text { \#BQP } \\ \text { cuts } \end{gathered}$ | \#iter. | \#outer loops |
| spinglass2pm_18 | 324 | 195 | -222 | -245 | 252.02 | -227 | 7200.07 | 7.35 | 78.26 | 12960 | 35081 | 8 |
| spinglass2pm_20 | 400 | 280 | -258 | -267 | 735.60 | -262 | 7200.39 | 1.87 | 55.56 | 14000 | 47002 | 7 |
| spinglass2pm_21 | 441 | 287 | -294 | -318 | 667.70 | -304 | 7200.25 | 4.40 | 58.33 | 15435 | 41725 | 7 |
| spinglass2pm_22 | 484 | 291 | -336 | -369 | 831.91 | -347 | 7200.64 | 5.96 | 66.67 | 24200 | 34795 | 10 |
| $G_{500,5}$ | 500 | 350 | 173 | 127 | 804.22 | 149 | 7200.63 | 17.32 | 47.83 | 20000 | 32734 | 10 |
| $G_{500,10}$ | 500 | 325 | 563 | 463 | 518.01 | 490 | 3101.66 | 5.83 | 27.00 | 11361 | 22746 | 8 |
| $G_{500,20}$ | 500 | 300 | 1682 | 1497 | 426.13 | 1534 | 2405.54 | 2.47 | 20.00 | 9646 | 21665 | 6 |
| $G_{1000,5}$ | 1000 | 650 | 387 | 260 | 5048.12 | 271 | 7200.38 | 4.23 | 8.66 | 8000 | 26970 | 2 |
| $G_{1000,10}$ | 1000 | 600 | 1304 | 1048 | 3400.21 | 1074 | 7200.49 | 2.48 | 10.16 | 16000 | 22496 | 4 |
| $G_{1000,20}$ | 1000 | 700 | 2844 | 2503 | 4952.74 | 2514 | 7200.66 | 0.44 | 3.23 | 12000 | 26158 | 3 |

Table 4.11: Large $G$ graphs from [228] and two-dimensional spinglass graphs, $m=\left(m_{1}, n-m_{1}\right)$.

# The Chvátal-Gomory procedure for integer SDPs with applications in combinatorial optimization 

## Chapter summary

In this chapter we study the well-known Chvátal-Gomory (CG) procedure for the class of integer semidefinite programs (ISDPs). We prove several results regarding the hierarchy of relaxations obtained by iterating this procedure. We also study different formulations of the elementary closure of spectrahedra. A polyhedral description of the elementary closure for a specific type of spectrahedra is derived by exploiting total dual integrality for SDPs. Moreover, we show how to exploit (strengthened) CG cuts in a branch-and-cut framework for ISDPs. Different from existing algorithms in the literature, the separation routine in our approach exploits both the semidefinite and the integrality constraints. We provide separation routines for several common classes of binary SDPs resulting from combinatorial optimization problems. In the second part of the chapter we present a comprehensive application of our approach to the quadratic traveling salesman problem (QTSP). Based on the algebraic connectivity of the directed Hamiltonian cycle, two ISDPs that model the QTSP are introduced. We show that the CG cuts resulting from these formulations contain several wellknown families of cutting planes. Numerical results illustrate the practical strength of the CG cuts in our branch-and-cut algorithm, which outperforms alternative ISDP solvers and is able to solve large QTSP instances to optimality.

### 5.1 Introduction

Convex integer nonlinear programs (CINLPs) are optimization problems in which the objective function is convex and the continuous relaxation of the feasible region is a convex set. Nonlinearities in CINLPs can appear in both the objective function and/or the constraints. Motivated by their numerous applications and their ability to generalize several well-known problem classes, CINLPs have been studied for decades. In this chapter we focus on a specific class of CINLPs: the integer semidefinite programs (ISDPs). These problems can be formulated as:

$$
\begin{array}{ll}
\text { sup } & b^{\top} x \\
\text { s.t. } & C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}, \quad x \in \mathbb{Z}^{m} \tag{5.1}
\end{array}
$$

with $b \in \mathbb{R}^{m}, C, A_{i} \in \mathcal{S}^{n}$, where $\mathcal{S}^{n}$ denotes the cone of symmetric matrices of order $n$. Since integer linear programs belong to the family of ISDPs, problems of the form (5.1) are generally $\mathcal{N} \mathcal{P}$-hard to solve.

Although CINLPs have been studied extensively, see e.g., the survey of Bonami et al. [51], the special case of ISDPs has received attention only very recently. This is remarkable, as the mixture of positive semidefiniteness and integrality leads naturally to a broad range of applications, e.g., in architecture [71, 376], signal processing [155, 302] and combinatorial optimization $[156,318]$. For a more detailed overview of applications of ISDPs, we refer the reader to [156, 234].

Only a few solution approaches for solving SDPs with integrality constraints have been considered. Gally et al. [156] propose a general framework called SCIP-SDP for solving mixed integer semidefinite programs (MISDPs) using a branch-and-bound ( $\mathrm{B} \& \mathrm{~B}$ ) procedure with continuous SDPs as subproblems. They show that strict duality of the relaxations is maintained in the $\mathrm{B} \& \mathrm{~B}$ tree and study several solver components. Alternatively, Kobayashi and Takano [234] propose a cutting-plane algorithm that initially relaxes the positive semidefinite (PSD) constraint and solves a mixed integer linear programming problem, where the PSD constraint is imposed dynamically via cutting planes. This leads to a general branch-and-cut (B\&C) algorithm for solving MISDPs. A third project that encounters general ISDPs is YALMIP [257]. However, it is noted by the authors of [156] and [234] that the branch-and-bound ISDP solver in YALMIP is not yet competitive to the performance of the other two methods. Recently, Matter and Pfetsch [270] study different presolving strategies for MISDPs for both the B\&B and B\&C approach.

Apart from solution methods for solving general ISDPs or MISDPs, there are several other approaches in the literature that aim to solve integer problems by utilizing SDP relaxations in a B\&B framework. Although these approaches are very related to problems of the form (5.1) in the sense that they also combine semidefinite programs with a branching strategy, they differ in the sense that the problem at hand is not necessarily formulated as a MISDP. Examples are the BiqCrunch solver for constrained binary quadratic problems [240] and the Biq Mac solver for unconstrained binary quadratic problems [318].

In the light of improving the performance of the $\mathrm{B} \& \mathrm{C}$ algorithm of [234], we consider the exploitation of cutting planes for ISDPs. Practical algorithms for CINLPs have benefited a lot from the addition of strong cutting planes, see e.g., [23, 24, 39, 349], where many of these cutting plane frameworks are based on generalizations from integer linear programming. Among the most well-known cutting planes for integer linear programs (ILPs) are
the Chvátal-Gomory (CG) cuts [77, 179]. Gomory [179] introduced these cuts to design the first finite cutting plane algorithm for ILPs. Chvátal [77] later generalized this notion and introduced the closure of all such cuts that leads to a hierarchy of relaxations of the ILP with increasing strength. Chvátal [77] and Schrijver [329] prove that this hierarchy is finite for bounded real polyhedra and rational polyhedra, respectively. Later on, the CG procedure is introduced for more general convex sets, see e.g., [ $56,88,89,98,112$ ]. In particular, Çezik and Iyengar [68] show how to generate CG cuts for CINLPs where the continuous relaxation of the feasible region is conic representable.

A leading application in this chapter is a combinatorial optimization problem that can be modeled as an ISDP: the quadratic traveling salesman problem (QTSP). Jäger and Molitor [224] introduce the QTSP as the problem of finding a Hamiltonian cycle in a graph that minimizes the total interaction costs among consecutive arcs. The problem is motivated by an important application in bioinformatics [135, 224], but has also applications in telecommunication, precision farming and robotics, see e.g., [6, 127, 370]. The QTSP is $\mathcal{N} \mathcal{P}$-hard in the strong sense and is currently considered as one of the hardest combinatorial optimization problems to solve in practice. For a more comprehensive overview of the background, applications and solution approaches proposed for the QTSP, we refer the reader to Section 1.4.1.

### 5.1.1 Main results and outline

In this chapter we consider the Chvátal-Gomory procedure for ISDPs from a theoretical as well as a practical point of view. On the theoretical side, we derive several results on the elementary closure of all CG cuts for spectrahedra. On the practical side, we show how to apply these cuts in a generic branch-and-cut algorithm for ISDPs that exploits both the positive semidefiniteness and the integrality of the problem. We extensively study the application of this new approach to the QTSP, which confirms the practical strength of the proposed method.

We start by reformulating a CG cut for a spectrahedron in terms of its data matrices in combination with the elements from the dual cone. This leads to a constructive description of the elementary closure of spectrahedra rather than the implicit description that is known for general convex sets. Equivalent to the case of polyhedra, the elementary closure operation can be repeated, leading to a hierarchy of stronger approximations of the integer hull of the spectrahedron. For the case of bounded spectrahedra, we provide a compact proof of a homogeneity property for the elementary closure operation that is based on a theorem of alternatives and Dirichlet's approximation theorem. We prove this property for halfspaces that are sufficient to describe any compact convex set. Homogeneity is the cornerstone in showing that the elementary closure of a bounded spectrahedron is polyhedral. Although the latter result is known in the literature, our proof significantly simplifies compared to the general proofs given in [56, 89]. Finally, we exploit the recently introduced notion of total dual integrality for SDPs [65] to derive a closed-form expression for the elementary closure of spectrahedra defined by a totally dual integral linear matrix inequality. We additionally provide a characterization of bounded spectrahedra with this property and several more general sufficient conditions.

It is known that the practical strength of CG cuts in integer linear programming is mainly due to their application in branch-and-bound methods. In this vein, we propose a generic branch-and-cut (B\&C) framework for ISDPs. Our algorithm initially relaxes the PSD constraint and solves a mixed integer linear program (MILP), where the PSD constraint is
imposed iteratively via CG and/or strengthened CG cuts. To derive strengthened CG cuts, we use a similar approach to the one for rational polyhedra by Dash et al. [93]. Our B\&C algorithm is an extension of the algorithm of [234], in which separation is only based on positive semidefiniteness without taking into account the integrality of the variables. Our approach also builds up on the work by Çezik and Iyengar [68], in which the authors leave the separation of CG cuts for conic problems as an open problem and do not include these cuts in their computational study. We provide an example of our approach for two common classes of binary SDPs that frequently appear in combinatorial optimization.

In the third part of this chapter we apply our results to a difficult-to-solve combinatorial optimization problem: the quadratic traveling salesman problem. We derive two ISDP formulations of this problem based on the notion of algebaic connectivity. To solve these models using our B\&C algorithm, we propose several CG separation routines and show that various of these routines lead to well-known cuts for the QTSP. Computational results on a large set of benchmark QTSP instances show that the practical potential of our new method is twofold. The method significantly outperforms the ISDP solvers from the literature, whereas it also provides competitive results to the state-of-the-art QTSP solution method of [135].

This chapter is organized as follows. In Section 5.2 we study the Chvátal-Gomory procedure for spectrahedra. Section 5.3 provides a CG-based B\&C framework for general ISDPs and provides specific CG separation routines for two classes of binary SDPs. In Section 5.4 we formally define the QTSP and present two ISDP formulations of this problem. Numerical results are given in Section 5.5.

### 5.2 The Chvátal-Gomory procedure for ISDPs

In this section we study the extension of the cutting-plane procedure by Chvátal [77] and Gomory [179] for integer linear programs to the class of integer semidefinite programs. We show that several concepts, such as the Chvátal-Gomory closure and the Chvátal rank, can be generalized to ISDPs. We start by recollecting the procedure for general convex sets.

### 5.2.1 The Chvátal-Gomory procedure

Let $C \subseteq \mathbb{R}^{m}$ be a nonempty closed convex set and let $C_{I}$ be its integer hull, i.e., $C_{I}:=$ $\operatorname{conv}\left(C \cap \mathbb{Z}^{m}\right)$. The Chvátal-Gomory cutting-plane procedure is introduced by Chvátal [77] and Gomory [179] and is regarded to be among the most celebrated results in integer programming. The CG procedure aims at systematically identifying valid inequalities for $C$ that cut off noninteger solutions. By adding these new cuts to the relaxation and repeating this process, one obtains a hierarchy of stronger relaxations that converges to $C_{I}$.

The CG procedure relies on the notion of rational halfspaces. A rational halfspace is of the form $H=\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$ for some $c \in \mathbb{Q}^{m}, d \in \mathbb{Q}$. It is known that all such halfspaces can be represented by $c \in \mathbb{Z}^{m}$ such that the entries of $c$ are relatively prime. If $H=\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$ with $c \in \mathbb{Z}^{m}, \operatorname{gcd}(c)=1$, then $H_{I}=\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq\lfloor d\rfloor\right\}$.

Definition 5.1. The elementary closure of a closed convex set $C$ is the set

$$
\begin{equation*}
\mathrm{cl}_{C G}(C):=\bigcap_{\substack{(c, d) \in \mathbb{Q}^{m} \times \mathbb{Q} \\ C \subseteq H=\left\{x: c^{\top} x \leq d\right\}}} H_{I} . \tag{5.2}
\end{equation*}
$$

Equivalently, the elementary closure of $C$ can be written as:

$$
\begin{equation*}
\operatorname{cl}_{C G}(C)=\bigcap_{\substack{(c, d) \in \mathbb{Z}^{m} \times \mathbb{R} \\ C \subseteq\left\{x: c^{\top} x \leq d\right\}}}\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq\lfloor d\rfloor\right\} \tag{5.3}
\end{equation*}
$$

and we will primarily use this form in this chapter. The inequalities that define $\mathrm{cl}_{C G}(C)$ in (5.3) are known as CG cuts [179]. One can verify that $C_{I} \subseteq \operatorname{cl}_{C G}(C)$. When $C$ is compact, we can exploit the following proposition due to Dadush et al. [88] and De Carli Silva and Tunçel [65].

Proposition 5.2. If $C \subseteq \mathbb{R}^{m}$ is a compact convex set, then

$$
C=\bigcap_{\substack{(c, d) \in \mathbb{Z}^{m} \times \mathbb{R} \\ C \subseteq\left\{x: c^{\top} x \leq d\right\}}}\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\} .
$$

It follows from Proposition 5.2 that for compact convex sets $C$ we have $\operatorname{cl}_{C G}(C) \subseteq C$. We can now repeat the procedure by defining $C^{(0)}:=C$ and $C^{(k+1)}:=\operatorname{cl}_{C G}\left(C^{(k)}\right)$ for all integer $k \geq 0$, where $C^{(k)}$ is referred to as the $k$ th CG closure of $C$. For any compact convex set $C$ this leads to the hierarchy $C_{I} \subseteq \ldots \subseteq C^{(k+1)} \subseteq C^{(k)} \subseteq \ldots \subseteq C^{(0)}=C$. The smallest $k$ for which $C_{I}=C^{(k)}$ is known as the Chvátal rank of $C$. In the same vein, the Chvátal rank of an inequality $c^{\top} x \leq d$ valid for $C_{I}$ is defined as the smallest $k$ such that $C^{(k)} \subseteq\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$.

Remark 5.3. Observe that for an unbounded closed convex set $C, \operatorname{cl}_{C G}(C) \subseteq C$ does not have to hold. For instance, the irrational halfspace $\left\{x \in \mathbb{R}^{2}: x_{1}+\sqrt{2} x_{2} \leq 0\right\}$ is not contained in any halfspace of the form $\left\{x \in \mathbb{R}^{2}: c^{\top} x \leq d\right\}$ with $c \in \mathbb{Z}^{2}$. Therefore, $\operatorname{cl}_{C G}(C)$ is the intersection over an empty set of halfspaces, resulting in $\operatorname{cl}_{C G}(C)=\mathbb{R}^{2}$.

The finiteness of the Chvátal rank is proven in the literature for bounded real polyhedra [77], unbounded rational polyhedra [329] and conic representable sets in the 0/1cube [68]. However, the Chvátal rank for unbounded real polyhedra can be infinite as shown by Schrijver [329]. Schrijver also shows that the elementary closure of a rational polyhedron is a rational polyhedron. This result is later generalized to irrational polytopes [112], bounded rational ellipsoids [98], strictly convex bodies [88] and general compact convex sets [56, 89]. As a consequence, the Chvátal rank of these sets is also known to be finite.

### 5.2.2 The elementary closure of spectrahedra

We now apply the notions from Section 5.2.1 to integer semidefinite programming problems in standard primal and dual forms. On top of the general definition given in the previous section, we derive alternative formulations of the elementary closure of spectrahedra.

Let $b \in \mathbb{R}^{m}, C \in \mathcal{S}^{n}$ and $A_{i} \in \mathcal{S}^{n}$ for all $i \in[m]$. An ISDP in standard primal form is given by:

$$
\left(P_{I S D P}\right) \begin{cases}\inf & \langle C, X\rangle  \tag{5.4}\\ \text { s.t. } & \left\langle A_{i}, X\right\rangle=b_{i} \quad \text { for all } i \in[m], \\ & X \succeq \mathbf{0}, X \in \mathbb{Z}^{n \times n},\end{cases}
$$

while an ISDP in standard dual form is given by:

$$
\left(D_{I S D P}\right) \begin{cases}\text { sup } & b^{\top} x  \tag{5.5}\\ \text { s.t. } & C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0} \\ & x \in \mathbb{Z}^{m}\end{cases}
$$

Using standard techniques, one can syntactically rewrite an integer SDP from primal form to dual form and vice versa. Consistent with most of the literature, we mainly consider, but not restrict ourselves to, ISDPs in dual form.

The continuous relaxation of the feasible set of (5.5) is defined as follows:

$$
\begin{equation*}
P:=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\} \tag{5.6}
\end{equation*}
$$

The set $P$ is a spectrahedron that is a closed, semialgebraic and convex set. We define the integer hull of $P$ to be $P_{I}:=\operatorname{conv}\left(P \cap \mathbb{Z}^{m}\right)$, i.e., the convex hull of the integral points in $P$. We briefly consider some illustrative examples of spectrahedra and their integer hulls.

Example 5.4 (Examples in $\mathbb{R}^{2}$ ). Let

$$
C=\left(\begin{array}{ll}
0 & 3 \\
3 & 3
\end{array}\right), A_{1}=\left(\begin{array}{cc}
-3 & 1 \\
1 & 1
\end{array}\right) \text { and } A_{2}=\left(\begin{array}{cc}
0.5 & 1 \\
1 & 0
\end{array}\right)
$$

Then, the induced spectrahedron $P$ in the dual form (5.6) is the semialgebraic set of points in $\mathbb{R}^{2}$ described by the quadratic inequality $4 x_{1}^{2}+x_{2}^{2} \leq 15 x_{1}+4 \frac{1}{2} x_{2}-1 \frac{1}{2} x_{1} x_{2}-9$. This spectrahedron is bounded and given in Figure 5.1a.

Let $Q$ be described by (5.6) with

$$
C=\left(\begin{array}{ll}
1 & 0 \\
0 & 0
\end{array}\right), A_{1}=\left(\begin{array}{rr}
0 & -1 \\
-1 & 0
\end{array}\right) \text { and } A_{2}=\left(\begin{array}{rr}
0 & 0 \\
0 & -2
\end{array}\right)
$$

The spectrahedron $Q$ is the unbounded semialgebraic set $\left\{x \in \mathbb{R}^{2}: x_{2} \geq \frac{1}{2} x_{1}^{2}\right\}$, see Figure 5.1 b .

Example 5.5 (Example in $\mathbb{R}^{3}$ ). Let

$$
\begin{aligned}
C & =\left(\begin{array}{ll}
1 & 2 \\
2 & 2
\end{array}\right) \oplus\left(\begin{array}{ll}
5 & 0 \\
0 & 5
\end{array}\right), & A_{1} & =\left(\begin{array}{cc}
-1 & 0.5 \\
0.5 & 1
\end{array}\right) \oplus\left(\begin{array}{ll}
0 & 0 \\
0 & 0
\end{array}\right), \\
A_{2} & =\left(\begin{array}{cc}
-0.6 & 0.3 \\
0.3 & 0
\end{array}\right) \oplus\left(\begin{array}{rr}
1 & 0 \\
0 & -1
\end{array}\right), & A_{3} & =\left(\begin{array}{cc}
0.5 & 2 \\
2 & -3
\end{array}\right) \oplus\left(\begin{array}{ll}
0 & 0 \\
0 & 0
\end{array}\right),
\end{aligned}
$$

and let $P$ be the induced spectrahedron of the form (5.6). Then, $P$ is the semialgebraic set in $\mathbb{R}^{3}$ described by $1 \frac{1}{4} x_{1}^{2}+\frac{9}{100} x_{2}^{2}+5 \frac{1}{2} x_{3}^{2} \leq-2+3 x_{1}+2 \frac{2}{5} x_{2}+10 x_{3}-\frac{9}{10} x_{1} x_{2}+\frac{3}{5} x_{2} x_{3}+1 \frac{1}{2} x_{1} x_{3}$, $1+x_{1}+\frac{3}{5} x_{2}-\frac{1}{2} x_{3} \geq 0,2-x_{1}+3 x_{3} \geq 0,-5 \leq x_{2}$ and $x_{2} \leq 5$, see Figure 5.2.


Figure 5.1: Spectrahedra $P$ and $Q$ defined in Example 5.4. Their corresponding integer hulls are given by the dark gradient areas.


Figure 5.2: Spectrahedron $P$ in $\mathbb{R}^{3}$ defined in Example 5.5.

Throughout this chapter, we make an additional assumption regarding the representation of the spectrahedron $P$ in the form (5.6). Namely, in case $P$ is not full-dimensional, we augment its representation matrices, without changing the spectrahedron itself, see also [314]. This assumption is required to obtain an alternative formulation of the elementary closure.
Assumption 5.6. In case $P$ is not full-dimensional, i.e., the subspace $\mathcal{L}:=\operatorname{Aff}(P)^{\perp}$ is nontrivial, we extend $C$ and $A_{i}, i \in[m]$, to

$$
C \oplus \operatorname{Diag}\left(L x_{0}\right) \oplus-\operatorname{Diag}\left(L x_{0}\right) \quad \text { and } \quad A_{i} \oplus \operatorname{Diag}\left(\ell_{i}\right) \oplus-\operatorname{Diag}\left(\ell_{i}\right) \text { for all } i \in[m]
$$

where $L:=\left[\begin{array}{lll}\ell_{1} & \ldots & \ell_{m}\end{array}\right] \in \mathbb{R}^{\operatorname{dim}(\mathcal{L}) \times m}$ is a matrix whose rows form a basis for $\mathcal{L}$ and $x_{0} \in P$.
Observe that the extended map in Assumption 5.6 does not change the spectrahedron $P$.

Indeed, if $\mathcal{L}=\operatorname{Aff}(P)^{\perp}$ is non-trivial, then $L\left(x-x_{0}\right)=\mathbf{0}$, or equivalently, $L x-L x_{0}=\mathbf{0}$ for all $x \in P$. Hence, the additional linear constraints are redundant for all $x \in P$. No additional constraints are needed if $P$ is full-dimensional, e.g., the spectrahedra in Example 5.4 and 5.5.

In the remaining part of this section we study the elementary closure, see Definition 5.1, of spectrahedra in primal and dual standard forms. The proofs of several results that will follow rely on the following semidefinite version of the theorem of alternatives, see e.g., Balakrishnan and Vandenberghe [26]. In this proposition, $X \nsucceq \mathbf{0}$ denotes that $X$ is positive semidefinite, but unequal to the zero matrix.
Proposition 5.7 (Theorem of the alternatives for $\operatorname{SDP}[26])$. Let $C, A_{1}, \ldots, A_{m} \in \mathcal{S}^{n}$. Then, at most one of the following is true:

1. There exists an $X \succ \mathbf{0},\left\langle A_{i}, X\right\rangle=0$ for all $i \in[m]$ and $\langle C, X\rangle \leq 0$;
2. There exists an $x \in \mathbb{R}^{m}$ such that $C-\sum_{i=1}^{m} A_{i} x_{i} \nsucceq \mathbf{0}$.

Moreover, if there exists no $x \in \mathbb{R}^{m}$ such that $\sum_{i=1}^{m} A_{i} x_{i} \nsucceq \mathbf{0}$, then exactly one of the statements above is true.

Since $C-\sum_{i=1}^{m} A_{i} x_{i}$ is positive semidefinite if and only if $\left\langle C-\sum_{i=1}^{m} A_{i} x_{i}, U\right\rangle \geq 0$ for all $U \in \mathcal{S}_{+}^{n}$, we can rewrite $P$ as follows:

$$
\begin{align*}
P & =\left\{x \in \mathbb{R}^{m}:\left\langle C-\sum_{i=1}^{m} A_{i} x_{i}, U\right\rangle \geq 0, U \in \mathcal{S}_{+}^{n}\right\} \\
& =\bigcap_{U \in \mathcal{S}_{+}^{n}}\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\langle A_{i}, U\right\rangle \leq\langle C, U\rangle\right\} \tag{5.7}
\end{align*}
$$

Moreover, since $P$ is a closed convex set, we can write $P$ as the intersection of the halfspaces that contain it:

$$
\begin{equation*}
P=\bigcap_{\substack{(c, d) \in \mathbb{R}^{m+1} \\ P \subseteq\left\{x: c^{\top} x \leq d\right\}}}\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\} \tag{5.8}
\end{equation*}
$$

It is clear that all halfspaces in the intersection of (5.7) are contained in the intersection (5.8). In order to show the converse statement, we consider the geometric and algebraic polar sets studied in [314].

The geometric polar of spectrahedron $P$ is defined as

$$
P^{\circ}:=\left\{y \in \mathbb{R}^{m}: y^{\top} x \leq 1 \text { for all } x \in P\right\}
$$

which coincides with the general definition of the polar set of a convex set. Ramana and Goldman [314] moreover define the algebraic polar of a spectrahedron $P$ in the form (5.6) as

$$
P^{*}:=\left\{\left[\left\langle A_{1}, U\right\rangle \ldots\left\langle A_{m}, U\right\rangle\right]^{\top}:\langle C, U\rangle \leq 1, U \in \mathcal{S}_{+}^{n}\right\}
$$

It is shown in [314] that if $\mathbf{0} \in P$, then $P^{\circ}=P^{*}+\operatorname{Aff}(P)^{\perp}$. Under Assumption 5.6, this implies that $P^{\circ}=P^{*}$ whenever $\mathbf{0} \in P$. This equality forms the key of the following result.
Theorem 5.8. Let $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be a nonempty spectrahedron. Let $(c, d) \in \mathbb{R}^{m+1}$ be such that $P \subseteq\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$. Then, there exists a matrix $U \in \mathcal{S}_{+}^{n}$ such that $\left\langle A_{i}, U\right\rangle=c_{i}$ for all $i \in[m]$ and $\langle C, U\rangle \leq d$.

Chapter 5. The Chvátal-Gomory procedure for ISDPs with applications in combinatorial optimization

Proof. First, suppose $(c, d)$ is such that $P$ is contained in the hyperplane $\left\{x \in \mathbb{R}^{m}: c^{\top} x=d\right\}$. Then, $c \in \operatorname{Aff}(P)^{\perp}$. This implies $\alpha^{\top} L=c^{\top}$ for some $\alpha \in \mathbb{R}^{\operatorname{dim}(\mathcal{L})}$ and $\alpha^{\top} L x_{0}=c^{\top} x_{0}=d$, where $L$ and $x_{0}$ are as defined in Assumption 5.6. Now, by taking $U=\mathbf{0} \oplus \operatorname{Diag}(\max (\alpha, \mathbf{0}) \oplus$ $\operatorname{Diag}(-\min (\alpha, \mathbf{0}))$, it follows directly from Assumption 5.6 that $\left\langle A_{i}, U\right\rangle=c_{i}$ for all $i \in[m]$ and $\langle C, U\rangle=d$.

Let us now assume that $(c, d)$ is such that $P \nsubseteq\left\{x \in \mathbb{R}^{m}: c^{\top} x=d\right\}$. Then, there exists a $x_{0} \in P$ such that $c^{\top} x_{0}<d$. Let $\hat{C}:=C-\sum_{i=1}^{m} A_{i}\left(x_{0}\right)_{i}$ and consider the translated spectrahedron $\hat{P}:=P-x_{0}=\left\{x \in \mathbb{R}^{m}: \hat{C}-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$. For this translated spectrahedron we know $\mathbf{0} \in \hat{P}$, from which it follows that $\hat{P}^{\circ}=\bar{P}^{*}$.

From $P \subseteq\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$, we obtain $\hat{P} \subseteq\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq \hat{d}\right\}$, where $\hat{d}:=$ $d-c^{\top} x_{0}>0$. Consequently, we know that $(1 / \hat{d}) c^{\top} x \leq 1$ for all $x \in \hat{P}$, which implies that $(1 / \hat{d}) c \in \hat{P}^{\circ}=\hat{P}^{*}$. Then, we know there exists an $\hat{U} \in \mathcal{S}_{+}^{n}$ such that

$$
\left\langle A_{i}, \hat{U}\right\rangle=\frac{1}{\hat{d}} c_{i} \quad \text { for all } i \in[m], \quad \text { and } \quad\langle\hat{C}, \hat{U}\rangle \leq 1 .
$$

Taking $U:=\hat{d} \hat{U}$ and using the definitions of $\hat{d}$ and $\hat{C}$, we obtain $\left\langle A_{i}, U\right\rangle=c_{i}$ for all $i \in[m]$ and $\langle C, U\rangle-\sum_{i=1}^{m}\left\langle A_{i}, U\right\rangle\left(x_{0}\right)_{i} \leq d-c^{\top} x_{0}$, yielding $\langle C, U\rangle \leq d$.

Using the representation of $P$ given by (5.7) and the result of Theorem 5.8, we now provide an alternative formulation of the elementary closure for spectrahedra of the form $P$. We have,

$$
\begin{equation*}
\operatorname{cl}_{C G}(P)=\bigcap_{\substack{U \in \mathcal{S}_{n}^{n} \text { s.t. } \\\left\langle A_{i}, U\right\rangle \in \mathbb{Z}, i \in[m]}}\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\langle A_{i}, U\right\rangle \leq\lfloor\langle C, U\rangle\rfloor\right\} . \tag{5.9}
\end{equation*}
$$

Hence, any possible CG cut for a spectrahedron is constructed by a matrix $U \in \mathcal{S}_{+}^{n}$ such that $\left\langle A_{i}, U\right\rangle \in \mathbb{Z}$ for $i \in[m]$.

A similar alternative definition of the elementary closure of spectrahedra in standard primal form can be obtained. Let $Q \subseteq \mathcal{S}^{n}$ denote the continuous relaxation of the feasible set of (5.4), i.e.,

$$
\begin{aligned}
Q & =\left\{X \in \mathcal{S}^{n}:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m], X \succeq \mathbf{0}\right\} \\
& =\left\{X \in \mathcal{S}^{n}:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m],\langle X, U\rangle \geq 0, U \in \mathcal{S}_{+}^{n}\right\} \\
& =\left\{X \in \mathcal{S}^{n}:\left\langle X, U+\sum_{i=1}^{m} A_{i} \lambda_{i}\right\rangle \geq \sum_{i=1}^{m} b_{i} \lambda_{i}, U \in \mathcal{S}_{+}^{n}, \lambda \in \mathbb{R}^{m}\right\},
\end{aligned}
$$

where the last equality follows from the fact that $(U, \lambda)=\left(\mathbf{0}, \mathbf{e}_{i}\right)$ and $(U, \lambda)=\left(\mathbf{0},-\mathbf{e}_{i}\right)$ lead to the cuts $\left\langle A_{i}, X\right\rangle \geq b_{i}$ and $\left\langle A_{i}, X\right\rangle \leq b_{i}$, respectively. Now, the elementary closure of $Q$ can be described by the following intersection of CG cuts:

$$
\begin{equation*}
\operatorname{cl}_{C G}(Q)=\bigcap_{\substack{(U, \lambda) \in \mathcal{S}_{+}^{n} \times \mathbb{R}^{m} \text { s.t. } \\ U+\sum_{i=1}^{m} A_{i} \lambda_{i} \in \mathbb{Z}^{n \times n}}}\left\{X \in \mathcal{S}^{n}:\left\langle X, U+\sum_{i=1}^{m} A_{i} \lambda_{i}\right\rangle \geq\left\lceil\sum_{i=1}^{m} b_{i} \lambda_{i}\right\rceil\right\} . \tag{5.10}
\end{equation*}
$$

For many SDPs resulting from applications the spectrahedra that define the feasible sets are contained in the cone of nonnegative vectors or matrices. In that case, alternative equivalent formulations of the elementary closure can be given, see also [68].

Theorem 5.9. Let $P=\left\{x \in \mathbb{R}_{+}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be a nonempty spectrahedron. Then, $\operatorname{cl}_{C G}(P)$ can equivalently be written as

$$
\begin{equation*}
\operatorname{cl}_{C G}(P)=\bigcap_{U \in \mathcal{S}_{+}^{n}}\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\lfloor\left\langle A_{i}, U\right\rangle\right\rfloor \leq\lfloor\langle C, U\rangle\rfloor\right\} \tag{5.11}
\end{equation*}
$$

Similarly, let $Q=\left\{X \in \mathcal{S}^{n}:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m], X \succeq \mathbf{0}, X \geq \mathbf{0}\right\}$. Then $\operatorname{cl}_{C G}(Q)$ can equivalently be written as

$$
\begin{equation*}
\operatorname{cl}_{C G}(Q)=\bigcap_{(U, \lambda) \in \mathcal{S}_{+}^{n} \times \mathbb{R}^{m}}\left\{X \in \mathcal{S}^{n}:\left\langle X,\left\lceil U+\sum_{i=1}^{m} A_{i} \lambda_{i}\right\rceil\right\rangle \geq\left\lceil\sum_{i=1}^{m} b_{i} \lambda_{i}\right\rceil\right\} \tag{5.12}
\end{equation*}
$$

Proof. We prove the statement for the dual form (5.11). The proof for the primal form is similar.

Let $\overline{\operatorname{cl}_{C G}(P)}:=\bigcap_{U \in \mathcal{S}_{+}^{n}}\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\lfloor\left\langle A_{i}, U\right\rangle\right\rfloor \leq\lfloor\langle C, U\rangle\rfloor\right\}$ and let $\operatorname{cl}_{C G}(P)$ be as given in (5.9). The inclusion $\overline{\mathrm{cl}_{C G}(P)} \subseteq \mathrm{cl}_{C G}(P)$ is obvious, as any halfspace in the intersection defining $\mathrm{cl}_{C G}(P)$ is also in the intersection defining $\overline{\mathrm{cl}_{C G}(P)}$. Now, given some $U \in \mathcal{S}_{+}^{n}$, consider the halfspace $\bar{H}=\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\lfloor\left\langle A_{i}, U\right\rangle\right\rfloor \leq\lfloor\langle C, U\rangle\rfloor\right\}$ that is included in the intersection defining $\overline{\mathrm{cl}_{C G}(P)}$. Since $P \subseteq \mathbb{R}_{+}^{n}$, we know

$$
\begin{aligned}
P \subseteq\left\{x \in \mathbb{R}_{+}^{m}: \sum_{i=1}^{m} x_{i}\left\langle A_{i}, U\right\rangle \leq\langle C, U\rangle\right\} & \subseteq\left\{x \in \mathbb{R}_{+}^{m}: \sum_{i=1}^{m} x_{i}\left\lfloor\left\langle A_{i}, U\right\rangle\right\rfloor \leq\langle C, U\rangle\right\} \\
& \subseteq\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\lfloor\left\langle A_{i}, U\right\rangle\right\rfloor \leq\langle C, U\rangle\right\}
\end{aligned}
$$

Now we apply Theorem 5.8 to the latter halfspace. It follows that there exists a matrix $V \in \mathcal{S}_{+}^{n}$ such that

$$
\left\langle A_{i}, V\right\rangle=\left\lfloor\left\langle A_{i}, U\right\rangle\right\rfloor \quad \text { for all } i \in[m], \quad \text { and } \quad\langle C, V\rangle \leq\langle C, U\rangle
$$

We define $H:=\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\langle A_{i}, V\right\rangle \leq\lfloor\langle C, V\rangle\rfloor\right\}$. Since $\lfloor\langle C, V\rangle\rfloor \leq\lfloor\langle C, U\rangle\rfloor$, it follows that the halfspace $\bar{H}$ contains the halfspace $H$, while $H$ is contained in the intersection of $\operatorname{cl}_{C G}(P)$ given in (5.9). Since this construction can be repeated for all halfspaces in the intersection (5.11) defining $\overline{\mathrm{cl}_{C G}(P)}$, it follows that $\operatorname{cl}_{C G}(P) \subseteq \overline{\operatorname{cl}_{C G}(P)}$.

Example 5.10. Let us reconsider the bounded spectrahedron $P$ defined in Example 5.4. The elementary closure $\operatorname{cl}_{C G}(P)$ of this spectrahedron is the intersection of six rational halfspaces, represented by the dashed lines in Figure 5.3. Each such halfspace is obtained from a rational halfspace $\left\{x \in \mathbb{R}^{2}: c^{\top} x \leq d\right\}$ containing $P$, where $d$ is shifted towards $P_{I}$ until the corresponding hyperplane hits an integral point. The integer hull $P_{I}$ is the intersection of only five halfspaces. Thus, for this example we have $P_{I} \subsetneq \operatorname{cl}_{C G}(P) \subsetneq P$.

In Section 5.2 .4 we provide a polyhedral description of the elementary closure of spectrahedra that satisfy the notion of total dual integrality.


Figure 5.3: Spectrahedron $P$, its integer hull $P_{I}$ and its elementary closure $\operatorname{cl}_{C G}(P)$.

### 5.2.3 The Chvátal rank of bounded spectrahedra

In this section we derive several results on the sequence of relaxations resulting from the Chvátal-Gomory procedure. Although some of these results are already known for general compact convex sets, we provide simplified proofs for the case of bounded spectrahedra. Throughout this section we assume $P$ to be a spectrahedron of the form (5.6) that is bounded. For unbounded sets it is in general not even clear whether $C^{(k+1)} \subseteq C^{(k)}$.

It is known that the Chvátal rank of compact convex sets is finite, including the special case of bounded spectrahedra.

Proposition 5.11. Let $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be bounded. Then, $P^{(k)}=P_{I}$ for some finite $k$.

Proof. See e.g., [77, 89].
Next, we aim to prove a homogeneity property of the CG procedure for bounded spectrahedra, which states that the elementary closure operation commutes with taking the intersection with supporting hyperplanes. This property plays a key role in showing that the elementary closure of $P$ is a rational polytope, following the proof of Braun and Pokutta [56]. We provide a simplified proof of this property for bounded spectrahedra, which can be seen as the conic analogue to a polyhedral result of Schrijver [329]. In the proof we restrict ourselves to halfspaces of the form $\left\{x \in \mathbb{R}^{m}: w^{\top} x \leq d\right\}$ where $w \in \mathbb{Z}^{m}$ and $d \in \mathbb{R}$. It follows from Proposition 5.2 that these halfspaces are sufficient to describe a compact convex set.

Before we show the main theorem, we need a chain of intermediate results, starting with a proposition regarding the condition of Proposition 5.7.

Proposition 5.12. Let $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be a nonempty and bounded spectrahedron. Then there does not exist an $x \in \mathbb{R}^{m}$ such that $\sum_{i=1}^{m} A_{i} x_{i} \nsucceq \mathbf{0}$.

Proof. Since $P$ is nonempty, there exists a point $x^{*} \in P$, i.e., $C-\sum_{i=1}^{m} A_{i} x_{i}^{*} \succeq \mathbf{0}$. Now suppose there exists a point $\hat{x}$ such that $\sum_{i=1}^{m} A_{i} \hat{x}_{i} \varsubsetneqq \mathbf{0}$. Then clearly $\hat{x} \neq \mathbf{0}_{m}$ and for
all $t \geq 0$ we have

$$
C-\sum_{i=1}^{m} A_{i} x_{i}^{*}+t \sum_{i=1}^{m} A_{i} \hat{x}_{i}=C-\sum_{i=1}^{m} A_{i}\left(x_{i}^{*}-t \hat{x}_{i}\right) \succeq \mathbf{0},
$$

i.e., $x^{*}-t \hat{x} \in P$ for all $t \geq 0$. Thus, $P$ is unbounded, so such $\hat{x}$ cannot exist.

We also need Dirichlet's approximation theorem and its weakened version.
Proposition 5.13 (Dirichlet's Approximation Theorem). Let $d \in \mathbb{R}$ and $N \geq 2$ be a positive integer. Then there exist integers $p$ and $q$ with $1 \leq p \leq N$ such that $|p d-q| \leq \frac{1}{N}$.

We now derive its one-sided variant below.
Corollary 5.14 (One-sided Approximation Theorem). Let $d \in \mathbb{R}$ and $N \geq 2$ be a positive integer number. Then there exists an integer $p \in \mathbb{Z}_{+}$such that

$$
p d-\lfloor p d\rfloor \leq \frac{1}{N} .
$$

Proof. By Dirichlet's Theorem, we know that for the given $d$ and $N$, there exist integers $q_{1}$ and $q_{2}$ with $1 \leq q_{1} \leq N$ such that $\left|q_{1} d-q_{2}\right| \leq \frac{1}{N}$. If $q_{1} d \geq q_{2}$, then we have

$$
q_{1} d-\left\lfloor q_{1} d\right\rfloor \leq q_{1} d-q_{2}=\left|q_{1} d-q_{2}\right| \leq \frac{1}{N}
$$

so the choice $p=q_{1}$ leads to the desired result. Next, we consider the case $q_{1} d<q_{2}$, for which we have $-\frac{1}{N} \leq q_{1} d-q_{2}<0$. Let $M \geq 1$ be the smallest integer that satisfies $M\left(q_{1} d-q_{2}\right) \leq-\frac{N-1}{N}$, which exists because $q_{1} d-q_{2}<0$. For this $M$ we must have $-1 \leq M\left(q_{1} d-q_{2}\right)$. Namely, if $M\left(q_{1} d-q_{2}\right)<-1$, then $(M-1)\left(q_{1} d-q_{2}\right) \leq-\frac{N-1}{N}$, contradicting the minimality of $M$. Thus,

$$
-1 \leq M\left(q_{1} d-q_{2}\right) \leq-\frac{N-1}{N} \quad \text { or equivalently, } \quad 0 \leq M q_{1} d-\left(M q_{2}-1\right) \leq \frac{1}{N}
$$

Since $M q_{2}-1$ is integer, it follows that

$$
M q_{1} d-\left\lfloor M q_{1} d\right\rfloor \leq M q_{1} d-\left(M q_{2}-1\right) \leq \frac{1}{N},
$$

so taking $p=M q_{1}$ gives the desired result.
We are now ready to present a simplified proof of Braun and Pokutta [56] for the homogeneity property of the elementary closure of bounded spectrahedra, see also Proposition 1 in [89].

Theorem 5.15 (Homogeneity property of elementary closure). Let a bounded spectrahedron $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be contained in a halfspace $\left\{x \in \mathbb{R}^{m}: w^{\top} x \leq d\right\}$ with $w \in \mathbb{Z}^{m}$ and $d \in \mathbb{R}$. Let $K:=\left\{x \in \mathbb{R}^{m}: w^{\top} x=d\right\}$. Then, $\operatorname{cl}_{C G}(P) \cap K=\operatorname{cl}_{C G}(P \cap K)$.

Proof. If $P$ is empty the claim is obvious, hence we assume that $P$ is nonempty.
The inclusion $\mathrm{cl}_{C G}(P \cap K) \subseteq \mathrm{cl}_{C G}(P) \cap K$ is trivial. To prove the converse statement, we assume that $H$ is a rational halfspace containing $P \cap K$, i.e., we may assume that $H=\{x \in$ $\left.\mathbb{R}^{m}: v^{\top} x \leq \alpha\right\}$ where $v$ is integer-valued with $\operatorname{gcd}(v)=1$. It suffices to show that there
exists a halfspace $\hat{H}$ containing $P$ such that $\hat{H}_{I} \cap K \subseteq H_{I}$. As $P \cap K$ is the intersection of all such halfspaces $H$, we establish $\operatorname{cl}_{C G}(P) \cap K \subseteq \operatorname{cl}_{C G}(P \cap K)$.

For each $i \in[m]$ we define the following extended matrix $\tilde{A}_{i} \in \mathcal{S}^{n+2}$ :

$$
\tilde{A}_{i}:=\left[\begin{array}{ccc}
A_{i} & \mathbf{0} & \mathbf{0} \\
\mathbf{0}^{\top} & -w_{i} & 0 \\
\mathbf{0}^{\top} & 0 & -v_{i}
\end{array}\right]
$$

We first show that there does not exist an $x \in \mathbb{R}^{m}$ such that $\sum_{i=1}^{m} \tilde{A}_{i} x_{i} \nsucceq \mathbf{0}$. For the sake of contradiction, suppose such a vector exists, i.e., we have $\sum_{i=1}^{m} A_{i} \tilde{x}_{i} \succeq \mathbf{0}, w^{\top} \tilde{x} \leq 0$ and $v^{\top} \tilde{x} \leq 0$ for some $\tilde{x}$, but not all of them are satisfied with equality. Since $P$ is nonempty and bounded, it follows from Proposition 5.12 that there does not exist an $x \in \mathbb{R}^{m}$ such that $\sum_{i=1}^{m} A_{i} x_{i} \varsubsetneqq \mathbf{0}$. Hence, we must have $\sum_{i=1}^{m} A_{i} \tilde{x}_{i}=\mathbf{0}$. This implies that either $w^{\top} \tilde{x}<0$ or $v^{\top} \tilde{x}<0$, or both.

Since $P$ is contained in $\left\{x \in \mathbb{R}^{m}: w^{\top} x \leq d\right\}$, it follows from Theorem 5.8 that there exists $T \succeq \mathbf{0}$ such that $\left\langle A_{i}, T\right\rangle=w_{i}$ for all $i \in[m]$. Since $\sum_{i=1}^{m} A_{i} \tilde{x}_{i}=\mathbf{0}$, we have

$$
\left\langle\sum_{i=1}^{m} A_{i} \tilde{x}_{i}, T\right\rangle=\sum_{i=1}^{m} \tilde{x}_{i}\left\langle A_{i}, T\right\rangle=w^{\top} \tilde{x}=0 .
$$

Since $P \cap K$ is contained in $H=\left\{x \in \mathbb{R}^{m}: v^{\top} x \leq \alpha\right\}$, we can apply Theorem 5.8 to the extended linear matrix inequality $(C \oplus d \oplus-d)-\sum_{i=1}^{m}\left(A_{i} \oplus w_{i} \oplus-w_{i}\right) x_{i} \succeq \mathbf{0}$ to show that $v_{i}=\left\langle A_{i}, S\right\rangle+\beta w_{i}$ for some $S \succeq \mathbf{0}$ and $\beta \in \mathbb{R}$. From this it follows that $v^{\top} \tilde{x}=0$. We conclude that there exists no $x \in \mathbb{R}^{m}$ such that $\sum_{i=1}^{m} \tilde{A}_{i} x_{i} \varsubsetneqq \mathbf{0}$.

Next, we define the following extended matrix $\tilde{C} \in \mathcal{S}^{n+2}$ and parameter $\epsilon>0$ :

$$
\tilde{C}:=\left[\begin{array}{ccc}
C & \mathbf{0} & \mathbf{0} \\
\mathbf{0}^{\top} & -d & 0 \\
\mathbf{0}^{\top} & 0 & -(\alpha+\epsilon)
\end{array}\right] \quad \text { and } \quad \epsilon:= \begin{cases}\frac{1}{2}(\lceil\alpha\rceil-\alpha) & \text { if } \alpha \text { is not integer, } \\
\frac{1}{2} & \text { otherwise. }\end{cases}
$$

Since $P \cap K$ is contained in $H$, it follows that $(P \cap K) \cap\left\{x \in \mathbb{R}^{m}: v^{\top} x \geq \alpha+\epsilon\right\}=\emptyset$. As $P \cap K$ can be expressed as $P \cap\left\{x \in \mathbb{R}^{m}: w^{\top} x \geq d\right\}$, we equivalently know that there does not exist an $x \in \mathbb{R}^{m}$ such that $\tilde{C}-\sum_{i=1}^{m} \tilde{A}_{i} x_{i} \nsucceq \mathbf{0}$. We can now apply Proposition 5.7 to this system, from where it follows that the first of the two alternative statements should be satisfied. Hence, there exist $\hat{U} \succ \mathbf{0}, \lambda>0$ and $\mu>0$ such that $\left\langle A_{i}, \hat{U}\right\rangle-w_{i} \lambda-v_{i} \mu=0$ for all $i \in[m]$ and $\langle C, \hat{U}\rangle-d \lambda-(\alpha+\epsilon) \mu \leq 0$. Without loss of generality, we may assume that $\mu=1$ and we define

$$
\hat{\alpha}:=\langle C, \hat{U}\rangle \quad \text { and } \quad \hat{v}_{i}:=\left\langle A_{i}, \hat{U}\right\rangle \text { for all } i \in[m] .
$$

It follows from above that this particular $\hat{\alpha}$ and $\hat{v}$ satisfy

$$
\begin{equation*}
\hat{\alpha} \leq \alpha+\epsilon+d \lambda \quad \text { and } \quad \hat{v}_{i}=v_{i}+w_{i} \lambda \text { for all } i \in[m] . \tag{5.13}
\end{equation*}
$$

Also, since $\hat{U} \succ \mathbf{0}$, we know that for all $x \in P$ we have

$$
\begin{equation*}
\hat{v}^{\top} x=\sum_{i=1}^{m}\left\langle A_{i}, \hat{U}\right\rangle x_{i}=\left\langle\sum_{i=1}^{m} A_{i} x_{i}, \hat{U}\right\rangle \leq\langle C, \hat{U}\rangle=\hat{\alpha} \tag{5.14}
\end{equation*}
$$

where we use the fact that $\left\langle C-\sum_{i=1}^{m} A_{i} x_{i}, \hat{U}\right\rangle \geq 0$. Observe that the tuple $(\lambda, \hat{v}, \hat{\alpha})$ can be replaced by $\left(\lambda+\lambda_{0}, \hat{v}+\lambda_{0} w, \hat{\alpha}+\lambda_{0} d\right)$ for all $\lambda_{0} \geq 0$ without affecting (5.13) and (5.14), where for the maintenance of (5.14) we use the fact that $P \subseteq\left\{x \in \mathbb{R}^{m}: w^{\top} x \leq d\right\}$. Now we choose $\lambda_{0}$ such that $\lambda+\lambda_{0} \in \mathbb{Z}_{+}$and $d_{f}:=d\left(\lambda+\lambda_{0}\right)-\left\lfloor d\left(\lambda+\lambda_{0}\right)\right\rfloor<\epsilon$, which can be done by Corollary 5.14.

Define $\hat{H}:=\left\{x \in \mathbb{R}^{m}:\left(\hat{v}+\lambda_{0} w\right)^{\top} x \leq \hat{\alpha}+\lambda_{0} d\right\}$. It follows from (5.14) that $P \subseteq \hat{H}$. Moreover, by exploiting (5.13) and the definitions of $d_{f}$ and $\epsilon$, we have

$$
\begin{aligned}
\hat{H}_{I} \cap K & \subseteq\left\{x \in \mathbb{R}^{m}:\left(\hat{v}+\lambda_{0} w\right)^{\top} x \leq\left\lfloor\hat{\alpha}+\lambda_{0} d\right\rfloor\right\} \cap\left\{x \in \mathbb{R}^{m}: w^{\top} x=d\right\} \\
& \subseteq\left\{x \in \mathbb{R}^{m}: v^{\top} x+w^{\top} x\left(\lambda+\lambda_{0}\right) \leq\left\lfloor\alpha+\epsilon+d\left(\lambda+\lambda_{0}\right)\right\rfloor\right\} \cap\left\{x \in \mathbb{R}^{m}: w^{\top} x=d\right\} \\
& \subseteq\left\{x \in \mathbb{R}^{m}: v^{\top} x+d_{f} \leq\left\lfloor\alpha+\epsilon+d_{f}\right\rfloor\right\} \\
& \subseteq\left\{x \in \mathbb{R}^{m}: v^{\top} x \leq\left\lfloor\alpha+\epsilon+d_{f}\right\rfloor\right\} \\
& =\left\{x \in \mathbb{R}^{m}: v^{\top} x \leq\lfloor\alpha\rfloor\right\}=H_{I},
\end{aligned}
$$

where the last inclusion follows from the fact that $d_{f} \geq 0$. For the first equality in the last line we exploit the fact that $\epsilon+d_{f}<2 \epsilon$, where $\epsilon$ is chosen such that $2 \epsilon=1$ if $\alpha$ is integer and $2 \epsilon=\lceil\alpha\rceil-\alpha$ otherwise. In both cases, we have $\lfloor\alpha\rfloor=\left\lfloor\alpha+\epsilon+d_{f}\right\rfloor$.

The result of Theorem 5.15 holds for any halfspace $\left\{x \in \mathbb{R}^{m}: w^{\top} x \leq d\right\}$ with $w \in \mathbb{Z}^{m}$ containing $P$. In particular, it holds for all such halfspaces that support $P$, meaning that $P \cap K \neq \emptyset$, where $K$ is the corresponding hyperplane. In such case, the set $P \cap K$ defines a face of the spectrahedron. It is known that all proper faces of spectrahedra are exposed [314], meaning that they can be obtained as the intersection of $P$ with a supporting hyperplane. Note, however, that for the faces of bounded spectrahedra these hyperplanes are not necessarily such that the entries in $w$ are integral, even if the data matrices describing the spectrahedron are rational (as is the case for polyhedra).

Homogeneity plays a key role in Braun and Pokutta's [56] proof for the polyhedrality of the elementary closure of compact convex sets. For the sake of completeness, we include this result here for the case of bounded spectrahedra.

Theorem 5.16 (Dadush et al. [89], Braun and Pokutta [56]). The elementary closure $\operatorname{cl}_{C G}(P)$ of a bounded spectrahedron $P$ is a rational polytope.

From Theorem 5.16 and the fact that the elementary closure of a rational polytope is again a rational polytope [329], it follows that the finite sequence

$$
P=P^{(0)} \supseteq P^{(1)} \supseteq \ldots \supseteq P^{(k)} \supseteq P^{(k+1)} \supseteq \ldots \supseteq P_{I}
$$

consists of rational polyhedra from the first closure onwards. Observe that the boundedness assumption cannot be relaxed. Indeed, if $P$ is unbounded, it is not even clear whether $P_{I}$ is a polyhedron, as the following example suggests.

Example 5.17. Consider the spectrahedron $Q$ in Example 5.4. The integer hull $Q_{I}$ is the convex hull of the integer points in the epigraph of $f\left(x_{1}\right)=\frac{1}{2} x_{1}^{2}$. This convex hull is not
polyhedral. To verify this, observe that the recession cone of $Q_{I}$ is contained in the recession cone of $Q$, which is $\operatorname{rec}(Q):=\left\{x \in \mathbb{R}^{2}: x_{2} \geq 0, x_{1}=0\right\}$. Since $Q_{I}$ is unbounded and $\operatorname{rec}(Q)$ has only one ray, the recession cone of $Q_{I}$ must also be $\operatorname{rec}(Q)$. If $Q_{I}$ would be polyhedral, this implies that the halfspace $x_{1} \leq N$ supports $Q_{I}$ for some finite value of $N$. However, this cannot be true as $Q_{I}$ contains integral points $\left(x_{1}, x_{2}\right) \in \mathbb{Z}^{2}$ for arbitrarily large $x_{1}$.

One can verify that $\operatorname{cl}_{C G}(Q)=Q_{I}$. Namely, each facet of $Q_{I}$ is induced by a line between the points $\left(2 k, 2 k^{2}\right),\left(2(k-1), 2(k-1)^{2}\right) \in \mathbb{Z}^{2}$ for any $k \in \mathbb{Z}$. Let such line for a fixed $k$ be described by $x_{2}=c x_{1}+d$ with $c, d \in \mathbb{Z}$. Then, the parallel line $x_{2}=c x_{1}+d-1$ lies strictly below $Q$. This implies that the halfspace $x_{2} \geq c x_{1}+d-1+\epsilon$ for any $\epsilon>0$ contains $Q$ and that its integer hull is $x_{2} \geq c x_{1}+d$. Therefore, all facet-defining inequalities of $Q_{I}$ have Chvátal rank one and $\operatorname{cl}_{C G}(Q)=Q_{I}$. This shows that $\mathrm{cl}_{C G}(Q)$ is not a polyhedron.

### 5.2.4 The elementary closure of spectrahedra and total dual integrality

In this section we derive a class of spectrahedra for which we can find an explicit expression for the elementary closure. For rational polyhedra such an expression can be derived from a totally dual integral representation of the linear system [329]. It is therefore not surprising that a similar construction can be applied for bounded spectrahedra, albeit with a bit more technicalities. After connecting total dual integrality for SDPs to the elementary closure, we derive a characterization and several sufficient conditions for a linear matrix inequality to be totally dual integral.

Recently, De Carli Silva and Tunçel [65] introduced a notion of total dual integrality for SDPs. The authors of [65] argue that the term integrality in SDPs should be defined with care. For instance, the rank-one property that is sometimes used in the literature as the notion of SDP integrality is proven to be primal-dual asymmetric and therefore not the favoured choice. Instead, the authors of [65] propose a notion of SDP integrality that is based on a set of integer generating matrices.

Definition 5.18 (Property $\left.(\mathrm{PZ})_{\mathcal{V}}\right)$. Let $\mathcal{V}:=\left\{V_{1}, \ldots, V_{k}\right\} \subseteq \mathcal{S}_{+}^{n}$ be a finite set of integer PSD matrices. A matrix $X \in \mathcal{S}_{n}^{+}$satisfies integrality property $(\mathrm{P} \mathbb{Z})_{\mathcal{V}}$ if

$$
\begin{equation*}
X=\sum_{j \in[k]} y_{j} V_{j} \quad \text { for some } y: \mathcal{V} \rightarrow \mathbb{Z}_{+} \tag{PZ}
\end{equation*}
$$

The authors of [65] restricted to the set $\mathcal{V}=\left\{\mathbb{1}_{S} \mathbb{1}_{S}^{\top}: S \subseteq[n]\right\}$, which could be seen as a natural embedding for the combinatorial problems that are considered in [65]. One could argue, however, that this embedding is rather arbitrary. For that reason, we consider a general set of generating matrices. Note that the matrices $X$ that satisfy property $(\mathrm{P} \mathbb{Z}) \mathcal{V}$ are also integral in the sense that $X \in \mathbb{Z}^{n \times n}$. To overcome confusion between these definitions, we will always explicitly refer to property $(\mathrm{P} \mathbb{Z})_{\mathcal{V}}$ if that notion is meant.

Now we present the definition of total dual integrality for SDPs, see also [65].
Definition 5.19 (Total dual integrality). Let $Z \subseteq \mathbb{Z}^{m}$. A linear matrix inequality (LMI) $C$ $\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}$ is called totally dual integral (TDI) on $Z$ if there exists some finite set of integer PSD matrices $\mathcal{V}$ such that, for every $b \in Z$, the SDP dual to $\sup \left\{b^{\top} x: C-\right.$ $\left.\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ has an optimal solution satisfying property $(\mathrm{P} \mathbb{Z})_{\mathcal{V}}$ whenever it has an optimal solution.

A main difference with the original definition of total dual integrality for polyhedra, see e.g. [121], is that we restrict the objective vectors for which dual integrality should hold to a
subset $Z$ of $\mathbb{Z}^{m}$. As explained in [65], this follows from the fact that semidefinite programs often follow from lifted formulations. For instance, $Z$ could be the range from a linear lifting map, e.g., $Z=\left\{0 \oplus b^{\prime}: b^{\prime} \in \mathbb{Z}^{m-1}\right\}$.

Based on this restriction to vectors in $Z$, it makes sense to consider a relaxed version of the CG closure in which we take the intersection of halfspaces induced by coefficient vectors in $Z$. More precisely, we define the CG closure with respect to $Z$ as

$$
\begin{equation*}
\operatorname{cl}_{C G}(P, Z):=\bigcap_{\substack{(c, d) \in Z \times \mathbb{R} \\ P \subseteq\left\{x: c^{\top} x \leq d\right\}}}\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq\lfloor d\rfloor\right\} . \tag{5.15}
\end{equation*}
$$

This relaxation of the CG closure is also considered in the literature, see e.g., [88, 89]. The standard CG closure $\operatorname{cl}_{C G}(P)$ that we considered so far equals $\mathrm{cl}_{C G}\left(P, \mathbb{Z}^{m}\right)$.

The following theorem shows that if a spectrahedron is defined by an LMI that is TDI on $Z$, its (relaxed) CG closure $\mathrm{cl}_{C G}(P, Z)$ can be explicitly defined.

Theorem 5.20. Let $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be such that $C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}$ is TDI on $Z$ and satisfies Slater's condition. Let $\mathcal{V}=\left\{V_{1}, \ldots, V_{k}\right\}$ denote the corresponding generating set of integer PSD matrices and suppose $\left[\left\langle V_{j}, A_{1}\right\rangle \ldots\left\langle V_{j}, A_{m}\right\rangle\right]^{\top} \in Z$ for all $j \in[k]$. Define $B \in \mathbb{Z}^{k \times m}$ and $d \in \mathbb{Z}^{k}$ such that:

$$
B_{j, i}:=\left\langle A_{i}, V_{j}\right\rangle \quad \text { and } \quad d_{j}:=\left\lfloor\left\langle C, V_{j}\right\rangle\right\rfloor,
$$

for all $j \in[k]$ and $i \in[m]$. Then,

$$
\operatorname{cl}_{C G}(P, Z)=Q:=\left\{x \in \mathbb{R}^{m}: B x \leq d\right\} .
$$

Proof. To prove that $\operatorname{cl}_{C G}(P, Z) \subseteq Q$, observe that $V_{j} \succeq \mathbf{0}$ with $\left[\left\langle V_{j}, A_{1}\right\rangle \ldots\left\langle V_{j}, A_{m}\right\rangle\right]^{\top} \in Z$ for all $j \in[k]$. Consequently, we know that $P \subseteq\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\langle A_{i}, V_{j}\right\rangle \leq\left\langle C, V_{j}\right\rangle\right\}$. It follows from (5.15) that $\operatorname{cl}_{C G}(P, Z) \subseteq\left\{x \in \mathbb{R}^{m}: \sum_{i=1}^{m} x_{i}\left\langle A_{i}, V_{j}\right\rangle \leq\left\lfloor\left\langle C, V_{j}\right\rangle\right\rfloor\right\}$. Since all inequalities in $B x \leq d$ are of this form, it follows that $\mathrm{cl}_{C G}(P, Z) \subseteq Q$.

To prove the converse direction, let $H:=\left\{x \in \mathbb{R}^{m}: b^{\top} x \leq q\right\}$ be a halfspace containing $P$ with $b \in Z$. Since $P \subseteq H$, we have

$$
\begin{align*}
q & \geq \sup _{x}\left\{b^{\top} x: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}  \tag{5.16}\\
& =\inf _{X}\left\{\langle C, X\rangle:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m], X \succeq \mathbf{0}\right\}, \tag{5.17}
\end{align*}
$$

where strong duality among (5.16) and (5.17) holds since the former problem has a Slater feasible point. By the same argument, we know that the infimum in (5.17) is attained. Since $C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}$ is TDI on $Z$, it follows that there exists an optimal solution $\hat{X}$ to (5.17) satisfying property $(\mathrm{PZ})_{\mathcal{V}}$. In other words, there exists an $\hat{y} \in \mathbb{Z}_{+}^{k}$ such that

$$
\hat{X}=\sum_{j \in[k]} \hat{y}_{j} V_{j}, \quad\left\langle A_{i}, \hat{X}\right\rangle=b_{i} \text { for all } i \in[m], \quad \hat{X} \succeq 0 .
$$

Consequently, we have

$$
\lfloor q\rfloor \geq\lfloor\langle C, \hat{X}\rangle\rfloor=\left\lfloor\sum_{j \in[k]} \hat{y}_{j}\left\langle C, V_{j}\right\rangle\right\rfloor \geq \sum_{j \in[k]} \hat{y}_{j}\left\lfloor\left\langle C, V_{j}\right\rangle\right\rfloor=d^{\top} \hat{y} .
$$

Now, consider the following linear optimization problem and its corresponding dual:

$$
\max \left\{b^{\top} x: B x \leq d\right\}=\min \left\{d^{\top} y: y \geq \mathbf{0}, y^{\top} B=b^{\top}\right\} .
$$

Since $\hat{y} \geq \mathbf{0}$ and $\left(\hat{y}^{\top} B\right)_{i}=\sum_{j \in[k]} \hat{y}_{j}\left\langle A_{i}, V_{j}\right\rangle=\left\langle A_{i}, \hat{X}\right\rangle=b_{i}$, the solution $\hat{y}$ is feasible for the minimization problem above. This yields

$$
\max \left\{b^{\top} x: B x \leq d\right\} \leq d^{\top} \hat{y} \leq\lfloor q\rfloor .
$$

Hence, $Q \subseteq\left\{x \in \mathbb{R}^{m}: b^{\top} x \leq\lfloor q\rfloor\right\}$. Since this holds for all halfspaces $H$ induced by coefficient vectors in $Z$, it follows that $Q \subseteq \operatorname{cl}_{C G}(P, Z)$.

For the special case $Z=\mathbb{Z}^{m}$, Theorem 5.20 provides a closed-form expression for $\mathrm{cl}_{C G}(P)$. Observe that for that case the condition that $\left[\left\langle V_{j}, A_{1}\right\rangle \ldots\left\langle V_{j}, A_{m}\right\rangle\right]^{\top} \in Z$ for all $j \in[k]$ can be simplified to $\left\langle A_{i}, V_{j}\right\rangle \in \mathbb{Z}$ for all $i \in[m]$ and $j \in[k]$.

Besides providing a closed-form expression for $\mathrm{cl}_{C G}(P)$, Theorem 5.20 can be used to identify bounded spectrahedra for which $P=P_{I}$. Namely, if the matrix $C$ is such that $\left\langle C, V_{j}\right\rangle \in \mathbb{Z}$ for all $j \in[k]$, then $P \subseteq Q$. For spectrahedra that are bounded, this implies that the chain $Q=\operatorname{cl}_{C G}(P) \subseteq P \subseteq Q$ holds with equality, hence $\mathrm{cl}_{C G}(P)=P$. As $P^{(k)}=P_{I}$ for some finite $k$ for all bounded spectrahedra, we must have $P=P_{I}$. De Carli Silva and Tunçel [65] show that this, for example, happens for the SDP formulation of the Lovász theta function when the underlying graph is perfect. Similar results cannot be extended to general $Z$, since $\mathrm{cl}_{C G}(P, Z)$ is not contained in $P$ anymore.

A natural question is under which conditions a linear matrix inequality is TDI on a certain set $Z$. Below we first derive a full characterization of LMIs that are totally dual integral on the full set $\mathbb{Z}^{m}$. The characterization relates to the faces of the spectrahedron induced by the LMI. It is well-known that the faces of $\mathcal{S}_{+}^{n}$ are associated with linear subspaces of $\mathbb{R}^{n}$, see e.g., [32]. In the same vein, the facial structure of a spectrahedron can be characterized as follows.

Lemma 5.21 (Ramana and Goldman [314]). Let $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be a spectrahedron and let $F \subseteq P$ be a nonempty face of $P$. Then, there exists a subspace $\mathcal{R}_{F} \subseteq \mathbb{R}^{n}$ such that

$$
F=\left\{x \in P: \mathcal{R}_{F} \subseteq \operatorname{Nul}\left(C-\sum_{i=1}^{m} A_{i} x_{i}\right)\right\}
$$

where any point $x$ in the relative interior of $F$ satisfies $\operatorname{Nul}\left(C-\sum_{i=1}^{m} A_{i} x_{i}\right)=\mathcal{R}_{F}$.
Lemma 5.21 implies that in the particular case where the face $F$ of $P$ is an extreme point $\bar{x}$, we have $\mathcal{R}_{\bar{x}}=\operatorname{Nul}\left(C-\sum_{i=1}^{m} A_{i} \bar{x}_{i}\right)$.

For any nonempty face $F$ of $P$, we define the cone of objective vectors $b$ for which the elements in $F$ maximize $b^{\top} x$ over $P$, i.e.,

$$
\begin{equation*}
K(F):=\left\{b \in \mathbb{R}^{m}: b^{\top} y=\max \left\{b^{\top} x: x \in P\right\} \text { for all } y \in F\right\} . \tag{5.18}
\end{equation*}
$$

For any proper face $F \subseteq P$, the cone $K(F)$ is nonempty and equals the intersection over all normal cones of $P$ at the points in $F$.

Next, we recall the definition of a so-called Hilbert basis.

Definition 5.22. A set $\left\{v_{1}, \ldots, v_{k}\right\} \subseteq \mathbb{Z}^{m}$ forms a Hilbert basis if every integral vector $x \in$ cone $\left(\left\{v_{1}, \ldots, v_{k}\right\}\right)$ can be written as $x=\sum_{j=1}^{k} \alpha_{j} v_{j}, \alpha_{j} \geq 0, \alpha_{j} \in \mathbb{Z}$, for all $j \in[k]$.

By abuse of terminology, we will refer to an LMI whose solution set is bounded as a bounded LMI. The following theorem provides a full characterization of bounded LMIs that are TDI on the full set of integer vectors.

Theorem 5.23. Let the linear matrix inequality $C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}$ be bounded and assume Slater's condition holds. Then, $C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}$ is totally dual integral on $\mathbb{Z}^{m}$ if and only if there exists some finite set of integer PSD matrices $\mathcal{V}=\left\{V_{1}, \ldots, V_{k}\right\}$ such that for each extreme point $\bar{x}$ of the induced spectrahedron $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ with $K(\bar{x}) \cap \mathbb{Z}^{m} \neq \emptyset$, the vectors

$$
g_{j}:=\left[\begin{array}{lll}
\left\langle A_{1}, V_{j}\right\rangle & \ldots & \left\langle A_{m}, V_{j}\right\rangle
\end{array}\right]^{\top} \quad \text { for } j \in J:=\left\{j \in[k]: \operatorname{Col}\left(V_{j}\right) \subseteq \mathcal{R}_{\bar{x}}\right\}
$$

form a Hilbert basis of $K(\bar{x})$.
Proof. Let $b \in \mathbb{Z}^{m}$. Since $P$ is bounded, the maximum of $b^{\top} x$ over $x \in P$ is attained at a face of $P$. Thus, there exists an extreme point $\bar{x}$ of $P$ with $b \in K(\bar{x})$. As $P$ contains a Slater feasible point, we have

$$
\begin{equation*}
\max _{x}\left\{b^{\top} x: x \in P\right\}=\min _{X}\left\{\langle C, X\rangle:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m], X \succeq \mathbf{0}\right\} \tag{5.19}
\end{equation*}
$$

The point $\bar{x}$ is optimal for the maximization problem above. Complementary slackness then implies that any $X$ optimal to the dual problem should satisfy $\left(C-\sum_{i=1}^{m} A_{i} \bar{x}_{i}\right) X=\mathbf{0}$, or equivalently, $\operatorname{Col}(X) \subseteq \operatorname{Nul}\left(C-\sum_{i=1}^{m} A_{i} \bar{x}_{i}\right)=\mathcal{R}_{\bar{x}}$. To show that $g_{j}$ is contained in $K(\bar{x})$ for $j \in J$, we first observe that $V_{j}$ is feasible for the minimization problem

$$
\min _{X}\left\{\langle C, X\rangle:\left\langle A_{i}, X\right\rangle=\left(g_{j}\right)_{i}, i \in[m], X \succeq \mathbf{0}\right\}
$$

Then, since $\operatorname{Col}\left(V_{j}\right) \subseteq \mathcal{R}_{\bar{x}}$, we know that $\left(C-\sum_{i=1}^{m} A_{i} \bar{x}_{i}\right) V_{j}=\mathbf{0}$. Therefore, $\bar{x}$ and $V_{j}$ are optimal solutions to $\max _{x}\left\{g_{j}^{\top} x: x \in P\right\}$ and $\min _{X}\left\{\langle C, X\rangle:\left\langle A_{i}, X\right\rangle=\left(g_{j}\right)_{i}, i \in\right.$ [ $m$ ], $X \succeq \mathbf{0}\}$, respectively. This implies that $g_{j}$ is indeed contained in $K(\bar{x})$ for $j \in J$.

Now, suppose that the vectors $g_{j}, j \in J$ form a Hilbert basis of $K(\bar{x})$. Then, we have $b=\sum_{j \in J} \alpha_{j} g_{j}$ for some $\alpha_{j} \geq 0, \alpha_{j} \in \mathbb{Z}, j \in J$. Consequently, $X:=\sum_{j \in J} \alpha_{j} V_{j}$ is feasible for the minimization problem in (5.19) with $\operatorname{Col}(X) \subseteq \mathcal{R}_{\bar{x}}$. Since this establishes complementary slackness between $X$ and $\bar{x}$, it follows that $X$ is a dual optimal solution that satisfies property $(\mathrm{P} \mathbb{Z})_{\mathcal{V}}$.

Conversely, if the LMI is totally dual integral on $\mathbb{Z}^{m}$, it follows that the dual problem in (5.19) has an optimal solution $X$ satisfying property $(\mathrm{P} \mathbb{Z}) \mathcal{v}$. Therefore, $X=\sum_{j=1}^{k} \alpha_{j} V_{j}$ for some $\alpha_{j} \geq 0, \alpha_{j} \in \mathbb{Z}, j \in[k]$. Now, let $J^{C}:=[k] \backslash J$. Then,

$$
X=\sum_{j \in J} \alpha_{j} V_{j}+\sum_{j \in J^{C}} \alpha_{j} V_{j}
$$

By complementary slackness, we have $\operatorname{Col}(X) \subseteq \mathcal{R}_{\bar{x}}$, implying that $\operatorname{Col}\left(\sum_{j \in J^{C}} \alpha_{j} V_{j}\right)=$ $\operatorname{Col}\left(X-\sum_{j \in J} \alpha_{j} V_{j}\right) \subseteq \mathcal{R}_{\bar{x}}$. Since the $V_{j}$ 's are positive semidefinite, we also know that $\operatorname{Col}\left(\alpha_{j} V_{j}\right) \subseteq \operatorname{Col}\left(\sum_{j \in J^{C}} \alpha_{j} V_{j}\right) \subseteq \mathcal{R}_{\bar{x}}$ for all $j \in J^{C}$. However, by the definition of $J^{C}$ we have $\operatorname{Col}\left(V_{j}\right) \nsubseteq \mathcal{R}_{\bar{x}}$, so we must have $\alpha_{j}=0$ for all $j \in J^{C}$. We conclude that $X$ is a nonnegative integer combination of the matrices $V_{j}$ with $j \in J$. By the constraints of the

Chapter 5. The Chvátal-Gomory procedure for ISDPs with applications in combinatorial optimization
minimization problem in (5.19), it finally follows that $b=\sum_{j \in J} \alpha_{j} g_{j}$. As the construction can be repeated for all $b \in \mathbb{Z}^{m}$ in $K(\bar{x})$, we conclude that $\left\{g_{j}: j \in J\right\}$ indeed forms a Hilbert basis of $K(\bar{x})$. The same holds for all other extreme points $\bar{x}$ for which $K(\bar{x}) \cap \mathbb{Z}^{m} \neq \emptyset$.

Theorem 5.23 has a significant implication on the structure of the induced spectrahedron of a bounded LMI that is TDI on $\mathbb{Z}^{m}$.

Corollary 5.24. If a bounded LMI C $-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}$ that satisfies Slater's condition is totally dual integral on $\mathbb{Z}^{m}$, the spectrahedron $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ is polyhedral.

Proof. Let $h_{P}: \mathbb{R}^{m} \rightarrow \mathbb{R}$ denote the support function of $P$, i.e., $h_{P}(x):=\sup _{a \in P}\left\{x^{\top} a\right\}$. Now, consider the polytope

$$
P^{\prime}:=\left\{x \in \mathbb{R}^{m}: g_{j}^{\top} x \leq h_{P}\left(g_{j}\right) \text { for all } j \in[k]\right\}
$$

where the vectors $g_{j}, j \in[k]$ are defined as in Theorem 5.23. We clearly have $P \subseteq P^{\prime}$. To prove the converse inclusion, let $(c, d) \in \mathbb{Z}^{m} \times \mathbb{R}$ be such that $P \subseteq\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$ and let $\bar{x}$ denote an extreme point such that $c \in K(\bar{x})$. By Theorem 5.23, it follows that there exists a subset $J \subseteq[k]$ and $\alpha_{j} \geq 0, \alpha_{j} \in \mathbb{Z}, j \in J$ such that $c=\sum_{j \in J} \alpha_{j} g_{j}$. Since $g_{j} \in K(\bar{x})$, we know $h_{P}\left(g_{j}\right)=g_{j}^{\top} \bar{x}$ for all $j \in J$. Moreover, we also have $h_{P}(c)=c^{\top} \bar{x}$. We now take a conical combination of the inequalities $g_{j}^{\top} x \leq h_{P}\left(g_{j}\right)$, each with weight $\alpha_{j}$, resulting in

$$
\sum_{j \in J} \alpha_{j} g_{j}^{\top} x \leq \sum_{j \in J} \alpha_{j} h_{P}\left(g_{j}\right), \quad \text { which implies, } \quad c^{\top} x \leq \sum_{j \in J} \alpha_{j} g_{j}^{\top} \bar{x}=c^{\top} \bar{x}=h_{P}(c) \leq d
$$

Hence, the halfspace $\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$ is implied by the inequalities defining $P^{\prime}$. Since this construction can be repeated for all halfspaces of the form $\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$ and $P$ equals the intersection of all such halfspaces, see Proposition 5.2, it follows that $P^{\prime} \subseteq P$. We conclude that $P$ is a polyhedral set.

Corollary 5.24 implies that the only bounded linear matrix inequalities that may be TDI on $\mathbb{Z}^{m}$ can be described by a finite number of linear inequalities, e.g., when $C$ and $A_{i}, i \in[m]$, are diagonal or simultaneously diagonalizable. Total dual integrality on $\mathbb{Z}^{m}$ therefore happens to be quite a rare event. In general, it is $\mathcal{N} \mathcal{P}$-hard to decide whether a spectrahedron is polyhedral, see Ramana [315]. The following result provides a characterization of polyhedral spectrahedra that are full-dimensional. Observe that any spectrahedron can be transformed to a full-dimensional spectrahedron by a restriction to its affine hull.
Theorem 5.25 (Ramana [315]). Let $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be a fulldimensional spectrahedron. Then, $P$ is polyhedral if and only if there exists a nonsingular matrix $M \in \mathbb{R}^{n \times n}$ and $d, a_{i} \in \mathbb{R}^{\ell}, C^{\prime}, A_{i}^{\prime} \in \mathcal{S}^{n-\ell}, i \in[m]$, with $1 \leq \ell \leq n$ such that for all $x \in \mathbb{R}^{m}$ we have

$$
M\left(C-\sum_{i=1}^{m} A_{i} x_{i}\right) M^{\top}=\left[\begin{array}{cc}
C^{\prime}-\sum_{i=1}^{m} A_{i}^{\prime} x_{i} & \mathbf{0}  \tag{5.20}\\
\mathbf{0} & \operatorname{Diag}(d)-\sum_{i=1}^{m} \operatorname{Diag}\left(a_{i}\right) x_{i}
\end{array}\right]
$$

with $P=\left\{x \in \mathbb{R}^{m}: \operatorname{Diag}(d)-\sum_{i=1}^{m} \operatorname{Diag}\left(a_{i}\right) x_{i} \succeq \mathbf{0}\right\}$.
It is well-known that any rational polyhedron can be described by a totally dual integral system of linear inequalities, see Giles and Pulleyblank [171]. Hence, if a spectrahedron
$P$ satisfies Theorem 5.25 with rational $d, a_{i}$ for all $i \in[m]$, then it is possible to rewrite $P$ such that its LMI is totally dual integral on $\mathbb{Z}^{m}$ with respect to generating matrices $\mathcal{V}=\left\{\operatorname{Diag}\left(\mathbf{e}_{1}\right), \ldots, \operatorname{Diag}\left(\mathbf{e}_{\ell}\right)\right\} \subseteq \mathcal{S}_{+}^{\ell}$.

By relaxing the notion of total dual integrality to a strict subset $Z$ of $\mathbb{Z}^{m}$, it might be possible to identify other conditions of TDIness that go beyond polyhedrality. In return, the best one can hope for is a description of $\mathrm{cl}_{C G}(P, Z)$, see Theorem 5.20.

As shown by Bhardwaj et al. [43], any full-dimensional spectrahedron $P$ can be expressed by a linear matrix inequality in the form of (5.20), even if $P$ is nonpolyhedral. When the residual linear matrix form $C^{\prime}-\sum_{i=1}^{m} A_{i}^{\prime} x_{i}$ cannot be further diagonalized, the form on the right-hand side of (5.20) is called the normal form of the linear matrix inequality. Intuitively speaking, the bottom right block of (5.20) can be viewed as the polyhedral part of the spectrahedron. As an extension of the result by Giles and Pulleyblank [171], the following result shows that the polyhedral part of a spectrahedron can, under mild conditions, be made totally dual integral on an appropriate set $Z$.

Theorem 5.26. Let $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ be a full-dimensional spectrahedron that can be written in the normal form (5.20) for some nonsingular matrix $M \in \mathbb{R}^{n \times n}$ and $d, a_{i} \in \mathbb{Q}^{\ell}, C^{\prime}, A_{i}^{\prime} \in \mathcal{S}^{n-\ell}, i \in[m]$ with $1 \leq \ell \leq n$. Let $Z \subseteq \mathbb{Z}^{m}$ be such that

$$
\max _{x}\left\{b^{\top} x: x \in P\right\}=\max _{x}\left\{b^{\top} x: \operatorname{Diag}(d)-\sum_{i=1}^{m} \operatorname{Diag}\left(a_{i}\right) x_{i} \succeq \mathbf{0}\right\}
$$

for all $b \in Z$. Then there exists a linear matrix inequality describing $P$ that is totally dual integral on $Z$.

Proof. Let $Q=\left\{x \in \mathbb{R}^{m}: \operatorname{Diag}(d)-\sum_{i=1}^{m} \operatorname{Diag}\left(a_{i}\right) x_{i} \succeq \mathbf{0}\right\}$. Since $d$ and $a_{i}$ are rational for all $i \in[\ell]$, it follows from Giles and Pulleyblank [171] that there exists some totally dual integral representation of $Q$, i.e., $Q=\left\{x \in \mathbb{R}^{m}: \hat{A} x \leq \hat{d}\right\}$ for some $\hat{A} \in \mathbb{Z}^{\ell^{\prime} \times m}, \hat{d} \in \mathbb{Q}^{\ell^{\prime}}$ with $\hat{A} x \leq \hat{d}$ TDI. For all $i \in[m]$, let $\hat{a}_{i}$ denote the $i$ th column of $\hat{A}$. Then, $P$ can be written as

$$
P=\left\{x \in \mathbb{R}^{m}:\left[\begin{array}{cc}
C^{\prime}-\sum_{i=1}^{m} A_{i}^{\prime} x_{i} & \mathbf{0}  \tag{5.21}\\
\mathbf{0} & \operatorname{Diag}(\hat{d})-\sum_{i=1}^{m} \operatorname{Diag}\left(\hat{a}_{i}\right) x_{i}
\end{array}\right] \succeq \mathbf{0}\right\} .
$$

We will show that the LMI in (5.21) is totally dual integral on $Z$. For any $b \in Z$, we have that

$$
\max _{x}\left\{b^{\top} x: x \in P\right\}=\max _{x}\left\{b^{\top} x: x \in Q\right\}=\min _{y}\left\{\hat{d}^{\top} y: y \geq \mathbf{0}, y^{\top} \hat{A}=b^{\top}\right\} .
$$

By construction, the minimization problem above has an optimal solution $\hat{y} \in \mathbb{Z}_{+}^{\ell^{\prime}}$. Now, we define

$$
\hat{X}:=\left[\begin{array}{cc}
\mathbf{0} & \mathbf{0} \\
\mathbf{0} & \operatorname{Diag}(\hat{y})
\end{array}\right] \in \mathbb{S}_{+}^{n-\ell} \oplus \mathbb{S}_{+}^{\ell^{\prime}} .
$$

Chapter 5. The Chvátal-Gomory procedure for ISDPs with applications in combinatorial optimization

It follows from above that

$$
\left\langle\left[\begin{array}{cc}
C^{\prime} & \mathbf{0} \\
\mathbf{0} & \operatorname{Diag}(\hat{d})
\end{array}\right], \hat{X}\right\rangle=\hat{d}^{\top} \hat{y} \quad \text { and }\left\langle\left[\begin{array}{cc}
A_{i}^{\prime} & \mathbf{0} \\
\mathbf{0} & \operatorname{Diag}\left(\hat{a}_{i}\right)
\end{array}\right], \hat{X}\right\rangle=b_{i} \text { for all } i \in[m]
$$

Therefore, $\hat{X}$ is optimal to the SDP dual to $\max _{x}\left\{b^{\top} x: x \in P\right\}$. By construction, $\hat{X}$ is an integer conical combination of matrices in the set $\mathcal{V}=\left\{\mathbf{0} \oplus \operatorname{Diag}\left(\mathbf{e}_{i}\right): i \in\left[\ell^{\prime}\right]\right\}$ of integer PSD matrices. We conclude that the LMI given in (5.21) is totally dual integral on $Z$.

Our final condition for total dual integrality on a set $Z$ is not related to the polyhedrality of the spectrahedron induced by the linear matrix inequality, but related to the feasible set of its corresponding dual problem to be polyhedral. It is possible for a spectrahedron to be nonpolyhedral, while the feasible set of its dual problem is polyhedral. For instance, consider the nonpolyhedral spectrahedron $Q=\left\{x \in \mathbb{R}^{2}: x_{2} \geq \frac{1}{2} x_{1}^{2}\right\}$ considered in Example 5.4. For any $b \in \mathbb{Z}_{-}^{2}$, its dual feasible set is given by

$$
\left\{\left[\begin{array}{ll}
x_{1} & x_{2} \\
x_{2} & x_{3}
\end{array}\right] \in \mathcal{S}^{2}: x_{1} \geq-\frac{b_{1}^{2}}{2 b_{2}}, x_{2}=-\frac{1}{2} b_{1}, x_{3}=-\frac{1}{2} b_{2}\right\}
$$

which is polyhedral. Let us formalize the criterion of polyhedrality of the dual feasible set.
Definition 5.27. The set $\left\{A_{1}, \ldots, A_{m}\right\}$ is called finitely generative on $Z \subseteq \mathbb{Z}^{m}$ if there exists a finite set of integer PSD matrices $\mathcal{V}=\left\{V_{1}, \ldots, V_{k}\right\}$ such that

$$
\left\{X:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m], X \succeq \mathbf{0}\right\}
$$

is contained in cone $(\mathcal{V}):=\left\{\sum_{j \in[k]} \alpha_{j} V_{j}: \alpha_{j} \geq 0 \forall j \in[k]\right\}$ for all integer vectors $b \in Z$.
The condition of the dual feasible set to be polyhedral is also considered in recent works on SDP exactness [360].

Note that if $\left\{A_{1}, \ldots, A_{m}\right\}$ is finitely generative, then $\left\{X:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m], X \succeq \mathbf{0}\right\}$ is polyhedral for all $b \in Z$ (since $\left.\operatorname{cone}(\mathcal{V}) \subseteq \mathcal{S}_{+}^{n}\right)$. Moreover, if $\left\{A_{1}, \ldots, A_{m}\right\}$ is finitely generative on $Z$, then $\left\{t A_{1}, \ldots, t A_{m}\right\}$ is also finitely generative on $Z$ for any scalar $t>0$.

As shown below, the constraint matrices being finitely generative and integer is a sufficient condition for the existence of a totally dual integral description of the spectrahedron.

Theorem 5.28. Let $C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}$ be a linear matrix inequality satisfying Slater's condition with $\left\{A_{1}, \ldots, A_{m}\right\} \subseteq \mathbb{Z}^{n \times n}$ being finitely generative on $Z$. Then, the spectrahedron $P=\left\{x \in \mathbb{R}^{m}: C-\sum_{i=1}^{m} A_{i} x_{i} \succeq \mathbf{0}\right\}$ can be described by a linear matrix inequality that is totally dual integral on $Z$.

Proof. Let $\mathcal{V}=\left\{V_{1}, \ldots, V_{k}\right\}$ denote the finite set of integer PSD matrices corresponding to $\left\{A_{1}, \ldots, A_{m}\right\}$ in Definition 5.27. Let $b \in Z$ and let $t>0$ be a positive rational number. We consider the following semidefinite program and its dual:

$$
\begin{equation*}
\sup _{x}\left\{b^{\top} x: t C-\sum_{i=1}^{m} t A_{i} x_{i} \succeq \mathbf{0}\right\}=\inf _{X}\left\{\langle t C, X\rangle:\left\langle t A_{i}, X\right\rangle=b_{i}, i \in[m], X \succeq \mathbf{0}\right\} \tag{5.22}
\end{equation*}
$$

Based on the fact that $\left\{A_{1}, \ldots, A_{m}\right\}$ is finitely generative, we know that the feasible set of the minimization problem in (5.22) is contained in cone $(\mathcal{V})$. Since we also know the minimum
is attained due to Slater's condition, we can rewrite the dual problem as follows:

$$
\begin{aligned}
& \min _{X}\left\{\langle t C, X\rangle:\left\langle t A_{i}, X\right\rangle=b_{i}, i \in[m], X=\alpha_{1} V_{1}+\cdots+\alpha_{k} V_{k}, \alpha \geq \mathbf{0}\right\} \\
= & \min _{X}\left\{\langle t C, X\rangle: \begin{array}{r}
t \operatorname{triu}\left(A_{i}\right)^{\top} \operatorname{svec}(X)=b_{i}, \quad i \in[m], \alpha \geq \mathbf{0} \\
\operatorname{svec}(X)-\alpha_{1} \operatorname{svec}\left(V_{1}\right)-\cdots-\alpha_{k} \operatorname{svec}\left(V_{k}\right)=\mathbf{0}
\end{array}\right\} \\
= & \min _{X}\left\{\langle t C, X\rangle:\left[\begin{array}{cc}
t A^{\prime} & \mathbf{0} \\
\mathbf{I} & -V^{\prime}
\end{array}\right]\left[\begin{array}{c}
\operatorname{svec}(X) \\
\alpha
\end{array}\right]=\left[\begin{array}{l}
b \\
0
\end{array}\right], \alpha \geq \mathbf{0}\right\} .
\end{aligned}
$$

Here $A^{\prime}:=\left[\operatorname{triu}\left(A_{1}\right) \cdots \operatorname{triu}\left(A_{m}\right)\right]^{\top}, V^{\prime}:=\left[\operatorname{svec}\left(V_{1}\right) \cdots \operatorname{svec}\left(V_{k}\right)\right]$, triu $: \mathcal{S}^{n} \rightarrow \mathbb{R}^{\frac{1}{2}\left(n^{2}+n\right)}$ is the operator that maps a matrix to a vector containing its upper-triangular entries and svec: $\mathcal{S}^{n} \rightarrow \mathbb{R}^{\frac{1}{2}\left(n^{2}+n\right)}$ is the symmetric vectorization operator that maps a matrix to a vector containing its upper-triangular part with weight two on the off-diagonal elements and weight one on the diagonal elements. The linear system in the dual problem above can be written as

$$
t\left[\begin{array}{cc}
A^{\prime} & \mathbf{0} \\
\mathbf{I} & -V^{\prime}
\end{array}\right]\left[\begin{array}{c}
\operatorname{svec}(X) \\
\alpha
\end{array}\right]=\left[\begin{array}{l}
b \\
\mathbf{0}
\end{array}\right], \quad \text { or equivalently, } \quad\left[\begin{array}{cc}
A^{\prime} & \mathbf{0} \\
\mathbf{I} & -V^{\prime}
\end{array}\right]\left[\begin{array}{c}
\operatorname{svec}(X) \\
\alpha
\end{array}\right]=\frac{1}{t}\left[\begin{array}{l}
b \\
\mathbf{0}
\end{array}\right] .
$$

Each basic feasible solution to this system with $\alpha \geq \mathbf{0}$ is the unique solution to one of its nonsingular subsystems. Following the proof by Giles and Pulleyblank [171], it is possible to find a rational number $t^{*}$ such that for all $b \in Z$, there exists an optimal solution that satisfies $\operatorname{svec}(X) \in \mathbb{Z}^{\frac{1}{2}\left(n^{2}+n\right)}$ and $\alpha \in \mathbb{Z}^{k}$. When mapping svec $(X)$ back to $X \in \mathcal{S}^{n}$, it follows that the SDP dual to $\max \left\{b^{\top} x: t^{*} C-\sum_{i=1}^{m} t^{*} A_{i} x_{i} \succeq \mathbf{0}\right\}$ for all $b \in Z$ has an optimal solution $X$ satisfying

$$
X=\sum_{j \in[k]} \alpha_{j} V_{j}, \quad \alpha_{j} \geq 0, j \in[k] .
$$

with $\alpha$ integer. Hence, property $(\mathrm{P} \mathbb{Z})_{\mathcal{V}}$ holds for $X$. We conclude that $t^{*} C-\sum_{i=1}^{m} t^{*} A_{i} x_{i} \succeq \mathbf{0}$ is a linear matrix inequality describing $P$ that is totally dual integral on $Z$.

### 5.2.5 Strengthened Chvátal-Gomory cuts

Dash et al. [93] consider a strengthening of the CG cuts for rational polyhedra. We briefly present here their approach that can be applied to general convex sets.

For all $c \in \mathbb{Z}^{m}$ such that $P \subseteq\left\{x \in \mathbb{R}^{m}: c^{\top} x \leq d\right\}$, the corresponding CG cut is $c^{\top} x \leq\lfloor d\rfloor$. The validity of this cut follows from the inequality

$$
\lfloor d\rfloor \geq \max \left\{c^{\top} x: c^{\top} x \leq d, x \in \mathbb{Z}^{m}\right\}
$$

where equality holds if the entries in $c$ are relatively prime. However, the gap between $\lfloor d\rfloor$ and $\max \left\{c^{\top} x: x \in P \cap \mathbb{Z}^{m}\right\}$ can generally be very large. In order to reduce this gap, suppose that we know that $P \cap \mathbb{Z}^{m}$ is contained in some set $S \subseteq \mathbb{Z}^{m}$. Given a valid inequality $c^{\top} x \leq d$ for $P$, we define

$$
\begin{equation*}
\lfloor d\rfloor_{S, c}:=\max \left\{c^{\top} x: c^{\top} x \leq d, x \in S\right\} \tag{5.23}
\end{equation*}
$$

Chapter 5. The Chvátal-Gomory procedure for ISDPs with applications in combinatorial optimization

By construction, $c^{\top} x \leq\lfloor d\rfloor_{S, c}$ is valid for $P \cap \mathbb{Z}^{m}$. We refer to these type of cuts as $S$ -Chvátal-Gomory ( $S$-CG) cuts. These cuts are at least as strong as standard CG cuts, since taking $S=\mathbb{Z}^{m}$ provides the standard CG cut. The geometric interpretation of an $S$-CG cut is that we shift the hyperplane $\left\{x \in \mathbb{R}^{m}: c^{\top} x=d\right\}$ in the direction of $P \cap \mathbb{Z}^{m}$ until it hits a point in $S$. An example for $S$ is the set $\{0,1\}^{m}$ in the case of binary optimization problems.

### 5.3 A CG-based branch-and-cut algorithm for ISDPs

Solving ISDPs is a relatively new field of research for which only a few general-purpose solution approaches have been proposed. Gally et al. [156] present a B\&B algorithm called SCIPSDP for solving (M)ISDPs with continuous SDPs as subproblems. Alternatively, Kobayashi and Takano [234] propose a B\&C algorithm that initially relaxes the PSD constraint and solves a mixed integer linear program (MILP), where the PSD constraint is imposed dynamically via cutting planes. Numerical results in [234] show that the B\&C algorithm of [234] outperforms the $\mathrm{B} \& \mathrm{~B}$ algorithm of [156]. The difference can be explained by the high performance of the current MILP solvers compared to the much less robust conic interior-point methods that are used in [156]. It has to be noted, however, that an older version of SCIPSDP with DSDP [41] as SDP solver was used in the computational results of [234]. The authors of [270] also compare the two approaches and conclude that SCIP-SDP is much faster on average than the approach by Kobayashi and Takano. However, they use Mosek [284] as an SDP solver and an improved implementation of SCIP-SDP. Another project that encounters MISDPs is YALMIP [257], although its performance is inferior compared to the other two methods [156, 234].

In this section we present a generic $\mathrm{B} \& \mathrm{C}$ algorithm for solving ISDPs that exploits CG cuts of the underlying spectrahedron. This algorithm can be seen as an extension of the works of [68, 234]. In Section 5.3.1 we provide a general B\&C framework for ISDPs which uses a cut generation routine based on $S$-CG cuts. Section 5.3 .2 presents a separation routine for the special class of binary SDPs.

### 5.3.1 Generic branch-and-cut framework

We start by presenting the B\&C framework proposed by Kobayashi and Takano [234] for ISDPs in standard dual form, see (5.5). However, the approach can be extended to problems in primal form in a straightforward way. We define

$$
\begin{equation*}
\mathcal{F}:=\left\{x \in \mathbb{R}^{m}: \operatorname{diag}\left(C-\sum_{i=1}^{m} A_{i} x_{i}\right) \geq 0\right\} \tag{5.24}
\end{equation*}
$$

which can be seen as the polyhedral part of the spectrahedron $P$, see (5.6). We assume that the problem of maximizing $b^{\top} x$ over $\mathcal{F}$ is bounded, which is a nonrestrictive assumption whenever the original ISDP is bounded.

The B\&C algorithm of [234] is based on a dynamic constraint generation known as a lazy constraint callback. The algorithm starts with optimizing over the set $\mathcal{F} \cap \mathbb{Z}^{m}$, i.e.,

$$
\begin{equation*}
\max \left\{b^{\top} x: x \in \mathcal{F} \cap \mathbb{Z}^{m}\right\} \tag{5.25}
\end{equation*}
$$

which can be solved using a $B \& B$ algorithm. Whenever an integer point $\hat{x}$ is found in the branching tree, it is verified whether $C-\sum_{i=1}^{m} A_{i} \hat{x}_{i} \succeq \mathbf{0}$ is satisfied. If so, the solution
is feasible for $\left(D_{I S D P}\right)$ and provides a possibly better lower bound to prune other nodes in the tree. If not, then $\left\langle C-\sum_{i=1}^{m} A_{i} \hat{x}_{i}, d d^{\top}\right\rangle<0$ where $d$ is a normalized eigenvector corresponding to the smallest eigenvalue of $C-\sum_{i=1}^{m} A_{i} \hat{x}_{i}$. This leads to the following valid constraint for ( $D_{I S D P}$ ):

$$
\begin{equation*}
\left\langle C-\sum_{i=1}^{m} A_{i} x_{i}, d d^{\top}\right\rangle \geq 0, \quad \text { or equivalently, } \quad \sum_{i=1}^{m}\left\langle A_{i}, d d^{\top}\right\rangle x_{i} \leq\left\langle C, d d^{\top}\right\rangle, \tag{5.26}
\end{equation*}
$$

which separates $\hat{x}$ from $P$. Now, the algorithm adds to $\mathcal{F}$ a cut of type (5.26) to cut off the current point and continues the branching scheme using this additional constraint. This process is iterated until the optimality of a solution for $\left(D_{I S D P}\right)$ is guaranteed by the $\mathrm{B} \& \mathrm{~B}$ procedure.

It follows from the Rayleigh principle that $\left\langle C-\sum_{i=1}^{m} A_{i} \hat{x}_{i}, U\right\rangle$ is minimized by taking $U=d d^{\top}$ with $d$ as defined above. In that sense, the cut (5.26) is the strongest cut with respect to violation in the PSD constraint. However, this type of separator ignores the fact that an optimal solution is also integer. We now propose an alternative stronger separator based on the CG procedure that exploits both the PSD and the integrality constraint.

Let $S \subseteq \mathbb{Z}^{m}$ be a set containing the feasible set of $\left(D_{I S D P}\right)$, with $S=\mathbb{Z}^{m}$ in case of no prior knowledge about the problem. If $\hat{x} \notin P$, and consequently $\hat{x} \notin \operatorname{cl}_{C G}(P)$, it follows from (5.9) that there exists a dual multiplier $U \in \mathcal{S}_{+}^{n}$ with $\left\langle A_{i}, U\right\rangle \in \mathbb{Z}$ for all $i \in[m]$, such that $\left.\sum_{i=1}^{m}\left\langle A_{i}, U\right\rangle \hat{x}_{i}\right\rangle\lfloor\langle C, U\rangle\rfloor$. Taking such $U$ and defining $v(U):=$ $\left(\left\langle A_{1}, U\right\rangle, \ldots,\left\langle A_{m}, U\right\rangle\right)^{\top}$, we obtain the following $S$-CG cut:

$$
\begin{equation*}
\sum_{i=1}^{m}\left\langle A_{i}, U\right\rangle x_{i} \leq\lfloor\langle C, U\rangle\rfloor_{S, v(U)}, \tag{5.27}
\end{equation*}
$$

see (5.23). The cut (5.27) exploits both the PSD and the integrality constraints in ( $D_{I S D P}$ ) by separating $\hat{x}$ from $\mathrm{cl}_{C G}(P)$ instead of only from $P$. As $\mathrm{cl}_{C G}(P) \subseteq P$ for bounded spectrahedra, this type of cut is possibly stronger than the eigenvalue cut (5.26) for all $S$ containing $P \cap \mathbb{Z}^{m}$. Figure 5.4 depicts a simplified example indicating the geometric difference between the cuts (5.26) and (5.27).

It is not clear in general how to find an appropriate cut (5.27) separating $\hat{\mathbf{x}}$ from $\mathrm{cl}_{C G}(P)$. Indeed, this is closely related to the CG separation problem, which was proven to be $\mathcal{N} \mathcal{P}$-hard even for polytopes contained in the unit hypercube, see Cornuéjols et al. [82]. Fischetti and Lodi [139] show how to solve the separation problem for polyhedra using a mixed integer programming problem. Extending their procedure to the class of spectrahedra, implies solving a MISDP. Instead, we can adopt problem-specific separation routines that are efficient and provide strong cuts. For instance, in the next subsection we present a separation routine for binary SDPs in primal form. Moreover, we later provide various separation routines for cuts of the form (5.27) for the quadratic traveling salesman problem.

Alongside extending the approach of Kobayashi and Takano [234], our framework also continues on the work of Çezik and Iyengar [68]. In [68] CG cuts for binary conic programs are introduced. It is noted that there is no method known for separating CG cuts from fractional points, and consequently the CG cuts are not included in the numerical experiments of [68]. Since our approach separates on integer points only, we partly resolve this issue for certain classes of problems by exploiting the underlying structure of the programs. As a result, we present the first practical algorithm that utilizes CG cuts in conic problems.

We end this section by providing a pseudocode of the B\&C framework, see Algorithm 5.1.


Figure 5.4: Simplified example of strengthened separation routine on spectrahedron $P$ from Example 5.4. The dotted line shows an eigenvalue cut (5.26) separating $\hat{x}$ from $P$, the solid line shows a CG cut (5.27) separating $\hat{x}$ from $\operatorname{cl}_{C G}(P)$, where $S=\mathbb{Z}^{m}$.

Suppose SeparationRoutine is a routine for constructing CG cuts of the form (5.27), where we assume this routine can generate multiple dual matrices at a time.

```
Algorithm 5.1 CG-based \(\mathrm{B} \& \mathrm{C}\) algorithm for solving ( \(D_{I S D P}\) )
Input: \(C, A_{i}, i \in[m], S, \epsilon>0\)
    Initialize \(\mathcal{F}\) as defined in (5.24).
    B\&B procedure: Start or continue the branch-and-bound algorithm for solving the MILP
    \(\max \left\{b^{\top} x: x \in \mathcal{F} \cap \mathbb{Z}^{m}\right\}\) incorporating the callback function below at each node in the branching
    tree.
    Callback procedure:
    if an integer point \(\hat{x} \in \mathcal{F}\) is found then
        if \(\lambda_{\text {min }}\left(C-\sum_{i=1}^{m} A_{i} \hat{x}_{i}\right)<-\epsilon\) then
            Call SeparationRoutine \(\left(C, A_{1}, \ldots, A_{m}, S, \hat{x}\right)\) which provides matrices \(U_{j}, j \in[K]\).
            Add the cuts \(\sum_{i=1}^{m}\left\langle A_{i}, U_{j}\right\rangle x_{i} \leq\left\lfloor\left\langle C, U_{j}\right\rangle\right\rfloor_{S, v\left(U_{j}\right)}\) for \(j \in[K]\) to \(\mathcal{F}\).
        else
            Use \(\hat{x}\) to cut off other nodes in the branching tree.
        end if
    end if
    Return to Step 2
Output: \(\hat{x}, O P T:=b^{\top} x\)
```


### 5.3.2 A separation routine for binary SDPs

We now focus on binary semidefinite programming problems in primal form, i.e.,

$$
\begin{cases}\inf & \langle C, X\rangle \\ \text { s.t. } & \left\langle A_{i}, X\right\rangle=b_{i} \quad \text { for all } i \in[m] \\ & X \succeq \mathbf{0}, X \in\{0,1\}^{n \times n}\end{cases}
$$

$$
\left(P_{B S D P}\right)
$$

In this section we present a separation routine for generating CG cuts for problems of the form $\left(P_{B S D P}\right)$ and provide two illustrative examples.

Suppose we solve $\left(P_{B S D P}\right)$ using the $\mathrm{B} \& \mathrm{C}$ algorithm presented in Section 5.3.1. In a certain node in the branching tree we have obtained a symmetric matrix $\hat{X} \in\{0,1\}^{n \times n}$ that satisfies $\left\langle A_{i}, \hat{X}\right\rangle=b_{i}$ for all $i \in[m]$. The separation oracle that we present below distinguishes two types of certificates for $\hat{X}$ not being positive semidefinite.

The first one is obtained by a so-called dominated diagonal, i.e., $\hat{X}_{i i}=0$, while $\hat{X}_{i j}=1$ for some $j$, which clearly implies that $\hat{X} \nsucceq \mathbf{0}$. For the second certificate, we exploit a wellknown result on binary positive semidefinite matrices: a matrix $X \in\{0,1\}^{n \times n}$ is positive semidefinite if and only if $X=\sum_{i=1}^{k} x_{i} x_{i}^{\top}$ for some $x_{i} \in\{0,1\}^{n}, i \in[k]$, see also Theorem 6.1 in this thesis. Each such vector $x_{i}$ may be thought of as the characteristic vector of a clique in the complete graph $K_{n}$. In that sense, $X$ represents the characteristic matrix of a set of nonoverlapping cliques in $K_{n}$. Now, our second certificate considers the existence of a so-called conflicting vertex, i.e., a vertex that is contained in two separate cliques implied by $\hat{X}$, which yields $\hat{X} \nsucceq \mathbf{0}$. The mentioned certificates correspond to the existence of the following induced submatrices in $\hat{X}$ (up to a permutation of the rows and columns):

$$
\left.\begin{array}{cc}
i & j \\
i\left[\begin{array}{ll}
0 & 1 \\
1 & \star
\end{array}\right]
\end{array} \quad \begin{array}{rll}
i & j & k \\
i
\end{array} \quad \begin{array}{ccc}
i \\
j & 1 & 1 \\
k & 1 & 0 \\
1 & 0 & 1
\end{array}\right]
$$

where $\star$ indicates a position that can be either 0 or 1 . The following result shows that these certificates are necessary and sufficient to characterize positive semidefiniteness.

Proposition 5.29. Let $\hat{X}=\left(\hat{x}_{i j}\right)$ be binary and symmetric. Then, $\hat{X}$ is positive semidefinite if and only if $\hat{X}$ contains no dominated diagonal or conflicting vertex.

Proof. Necessity follows from the discussion above. Conversely, let $D(i):=\left\{j \in[n]: \hat{x}_{i j}=1\right\}$ for all $i \in[n]$ with $\hat{x}_{i i}=1$. If $\hat{x}_{i j}=1$ and $\hat{x}_{i k}=1$, it must follow that $\hat{x}_{j k}=1$, otherwise $i$ would be conflicting. Hence, the sets $D(i)$ for all $i$ with $\hat{X}_{i i}=1$ are cliques. Since $i \in D(j)$ if and only if $j \in D(i)$, it follows that the collection $\mathcal{D}$ of all distinct sets $D(i)$ is a set of nonoverlapping cliques. Then, $\hat{X}=\sum_{D \in \mathcal{D}} \mathbb{1}_{D} \mathbb{1}_{D}{ }^{\top}$, hence $\hat{X} \succeq \mathbf{0}$.

In case of a dominated diagonal, i.e., indices $i, j \in[n], i \neq j$ with $\hat{x}_{i i}=0$ and $\hat{x}_{i j}=1$, the dual matrix $U=\left(\mathbf{e}_{i}-\mathbf{e}_{j}\right)\left(\mathbf{e}_{i}-\mathbf{e}_{j}\right)^{\top}$ separates $\hat{X}$ from $\mathcal{S}_{+}^{n}$. In case of a conflicting vertex, say $i$, with $\hat{x}_{i j}=1, \hat{x}_{i k}=1$, but $\hat{x}_{j k}=0$, the dual matrix $U=\left(\mathbf{e}_{j}+\mathbf{e}_{k}-\mathbf{e}_{i}\right)\left(\mathbf{e}_{j}+\mathbf{e}_{k}-\mathbf{e}_{i}\right)^{\top}$ provides a separating hyperplane. Since dominated diagonals and conflicting vertices can be found efficiently by enumeration, this approach defines an efficient separation routine for binary SDPs in primal form.

The cuts $\langle U, X\rangle \geq 0$ can be further strengthened by exploiting the affine constraints in a CG rounding step. We show how this can be done for two classes of binary semidefinite programming problems that often appears in relaxations of combinatorial problems.

Example 5.30 (Binary SDPs over the elliptope). Suppose we have the following binary SDP:

$$
\left\{\begin{align*}
\inf & \langle C, X\rangle  \tag{1}\\
\text { s.t. } & \left\langle A_{i}, X\right\rangle=b_{i} \quad \text { for all } i \in[m] \\
& \operatorname{diag}(X)=\mathbf{1} \\
& X \succeq \mathbf{0}, X \in\{0,1\}^{n \times n}
\end{align*}\right.
$$

The problem $\left(P_{1}\right)$ can be solved using the $\mathrm{B} \& \mathrm{C}$ algorithm of Section 5.3 .1 by initially setting $\mathcal{F}:=\left\{X \in \mathcal{S}^{n}:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m], \operatorname{diag}(X)=\mathbf{1}, \mathbf{0} \leq X \leq \mathbf{J}\right\}$. At a certain node in the branching tree, a matrix $\hat{X} \in \mathcal{F} \cap\{0,1\}^{n \times n}$ is considered. If $\hat{X}$ is not positive semidefinite, then an integer dual matrix $U$ needs to be provided. Since $\operatorname{diag}(X)=\mathbf{1}$ is included in $\mathcal{F}$, this dual matrix results from a conflicting vertex certificate. Suppose this dual matrix is $U=\left(\mathbf{e}_{j}+\mathbf{e}_{k}-\mathbf{e}_{i}\right)\left(\mathbf{e}_{j}+\mathbf{e}_{k}-\mathbf{e}_{i}\right)^{\top}$ for some distinct $i, j, k$. We can now further strengthen the cut $\langle U, X\rangle \geq 0$ using the constraint $\operatorname{diag}(X)=\mathbf{1}$ and the fact that $U$ is integral. Namely, taking the linear combination of $\langle U, X\rangle \geq 0, x_{i i}=1, x_{j j}=1$ and $x_{k k}=1$, each with weight $\frac{1}{2}$, yields:

$$
\begin{aligned}
\frac{1}{2}\left(\langle U, X\rangle+x_{i i}+x_{j j}+x_{k k}\right) & \geq \frac{1}{2}(0+1+1+1) \\
\Longleftrightarrow \quad\left\langle\frac{1}{2} U+\frac{1}{2}\left(\mathbf{E}_{i i}+\mathbf{E}_{j j}+\mathbf{E}_{k k}\right), \mathbf{X}\right\rangle & \geq 1 \frac{1}{2}
\end{aligned}
$$

Since $X \in \mathcal{S}_{n}$ and all coefficients on the left-hand side are integral, we can strengthen the cut using a CG rounding step:

$$
\left\langle\frac{1}{2} U+\frac{1}{2}\left(\mathbf{E}_{i i}+\mathbf{E}_{j j}+\mathbf{E}_{k k}\right), X\right\rangle \geq\left\lceil 1 \frac{1}{2}\right\rceil=2 .
$$

This cut is equivalent to $x_{j k}+1 \geq x_{i k}+x_{i j}$, which is one of the triangle inequalities resulting from the boolean quadric polytope [297]. These cuts are facet defining for the binary PSD polytope [249].

Example 5.31 (Binary SDPs over the simplex). Many combinatorial optimization problems have formulations including a constraint on the trace of the matrix variable, i.e.,

$$
\left\{\begin{align*}
\text { inf } & \langle C, X\rangle  \tag{2}\\
\text { s.t. } & \left\langle A_{i}, X\right\rangle=b_{i} \quad \text { for all } i \in[m] \\
& \operatorname{tr}(X)=K \\
& X \succeq \mathbf{0}, X \in\{0,1\}^{n \times n},
\end{align*}\right.
$$

for some $K \in \mathbb{N}$. Using $\mathcal{F}:=\left\{X \in \mathcal{S}^{n}:\left\langle A_{i}, X\right\rangle=b_{i}, i \in[m], \operatorname{tr}(X)=K, \mathbf{0} \leq X \leq \mathbf{J}\right\}$, one can solve ( $P_{2}$ ) by Algorithm 5.1. Assume that the separation routine provides a dual matrix $U=\left(\mathbf{e}_{j}+\mathbf{e}_{k}-\mathbf{e}_{i}\right)\left(\mathbf{e}_{j}+\mathbf{e}_{k}-\mathbf{e}_{i}\right)^{\top}$ for some distinct $i, j, k$. Taking the linear combination of $\langle U, X\rangle \geq 0, \operatorname{tr}(X)=K$ and $x_{l l} \geq 0$ for all $l \notin\{i, j, k\}$, each with weight $\frac{1}{2}$, yields:

$$
\left\langle\frac{1}{2} U+\frac{1}{2} \mathbf{I}+\frac{1}{2} \sum_{l \notin\{i, j, k\}} \mathbf{E}_{\mathbf{l l}}, X\right\rangle \geq \frac{1}{2} K .
$$

For $K$ odd, we strengthen the cut by replacing the right-hand side by $\left\lceil\frac{1}{2} K\right\rceil$. This procedure can be repeated for dual matrices resulting from a dominated diagonal certificate.

### 5.4 The Chvátal-Gomory procedure for ISDP formulations of the QTSP

In this section we provide an in-depth study on solving the quadratic traveling salesman problem using our B\&C approach. We formally define the QTSP in Section 5.4.1. In

Section 5.4.2 we derive two ISDP formulations of the QTSP. Our first ISDP model exploits the algebraic connectivity of a directed tour. Our second formulation exploits the algebraic connectivity of a directed tour and the distance two matrix that originates from the product of a tour matrix with itself. Finally, in Section 5.4 .3 we derive CG cuts for the two ISDPs and show that we can obtain various classes of well-known cuts in this way.

### 5.4.1 The quadratic traveling salesman problem

Let $G=(N, A)$ be a directed simple graph on $n:=|N|$ nodes and $m:=|A|$ arcs. A directed cycle $C$ in $G$ that visits all the nodes exactly once is called a directed Hamiltonian cycle or a directed tour in $G$. For the sake of simplicity, we often omit the adjective 'directed' in the sequel.

A tour in $G$ can be represented by a binary matrix $X=\left(x_{i j}\right) \in\{0,1\}^{n \times n}$ such that $x_{i j}=1$ if and only if arc $(i, j)$ is used in the tour. We refer to such a matrix as a tour matrix. The set of all tour matrices in $G$ is defined as follows:

$$
\begin{equation*}
\mathcal{T}_{n}(G):=\left\{X^{C} \in\{0,1\}^{n \times n}: x_{i j}^{C}=1 \text { if and only if }(i, j) \in C \text { for tour } C\right\} . \tag{5.28}
\end{equation*}
$$

It follows from (5.28) that for all $X \in \mathcal{T}_{n}(G)$ we have $x_{i j}=0$ if $(i, j) \notin A$. In partic$\operatorname{ular}, \operatorname{diag}(X)=\mathbf{0}_{n}$. Given a distance matrix $D=\left(d_{i j}\right) \in \mathbb{R}^{n \times n}$, the (linear) traveling salesman problem (TSP) is the problem of finding a Hamiltonian cycle $C$ of $G$ that minimizes $\sum_{(i, j) \in C} d_{i j}$. As $G$ is directed and $D$ is not necessarily symmetric, this version of the problem is sometimes referred to as the asymmetric traveling salesman problem. Using the set defined in (5.28), we can state the TSP as follows:

$$
\begin{equation*}
T S P(D, G):=\min \left\{\sum_{i=1}^{n} \sum_{j=1}^{n} d_{i j} x_{i j}: X \in \mathcal{T}_{n}(G)\right\} \tag{5.29}
\end{equation*}
$$

We now define the quadratic version of the TSP, where the total cost is given by the sum of interaction costs between arcs used in the tour. In accordance with most of the literature, we assume that a quadratic cost is incurred only if two arcs are placed in succession on the tour, see e.g., $[133,134,135,224,326]$. To model this problem, we define the set of the so-called 2 -arcs of $G$, i.e.,

$$
\begin{equation*}
\mathcal{A}:=\{(i, j, k):(i, j),(j, k) \in A,|\{i, j, k\}|=3\}, \tag{5.30}
\end{equation*}
$$

which consists of all node triples of $G$ that can be placed in succession on a cycle. Now, let $Q=\left(q_{i j k}\right) \in \mathbb{R}^{n \times n \times n}$ be a cost matrix such that $q_{i j k}=0$ if $(i, j, k) \notin \mathcal{A}$. Then the quadratic traveling salesman problem (QTSP) is formulated as:

$$
\begin{equation*}
\operatorname{QTSP}(Q, G):=\min \left\{\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} q_{i j k} x_{i j} x_{j k}: X \in \mathcal{T}_{n}(G)\right\} . \tag{5.31}
\end{equation*}
$$

Since the in- and outdegree of each node on a tour is exactly one, the equalities $X \mathbf{1}=\mathbf{1}$ and $X^{\top} \mathbf{1}=\mathbf{1}$ hold for all $X \in \mathcal{T}_{n}(G)$. The set of square binary matrices that satisfy this property is known as the set of permutation matrices $\Pi_{n}$, i.e.,

$$
\Pi_{n}:=\left\{X \in\{0,1\}^{n \times n}: X \mathbf{1}=\mathbf{1}, X^{\top} \mathbf{1}=\mathbf{1}\right\} .
$$

The permutation matrices that additionally satisfy $\operatorname{diag}(X)=\mathbf{0}_{n}$ induce a disjoint cycle cover in $K_{n}$.

Similar to the definition of $\mathcal{T}_{n}(G)$, we can also restrict $\Pi_{n}$ to the entries induced by $G$. That is, $\Pi_{n}(G)$ has a zero on position $(i, j)$ whenever $(i, j) \notin A$.

### 5.4.2 ISDP based on algebraic connectivity in directed graphs

Cvetković et al. [86] derive an ISDP formulation of the symmetric linear TSP based on algebraic connectivity. We now exploit the equivalent of this notion for directed graphs to derive two ISDP formulations of the QTSP. Different from our approach, there was no attempt in [86] to solve the ISDP itself, only its SDP relaxation.

Let $D_{G}$ be an $n \times n$ diagonal matrix that contains the outdegrees of the nodes of $G$ on the diagonal. Moreover, let $A_{G}$ denote the adjacency matrix of $G$. That is, $\left(A_{G}\right)_{i j}=1$ if there exists an arc from $i$ to $j$ in $G$, and $\left(A_{G}\right)_{i j}=0$ otherwise. We define the directed out-degree Laplacian matrix of $G$ as $L_{G}:=D_{G}-A_{G}$. The matrix $L_{G}$ can be asymmetric and has a zero eigenvalue with corresponding eigenvector $\mathbf{1}_{n}$. Observe that there exist also other ways for defining the directed graph Laplacian of $G$, see e.g., [67]. Wu [375] generalized Fiedler's notion of algebraic connectivity of an undirected graph [132] to directed graphs, by exploiting the out-degree Laplacian matrix.
Definition 5.32. The algebraic connectivity of a directed graph $G$ is given by

$$
a(G):=\min _{x \in S} x^{\top} L_{G} x=\min _{\substack{x \in \mathbb{R}^{n} \\ x \neq 0, x \perp 1_{n}}} \frac{x^{\top} L_{G} x}{x^{\top} x}=\lambda_{\min }\left(\frac{1}{2} W^{\top}\left(L_{G}+L_{G}^{\top}\right) W\right),
$$

where $S:=\left\{x \in \mathbb{R}^{n}: x \perp \mathbf{1}_{n},\|x\|_{2}=1\right\}$ and $W \in \mathbb{R}^{n \times(n-1)}$ is a matrix whose columns form an orthonormal basis for $\mathbf{1}_{n}^{\perp}$.

The last equality in Definition 5.32 follows from the Courant-Fischer theorem. Observe that $a(G)$ is not necessarily equal to the second smallest eigenvalue of the directed Laplacian matrix, which is the definition of its undirected counterpart. The algebraic connectivity $a(G)$ as defined in Definition 5.32 is a real number that can be negative.

A directed graph is called balanced if for each node its indegree is equal to its outdegree. Let $B \in\{-1,0,1\}^{n \times m}$ be the signed incidence matrix of $G$, i.e., $B_{i, e}=-1$ if arc leaves node $i, B_{i, e}=1$ if $e$ enters node $i$ and $B_{i, e}=0$ otherwise. One can verify that $G$ is balanced if and only if $L_{G}+L_{G}{ }^{\top}=B B^{\top}$. This implies that for balanced graphs the matrix $\frac{1}{2}\left(L_{G}+L_{G}{ }^{\top}\right)$ is positive semidefinite. Wu [375] observes that if $G$ is balanced, then

$$
a(G)=\lambda_{2}\left(\frac{1}{2}\left(L_{G}+L_{G}^{\top}\right)\right) \geq 0
$$

A directed graph is called strongly connected if for every pair of distinct nodes $u, v \in N$ there exists a directed path from $u$ to $v$ in $G$. The balanced graphs that are strongly connected are characterized by their algebraic connectivity, see Proposition 5.33 below. Connectedness of directed graphs is also studied in [67, 357].

Proposition 5.33 (Wu [375]). Let a directed graph $G$ be balanced. Then, $a(G)>0$ if and only if $G$ is strongly connected.

This characterization can be exploited to derive a certificate for a tour matrix via a linear matrix inequality. In order to do so, we consider the spectrum of a Hamiltonian
cycle on $n$ nodes. Let $C$ be a Hamiltonian cycle in $G$ corresponding to the tour matrix $X \in \mathcal{T}_{n}(G)$, see (5.28). We then have $\frac{1}{2}\left(L_{C}+L_{C}{ }^{\top}\right)=\mathbf{I}_{n}-\frac{1}{2}\left(X+X^{\top}\right)$. The matrix $X+X^{\top}$ with $X \in \mathcal{T}_{n}(G)$ has the same spectrum as the adjacency matrix of the standard undirected $n$-cycle. As a result, the spectrum of $\frac{1}{2}\left(X+X^{\top}\right)$ is given by $\cos \left(\frac{2 \pi j}{n}\right)$ for $j \in[n]$ see e.g., [86]. From this, it follows that the spectrum of $\frac{1}{2}\left(L_{C}+L_{C}{ }^{\top}\right)$ is given by

$$
1-\cos \left(\frac{2 \pi j}{n}\right) \quad \text { for } j \in[n]
$$

and the algebraic connectivity of a directed Hamiltonian cycle $C$ is $a(C)=1-\cos (2 \pi / n)$. We define:

$$
\begin{equation*}
k_{n}:=\cos \left(\frac{2 \pi}{n}\right) \quad \text { and } \quad h_{n}:=1-k_{n} \tag{5.32}
\end{equation*}
$$

Next, we extend a result by Cvetković et al. [86] from undirected to directed Hamiltonian cycles.

Theorem 5.34. Let $H$ be a spanning subgraph of a directed graph $G$ where the in- and outdegree equals one for all nodes in $H$. Let $X$ be its adjacency matrix and let $\alpha, \beta \in \mathbb{R}$ be such that $\alpha \geq h_{n} / n$ and $k_{n} \leq \beta<1$, with $k_{n}, h_{n}$ as defined in (5.32). Then, $H$ is a directed Hamiltonian cycle if and only if

$$
Z:=\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right) \succeq \mathbf{0}
$$

Proof. Let $L_{H}$ be the Laplacian matrix of $H$ and let $W$ be as given in Definition 5.32. Then, $a(H)=\lambda_{\min }\left(\frac{1}{2} W^{\top}\left(L_{H}+L_{H}^{\top}\right) W\right)$.

Let $Z \succeq \mathbf{0}$. This implies that $W^{\top} Z W \succeq \mathbf{0}$, i.e.,

$$
\begin{aligned}
W^{\top} Z W & =W^{\top}\left(\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right)\right) W \\
& =\beta W^{\top} W+\alpha W^{\top} \mathbf{J}_{n} W-\frac{1}{2} W^{\top}\left(X+X^{\top}\right) W \\
& =\beta \mathbf{I}_{n-1}-\frac{1}{2} W^{\top}\left(X+X^{\top}\right) W \\
& =(\beta-1) \mathbf{I}_{n-1}+\frac{1}{2} W^{\top}\left(L_{H}+L_{H}^{\top}\right) W \succeq \mathbf{0}
\end{aligned}
$$

where we used the fact that $\mathbf{J}_{n} W=\mathbf{0}$ and $\frac{1}{2}\left(L_{H}+L_{H}{ }^{\top}\right)=\mathbf{I}_{n}-\frac{1}{2}\left(X+X^{\top}\right)$. The linear matrix inequality above can be rewritten as

$$
\frac{1}{2} W^{\top}\left(L_{H}+L_{H}^{\top}\right) W \succeq(1-\beta) \mathbf{I}_{n-1}
$$

implying

$$
a(H)=\lambda_{\min }\left(\frac{1}{2} W^{\top}\left(L_{H}+L_{H}^{\top}\right) W\right) \geq 1-\beta
$$

Since $\beta<1$, we have $\alpha(H)>0$. Because $H$ is balanced, it follows from Proposition 5.33 that $H$ is strongly connected and, thus, $H$ is a directed Hamiltonian cycle.

Conversely, let $H$ be a directed Hamiltonian cycle. Then, $a(H)=\lambda_{\min }\left(\frac{1}{2} W^{\top}\left(L_{H}+\right.\right.$

Chapter 5. The Chvátal-Gomory procedure for ISDPs with applications in combinatorial optimization
$\left.\left.L_{H}{ }^{\top}\right) W\right)=1-k_{n}$. Since $\beta \geq k_{n}$, we have

$$
\frac{1}{2} W^{\top}\left(L_{H}+L_{H}^{\top}\right) W-(1-\beta) \mathbf{I}_{n-1} \succeq \mathbf{0} \quad \Longleftrightarrow \quad W^{\top} Z W \succeq \mathbf{0}
$$

following the same derivation as above. Now, let $x \in \mathbb{R}^{n}$. Since the columns of $W$ form a basis for $\mathbf{1}_{n}^{\perp}, x$ can be written as $x=W y+\delta \mathbf{1}_{n}$ for some $y \in \mathbb{R}^{n-1}$ and $\delta \in \mathbb{R}$. This yields:

$$
\begin{aligned}
x^{\top} Z x & =y^{\top} W^{\top} Z W y+2 \delta y^{\top} W^{\top} Z \mathbf{1}_{n}+\delta^{2} \mathbf{1}_{n}^{\top} Z \mathbf{1}_{n} \\
& =\underbrace{y^{\top} W^{\top} Z W y}_{\geq 0}+\underbrace{2 \delta y^{\top} W^{\top}\left((\beta-1) \mathbf{1}_{n}+\alpha n \mathbf{1}_{n}\right)}_{=0}+\underbrace{\delta^{2} n((\beta-1)+\alpha n)}_{\geq 0},
\end{aligned}
$$

where we used the facts that $W^{\top} Z W \succeq \mathbf{0}, W^{\top} \mathbf{1}_{n}=\mathbf{0}$ and $\beta-1+\alpha n \geq k_{n}-1+n \frac{1-k_{n}}{n}=0$. Thus, $Z \succeq \mathbf{0}$.

In order to present our first ISDP formulation of the QTSP, we derive an explicit expression for the set $\mathcal{T}_{n}(G)$ and linearize the objective function. The former can be done using Theorem 5.34. The set $\mathcal{T}_{n}(G)$ can be fully characterized by the permutation matrices that satisfy a linear matrix inequality. That is,

$$
\begin{equation*}
\mathcal{T}_{n}(G)=\Pi_{n}(G) \cap\left\{X \in \mathcal{S}^{n}: \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right) \succeq \mathbf{0}\right\} \tag{5.33}
\end{equation*}
$$

for all $\alpha \geq h_{n} / n$ and $k_{n} \leq \beta<1$. Recall that $\Pi_{n}(G)$ is the set of permutation matrices implied by $G$, see Section 5.4.1.

To linearize the objective function, we follow the same construction as proposed by Fischer et al. [135]. For all two-arcs $(i, j, k) \in \mathcal{A}$, see (5.30), we define a variable $y_{i j k}:=x_{i j} x_{j k}$. This equality can be guaranteed by the introduction of the following set of linear coupling constraints:

$$
x_{i j}=\sum_{\substack{k \in N: \\(k, i, j) \in \mathcal{A}}} y_{k i j}=\sum_{\substack{k \in N: \\(i, j, k) \in \mathcal{A}}} y_{i j k} \text { for all }(i, j) \in A \quad \text { and } \quad y_{i j k} \geq 0 \text { for all }(i, j, k) \in \mathcal{A} \text {. }
$$

We define the following set:

$$
\begin{equation*}
\mathcal{F}_{1}:=\left\{(y, X) \in\{0,1\}^{\mathcal{A}} \times \Pi_{n}(G): x_{i j}=\sum_{\substack{k \in N: \\(k, i, j) \in \mathcal{A}}} y_{k i j}=\sum_{\substack{k \in N: \\(i, j, k) \in \mathcal{A}}} y_{i j k} \forall(i, j) \in A\right\} . \tag{5.34}
\end{equation*}
$$

Now, our first ISDP formulation of the QTSP is as follows:

$$
\begin{cases}\min & \sum_{(i, j, k) \in \mathcal{A}} q_{i j k} y_{i j k}  \tag{1}\\ \text { s.t. } & \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right) \succeq \mathbf{0} \\ & (y, X) \in \mathcal{F}_{1},\end{cases}
$$

where $\alpha \geq h_{n} / n$ and $k_{n} \leq \beta<1$. One can verify that setting $\alpha=h_{n} / n$ and $\beta=k_{n}$ leads to the strongest linear matrix inequality among all possible values for $\alpha$ and $\beta$. Thus, we use these values in the computational results of Section 5.5.

Remark 5.35. In fact, we do not need to enforce integrality on $y$ explicitly. Namely, if $X \in \mathcal{T}_{n}(G)$, it follows from the integrality of $X$ and the coupling constraints that $y_{i j k}=1$ if $(i, j, k) \in \mathcal{A}$ is used in the tour and 0 otherwise. Hence, when optimizing over $\mathcal{F}_{1}$ using a $\mathrm{B} \& \mathrm{~B}$ or $\mathrm{B} \& \mathrm{C}$ algorithm, we relax the integrality constraint on $y$ and branch on $X$ only.

In what follows, we further exploit properties of tour matrices to derive our second ISDP formulation of the QTSP. Let $X \in \mathcal{T}_{n}(G)$ be a tour matrix and define $X^{(2)}=\left(x_{i j}^{(2)}\right):=X \cdot X$. For $i, k \in N$ we have $x_{i k}^{(2)}=\sum_{j=1}^{n} x_{i j} x_{j k}=\sum_{j \in N:(i, j, k) \in \mathcal{A}} y_{i j k}$, where the last equality follows from the definition of $y$. Thus, $X^{(2)}$ is a binary matrix and $x_{i k}^{(2)}=1$ if and only if the length of the shortest directed path from $i$ to $k$ in the subgraph induced by $X$ is equal to two.

We can again characterize a tour matrix as in Theorem 5.34 by combining the variables $X$ and $X^{(2)}$. Observe that the directed graph induced by $X^{(2)}$ is balanced with inand outdegree one, and circulant (but not strongly connected for even $n$ ). Moreover, the circulant graph $C^{(2)}$ corresponding to $X+X^{(2)}$ is strongly connected and balanced with inand outdegree two.

The spectrum of $\frac{1}{2}\left(\left(X+X^{(2)}\right)+\left(X+X^{(2)}\right)^{\top}\right)$ for any $X \in \mathcal{T}_{n}(G)$ and $X^{(2)}=X \cdot X$ is given by

$$
\begin{equation*}
\cos \left(\frac{2 \pi j}{n}\right)+\cos \left(\frac{4 \pi j}{n}\right) \quad \text { for } j \in[n] \tag{5.35}
\end{equation*}
$$

which results in the algebraic connectivity of $C^{(2)}$ being $a\left(C^{(2)}\right)=2-(\cos (2 \pi / n)+\cos (4 \pi / n))$. We define

$$
\begin{equation*}
k_{n}^{(2)}:=\cos \left(\frac{2 \pi}{n}\right)+\cos \left(\frac{4 \pi}{n}\right) \quad \text { and } \quad h_{n}^{(2)}:=2-k_{n}^{(2)} . \tag{5.36}
\end{equation*}
$$

Now, we are ready to state the following theorem.
Theorem 5.36. Let $H$ be a spanning subgraph of a directed graph $G$ where the in- and outdegree equals one for all nodes in $H$. Let $X$ be its adjacency matrix and define $X^{(2)}:=X \cdot X$. Let $\alpha^{(2)}, \beta^{(2)} \in \mathbb{R}$ be such that $\alpha^{(2)} \geq h_{n}^{(2)} / n$ and $k_{n}^{(2)} \leq \beta^{(2)}<2$, with $k_{n}^{(2)}, h_{n}^{(2)}$ as defined in (5.36). Then, $H$ is a directed Hamiltonian cycle if and only if

$$
Z:=\beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}-\frac{1}{2}\left(\left(X+X^{(2)}\right)+\left(X+X^{(2)}\right)^{\top}\right) \succeq \mathbf{0} .
$$

Proof. Let $\tilde{H}$ be the subgraph of $G$ that has adjacency matrix $X+X^{(2)}$. Observe that $\tilde{H}$ is balanced, and thus, $\tilde{H}$ is strongly connected if and only if $a(\tilde{H})>0$.

Let $Z \succeq \mathbf{0}$, which implies that $W^{\top} Z W \succeq \mathbf{0}$. Now we can use a similar derivation as in the proof of Theorem 5.34, which results in the following:

$$
\frac{1}{2} W^{\top}\left(L_{\tilde{H}}+L_{\tilde{H}}^{\top}\right) W \succeq\left(2-\beta^{(2)}\right) \mathbf{I}_{n-1}
$$

implying

$$
a(\tilde{H})=\lambda_{\min }\left(\frac{1}{2} W^{\top}\left(L_{\tilde{H}}+L_{\tilde{H}}^{\top}\right) W\right) \geq 2-\beta^{(2)}
$$

Since $\beta^{(2)}<2$, we have $a(\tilde{H})>0$, and thus, $\tilde{H}$ is strongly connected. As $\tilde{H}$ is the union of a directed cycle cover and its implied distance two graph, $\tilde{H}$ can only be strongly connected if $H$ is strongly connected. We conclude that $H$ is a Hamiltonian cycle.

Conversely, let $H$ be a Hamiltonian cycle. In that case, the algebraic connectivity of $\tilde{H}$ is $a(\tilde{H})=2-k_{n}^{(2)}$, i.e., $\lambda_{\min }\left(\frac{1}{2} W^{\top}\left(L_{\tilde{H}}+L_{\tilde{H}}{ }^{\top}\right) W\right)=2-k_{n}^{(2)}$. Since $\beta^{(2)} \geq k_{n}^{(2)}$, this yields

$$
\frac{1}{2} W^{\top}\left(L_{\tilde{H}}+L_{\tilde{H}}^{\top}\right) W-\left(2-\beta^{(2)}\right) \mathbf{I}_{n-1} \succeq \mathbf{0}, \quad \text { or equivalently, } \quad W^{\top} Z W \succeq \mathbf{0} .
$$

Now we can use the same argument as in the proof of Theorem 5.34 to show that $Z \succeq 0$ where $\beta, \alpha$ and $k_{n}$ are replaced by $\beta^{(2)}, \alpha^{(2)}$ and $k_{n}^{(2)}$, respectively.

We define the set $\mathcal{F}_{2}$ as follows:

$$
\begin{equation*}
\mathcal{F}_{2}:=\left\{\left(y, X, X^{(2)}\right) \in \mathcal{F}_{1} \times \Pi_{n}\left(G^{2}\right): x_{i k}^{(2)}=\sum_{\substack{j \in N: \\(i, j, k) \in \mathcal{A}}} y_{i j k} \forall(i, k) \in A^{2}\right\} \tag{5.37}
\end{equation*}
$$

where

$$
\Pi_{n}\left(G^{2}\right):=\left\{X^{(2)} \in\{0,1\}^{n \times n}: \begin{array}{rl}
\operatorname{diag}\left(X^{(2)}\right) & =\mathbf{0}, X^{(2)} \mathbf{1}=\mathbf{1} \\
\left(X^{(2)}\right)^{\top} \mathbf{1}=\mathbf{1}, & x_{i j}^{(2)}
\end{array}=0 \quad \forall(i, j) \notin A^{2}\right\},
$$

and $A^{2}$ is the set of node pairs $(i, j)$ for which there exists a directed path from $i$ to $j$ of length 2. The set $\mathcal{F}_{2}$ and the result of Theorem 5.36 lead to our second ISDP formulation of the QTSP:

$$
\begin{cases}\min & \sum_{(i, j, k) \in \mathcal{A}} Q_{i j k} y_{i j k}  \tag{2}\\ \text { s.t. } & \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right) \succeq \mathbf{0} \\ & \beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}-\frac{1}{2}\left(\left(X+X^{(2)}\right)+\left(X+X^{(2)}\right)^{\top}\right) \succeq \mathbf{0} \\ & \left(y, X, X^{(2)}\right) \in \mathcal{F}_{2},\end{cases}
$$

where $\alpha \geq h_{n} / n, k_{n} \leq \beta<1, \alpha^{(2)} \geq h_{n}^{(2)} / n$ and $k_{n}^{(2)} \leq \beta^{(2)}<2$. Again the choice of $\alpha, \beta, \alpha^{(2)}$ and $\beta^{(2)}$ equal to their lower bounds provides the strongest continuous relaxation.

It follows from Theorem 5.36 that one can remove the first linear matrix inequality in $\left(I S D P_{2}\right)$ and still obtain an exact formulation of the QTSP. However, the bound obtained from the SDP relaxation of $\left(I S D P_{2}\right)$ dominates the bound obtained from the SDP relaxation of $\left(I S D P_{1}\right)$. In that sense, the formulation $\left(I S D P_{2}\right)$ can be seen as a level two formulation of the QTSP, whose continuous relaxation is stronger than that of the first level formulation. An additional advantage of the level two formulation is that both linear matrix inequalities may be used to generate CG cuts, as we show in the following section.

In the same vein, one can construct level $k$ formulations of the QTSP for $k=3, \ldots, n$. This leads to a hierarchy of formulations, whose SDP relaxations are of increasing strength and complexity.

### 5.4.3 Chvátal-Gomory cuts for the ISDPs of the QTSP

In order to solve $\left(I S D P_{1}\right)$ and $\left(I S D P_{2}\right)$ using our $\mathrm{B} \& \mathrm{C}$ algorithm, we study various CGbased separation routines for the QTSP. We first derive a general CG cut generator for the formulations $\left(I S D P_{1}\right)$ and $\left(I S D P_{2}\right)$. Thereafter, we show how different types of wellknown inequalities for the QTSP can be derived as CG cuts of the formulations (ISDP $P_{1}$ ) and $\left(I S D P_{2}\right)$.

Let us consider $\left(I S D P_{1}\right)$. The set $\mathcal{F}_{1}$, see (5.34), consists of all tuples $(y, X)$ where $X$ represents a node-disjoint cycle cover in $G$. Our B\&C algorithm starts with optimizing over the set $\mathcal{F}_{1}$, where we are allowed to relax the integrality of $y$ at no cost, see Remark 5.35. If an integer point $(\hat{y}, \hat{X})$ is found in the branching tree, it is verified whether we have $\lambda_{\min }\left(\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(\hat{X}+\hat{X}^{\top}\right)\right) \geq 0$. If so, then $\hat{X} \in \mathcal{T}_{n}(G)$ and we have found a possibly new incumbent solution. If not, then $\hat{X}$ is the adjacency matrix of a node-disjoint cycle cover that is not a Hamiltonian cycle. Therefore we have to generate dual matrices that cut off the current point.

The first separation routine that we present is based on finding a set of integer eigenvectors corresponding to a negative eigenvalue of $\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(\hat{X}+\hat{X}^{\top}\right)$.
Proposition 5.37. Let $X \in \Pi_{n}(G)$ be the adjacency matrix of a directed node-disjoint cycle cover consisting of $k \geq 2$ cycles. Let $\left\{S_{1}, \ldots, S_{k}\right\}$ be the partition of the nodes implied by the cycle cover and define for each $l \in[k]$ the vector

$$
v_{i}^{l}:= \begin{cases}n-\left|S_{l}\right| & \text { if } i \in S_{l} \\ -\left|S_{l}\right| & \text { if } i \notin S_{l}\end{cases}
$$

Then, $\left\langle v^{l}\left(v^{l}\right)^{\top}, \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right)\right\rangle<0$ for all $l \in[k]$.
Proof. The vectors $v^{l}$ are eigenvectors of $X$ and $X^{\top}$ corresponding to eigenvalue 1. Therefore we have:

$$
\begin{aligned}
\left(\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right)\right) v^{l} & =\beta v^{l}+\alpha\left(\left(n-\left|S_{l}\right|\right)\left|S_{l}\right|+\left(n-\left|S_{l}\right|\right)\left(-\left|S_{l}\right|\right)\right) \mathbf{1}-\frac{1}{2} v^{l}-\frac{1}{2} v^{l} \\
& =(\beta-1) v^{l}
\end{aligned}
$$

from where it follows that $v^{l}$ is an eigenvector of $\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right)$ corresponding to eigenvalue $\beta-1$. Since we assume $\beta<1$, this eigenvalue is negative, from which the conclusion follows.

The result of Proposition 5.37 can be used within our B\&C algorithm in the following way. Let $\left\{S_{1}, \ldots, S_{k}\right\}$ be the partition of the nodes implied by the current solution $\hat{X}$ and let $U^{l}:=v^{l}\left(v^{l}\right)^{\top}$ where $v^{l}$ is as defined in Proposition 5.37. Then for each $l \in[k]$ we construct the following CG cuts:

$$
\begin{equation*}
\left\langle U^{l}, \frac{1}{2}\left(X+X^{\top}\right)\right\rangle \leq\left\lfloor\left\langle U^{l}, \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}\right\rangle\right\rfloor \tag{5.38}
\end{equation*}
$$

or equivalently,

$$
\begin{equation*}
\left\langle U^{l}, X\right\rangle \leq\left\lfloor\left\langle U^{l}, \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}\right\rangle\right\rfloor \tag{5.39}
\end{equation*}
$$

which cut off the current point. Observe that the choice $\alpha=h_{n} / n$ and $\beta=k_{n}$ leads to noninteger values for $\alpha$ and $\beta$, i.e., the CG rounding step provides a strengthened eigenvalue cut.

Since the result of Proposition 5.37 can be repeated for the extended linear matrix inequality in Theorem 5.36, we also obtain the following CG cuts with respect to ( $I S D P_{2}$ ):

$$
\begin{equation*}
\left\langle U^{l}, X+X^{(2)}\right\rangle \leq\left\lfloor\left\langle U^{l}, \beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}\right\rangle\right\rfloor \quad \forall l \in[k] \tag{5.40}
\end{equation*}
$$

Next, we consider the class of subtour elimination constraints. It has been shown by Çezik and Iyengar [68] that the ordinary subtour elimination constraints defined by Dantzig et al. [91] can be obtained as CG cuts for the symmetric TSP, provided that $\alpha$ and $\beta$ equal their lower bounds. We extend the result from [68] and present five types of subtour elimination constraints that are in fact (strengthened) CG cuts of (ISDP1) and/or $\left(I S D P_{2}\right)$, see Table 5.1. Many of these constraints do not follow directly from the linear matrix inequalities, but require the addition of a positive multiple of a subset of the affine constraints. It is shown by Fischer [134] that the inequalities IV and V of Table 5.1 define facets of the asymmetric quadratic traveling salesman polytope.

In Appendix A.3, we explicitly derive these inequalities as (strengthened) CG cuts.

### 5.5 Computational results

In this section we test our ISDP formulations of the QTSP, see Section 5.4. We solve the ISDPs using various settings of our CG-based B\&C framework, see Algorithm 5.1, where we include different sets of cuts from Section 5.4.3 in the separation routines. We compare the performance of our approach with the two other ISDP solvers from the literature.

### 5.5.1 Design of computational experiments

In total we compare seven different approaches, among which two from the literature and five variants of our B\&C approach. The former class consists of the following:

- $\boldsymbol{K} \boldsymbol{T}$ : The B\&C algorithm of Kobayashi and Takano [234], see Section 5.3.1.
- SCIP-SDP: The general ISDP solver of Gally et al. [156]. This approach is based on solving continuous SDPs in a B\&B framework.

Another project that is known for its ability to solve ISDPs is YALMIP [257]. Preliminary experiments show, however, that the solver of [257] is significantly outperformed by the solvers from [156] and [234]. Therefore, we do not take the solver of YALMIP into account. Another notable approach is the branch-and-bound LP outer approximation algorithm Pajarito introduced in [78]. Pajarito is designed for solving mixed-integer convex programs, hence also covering ISDPs. Numerical experiments in [157] indicate that SCIP-SDP beats Pajarito in both time and number of solved instances on almost all of the tested data. An argument for this observation given in [157] is that the considered instances are provided in dual ISDP form instead of the primal form normally being used by Pajarito. For that reason, we do not take this solver into account and only consider the tailor-made ISDP solvers.

On top of the approaches from the literature, we consider five variants of our $\mathrm{B} \& \mathrm{C}$ procedure that differ in the initial feasible set and the type of cuts that we add in the separation routine:

|  | Inequality | Description |
| :---: | :---: | :---: |
| I | $\sum_{\substack{i \in S \\ j \in S}} x_{i j} \leq\|S\|-1, \quad \forall S \subset N, 2 \leq\|S\|<n$ | CG cut of $\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right) \succeq \mathbf{0}$ with dual multiplier $\mathbf{U}=\mathbb{1}_{S} \mathbb{1}_{S}{ }^{\top}$. |
| II | $\sum_{\substack{i \in S \\ j \notin S}} x_{i j} \geq 1, \quad \forall S \subset N, 2 \leq\|S\|<n$ | CG cut of $\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right) \succeq \mathbf{0}$ with dual multiplier $U=\mathbb{1}_{S} \mathbb{1}_{S}{ }^{\top}$ and $-X \mathbf{1}=-\mathbf{1}$ with dual multiplier $\mathbb{1}_{S}$. |
| III | $\begin{aligned} & \sum_{l=1}^{k} \sum_{\substack{i \in S_{l} \\ j \in S_{l}}} x_{i j}-\sum_{l \neq p} \sum_{\substack{i \in S_{l} l \\ j \in S_{p}}} x_{i j} \leq n-2 k \\ & \forall\left(S_{1}, \ldots, S_{k}\right), \cup_{l=1}^{k} S_{l}=N, S_{l} \cap S_{p}=\emptyset \forall l \neq p \end{aligned}$ | CG cut of $\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right) \succeq \mathbf{0}$ with dual multiplier $U=2 \sum_{l=1}^{k} \mathbb{1}_{S_{l}} \mathbb{1}_{S_{l}}{ }^{\top}$ and $-X \mathbf{1}=-\mathbf{1}$ with dual multiplier $\mathbf{1}$. |
| IV | $\begin{aligned} & x_{i j}+x_{j i}+\sum_{\substack{k \in N: \\ (i, k, j) \in \mathcal{A}}} y_{i k j}+\sum_{\substack{k \in N: \\ (j, k, i) \in \mathcal{A}}} y_{j k i} \leq 1 \\ & \forall i, j \in N, i \neq j, n \geq 5 \end{aligned}$ | $S$-CG cut of $\beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}-\frac{1}{2}\left(\left(X+X^{(2)}\right)\right.$ $\left.+\left(X+X^{(2)}\right)^{\top}\right) \succeq \mathbf{0}$ with dual multiplier $U=$ $\mathbb{1}_{\{i, j\}} \mathbb{1}_{\{i, j\}}{ }^{\top}$ and $\sum_{\substack{k \in N: \\(i, k, j) \in \mathcal{A}}} y_{i k j}-x_{i j}^{(2)}=0$, $\sum_{\substack{k, k \in N: \\(j, k, i) \in \mathcal{A}}} y_{j k i}-x_{j i}^{(2)}=0,-x_{i i}=0$, $-x_{j j}=0,-x_{i i}^{(2)}=0$ and $-x_{j j}^{(2)}=0$, each with dual multiplier 1. |
| V | $\begin{aligned} & \sum_{\substack{i \in S \\ j \in S}} x_{i j}+\sum_{\substack{i \in S \\ j \in S}} \sum_{\substack{k \in N \backslash S: \\ (i, k, j) \in \mathcal{A}}} y_{i k j} \leq\|S\|-1 \\ & \forall S \subset N, 2 \leq\|S\|<\frac{1}{2} n \end{aligned}$ | $S$-CG cut of $\beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}-\frac{1}{2}\left(\left(X+X^{(2)}\right)\right.$ $\left.+\left(X+X^{(2)}\right)^{\top}\right) \succeq \mathbf{0}$ with dual multiplier $U=\mathbb{1}_{S} \mathbb{1}_{S}{ }^{\top}$ and $\sum_{k \in N:(i, k, j) \in \mathcal{A}} y_{i k j}-x_{i j}^{(2)}=0$, for all $i, j \in S$, each with dual multiplier 1 , and $-y_{i k j} \leq 0$ for all $(i, k, j) \in \mathcal{A}$ with $i, k, j \in S$, each with dual multiplier 1 . |

Table 5.1: Five types of subtour elimination constraints for the QTSP that can be obtained as (strengthened) CG cuts of $\left(I S D P_{1}\right)$ and/or $\left(I S D P_{2}\right)$. The third column describes which (in)equalities and dual multipliers are used to construct the inequality.

- CG1: In this setting we solve $\left(I S D P_{1}\right)$ where we initially optimize over $\mathcal{F}_{1}$, see (5.34). In the separation routine we add the CG cut of the form (5.38) for each subtour present in the current candidate solution.
- CG2: In this setting we solve the second QTSP formulation ( $I S D P_{2}$ ). We initially optimize over $\mathcal{F}_{2}$, see (5.37), and in each callback iteration we add the CG cuts of the form (5.38) and (5.40) for each subtour in the current candidate solution.
- SEC-simple: In this setting we solve $\left(I S D P_{1}\right)$ by starting from optimizing over $\mathcal{F}_{1}$, see (5.34). In the callback procedure, we add the ordinary subtour elimination constraints, see Type I in Table 5.1, for all subtours in the current candidate solution.
- SEC: This setting solves $\left(I S D P_{2}\right)$ with subtour elimination constraints of Type I, IV and V from Table 5.1. The latter type of constraint is added only for the subtours of size less than $\frac{1}{2} n$. Since the order two variables $X^{(2)}$ in this setting do not appear
directly in the cutting planes, we eliminate them also from the initial MILP based on preliminary tests. That is, we start optimizing over $\mathcal{F}_{1}$, see (5.34). Moreover, based on a result by Fischer et al. [135] we also add additional cuts to forbid subtours of three nodes. For a triple $i, j, k$ of distinct nodes, the following cut is valid for any tour:

$$
y_{i j k}+y_{k i j} \leq x_{i j} .
$$

We add this cut for all distinct $i, j, k \in S$ in the separation routine whenever a subtour on $S$ with $|S|=3$ is present in the current candidate solution. Observe that there are six of them for each triple of nodes.

- SEC-CG: This setting solves $\left(I S D P_{2}\right)$, starting from $\mathcal{F}_{2}$, see (5.37). In the separation routines, we add all the cuts that are included in the previous setting SEC. Moreover, on top of that we also add the CG cuts (5.38) and (5.40) in the callback procedure.
Recall that the separation routines are only called at integer points, which represent cycle covers of $G$. Therefore, the separation of all mentioned cuts boils down to identifying the subtours in the cycle cover. Also, recall that the integrality of $\mathbf{y}$ is relaxed in all settings, see Remark 5.35.

The setting SEC looks similar to the best exact QTSP solving strategy of Fischer et al. [134]. However, there are two main differences between the methods. First, our separation routine is only called on integer points, while the algorithm of [135] separates on fractional points. The separation on integer points is computationally very cheap compared to the fractional separation method applied by [135]. Consequently, the former separation can lead to superior behavior, as observed by Aichholzer et al. [7] for the symmetric QTSP. Second, our approach results from a more general B\&C framework for solving integer SDPs, which is not limited to the QTSP.

Notice that the derived CG cuts of Type II and III from Table 5.1 are not added in the test settings. Preliminary experiments have shown that the cut-set subtour elimination constraints (Type II of Table 5.1) have similar practical behaviour compared to the ordinary subtour elimination constraints. Also, preliminary tests show that the addition of one merged Type III cut instead of all separate Type I cuts leads to worse behaviour in terms of overall computation time. We expect this difference to be caused by the sparsity of the Type I cuts, compared to the very dense Type III cuts.

For our tests, we consider three types of instances:

- Real instances from bioinformatics: Jäger and Molitor [224], Fischer [134] and Fischer et al. $[135,136]$ consider an important application of the QTSP in computational biology. In order to recognise transcription factor binding sites or RNA splice sites in a given set of DNA sequences, Permuted Markov (PM) models [123] or Permuted Variable Length Markov (PVLM) models [381] can be used. Finding the optimal order two PM or PVLM model boils down to solving a QTSP instance. We consider three classes of bioinformatics instances used in [133, 134], which are denoted by 'bma', 'map' and 'ml'. Each class consists of 38 instances with $n \in\{3, \ldots, 40\}$.
- Reload instances: The reload instances are the same as the ones used by Rostami et al. [326] and De Meijer and Sotirov [275]. The reload model [370] is inspired by logistics and energy distribution, where a certain cost is incurred whenever the underlying type of arc in a network changes, e.g., the means of transport. Let $G$ be a directed graph where each arc $(i, j)$ is present with probability $p$. Each arc in $G$ is randomly
assigned a color from a color set $L$ with cardinality $c$. If two successive arcs $e$ and $f$ have colors $s$ and $t$, respectively, the quadratic cost among $e$ and $f$ equals $r(s, t)$, where $r: L \times L \rightarrow \mathbb{R}$ is a reload cost function such that $r(s, s)=0$ for all $s \in L$. We consider two types of reload classes:
- Reload class 1: For each pair of distinct colors $s, t \in L$ the reload $\operatorname{cost}$ is $r(s, t)=1$;
- Reload class 2: For each pair of distinct colors $s, t \in L$, the reload cost $r(s, t)$ is chosen uniformly at random from $\{1, \ldots, 10\}$.

For each class, we consider 10 distinct instances for each possible combination of $n \in$ $\{10,15,20,25\}, p \in\{0.5,1\}$ and $c \in\{5,10,20\}$, except for the combination between $n=$ 25 and $p=1$ due to extremely large computation times. Thus, in total we consider 420 reload instances.

- Turn cost instances: The special case of the QTSP where the nodes are points in Euclidean space and the angle cost of a tour is the sum of the direction changes at the points is called the angular-metric traveling salesman problem [6]. The angular-metric TSP is motivated by VLSI design and proven to be $\mathcal{N} \mathcal{P}$-hard [6]. The problem is in the literature also known as the minimum bends traveling salesman problem [348]. We consider two classes of this type:
- TSPLIB instances: The TSP library (TSPLIB) [317] contains a broad set of TSP test instances, among which a large number of Euclidean instances. We construct a corresponding QTSP instance as follows: Given points $v_{1}, \ldots, v_{n}$ in $\mathbb{R}^{2}$, we let $G$ be the complete graph on $n$ vertices. For $i, j, k, i \neq j, j \neq k, i \neq k$, we define $q_{i j k}$ to be proportional to the angle between edges $\{i, j\}$ and $\{j, k\}$. More precisely,

$$
q_{i j k}:=\left\lceil 10 \cdot\left(1-\frac{1}{\pi} \arccos \left(\frac{\left(v_{i}-v_{j}\right)^{\top}\left(v_{k}-v_{j}\right)}{\left\|v_{i}-v_{j}\right\| \cdot\left\|v_{j}-v_{k}\right\|}\right)\right)\right] .
$$

This cost structure is similar to the angle-distance costs considered in Fischer et al. [135] and De Meijer and Sotirov [274]. In total, we consider 9 TSPLIB instances with $n$ ranging from 15 to 70 . Figure 5.5a depicts one of the TSPLIB instances including its optimal tour with respect to the defined quadratic cost structure.

- Grid instances: Fekete and Krupke [127, 128] consider problems of computing optimal covering tours and cycle covers under a turn cost model, see also Arkin et al. [19]. These problems have many practical applications, such as pest control and precision farming. Following this line, we consider the angular-metric TSP on grid graphs. We construct a 2D connected grid graph using the Type II instance generator of [128]. Given the vertex coordinate vectors $v^{1}, \ldots, v^{n} \in$ $\left\{0, \ldots, N_{1}\right\} \times\left\{0, \ldots, N_{2}\right\}$ for integers $N_{1}, N_{2}$, we include an edge between vertex $i$ and $j$ if and only if ( $v_{1}^{i}=v_{1}^{j}$ and $\left|v_{2}^{i}-v_{2}^{j}\right|=1$ ) or ( $v_{2}^{i}=v_{2}^{j}$ and $\left|v_{1}^{i}-v_{1}^{j}\right|=1$ ). If two edges $\{i, j\}$ and $\{j, k\}$ are present, the quadratic costs are computed similar as for the TSPLIB instances. In total we consider 9 grid instances with $N_{1}$ and $N_{2}$ running from 20 to 80 , corresponding to $n$ ranging from 430 to 2646 . An example of a grid instance including its minimum bend tour is given in Figure 5.5b.

Chapter 5. The Chvátal-Gomory procedure for ISDPs with applications in combinatorial optimization

Both types of turn cost instances are in fact instances of the symmetric QTSP, as they are defined on undirected graphs. To account for this, we use symmetrized versions of $\left(I S D P_{1}\right)$ and $\left(I S D P_{2}\right)$ instead. We refer to Appendix A. 4 for the construction of these formulations.


Figure 5.5: Optimal tours of two turn instances: the TSPLIB instance 'kn57' $(n=57)$ and the instance 'grid1' $(n=430)$. Each square in (b) represents a vertex in the grid graph.

All our algorithms, including the algorithm of [234], are implemented in Julia 1.5.3 using JuMP v0.21.10 [113] to model the mathematical optimization problems. In particular, we exploit the solver-independent lazy constraint callback option of JuMP to include the separation routines. Solving the underlying MILP in the subproblems is done using Gurobi v9.10 [189] in the default settings including built-in cuts. Experiments are carried out on a PC with an $\operatorname{Intel}(\mathrm{R})$ Core(TM) i7-8700 CPU, $3.20 \mathrm{GHz}, 8 \mathrm{~GB}$ RAM. To run SCIP-SDP, we use SCIP-SDP version 3.2 .0 on the NEOS Server [87], where the B\&B framework of SCIP 7.0.0 [159] and the SDP solver Mosek 9.2 [284] are combined in the default setting.

Observe that an older version of SCIP-SDP with DSDP [41] as SDP solver was used in the numerical experiments of [234], which partly explains the poor behaviour of SCIPSDP compared to the B\&C algorithm of [234]. However, our computational study that uses SCIP-SDP with the state-of-the-art SDP solver Mosek [284] also shows superior behaviour of the $\mathrm{B} \& \mathrm{C}$ algorithms.

We test all seven settings on the bioinformatics and reload instances. Since these instance classes give a clear and consistent overview of the superior approaches, we restrict ourselves to the best three settings for the turn cost instances. The maximum computation time for all our approaches is set to 8 hours, which is in correspondence with the maximum computation time on the NEOS Server [87].

### 5.5.2 Comparison of approaches

Table 5.2 and Figure 5.6 provide an overview of the performance on the instances from bioinformatics. For each setting, the average values in Table 5.2 are only computed over the instances that could be solved to optimality for that setting. An extended table on the results per instance can be found in Appendix A.5. Observe that the percentage of instances solved is quite similar over the three instance classes. This indicates that it is mainly the size rather than the cost structure that determines whether a bioinformatics instance can be solved or not. It is clear that our B\&C settings significantly outperform the other two

ISDP solvers SCIP-SDP and KT, which can solve at most $60 \%$ of the instances to optimality. Since the separation routine of CG1 is based on the identification of an integer eigenvector corresponding to a negative eigenvalue, the settings KT and CG1 are almost identical apart from the CG rounding step. The large decrease in the number of branching nodes of CG1 compared to KT is remarkable. This indicates that the effect of deeper cuts as shown in Figure 5.4 is not solely theoretical, but is also substantial from a practical point of view.

When comparing the five different separation routines of our B\&C approach, we also see a clear pattern. The settings SEC and SEC-CG turn out to be superior, being able to solve all instances within short computation times. Although SEC generally provides the fastest algorithm, it sometimes happens that SEC-CG solves the instance faster, see Figure 5.6, due to the smaller number of $\mathrm{B} \& \mathrm{C}$ nodes. This shows that the additional CG cuts can sometimes improve on the subtour elimination constraints. The two approaches are followed by SEC-simple, which is able to solve instances up to $n=35$ to optimality. This difference is mainly due to the strengthened subtour elimination cuts (type IV and V in Table 5.1) that work well for the bioinformatics instances, as also noted by Fischer et al. [135]. Finally, the settings CG1 and CG2 are only able to solve instances up to $n=32$ and $n=27$, respectively. Although the distance two CG cuts (5.40) significantly reduce the number of needed branching steps, the overall computation time is larger due to the increase in the number of variables and constraints in CG2.

| Type | Statistic | SCIP-SDP | KT | CG1 | CG2 | SEC-simple | SEC | SEC-CG |
| :--- | :--- | ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| $\mathbf{b m a}$ | Instances solved (\%) | 34.21 | 60.53 | 78.95 | 65.79 | 84.21 | $\mathbf{1 0 0}$ | $\mathbf{1 0 0}$ |
|  | Average comp. time | 1519 | 1581 | 846.1 | 639.40 | 817.43 | $\mathbf{2 8 . 3 1}$ | 182.3 |
|  | Average B\&C nodes | 30308 | 854964 | 119144 | 42515 | 85984 | 200.4 | 142.5 |
|  | Average time per node | 0.025 | 0.001 | 0.002 | 0.006 | 0.005 | 0.184 | 1.158 |
| map | Instances solved (\%) | 34.21 | 57.89 | 78.95 | 65.79 | 81.58 | $\mathbf{1 0 0}$ | $\mathbf{1 0 0}$ |
|  | Average comp. time | 2247 | 1721 | 1768 | 806.6 | 911.4 | $\mathbf{2 5 . 8 3}$ | 199.6 |
|  | Average B\&C nodes | 30385 | 896340 | 245732 | 56197 | 79869 | 244 | 173 |
|  | Average time per node | 0.037 | 0.001 | 0.002 | 0.009 | 0.004 | 0.496 | 2.094 |
|  | Instances solved (\%) | 34.21 | 57.89 | 76.32 | 65.79 | 81.58 | $\mathbf{1 0 0}$ | $\mathbf{1 0 0}$ |
| $\mathbf{m l}$ | 2891 | 1315 | 460.9 | 805.6 | 520.2 | $\mathbf{2 7 . 3 4}$ | 221.7 |  |
|  | Average comp. time | 33185 | 658640 | 86743 | 44342 | 51495 | 252.0 | 186.3 |
|  | Average B\&C nodes | 0.034 | 0.001 | 0.002 | 0.007 | 0.005 | 0.096 | 0.961 |
|  | Average time per node | 0.034 |  |  |  |  |  |  |

Table 5.2: Summary table of the performance on the bioinformatics instances per setting and per instance type. The best performing setting per row is given in bold.

Next, we discuss the results on the set of reload instances. For both class 1 and 2 and for each value of $n, p$ and $c$ we consider 10 randomly generated instances. The averaged results for each combination of parameters can be found in Appendix A.5, see Table A.10, A. 11 and A.12. In general, we see that the computation times increase with the number of nodes $n$ and the graph density $p$. On the other hand, if the number of colors $c$ increases, the instances become easier to solve as the number of (optimal) solutions will decrease. Table 5.3 shows a summary of the results accumulated over the number of colors $c$. Accordingly, Figure 5.7 shows the spread of the computation times, where we also accumulate both reload classes.

When comparing the different settings, we draw similar conclusions as before. Note that SCIP-SDP performs very poorly on the reload instances. The difference between KT and CG1 is not as significant as before, although CG1 is still favourable above KT on almost all instance types. The settings that involve the variables $X^{(2)}$ in the root node, i.e., CG2 and

SEC-CG, are outperformed by SEC-simple and SEC. Apparently, the increase in the number of variables does not contribute much to the pruning of the branching tree. In fact, the results in Appendix A. 5 even suggest that the number of branching nodes sometimes becomes larger. The large spread in computation times for these settings, see Figure 5.7, also suggests that $\left(I S D P_{2}\right)$ leads to a search process that is less robust and that this effect becomes more visible as the instances become larger. However, the $S$-CG cuts resulting from ( $I S D P_{2}$ ) do contribute to the pruning of the tree, as is suggested by the strong performance of SEC. The settings SEC and SEC-simple overall perform best. None of the two algorithms outperforms the other in terms of computation time, even when the problem size goes up, see the additional numerical results in Table A. 12 of Appendix A.5.


Figure 5.6: Computation times versus instance size for the bioinformatics classes 'bma' (top), 'map' (middle) and 'ml' (bottom). The computation times are given on a logarithmic scale.


Figure 5.7: Boxplots showing the computation times for the reload instances for different values of $n$ and $p$, accumulated over the reload class and the number of colors $c$. We omit the results of SCIP-SDP, since these computation times are several magnitudes larger.

Chapter 5. The Chvátal-Gomory procedure for ISDPs with applications in combinatorial optimization

Finally, we consider the turn cost instances. From the class of bioinformatics and reload instances it is clear that the settings SEC-simple, SEC and SEC-CG generally perform best. Hence, we restrict the numerical results on the turn cost instances to these three settings. Table 5.4 and 5.5 show the computation times and number of branching nodes for the TSPLIB and grid instances, respectively.

The TSPLIB graphs are complete graphs, and hence we can only solve up to $n=70$ for this instance type. We are able to solve all TSPLIB instances in a time span 900 seconds. Since the grid instances are more sparse, we can solve much larger instance sizes to optimality. For this type, instances up to 2646 nodes (!) can be solved to optimality within 15 seconds. These are currently the largest solved QTSP instances in the literature.

When comparing the three settings, we see that SEC-simple and SEC perform slightly better than SEC-CG on the turn cost instances. Since the different separation routines lead to different relaxations, the branching strategy between the methods can differ. Not surprisingly, the favourable setting is often the one with the smallest number of B\&C nodes, regardless of the time per branching node. Taking both the TSPLIB and grid instances into account, this happens slightly more often for the setting SEC-simple.

| Instance |  |  |  | Average computation times (s) |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| Class | $n$ | $p$ | OPT | SCIP-SDP | KT | CG1 | CG2 | SEC-simple | SEC | SEC-CG |
| 1 | 10 | 0.5 | 6.233 | 0.161 | 0.035 | 0.028 | 0.035 | 0.024 | 0.019 | 0.031 |
|  | 10 | 1 | 3.3 | 1.627 | 0.133 | 0.127 | 0.165 | 0.128 | 0.113 | 0.171 |
|  | 15 | 0.5 | 6.367 | 2.256 | 0.158 | 0.160 | 0.251 | 0.142 | 0.139 | 0.223 |
|  | 15 | 1 | 2.8 | 244.0 | 1.426 | 1.124 | 4.503 | 1.095 | 1.040 | 2.825 |
|  | 20 | 0.5 | 6.2 | 82.08 | 0.610 | 0.625 | 1.510 | 0.483 | 0.465 | 1.237 |
|  | 20 | 1 | 2.314 | 3908 | 183.8 | 91.56 | 1910 | 162.5 | 43.21 | 3278 |
|  | 25 | 0.5 | 6.6 | - | 76.20 | 35.70 | 1141 | 17.10 | 16.30 | 249.1 |
| 2 | 10 | 0.5 | 22.74 | 0.185 | 0.036 | 0.039 | 0.049 | 0.029 | 0.029 | 0.044 |
|  | 10 | 1 | 8.2 | 0.962 | 0.164 | 0.148 | 0.152 | 0.116 | 0.139 | 0.157 |
|  | 15 | 0.5 | 22.73 | 2.989 | 0.193 | 0.174 | 0.255 | 0.173 | 0.172 | 0.261 |
|  | 15 | 1 | 6.767 | 277.5 | 1.363 | 1.768 | 4.643 | 1.293 | 1.190 | 3.290 |
|  | 20 | 0.5 | 18.1 | 58.68 | 0.575 | 0.585 | 1.246 | 0.552 | 0.576 | 1.352 |
|  | 20 | 1 | 4.745 | 2689 | 43.99 | 20.88 | 1187 | 11.89 | 16.88 | 850.2 |
|  | 25 | 0.5 | 16.37 | - | 1298 | 315.1 | 5159 | 94.81 | 75.81 | 1701 |

Table 5.3: Overview of average computation times for the reload instances. Each row provides averages of 30 instances, namely 10 random instances for each value of $c=5,10,20$.

| Instance | $n$ | $m$ | OPT | SEC-simple |  | SEC |  | SEC-CG |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | Comput. time (s) | Number of nodes | Comput. time (s) | Number of nodes | Comput. time (s) | Number of nodes |
| lau15 | 15 | 105 | 47 | 0.278 | 1 | 0.458 | 1 | 0.115 | 1 |
| wg22 | 22 | 231 | 63 | 0.643 | 1 | 0.436 | 1 | 0.490 | 1 |
| bays29 | 29 | 406 | 78 | 1.519 | 96 | 0.905 | 93 | 0.949 | 78 |
| dantzig42 | 42 | 861 | 96 | 11.25 | 994 | 12.04 | 1059 | 21.20 | 1458 |
| att48 | 48 | 1128 | 105 | 53.07 | 4104 | 47.64 | 3627 | 55.52 | 3375 |
| berlin52 | 52 | 1326 | 118 | 702.9 | 36523 | 1115 | 49265 | 1070 | 41041 |
| kn57 | 57 | 1596 | 120 | 153.1 | 2425 | 110.8 | 1539 | 138.6 | 1804 |
| wg59 | 59 | 1711 | 121 | 391.2 | 10503 | 898.7 | 13627 | 650.1 | 10269 |
| st70 | 70 | 2415 | 137 | 861.8 | 8596 | 838.1 | 4649 | 1862 | 12222 |

Table 5.4: Computation times and number of branching nodes for the TSPLIB instances.

| Instance | $n$ | $m$ | OPT | SEC-simple |  | SEC |  | SEC-CG |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | Comput. time (s) | Number of nodes | Comput. time (s) | Number of nodes | Comput. time (s) | Number of nodes |
| grid1 | 430 | 795 | 620 | 1.431 | 6 | 1.538 | 1 | 1.020 | 1 |
| grid2 | 734 | 1393 | 460 | 14.86 | 2781 | 20.29 | 7942 | 19.33 | 2562 |
| grid3 | 880 | 1672 | 590 | 3.303 | 30 | 5.019 | 78 | 5.945 | 207 |
| grid4 | 960 | 1802 | 840 | 4.954 | 3 | 3.731 | 1 | 7.507 | 1 |
| grid5 | 1038 | 1965 | 440 | 8.452 | 24 | 4.514 | 16 | 8.192 | 10 |
| grid6 | 1214 | 2335 | 480 | 19.67 | 57 | 15.61 | 25 | 23.27 | 55 |
| grid 7 | 1302 | 2493 | 730 | 9.121 | 330 | 17.83 | 177 | 14.65 | 181 |
| grid8 | 1788 | 3469 | 540 | 4.800 | 1 | 4.917 | 1 | 4.619 | 1 |
| grid9 | 2646 | 5172 | 760 | 13.79 | 1 | 13.80 | 1 | 13.39 | 1 |

Table 5.5: Computation times and number of branching nodes for the grid instances.

### 5.6 Conclusions

In this chapter we study the Chvátal-Gomory cuts for spectrahedra and their strength in solving integer semidefinite programs resulting from combinatorial optimization problems. Accordingly, this chapter increases the theoretical understanding of integer semidefinite programming, which in turn contributes to new solution techniques for this type of problems.

In Section 5.2 we study the elementary closure of spectrahedra and the hierarchy obtained by iterating this procedure. Using an alternative formulation of the elementary closure, see (5.9), we provide simple proofs of several properties, including a homogeneity property for bounded spectrahedra, see Theorem 5.15. Although some of the here presented results are already known in the literature, the proofs we present are considerably simpler and are mainly based on concepts from mathematical optimization and number theory. We also present the polyhedral description of the elementary closure of spectrahedra whose defining linear matrix inequality is totally dual integral, see Theorem 5.20. To the best of our knowledge, this is the first such description for the elementary closure of a nonpolyhedral set. A full characterization of bounded LMIs that are TDI on $\mathbb{Z}^{m}$ is given in Theorem 5.23. Sufficient conditions for TDI-ness on a set $Z \subseteq \mathbb{Z}^{m}$ are given in Theorem 5.26 and 5.28.

A generic $B \& C$ algorithm for ISDPs based on strengthened CG cuts is presented in Section 5.3, see Algorithm 5.1. Our algorithm is a refinement of the algorithm from [234],

Chapter 5. The Chvátal-Gomory procedure for ISDPs with applications in combinatorial optimization
where eigenvector-based inequalities are used to separate infeasible integer points. Moreover, our approach can be seen as an extension of [68], in which the authors introduce CG cuts for conic programs, but leave the separation of CG cuts as an open problem. Our numerical results indicate the effectiveness of the use of deeper CG cuts. We also provide a separation routine for binary SDPs originating from combinatorial problems, see Section 5.3.2.

In Section 5.4 we extensively study the application of our approach to the quadratic traveling salesman problem. Based on a generalization of the notion of algebraic connectivity to directed graphs, we present two exact ISDP formulations of the QTSP, see (ISDP ${ }_{1}$ ) and $\left(I S D P_{2}\right)$. We show that the simplest CG separation routine boils down to finding integer eigenvectors of the adjacency matrix of a node-disjoint cycle cover, see Proposition 5.37. However, more intricate dual multipliers lead to some well-known families of cuts, e.g., the ordinary and strengthened versions of the subtour elimination constraints, see Table 5.1. We test several variants of our B\&C procedure that involve different separation routines.

Numerical results on the QTSP show that our B\&C algorithm significantly outperforms the two alternative ISDP solvers of [156] and [234]. For the real instances from bioinformatics $[135,136]$, these solvers are able to solve instances up to only $n=15$ and $n=25$, respectively, whereas our method can solve all instances up to $n=40$ in a short timespan. As one would expect, the extension to CG inequalities leads to deeper cuts, which successfully reduces the size of the branching tree compared to [234]. From all considered separation routines, it turns out that the setting SEC, see page 176, is overall most effective. This setting was able to solve almost all of the 552 tested QTSP instances to optimality within 5 minutes, where the largest instance contains $m=5172$ arcs.

The contents in this chapter inspires several future research directions. It would be interesting to study the performance of our B\&C algorithm when applied to other optimization problems that can be formulated as ISDPs. We expect the exploitation of CG cuts in the branching scheme to be effective for such ISDPs. Moreover, as for the QTSP many known classes of cuts turned out to be (strengthened) CG cuts with respect to the ISDP formulation, it would be interesting to know whether this also holds for other problems.

On integrality in semidefinite programming for combinatorial optimization

## Chapter summary

It is well-known that by adding integrality constraints to the semidefinite programming (SDP) relaxation of the max-cut problem, the resulting integer semidefinite program is an exact formulation of the problem. In this chapter we show similar results for a wide variety of combinatorial optimization problems for which SDP relaxations have been derived. Based on a comprehensive study on discrete positive semidefinite matrices, we follow a generic approach to derive mixed-integer semidefinite programming (MISDP) formulations for binary quadratically constrained quadratic programs and binary quadratic matrix programs. Applying a problem-specific approach, we derive more compact MISDP formulations for several problems, such as the quadratic assignment problem, the graph partition problem and the integer matrix completion problem. We also show that several structured problems allow for novel compact MISDP formulations through the notion of algebraic connectivity. Although solving mixed-integer semidefinite programs is still practically challenging, the here presented formulations induce new bounds based on integer Lagrangian duality that are at least as good as their continuous counterparts. By introducing a MISDPbased projected subgradient algorithm, we show that the resulting Lagrangian dual bounds for the max-cut problem are substantially stronger than the standard SDP bound.

### 6.1 Introduction

Semidefinite programming (SDP) deals with the optimization of a linear function over the cone of positive semidefinite matrices under the presence of affine constraints. Over the last decades, semidefinite programs (SDPs) have proven themselves particularly useful in providing tight relaxations of combinatorial optimization problems. Following the extension from linear programming to integer linear programming initiated in the 1960s, a recent interest in incorporating integer variables in SDPs has arisen. Indeed, many real-world decision problems are most naturally modeled by including integer variables in optimization problems. When the variables in an SDP are required to be integer, we refer to the problem as an integer semidefinite program (ISDP). When an SDP contains both integer and continuous variables, we refer to the program as a mixed-integer semidefinite program (MISDP). As mixed-integer linear programs (MILPs) form a subclass of MISDPs, mixed-integer semidefinite programming is in general $\mathcal{N} \mathcal{P}$-hard.

The combination of positive semidefiniteness and integrality induces a lot of structure in matrices. Exploiting that fact, it has been shown that several structured combinatorial optimization problems allow for a formulation as a (M)ISDP. To the best of our knowledge, the first ISDP formulation of a combinatorial optimization problem is derived for the symmetric traveling salesman problem by Cvetković et al. [86]. Eisenblätter [122] derives an ISDP formulation of the minimum $k$-partition problem, which asks for a partition of the vertex set of a graph into at most $k$ sets such that the total weight of the edges within the same set is minimized. Anjos and Wolkowicz [16] show that the standard SDP relaxation of the max-cut problem becomes exact when adding integrality constraints. As an immediate consequence, also the SDP relaxation of the max-2-sat problem, i.e., the maximum satisfiability problem where each clause has at most two literals, see e.g., [176], can be modeled as an ISDP. An ISDP formulation of the chromatic number of a graph is derived by Meurdesoif [279]. The quadratic traveling salesman problem (QTSP) is formulated as an ISDP in [276], see also Chapter 5. Next to these classical textbook problems, integrality in SDPs has also been at consideration in more applied problems. Yonekura and Kanno [376] formulate an optimization problem in robust truss topology design as a MISDP, see also [71, 239, 267]. The problem of computing restricted isometry constants also allows for a MISDP formulation, see [155]. A MISDP formulation of the regularized cardinality constrained least squares problem is derived in [305]. Gil-González et al. [170] use a MISDP to formulate an optimal location problem in power system analysis. Zheng et al. [383] model a robust version of a power system unit commitment problem using a mixture of semidefinite constraints and integer variables. Finally, Duarte [109] exploits MISDPs to find exact optimal designs of experiments in the domain of surface response modeling in statistics.

Despite the literature on these particular problems, a generic approach for deriving problem formulations based on mixed-integer semidefinite programming has not been followed. Although there do exist several approaches in the literature where SDP relaxations are used in a branching scheme, see e.g., [240, 364], the branching strategies are based on the problem structure rather than on the matrix variables being integer. Accordingly, exploiting integrality in the MISDP models itself has not been the method of choice so far. This might be due to the fact that solving SDPs of large sizes is still practically challenging, discouraging to look into the extension of adding hard integrality constraints to the model. Given that many problems can be modeled as a MILP, why would one have an incentive to model the problem in a form that is seemingly more complex to solve in practice?

In this chapter we refute these objections to consider MISDPs as a general modeling tech-
nique, advocating that they have a great potential to be also numerically advantageous. We particularly focus on binary quadratic programs (BQPs), which aim to optimize a quadratic objective function $g(x)$ over a feasible set $\mathcal{X}$ defined by quadratic or linear constraints, where $x$ is required to be binary, see Figure 6.1. A common approach to solve these programs is by exploiting standard linearization techniques to model them as a MILP. This is often done in a branch-and-bound setting, where the subproblems correspond to the linear programming relaxations of the MILP. This research line is depicted in the top stream of Figure 6.1. An alternative approach is to lift the vector variable $x$ in a BQP to a matrix variable $X=x x^{\top}$ so as to model the problem as an SDP with a nonconvex rank constraint. After relaxing this rank constraint, we obtain an SDP relaxation of the problem, see e.g., [15]. This relaxation approach corresponds to the bottom arrow in Figure 6.1. Apart from the particular problems mentioned earlier, it is disregarded up to this point that this relaxation can also be obtained via relaxing integrality in a MISDP model that is equivalent to the BQP. More precisely, there exists a bijection between the elements in $\mathcal{X}$ and the integer points in the feasible set of the SDP relaxation. Realizing that fact, this provides a systematic way of approaching BQPs via mixed-integer semidefinite programming. Comparing the three equivalent formulations given in Figure 6.1, the MISDP formulation has the advantages to have both a linear objective function (compared to the BQP formulation) and a convex relaxation that is often stronger than standard linear programming relaxations. After reformulating the mixed-integer nonlinear program as a convex mixed-integer nonlinear program possessing a tight relaxation, all solution techniques from convex mixed-integer nonlinear programming can be applied to tackle the problem. With the advancing state of the solution approaches in this field, the perspectives of this generic solution approach are hopeful.

The focus of this chapter is primarily on the modeling aspect of combinatorial optimization problems as (M)ISDPs, and less on the algorithmic aspects of solving these. With respect to the computational side, several general-purpose solution approaches have been considered recently. Gally et al. [156] propose a branch-and-bound framework for solving MISDPs, with the characteristic that strict duality is maintained throughout the branching tree. Solver ingredients, such as dual fixing and branching rules, are also considered in [156]. Kobayashi and Takano [234] propose a cutting-plane and a branch-and-cut algorithm for solving generic MISDPs, where it is shown that the branch-and-cut algorithm performs best. This branch-and-cut algorithm is strengthened in [276], where specialized cuts, such as Chvátal-Gomory cuts, are incorporated in the approach. Presolving techniques for MISDPs have been studied by Matter and Pfetsch [270]. Hojny and Pfetsch [211] consider reduction techniques for solving MISDPs based on permutation symmetries. The computational ingredients of the above-mentioned approaches combined with the theoretical framework of modeling problems as (M)ISDPs that we derive in this chapter, provide a balanced foundation of mixed-integer semidefinite programming in combinatorial optimization.

## Main results and outline

This chapter studies the theoretical role of mixed-integer semidefinite programming in combinatorial optimization. We show that many problems can be modeled as a (M)ISDP, either by a generic approach for certain large problem classes, or by a more problem-specific approach. Gradually, we cover and exploit results from matrix theory, combinatorics, algebraic graph theory and Lagrangian duality theory. Our approach is accompanied with a large number of examples of various combinatorial optimization problems.

We start our approach with an extensive overview of results on the matrix theory of


Figure 6.1: Overview of various exact formulations of binary quadratic program and their relaxations. A double arrow $(\Longleftrightarrow)$ denotes equivalence between the formulations, while a solid arrow $(\rightarrow)$ denotes that the formulation is relaxed from the former to the latter. The sets $\mathcal{X}, \mathcal{X}_{M I L P}$ and $\mathcal{X}_{M I S D P}$ are defined by nonconvex integer constraints, while $\overline{\mathcal{X}}_{\text {MILP }}$ and $\overline{\mathcal{X}}_{\text {MISDP }}$ are convex relaxations.
discrete positive semidefinite (PSD) matrices. Without considering explicit optimization problems, we focus on the structure of PSD $\{0,1\}$-, $\{ \pm 1\}-$ and $\{0, \pm 1\}$-matrices. This overview reviews results from [42, 110, 249, 254], but also introduces new results and formulations of these matrix sets, such as a combinatorial viewpoint of $\operatorname{PSD}\{0,1\}$-matrices of rank at most $r$. We also extend results that are known for $\{0,1\}$-matrices to the other two matrix sets.

These matrix theoretical results are exploited when proving that many binary quadratic problems allow for a formulation as a binary semidefinite program (BSDP). We establish this result for binary quadratically constrained quadratic programs and, in particular, for binary quadratic matrix programs. Problems that allow for a formulation as a binary quadratic matrix program, e.g., quadratic clustering or packing problems, can be modeled as a compact BSDP with a PSD matrix variable of relatively low order.

After that, we treat several specific problem classes for which we obtain MISDP formulations that do not follow from the above-mentioned framework or for which it is possible to obtain a more efficient formulation. Among these are the quadratic assignment problem, including its extensive number of special cases, see e.g., [62], several graph partition problems and graph problems that can be modeled based on algebraic connectivity. Also, as most formulations that we discuss include binary variables, we present several problems that have a MISDP formulation where the variables are integer, but nonbinary. Among those are the integer matrix completion problem [338], the graph coloring problem [279] and the sparse integer least squares problem [303].

As explained before, the results in this chapter can be exploited in an algorithmic framework for solving MISDPs, e.g., [156, 234, 276]. We finalize this chapter by considering an alternative way of utilizing our formulations, namely in terms of Lagrangian dual bounds. We derive several results with respect to integer Lagrangian duality theory for MISDPs and show how this results in bounds that are at least as strong as the SDP relaxations obtained from relaxing integrality. The obtained bounds are related to the so-called exact subgraph approach, see e.g., $[1,146,147,318]$, and can, for example, be derived by a projected subgradient algorithm. Several implementations of this algorithm are tested on the max-cut problem. Although the presented experiments are only preliminary, the Lagrangian dual bounds turn out to be significantly stronger than the standard SDP bounds, while they are efficiently computable.

This chapter is structured as follows. In Section 6.2 we present results on the matrix theory of discrete PSD matrices. These results are exploited in Section 6.3, where MISDP formulations of generic quadratically constrained quadratic programs and quadratic matrix programs are derived. In Section 6.4 we treat problem-specific MISDP formulations that do not follow from the previous section. Lagrangian duality for MISDPs is the topic of Section 6.5.

### 6.2 Theory on discrete PSD matrices

Most combinatorial optimization problems that we consider in this chapter are defined using binary variables, i.e., variables taking values in $\{0,1\}$ or $\{ \pm 1\}$, or ternary variables, i.e., variables whose values are in $\{0, \pm 1\}$. In this section we derive several useful results on these matrix sets with respect to positive semidefiniteness. We start by considering the PSD $\{0,1\}$ matrices, after which we extend these results to $\operatorname{PSD}\{ \pm 1\}$ - and $\{0, \pm 1\}$-matrices.

### 6.2.1 Theory on PSD $\{0,1\}$-matrices

In this section we consider the set of positive semidefinite $\{0,1\}$-matrices. As these matrices are the main objects in the remainder of the chapter, we study this set extensively. We derive and recall several formulations of this matrix set, including a combinatorial, polyhedral and a set-completely positive description.

Positive semidefinite $\{0,1\}$-matrices are studied explicitly by Letchford and Sørensen [249]. They derive the following decomposition result on PSD $\{0,1\}$-matrices.

Theorem 6.1 ([249]). Let $X \in\{0,1\}^{n \times n}$ be a symmetric matrix. Then $X \succeq \mathbf{0}$ if and only if $X=\sum_{j=1}^{r} x_{j} x_{j}^{\top}$ for some $x_{j} \in\{0,1\}^{n}, j \in[r]$.

Given a subset $S \subseteq \mathbb{R}_{+}$, a matrix $X$ is called $S$-completely positive if $X=P P^{\top}$ for some $P \in S^{n \times k}$. In case $S=\mathbb{R}_{+}$, we call $X$ completely positive. It follows from Theorem 6.1 that any $\operatorname{PSD}\{0,1\}$-matrix is $\{0,1\}$-completely positive, see also Berman and $\mathrm{Xu}[42]$.

The decomposition of PSD $\{0,1\}$-matrices gives rise to a useful combinatorial interpretation on the complete graph $K_{n}$. Viewing each vector $x_{j} \in\{0,1\}^{n}$ as an indicator vector on the vertices of $K_{n}$, the matrix $x_{j} x_{j}^{\top}$ can be seen as the characteristic matrix of a clique in $K_{n}$. Given a decomposition $X=\sum_{j=1}^{k} x_{j} x_{j}^{\top}$, the cliques indexed by $j \in[k]$ are pairwise disjoint, since the diagonal of $X$ is at most one. Therefore, each PSD $\{0,1\}$-matrix can be seen as the characteristic matrix of a set of pairwise disjoint cliques in $K_{n}$. This combinatorial structure is in the literature also known as a clique packing.

As we will see in the next section, many $\{0,1\}$-SDP formulations arise from a lifting $P P^{\top}$, where $P$ is an appropriate $n \times k\{0,1\}$-matrix. Consequently, the resulting PSD $\{0,1\}$-matrix has rank at most $k$. From that perspective, it makes sense to consider the set of PSD $\{0,1\}$ matrices that have an upper bound on the rank. For positive integers $r, n$ with $r \leq n$, let us define the discrete set

$$
\begin{equation*}
\mathcal{D}_{r}^{n}:=\left\{X \in\{0,1\}^{n \times n}: X \succeq \mathbf{0}, \operatorname{rank}(X) \leq r\right\} . \tag{6.1}
\end{equation*}
$$

Theorem 6.1 induces the following $\{0,1\}$-completely positive description of $\mathcal{D}_{r}^{n}$ :

$$
\begin{equation*}
\mathcal{D}_{r}^{n}=\left\{P P^{\top}: P \in\{0,1\}^{n \times r}, P \mathbf{1}_{r} \leq \mathbf{1}_{n}\right\} . \tag{6.2}
\end{equation*}
$$

Next, we will derive another formulation of $\mathcal{D}_{r}^{n}$, where the constraint $\operatorname{rank}(X) \leq r$ is established by an appropriate linear matrix inequality. To that end, we exploit the following result that is implicitly proved in many sources and explicitly by Dukanovic and Rendl [110].

Proposition 6.2 ([110]). Let $X \in\{0,1\}^{n \times n}$ be a symmetric matrix. Then the following statements are equivalent.
(i) $\operatorname{diag}(X)=\mathbf{1}_{n}, \operatorname{rank}(X)=r, X \succeq \mathbf{0}$.
(ii) There exists a permutation matrix $Q$ such that $Q X Q^{\top}=\mathbf{J}_{n_{1}} \oplus \cdots \oplus \mathbf{J}_{n_{r}}$ with $n=$ $n_{1}+\cdots+n_{r}$.
(iii) $\operatorname{diag}(X)=\mathbf{1}_{n}, \operatorname{rank}(X)=r$ and $X$ satisfies the triangle inequalities:

$$
X_{i j}+X_{i k}-X_{j k} \leq 1 \quad \forall(i, j, k) \in[n] \times[n] \times[n] .
$$

(iv) $\operatorname{diag}(X)=\mathbf{1}_{n}$ and $(t X-\mathbf{J} \succeq \mathbf{0} \Longleftrightarrow t \geq r)$.

Proposition 6.2 establishes the equivalence between four useful characterizations of rank- $r$ $\operatorname{PSD}\{0,1\}$-matrices that have ones on the diagonal. In the following corollary we generalize this result by relaxing the condition $\operatorname{diag}(X)=\mathbf{1}_{n}$.
Corollary 6.3. Let $X \in\{0,1\}^{n \times n}$ be a symmetric matrix. Then the following statements are equivalent.
(i) $\operatorname{rank}(X)=r, X \succeq \mathbf{0}$.
(ii) There exists a permutation matrix $Q$ such that $Q X Q^{\top}=\mathbf{J}_{n_{1}} \oplus \cdots \oplus \mathbf{J}_{n_{r}} \oplus \mathbf{0}_{n_{z} \times n_{z}}$ with $n=n_{1}+\cdots+n_{r}+n_{z}$.
(iii) $\operatorname{rank}(X)=r$ and $X$ satisfies the triangle inequalities:

$$
\left\{\begin{array}{l}
X_{i j} \leq X_{i i} \quad \forall i \neq j \\
X_{i j}+X_{i k} \leq X_{i i}+X_{j k} \quad \forall j<k, i \neq j, k
\end{array}\right.
$$

(iv) $t X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}$ if and only if $t \geq r$.

Proof. Let $X \in\{0,1\}^{n \times n}$ be symmetric. Throughout the proof, let $N_{1}:=\left\{i \in[n]: X_{i i}=1\right\}$ and let $N_{2}:=[n] \backslash N_{1}$. Moreover, let $Q^{\prime}$ denote a permutation matrix corresponding to a permutation of $[n]$ that maps the ordered set $(1,2, \ldots, n)$ to an ordered set where the elements in $N_{1}$ occupy the first $N_{1}$ positions.
$(i) \Longleftrightarrow(i i): \quad$ Let $X$ be positive semidefinite with $\operatorname{rank}(X)=r$. Then, the rows and columns indexed by $N_{2}$ only contain zeros. As a consequence, $Q^{\prime} X\left(Q^{\prime}\right)^{\top}$ is of the form $Y \oplus \mathbf{0}_{\left|N_{2}\right| \times\left|N_{2}\right|}$ with $\operatorname{diag}(Y)=\mathbf{1}_{\left|N_{1}\right|}$ and $Y \succeq \mathbf{0}$. By Proposition 6.2 there exists a permutation matrix $\bar{Q}$ such that $\bar{Q} Y \bar{Q}^{\top}=\mathbf{J}_{n_{1}} \oplus \cdots \oplus \mathbf{J}_{n_{r}}$ with $\left|N_{1}\right|=n_{1}+\cdots+n_{r}$. Let $Q:=\left(\bar{Q} \oplus \mathbf{I}_{\left|N_{2}\right|}\right) Q^{\prime}$, then $Q X Q^{\top}=\mathbf{J}_{n_{1}} \oplus \cdots \oplus \mathbf{J}_{n_{r}} \oplus \mathbf{0}_{\left|N_{2}\right| \times\left|N_{2}\right|}$.

Conversely, suppose that $Q X Q^{\top}=\mathbf{J}_{n_{1}} \oplus \cdots \oplus \mathbf{J}_{n_{r}} \oplus \mathbf{0}_{n_{z} \times n_{z}}$ with $n=n_{1}+\cdots+n_{r}+n_{z}$ for some permutation matrix $Q$. Then, obviously, $Q X Q^{\top}$ is positive semidefinite with $\operatorname{rank}\left(Q X Q^{\top}\right)=r$, from which it follows that $X \succeq \mathbf{0}$ with $\operatorname{rank}(X)=r$.
$(i) \Longleftrightarrow(i i i): \quad$ For $n=2$, the inequalities $X_{i j} \leq X_{i i}, i \neq j$, are trivially necessary and sufficient for $X \succeq \mathbf{0}$. For $n \geq 3$, the result follows from Letchford and Sørensen [249, Proposition 3]
$(i) \Longleftrightarrow(i v): \quad$ Suppose $X$ is PSD with $\operatorname{rank}(X)=r$. Then $Q^{\prime} X\left(Q^{\prime}\right)^{\top}=Y \oplus \mathbf{0}_{\left|N_{2}\right| \times\left|N_{2}\right|}$ with $\operatorname{diag}(Y)=\mathbf{1}_{\left|N_{1}\right|}$ and $Y \succeq \mathbf{0}$. This leads to the following sequence of equivalences:

$$
\begin{aligned}
t X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0} & \Longleftrightarrow t Q^{\prime} X\left(Q^{\prime}\right)^{\top}-Q^{\prime} \operatorname{diag}(X) \operatorname{diag}(X)^{\top}\left(Q^{\prime}\right)^{\top} \succeq \mathbf{0} \\
& \Longleftrightarrow t\left(Y \oplus \mathbf{0}_{\left|N_{2}\right| \times\left|N_{2}\right|}\right)-\binom{\mathbf{1}_{\left|N_{1}\right|}}{\mathbf{0}_{\left|N_{2}\right|}}\binom{\mathbf{1}_{\left|N_{1}\right|}}{\mathbf{0}_{\left|N_{2}\right|}}^{\top} \succeq \mathbf{0} \\
& \Longleftrightarrow t Y-\mathbf{J} \succeq \mathbf{0} \\
& \Longleftrightarrow t \geq r
\end{aligned}
$$

where the last line follows from statement (iv) of Proposition 6.2.
Conversely, suppose that $t X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}$ if and only if $t \geq r$. If $r=0$, then $t=0$ induces $\operatorname{diag}(X)=\mathbf{0}_{n}$, while $t=1$ implies $X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top}=X \succeq \mathbf{0}$. Hence, $X$ must be the zero matrix, which is positive semidefinite with rank zero. Now, assume that $r \geq 1$. Then, $r X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}$ can be written as $X \succeq \frac{1}{r} \operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}$. Let $r^{*}:=\operatorname{rank}(X)$. Then, it follows from the previously proven implication, $(i) \Longrightarrow(i v)$, that $r^{*}=\min \left\{t: t X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}\right\}$. By assumption the value equals $r$, so $r^{*}=r$. We conclude that $X$ is positive semidefinite with $\operatorname{rank}(X)=r$.

Corollary 6.3 can be exploited to prove the following result.
Corollary 6.4. Let $X \in\{0,1\}^{n \times n}$ be symmetric. If $Y=\left(\begin{array}{cc}r & \operatorname{diag}(X)^{\top} \\ \operatorname{diag}(X) & X\end{array}\right) \succeq \mathbf{0}$, then $X \succeq \mathbf{0}$ with $\operatorname{rank}(X) \leq r$.

Proof. The assertion $X \succeq \mathbf{0}$ is trivial, so it suffices to show that $Y \succeq \mathbf{0}$ implies $\operatorname{rank}(X) \leq r$. If $r=0$, then $\operatorname{diag}(X)=\mathbf{0}_{n}$. Since $X \succeq \mathbf{0}, X$ must be the zero matrix and, thus, $\operatorname{rank}(X)=0$.

Now, let $r \geq 1$. The Schur complement lemma implies that $r X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}$. Let $r^{*}:=\min \left\{t: t X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}\right\} \leq r$. Since $r^{*} X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}$ and $X \succeq \mathbf{0}$, it follows that $t X-\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}$ for all $t \geq r^{*}$. Therefore, $t X-$ $\operatorname{diag}(X) \operatorname{diag}(X)^{\top} \succeq \mathbf{0}$ if and only if $t \geq r^{*}$. Corollary 6.3 then implies $\operatorname{rank}(X)=r^{*} \leq r$.

Corollary 6.4 implies the following characterization of $\mathcal{D}_{r}^{n}$, where the rank constraint is merged into a lifted linear matrix inequality:

$$
\mathcal{D}_{r}^{n}=\left\{X \in\{0,1\}^{n \times n}:\left(\begin{array}{cc}
r & \operatorname{diag}(X)^{\top}  \tag{6.3}\\
\operatorname{diag}(X) & X
\end{array}\right) \succeq \mathbf{0}\right\} .
$$

For some optimization problems, the upper bound constraint on the rank of $X$ is not sufficient, as we require that $X$ is exactly of rank $r$. The max $k$-cut problem, for instance, requires to partition the vertex set of a graph into exactly $k$ nonempty and pairwise disjoint subsets. The following two results show that the description given in (6.3) can be extended to also include a lower bound on the rank of $X$.
Theorem 6.5. Let $X \in\{0,1\}^{n \times n}$ be symmetric. If there exists a matrix $P \in\{0,1\}^{n \times r}$ with $P^{\top} \mathbf{1} \geq \mathbf{1}$ such that $Y=\left(\begin{array}{cc}\mathbf{I}_{r} & P^{\top} \\ P & X\end{array}\right) \succeq \mathbf{0}$, then $X \succeq \mathbf{0}$ with $\operatorname{rank}(X) \geq r$.
Proof. The assertion $X \succeq \mathbf{0}$ is trivial. It suffices to show that $\operatorname{rank}(X) \geq r$. As $Y \succeq \mathbf{0}$ and $Y$ has binary entries, it follows from Theorem 6.1 that

$$
Y=\sum_{j=1}^{k}\binom{u_{j}}{x_{j}}\binom{u_{j}}{x_{j}}^{\top}
$$

for some $u_{j} \in\{0,1\}^{r}$ and $x_{j} \in\{0,1\}^{n}, j \in[k]$. Since $\sum_{j=1}^{k} u_{j} u_{j}^{\top}=\mathbf{I}_{r}$, we must have $k \geq r$. Moreover, the set $\left\{u_{j}: j \in[k]\right\}$ must contain $\mathbf{e}_{1}, \ldots, \mathbf{e}_{r}$ and $k-r$ copies of $\mathbf{0}_{r}$. Without loss of generality, let us assume that the first $r$ vectors in $\left\{u_{j}: j \in[k]\right\}$ correspond to the elementary vectors. Then, it follows that

$$
P=\sum_{j=1}^{k} x_{j} u_{j}^{\top}=\sum_{j=1}^{r} x_{j} \mathbf{e}_{j}^{\top}=\left(\begin{array}{ccc}
\mid & & \mid \\
x_{1} & \ldots & x_{r} \\
\mid & & \mid
\end{array}\right) .
$$

Since $P^{\top} \mathbf{1} \geq \mathbf{1}$, it follows that the vectors $x_{j}, j \in[r]$ cannot be the zero vector. Since these are moreover linearly independent, we have $\operatorname{rank}(X) \geq \operatorname{rank}\left(\sum_{j=1}^{r} x_{j} x_{j}^{\top}\right)=r$.

Theorem 6.5 and Corollary 6.4 together impose the following integer semidefinite characterization of PSD $\{0,1\}$-matrices of rank $r$.
Corollary 6.6. Let $X \in\{0,1\}^{n \times n}$ be symmetric. If there exists a matrix $P \in\{0,1\}^{n \times r}$ with $P^{\top} \mathbf{1} \geq \mathbf{1}, P \mathbf{1}=\operatorname{diag}(X)$, such that $Y=\left(\begin{array}{cc}\mathbf{I}_{r} & P^{\top} \\ P & X\end{array}\right) \succeq \mathbf{0}$, then $X \succeq \mathbf{0}$ with $\operatorname{rank}(X)=r$.

Proof. It immediately follows from Theorem 6.5 that $\mathbf{X} \succeq \mathbf{0}$ with $\operatorname{rank}(X) \geq r$. Moreover, since $Y \succeq \mathbf{0}$, we also know that

$$
\left(\mathbf{1}_{r}^{\top} \oplus \mathbf{I}_{n}\right) Y\left(\mathbf{1}_{r}^{\top} \oplus \mathbf{I}_{n}\right)^{\top}=\left(\begin{array}{cc}
\mathbf{1}_{r}^{\top} \mathbf{I}_{r} \mathbf{1}_{r} & \mathbf{1}_{r}^{\top} P^{\top} \mathbf{I}_{n} \\
\mathbf{I}_{n} P \mathbf{1}_{r} & \mathbf{I}_{n} X \mathbf{I}_{n}
\end{array}\right)=\left(\begin{array}{cc}
r & \operatorname{diag}(X)^{\top} \\
\operatorname{diag}(X) & X
\end{array}\right) \succeq \mathbf{0}
$$

It then follows from Corollary 6.4 that $\operatorname{rank}(X) \leq r$.
The integer semidefinite characterization of $\mathcal{D}_{r}^{n}$ given in (6.3) shows that if a $\{0,1\}$-matrix satisfies a certain linear matrix inequality, then a rank condition is implied. For the case of rank-one matrices, we can show that the converse implication does also hold, i.e., if a rank-one matrix satisfies a certain linear matrix inequality, then its entries must be in $\{0,1\}$.

Theorem 6.7. Let $Y=\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right) \succeq \mathbf{0}$ with $\operatorname{diag}(X)=x$. Then $\operatorname{rank}(Y)=1$ if and only if $X \in\{0,1\}^{n \times n}$.

Proof. $(\Longrightarrow)$ : If $\operatorname{rank}(Y)=1$, then $Y=\bar{x} \bar{x}^{\top}$ for some $\bar{x}=\left[1 x^{\top}\right]^{\top} \in \mathbb{R}^{n+1}$ and $X=x x^{\top}$. Since $\operatorname{diag}\left(x x^{\top}\right)=x$, we have $x_{i}^{2}=x_{i}$ for all $i \in[n]$, so $x \in\{0,1\}^{n}$. We conclude that $X=x x^{\top} \in\{0,1\}^{n \times n}$.
$(\Longleftarrow):$ Since $X \in\{0,1\}^{n \times n}$ and $x=\operatorname{diag}(X)$, it follows that $Y \in\{0,1\}^{(n+1) \times(n+1)}$. From Theorem 6.1 it follows that $Y=\sum_{j=1}^{k} x_{j} x_{j}^{\top}$ for some $x_{j} \in\{0,1\}^{n+1}, j \in[k]$, i.e., $Y$ can be decomposed in terms of cliques. Since $Y_{11}=1$ and $\operatorname{diag}(Y)=\left(1, x^{\top}\right)^{\top}$, all indices $i \in[n+1]$ for which $Y_{i i}=1$ must be in the same clique as the first index. Hence, the decomposition consists of only one clique and $\operatorname{rank}(Y)=1$.

Theorem 6.7 plays a central role in deriving integer SDP formulations of binary quadratic problems defined over vectors of variables in Section 6.3.1. However, Theorem 6.7 cannot be extended to matrices with a rank larger than one. For example, the matrix

$$
Y=\frac{1}{2}\left(\begin{array}{lll}
4 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1
\end{array}\right)
$$

satifies $Y \succeq \mathbf{0}, \operatorname{diag}(Y)=Y \mathbf{e}_{1}$ and $\operatorname{rank}(Y)=2$, but $Y$ is not integer.
The characterizations given in (6.2) and (6.3) rely on conditions involving discreteness. Let us now move on to continuous descriptions. Of course, since $\mathcal{D}_{r}^{n}$ is itself a discrete set, a continuous description does not aim at describing $\mathcal{D}_{r}^{n}$, but rather its convex hull, i.e.,

$$
\begin{equation*}
\mathcal{P}_{r}^{n}:=\operatorname{conv}\left(\mathcal{D}_{r}^{n}\right) . \tag{6.4}
\end{equation*}
$$

Observe that although the matrices in $\mathcal{D}_{r}^{n}$ have an upper bound on the rank, the polytopes $\mathcal{P}_{r}^{n}$ are full-dimensional, since $\frac{1}{n} \mathbf{I}_{n} \in \mathcal{P}_{r}^{n}$ for all $1 \leq r \leq n$. In order to gain more insight in the structure of $\mathcal{P}_{r}^{n}$, we introduce the notion of a so-called packing family.
Definition 6.8. Let $T$ be a finite set of elements. A collection $\mathcal{F}$ of nonempty subsets of $T$ is called a packing of $T$ if the subsets in $\mathcal{F}$ are pairwise disjoint. The family of all packings of $T$ is called the packing family of $T$, denoted by $\mathbf{F}(T)$.

Observe that $\mathcal{F}=\emptyset$ also belongs to $\mathbf{F}(T)$ for all sets $T$. Next, we define the notion of an $r$-packing of $T$.

Definition 6.9. Let $T$ be a finite set of elements. A packing $\mathcal{F}$ of $T$ is called an $r$-packing of $T$ if $|\mathcal{F}| \leq r$. The family of all $r$-packings of $T$ is called the $r$-packing family of $T$, denoted by $\mathbf{F}_{r}(T)$.

The $r$-packing family of $[n]$ can be exploited to describe $\mathcal{P}_{r}^{n}$. Let $X \in \mathcal{D}_{r}^{n}$. By Theorem 6.1 we know that $X$ is the sum of at most $r$ rank-one $\operatorname{PSD}\{0,1\}$-matrices. From a combinatorial point of view, this implies that $X$ corresponds to an $r$-packing of $[n]$. In fact, there is a bijection between the matrices in $\mathcal{D}_{r}^{n}$ and the $r$-packings in $\mathbf{F}_{r}([n])$. For any $r$-packing $\mathcal{F}$, let $\mathbf{E}_{\mathcal{F}}:=\sum_{S \in \mathcal{F}} \mathbb{1}_{S} \mathbb{1}_{S}^{\top}$. Then, we obtain the following polyhedral description of $\mathcal{P}_{r}^{n}$ for all positive integers $r \leq n$ :

$$
\mathcal{P}_{r}^{n}=\left\{X \in \mathcal{S}^{n}: \begin{array}{rl}
X=\sum_{\mathcal{F} \in \mathbf{F}_{r}([n])} \lambda_{\mathcal{F}} \mathbf{E}_{\mathcal{F}}, \sum_{\mathcal{F} \in \mathbf{F}_{r}([n])} \lambda_{\mathcal{F}}=1,  \tag{6.5}\\
\lambda_{\mathcal{F}} \geq 0 \text { for all } \mathcal{F} \in \mathbf{F}_{r}([n])
\end{array}\right\}
$$

We denote the description above by the packing description of $\mathcal{P}_{r}^{n}$. Let us now consider the cardinality of the vertices of $\mathcal{P}_{r}^{n}$.

In the vein of Definition 6.9, we call $\mathcal{F} \subseteq \mathbb{P}([n])$ an $r$-partition of $[n]$ if it is an $r$-packing with $\bigcup_{S \in \mathcal{F}}=[n]$. Here, $\mathbb{P}([n])$ denotes the power set of $[n]$. The number of partitions of the set $[n]$ into $k$ nonempty subsets is in the literature known as the Stirling number of the second kind, denoted by $\left\{\begin{array}{l}n \\ k\end{array}\right\}$. The total number of partitions of [ $n$ ] equals the Bell number $B_{n}$ [38], for which we have

$$
B_{n}=\sum_{k=0}^{n}\left\{\begin{array}{l}
n \\
k
\end{array}\right\} .
$$

We can now show the following result regarding the cardinality of $\mathcal{D}_{r}^{n}$.
Theorem 6.10. For $n \geq 1$ and $0 \leq r \leq n$, we have $\left|\mathcal{D}_{r}^{n}\right|=\sum_{k=1}^{r+1}\left\{\begin{array}{c}n+1 \\ k\end{array}\right\}$. In particular, $\left|\mathcal{D}_{1}^{n}\right|=2^{n}$ and $\left|\mathcal{D}_{n}^{n}\right|=B_{n+1}$.

Proof. It follows from the discussion above that $\left|\mathcal{D}_{r}^{n}\right|$ equals the number of $r$-packings in $\mathbf{F}_{r}([n])$. In order to count these, we count the number of packings that consist of exactly $k$ subsets, while $k$ ranges from 0 to $r$.

Now, any packing of [ $n$ ] into $k$ subsets corresponds to a partition of $[n+1]$ into $k+1$ subsets. To see this, observe that to each packing $\mathcal{F}$ of $[n]$ into $k$ subsets one can add a $(k+1)$ th set containing the element $n+1$ and the elements not covered by $\mathcal{F}$. Conversely, given a partition of $[n+1]$ into $k+1$ subsets, dropping the set containing the element $n+1$ yields a packing of $[n]$ consisting of exactly $k$ subsets. Hence, the number of packings of $[n]$ consisting of exactly $k$ subsets equals $\left\{\begin{array}{c}n+1 \\ k+1\end{array}\right\}$ and

$$
\left|\mathcal{D}_{r}^{n}\right|=\sum_{k=0}^{r}\left\{\begin{array}{l}
n+1 \\
k+1
\end{array}\right\}=\sum_{k=1}^{r+1}\left\{\begin{array}{c}
n+1 \\
k
\end{array}\right\} .
$$

For the special case $r=1$, we obtain $\left|\mathcal{D}_{1}^{n}\right|=\left\{\begin{array}{c}n+1 \\ 1\end{array}\right\}+\left\{\begin{array}{c}n+1 \\ 2\end{array}\right\}=1+\frac{2^{n+1}-2}{2}=2^{n}$. When $r=n$, we exploit $\left\{\begin{array}{c}n+1 \\ 0\end{array}\right\}=0$ to conclude that $\left|\mathcal{D}_{n}^{n}\right|=\sum_{k=1}^{n+1}\left\{\begin{array}{c}n+1 \\ k\end{array}\right\}=\sum_{k=0}^{n+1}\left\{\begin{array}{c}n+1 \\ k\end{array}\right\}=B_{n+1}$.

The polytope $\mathcal{P}_{r}^{n}$ has several relationships with other well-known polytopes from the literature. Letchford and Sørensen [249] study the polytope $\mathcal{P}_{n}^{n}$, albeit in a different embedding, and refer to it as the binary PSD polytope of order $n$. They emphasize its relationship with the clique partitioning polytope that was introduced by Grötschel and Wakabayashi [186] and later studied in [28, 293]. Given the complete graph $G=(V, E)$, a clique partition is a subset $A \subseteq E$ such that there is a partition of $V$ into nonempty disjoint sets $V_{1}, \ldots, V_{k}$ such that each $V_{j}, j \in[k]$, induces a clique in $G$ and $A=\bigcup_{j \in[k]}\left\{\{i, \ell\}: i, \ell \in V_{j}, i \neq \ell\right\}$. The incidence vectors of clique partitions are only defined on the edge set, and therefore the clique partition polytope can be seen as a projection of $\mathcal{P}_{n}^{n}$.

Among one of the first graph partition problems is the one considered by Chopra and Rao [76]. Given an undirected graph $G$, the vertices need to be partitioned into at most $k$ subsets so as to minimize the total cost of edges cut by the partition. When we add to $\mathcal{P}_{r}^{n}$ the additional constraint that $\operatorname{diag}(X)=\mathbf{1}_{n}$, we obtain the partition polytope $P 1(r)$ of the complete graph considered in [76] (apart from the embedding).

The polytope $\mathcal{P}_{n}^{n}$ can also be related to the stable set polytope. Let $G_{\mathbb{P}}=\left(V_{\mathbb{P}}, E_{\mathbb{P}}\right)$ be the power set graph, i.e., each vertex in $V_{\mathbb{P}}$ corresponds to a nonempty subset of $[n]$ and $E_{\mathbb{P}}:=\left\{\{S, T\} \in V_{\mathbb{P}}^{(2)}: S \cap T \neq \emptyset\right\}$. A set of vertices is stable in $G_{\mathbb{P}}$ if and only if its corresponding collection of subsets is a packing of $[n]$. Hence, the packing family $\mathbf{F}_{n}([n])$ is the collection of all stable sets in $G_{\mathbb{P}}$. It follows that there is a bijection between the elements in $\mathcal{P}_{n}^{n}$ and the stable set polytope of $G_{\mathbb{P}}$.

Finally, for $r=1$, the $r$-packings of $[n]$ are subsets of $[n]$, so the polytope $\mathcal{P}_{1}^{n}$ reduces to

$$
\begin{equation*}
\mathcal{R}_{1}^{n}:=\left\{X \in \mathcal{S}^{n}: X=\sum_{S \subseteq[n]} \theta_{S} \mathbb{1}_{S} \mathbb{1}_{S}^{\top}, \sum_{S \subseteq[n]} \theta_{S}=1, \theta_{S} \geq 0 \text { for all } S \subseteq[n]\right\} . \tag{6.6}
\end{equation*}
$$

The polytope $\mathcal{R}_{1}^{n}$ relates to the convex hull of the characteristic vectors of all cliques in $K_{n}$, i.e., the clique polytope of $K_{n}$, which is in the literature also known as the complete set packing polytope, see [58]. Finally, apart from the embedding, the polytope $\mathcal{R}_{1}^{n}$ also coincides with the Boolean quadric polytope [297], see also Section 3.3.

Another continuous formulation of the convex hull of PSD $\{0,1\}$-matrices is given by a conic description. The cone of completely positive matrices is defined as:

$$
\begin{equation*}
\mathcal{C P}^{n}:=\operatorname{conv}\left(\left\{x x^{\top}: x \in \mathbb{R}_{+}^{n}\right\}\right) . \tag{6.7}
\end{equation*}
$$

An extension of the cone of completely positive matrices are the so-called set-completely positive matrices, see e.g., [50, 254], where the membership condition $x \in \mathbb{R}_{+}^{n}$ is replaced by $x \in \mathcal{K}$ for a general convex cone $\mathcal{K}$. Lieder et al. [254] considered the following setcompletely positive matrix cone:

$$
\begin{equation*}
\mathcal{S C P}^{n}:=\operatorname{conv}\left(\left\{x x^{\top}: x \in \mathbb{R}_{+}^{n}, x_{1} \geq x_{i} \text { for all } i \in\{2, \ldots, n\}\right\}\right) \tag{6.8}
\end{equation*}
$$

We clearly have $\mathcal{S C} \mathcal{P}^{n} \subsetneq \mathcal{C} \mathcal{P}^{n}$. Let us now consider the following set-completely positive matrix set:

$$
\mathcal{C}_{1}^{n}:=\left\{X \in \mathcal{S}^{n}:\left(\begin{array}{cc}
1 & x^{\top}  \tag{6.9}\\
x & X
\end{array}\right) \in \mathcal{S C P}^{n+1}, \quad \operatorname{diag}(X)=x\right\} .
$$

The following result follows from Lieder et al. [254].
Theorem 6.11 ([254]). We have $\mathcal{P}_{1}^{n}=\mathcal{C}_{1}^{n}$.
A natural question is whether the exact descriptions $\mathcal{R}_{1}^{n}$ and $\mathcal{C}_{1}^{n}$ given in (6.6) and (6.9), respectively, for $\mathcal{P}_{1}^{n}$ can be extended to higher ranks. The rank extensions of these sets are as follows:

$$
\begin{align*}
\mathcal{R}_{r}^{n} & :=\left\{\begin{array}{c}
X=\sum_{S \subseteq[n]} \theta_{S} \mathbb{1}_{S} \mathbb{1}_{S}^{\top}, \sum_{S \subseteq[n]} \theta_{S}=r \\
X \in \mathcal{S}^{n}: \\
\sum_{S: i \in S} \theta_{S} \leq 1 \forall i \in[n], \theta_{S} \geq 0 \forall S \subseteq[n]
\end{array}\right\},  \tag{6.10}\\
\mathcal{C}_{r}^{n} & :=\left\{X \in \mathcal{S}^{n}:\left(\begin{array}{cc}
r & \operatorname{diag}(X)^{\top} \\
\operatorname{diag}(X) & X
\end{array}\right) \in \mathcal{S C P}^{n+1}, \operatorname{diag}(X) \leq \mathbf{1}_{n}\right\} . \tag{6.11}
\end{align*}
$$

The extension from $\mathcal{C}_{1}^{n}$ to $\mathcal{C}_{r}^{n}$ follows from the intersection of the Minkowski sum of $r$ copies of $\mathcal{C}_{1}^{n}$ with the upper bound constraint $X \leq \mathbf{J}_{n}$. Since $X_{i i} \geq X_{i j}$ for all $i, j \in[n]$ if $X \in \mathcal{C}_{1}^{n}$, it suffices to add $\operatorname{diag}(X) \leq \mathbf{1}_{n}$. The extension from $\mathcal{R}_{1}^{n}$ to $\mathcal{R}_{r}^{n}$ is derived as follows. If $X \in \mathcal{P}_{r}^{n}$, then $X=\sum_{\mathcal{F} \in \mathbf{F}_{r}([n])} \lambda_{\mathcal{F}} \mathbf{E}_{\mathcal{F}}$ for some nonnegative weights $\lambda_{\mathcal{F}}$. By splitting each $r$-packing into its separate subsets, we obtain

$$
X=\sum_{\mathcal{F} \in \mathbf{F}_{r}([n])} \lambda_{\mathcal{F}} \mathbf{E}_{\mathcal{F}}=\sum_{\mathcal{F} \in \mathbf{F}_{r}([n])} \lambda_{\mathcal{F}} \sum_{S \in \mathcal{F}} \mathbb{1}_{S} \mathbb{1}_{S}^{\top}=\sum_{S \subseteq[n]} \sum_{\mathcal{F} \in \mathbf{F}_{r}([n]):}^{\substack{\mathcal{F}}} \lambda_{\mathcal{F}} \mathbb{1}_{S} \mathbb{1}_{S}^{\top}=\sum_{S \subseteq[n]} \theta_{S} \mathbb{1}_{S} \mathbb{1}_{S}^{\top},
$$

where $\theta_{S}:=\sum_{\mathcal{F} \in \mathbf{F}_{r}([n]): S \in \mathcal{F}} \lambda_{\mathcal{F}}$. Moreover,

$$
\sum_{S \subseteq[n]} \theta_{S}=\sum_{\mathcal{F} \in \mathbf{F}_{r}([n])} \lambda_{F}|\mathcal{F}| \leq r \sum_{\mathcal{F} \in \mathbf{F}_{r}([n])} \lambda_{F}=r .
$$

By increasing $\theta_{\emptyset}$, we obtain $\sum_{S \subseteq[n]} \theta_{S}=r$. Finally, since $X_{i i} \leq 1$ for all $i \in[n]$, we have $\sum_{S: i \in S} \theta_{S} \leq 1$. We conclude that $\mathcal{P}_{r}^{n} \subseteq \mathcal{R}_{r}^{n}$.

Unfortunately, for $r \geq 2$, the sets $\mathcal{R}_{r}^{n}$ and $\mathcal{C}_{r}^{n}$ do no longer exactly describe $\mathcal{P}_{r}^{n}$. Namely, consider the matrix

$$
X=\left(\begin{array}{cccc}
\frac{1}{2} & 0 & \frac{1}{2} & \frac{1}{2} \\
0 & 1 & \frac{1}{2} & \frac{1}{2} \\
\frac{1}{2} & \frac{1}{2} & 1 & \frac{1}{2} \\
\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 1
\end{array}\right)=\frac{1}{2}\left(\begin{array}{lll}
1 & 0 & 0 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
1 & 0 & 1
\end{array}\right)\left(\begin{array}{lll}
1 & 0 & 0 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
1 & 0 & 1
\end{array}\right)^{\top}
$$

For this matrix one can verify that $X \in \mathcal{R}_{2}^{4}$ and $X \in \mathcal{C}_{2}^{4}$, while $X \notin \mathcal{P}_{2}^{4}$. For $r \geq 2$, the following relationship between $\mathcal{P}_{r}^{n}, \mathcal{C}_{r}^{n}, \mathcal{R}_{r}^{n}$ holds.

Theorem 6.12. We have $\mathcal{P}_{r}^{n} \subseteq \mathcal{C}_{r}^{n}=\mathcal{R}_{r}^{n}$, while for $r=1$ the three sets are equal.

Proof. Since $\mathcal{P}_{r}^{n}=\operatorname{conv}\left(\mathcal{D}_{r}^{n}\right)$, it suffices to consider membership of the elements in $\mathcal{D}_{r}^{n}$ in $\mathcal{C}_{r}^{n}$. Let $X \in \mathcal{D}_{r}^{n}$, then $X=\sum_{j=1}^{r} x_{j} x_{j}^{\top}$ for some $x_{j} \in\{0,1\}^{n}, j \in[r]$. Let $Y^{j}:=x_{j} x_{j}^{\top}$ for
all $j \in[r]$. We clearly have

$$
\left(\begin{array}{cc}
1 & \operatorname{diag}\left(Y^{j}\right) \\
\operatorname{diag}\left(Y^{j}\right) & Y^{j}
\end{array}\right) \in \mathcal{S C} P^{n+1} \text { for all } j \in[r],
$$

from where it follows that

$$
\sum_{j=1}^{r}\left(\begin{array}{cc}
1 & \operatorname{diag}\left(Y^{j}\right) \\
\operatorname{diag}\left(Y^{j}\right) & Y^{j}
\end{array}\right)=\left(\begin{array}{cc}
r & \operatorname{diag}(X) \\
\operatorname{diag}(X) & X
\end{array}\right) \in \mathcal{S C P}^{n+1} .
$$

Moreover, $X \in\{0,1\}^{n \times n}$, so $\operatorname{diag}(X) \leq \mathbf{1}_{n}$. We conclude that $X \in \mathcal{C}_{r}^{n}$.
To prove $\mathcal{C}_{r}^{n}=\mathcal{R}_{r}^{n}$, let $X \in \mathcal{C}_{r}^{n}$. We define the matrix $Y$ as

$$
Y:=\frac{1}{r}\left(\begin{array}{cc}
r & \operatorname{diag}(X)^{\top}  \tag{6.12}\\
\operatorname{diag}(X) & X
\end{array}\right)=\left(\begin{array}{cc}
1 & \operatorname{diag}\left(\frac{1}{r} X\right)^{\top} \\
\operatorname{diag}\left(\frac{1}{r} X\right) & \frac{1}{r} X
\end{array}\right)
$$

From the fact that $X \in \mathcal{C}_{r}^{n}$, it follows that $Y \in \mathcal{S C} \mathcal{P}^{n+1}$. Applying Theorem 6.11 to the matrix $Y$ implies that $\frac{1}{r} X$ is a convex combination of rank one binary PSD matrices, i.e., there exist $\theta_{S}^{\prime} \geq 0$ for all $S \subseteq[n]$ with $\sum_{S \subseteq[n]} \theta_{S}^{\prime}=1$, such that

$$
\frac{1}{r} X=\sum_{S \subseteq[n]} \theta_{S}^{\prime} \mathbb{1}_{S} \mathbb{1}_{S}^{\top} \quad \text { or equivalently, } \quad X=\sum_{S \subseteq[n]} r \theta_{S}^{\prime} \mathbb{1}_{S} \mathbb{1}_{S}^{\top} .
$$

Now, let $\theta_{S}:=r \theta_{S}^{\prime}$ for all $S \subseteq[n]$, from where it follows $\sum_{S \subseteq[n]} \theta_{S}=r$. Moreover, since $\operatorname{diag}(X) \leq \mathbf{1}_{n}$, it follows that $X_{i i}=\sum_{S: i \in S} \theta_{S} \leq 1$. We conclude that $X \in \mathcal{R}_{r}^{n}$.

Finally, observe that the argument above can also be followed in the converse direction. That is, given $X \in \mathcal{R}_{r}^{n}$ with corresponding weights $\theta_{S}$ for all $S \subseteq[n]$, we define $\theta_{S}^{\prime}:=\frac{1}{r} \theta_{S}$, $S \subseteq[n]$, from where it follows that $\frac{1}{r} X \in \mathcal{P}_{1}^{n}$. By Theorem 6.11, we know that $Y$, see (6.12), is contained in $\mathcal{S C} P^{n+1}$ and, consequently, $X \in \mathcal{C}_{r}^{n}$.

### 6.2.2 Theory on PSD $\{ \pm 1\}$-matrices

In this section we present several results for PSD matrices that have entries in $\{ \pm 1\}$. Let us first state the following well-known result.

Proposition 6.13 ([16]). Let $X$ be symmetric. Then, $X \succeq \mathbf{0}, X \in\{ \pm 1\}^{n \times n}$ if and only if $X=x x^{\top}$ for some $x \in\{ \pm 1\}^{n}$.

A simple necessary condition for $X \in\{ \pm 1\}^{n \times n}$ to be positive semidefinite is that $\operatorname{diag}(X)=1$. Next result establishes the equivalence between $\{0,1\}$ - and $\{ \pm 1\}$-PSD matrices by exploiting their rank.

Proposition 6.14. Let $X \in\{ \pm 1\}^{n \times n}$ be symmetric and $Y:=\frac{1}{2}(X+\mathbf{J}) \in\{0,1\}^{n \times n}$. Then, $X \succeq \mathbf{0}$ if and only if $\operatorname{diag}(Y)=\mathbf{1}, Y \succeq \mathbf{0}$ and $\operatorname{rank}(Y) \leq 2$.

Proof. $(\Longrightarrow)$ : Let $X \succeq \mathbf{0}$. Since $\mathbf{J} \succeq \mathbf{0}$, it follows that $Y \succeq \mathbf{0}$. Moreover, since $\operatorname{diag}(X)=$ $\operatorname{diag}(\mathbf{J})=\mathbf{1}$, we also have that $\operatorname{diag}(Y)=\mathbf{1}$. Finally, by Proposition 6.13 we know that $X=x x^{\top}$ for some $x \in\{ \pm 1\}^{n}$. Therefore $Y$ is the weighted sum of two rank one matrices, so $\operatorname{rank}(Y) \leq 2$.
$(\Longleftarrow)$ : Let $Y=\frac{1}{2}(X+\mathbf{J}) \succeq \mathbf{0}, \operatorname{diag}(Y)=\mathbf{1}$ and $\operatorname{rank}(Y) \leq 2$ for some symmetric matrix $X \in\{ \pm 1\}^{n \times n}$. Since $Y$ is binary positive semidefinite with rank at most two, it follows from Theorem 6.1 that $Y=x_{1} x_{1}^{\top}+x_{2} x_{2}^{\top}$ for some $x_{1}, x_{2} \in\{0,1\}^{n}$. By $\operatorname{diag}(Y)=\mathbf{1}$ and $x_{1}+x_{2}=\mathbf{1}$,

$$
X=2 Y-\mathbf{J}=2\left(x_{1} x_{1}^{\top}+x_{2} x_{2}^{\top}\right)-\left(x_{1}+x_{2}\right)\left(x_{1}+x_{2}\right)^{\top}=\left(x_{1}-x_{2}\right)\left(x_{1}-x_{2}\right)^{\top}
$$

which implies that $X \succeq \mathbf{0}$.
Note that the matrix $Y$ from the previous theorem has rank one if and only if $Y=X=\mathbf{J}$. Similar to (6.1), we define the discrete set of all $\{ \pm 1\}$-matrices as

$$
\begin{equation*}
\widehat{\mathcal{D}}^{n}:=\left\{X \in\{ \pm 1\}^{n \times n}: X \succeq \mathbf{0}\right\}, \tag{6.13}
\end{equation*}
$$

where the subscript $r$ is not present anymore, as all matrices in $\widehat{\mathcal{D}}^{n}$ have rank one. Based on Proposition 6.13, we can easily establish that $\left|\widehat{\mathcal{D}}^{n}\right|=2^{n-1}$. Next, we summarize known results on sets related to $\{ \pm 1\}$-matrices. The convex hull of all PSD $\{ \pm 1\}$-matrices is known as the (matrix) cut polytope:

$$
\begin{equation*}
\widehat{\mathcal{P}}^{n}:=\operatorname{conv}\left(\widehat{\mathcal{D}}^{n}\right), \tag{6.14}
\end{equation*}
$$

see e.g., [245]. Also, we define the following set-completely positive matrix cone:

$$
\begin{equation*}
\mathcal{S C} \widehat{\mathcal{P}}^{n}:=\operatorname{conv}\left(\left\{x x^{\top}: x \in \mathbb{R}^{n}, x_{1}+x_{i} \geq 0, x_{1}-x_{i} \geq 0 \text { for all } i \in\{2, \ldots, n\}\right\}\right) . \tag{6.15}
\end{equation*}
$$

The cone $\mathcal{S C} \widehat{\mathcal{P}}^{n}$ is considered in [254], where the authors show that $\mathcal{S C} \widehat{\mathcal{P}}^{n}$ and $\mathcal{S C} \mathcal{P}^{n}$, see (6.8), are related as follows:

$$
\mathcal{T}\left(\mathcal{S C P}{ }^{n}\right)=\mathcal{S C} \widehat{\mathcal{P}}^{n} \quad \text { and } \quad \mathcal{T}^{-1}\left(\mathcal{S C} \widehat{\mathcal{P}}^{n}\right)=\mathcal{S C P}{ }^{n}
$$

where $\mathcal{T}$ is an appropriate linear mapping. Lieder et al. [254] consider the following setcompletely positive matrix set:

$$
\widehat{\mathcal{C}}^{n}:=\left\{X \in \mathcal{S}^{n}:\left(\begin{array}{cc}
1 & x^{\top}  \tag{6.16}\\
x & X
\end{array}\right) \in \mathcal{S C} \widehat{\mathcal{P}}^{n+1}, \operatorname{diag}(X)=\mathbf{1}_{n}\right\}
$$

which is the analogue of the set $\mathcal{C}_{1}^{n}$ for $\{0,1\}$-matrices, see (6.9).
Theorem 6.15 ([254]). We have $\widehat{\mathcal{P}}^{n}=\widehat{\mathcal{C}}^{n}$.
This theorem is the analogue of Theorem 6.11 that provides a result for $\{0,1\}$-matrices. For the equivalence transformation between $\{ \pm 1\}$ - and $\{0,1\}$-representations of SDP relaxations of binary quadratic optimization problems, we refer the interested reader to [200].

### 6.2.3 Theory on PSD $\{0, \pm 1\}$-matrices

In the sequel we generalize several results from the previous sections to $\operatorname{PSD}\{0, \pm 1\}$-matrices. The following result shows that a PSD $\{0, \pm 1\}$-matrix is block-diagonalizable, and it is the analogue of Proposition 6.2 for $\{0,1\}$-matrices.
Proposition 6.16. Let $X \in\{0, \pm 1\}^{n \times n}$ be a symmetric matrix. Then the following statements are equivalent.
(i) $\operatorname{diag}(X)=\mathbf{1}_{n}, \operatorname{rank}(X)=r, X \succeq \mathbf{0}$.
(ii) There exists a permutation matrix $Q$ such that $Q X Q^{\top}=\mathbf{B}_{n_{1}} \oplus \cdots \oplus \mathbf{B}_{n_{r}}$, where $\mathbf{B}_{n_{i}}=$ $b_{i} b_{i}^{\top}, b_{i} \in\{ \pm 1\}^{n_{i}}$ for $i \in[r], n=n_{1}+\cdots+n_{r}$.

Proof. Suppose that $Q X Q^{\top}$ is in the block form given in (ii), then it trivially satisfies the conditions given in $(i)$. Conversely, let $X \in\{0, \pm 1\}^{n \times n}$ satisfy $(i)$. Let us consider the $i$ th row in $X$. Suppose $j$ and $k$ are two distinct indices not equal to $i$ in the support of this row, i.e., $X_{i j}, X_{i k} \neq 0$. For the sake of contradiction, suppose that $X_{j k}=0$. Then, the submatrix of $X$ induces by $i, j$ and $k$ is either one of the following matrices:

$$
\begin{aligned}
& \begin{array}{llllllllllllllll}
i & j & k & i & j & k & i & j & k & i & j
\end{array} \\
& \left.\begin{array}{l}
i \\
j \\
k
\end{array}\left(\begin{array}{ccc}
1 & 1 & 1 \\
1 & 1 & 0 \\
1 & 0 & 1
\end{array}\right), \quad \begin{array}{l}
i \\
j \\
k
\end{array}\left(\begin{array}{ccc}
1 & -1 & -1 \\
-1 & 1 & 0 \\
-1 & 0 & 1
\end{array}\right), \quad \begin{array}{c}
i \\
j \\
k
\end{array}\left(\begin{array}{ccc}
1 & 1 & -1 \\
1 & 1 & 0 \\
-1 & 0 & 1
\end{array}\right), \quad \begin{array}{c}
i
\end{array} \quad \begin{array}{c}
i \\
j \\
-1
\end{array} \begin{array}{ccc}
1 & -1 & 1 \\
1 & 0 & 1
\end{array}\right) .
\end{aligned}
$$

One easily checks that the determinants of these matrices are all negative, contradicting that $X \succeq \mathbf{0}$. Hence, $X_{j k} \neq 0$. This argument can be repeated to conclude that the submatrix of $X$ indexed by the support of row $i$ has entries in $\{ \pm 1\}$. Since it is also positive semidefinite, it follows from Proposition 6.13 that the submatrix is of the form $b b^{\top}$ with $b \in\{ \pm 1\}^{n_{i}}$ for some positive integer $n_{i}$.

By the same argument, it follows that the other indices in the submatrix induced by row $i$ have the same support as row $i$. Indeed, if this would not be the case, one of the four matrices above should be a submatrix of $X$. We conclude that $X$ can be fully constructed from nonoverlapping submatrices of the form $b b^{\top}$ with $b \in\{ \pm 1\}^{n_{i}}$ for some positive integer $n_{i}$. Since its rank equals $r$, there must be $r$ of those submatrices. From here the claim follows.

Proposition 6.16 extends easily to the following result.
Corollary 6.17. Let $X \in\{0, \pm 1\}^{n \times n}$ be a symmetric matrix. Then the following statements are equivalent.
(i) $\operatorname{rank}(X)=r, X \succeq \mathbf{0}$.
(ii) There exists a permutation matrix $Q$ such that $Q X Q^{\top}=\mathbf{B}_{n_{1}} \oplus \cdots \oplus \mathbf{B}_{n_{r}} \oplus \mathbf{0}_{n_{z} \times n_{z}}$ where $\mathbf{B}_{n_{i}}=b_{i} b_{i}^{\top}, b_{i} \in\{ \pm 1\}^{n_{i}}$ for $i \in[r], n=n_{1}+\cdots+n_{r}+n_{z}$.

Proof. See the first part of the proof of Corollary 6.3.
Let $X \in\{0, \pm 1\}^{n \times n}$ be given as in Corollary 6.17, then

$$
Q X Q^{\top}=\mathbf{B}_{n_{1}} \oplus \cdots \oplus \mathbf{B}_{n_{r}} \oplus \mathbf{0}_{n_{z} \times n_{z}}=b_{1} b_{1}^{\top} \oplus \cdots \oplus b_{r} b_{r}^{\top} \oplus \mathbf{0}_{n_{z} \times n_{z}}=\sum_{i=1}^{r} \bar{x}_{i} \bar{x}_{i}^{\top},
$$

where $\bar{x}_{1}^{\top}=\left[\begin{array}{ll}b_{1}^{\top} & \mathbf{0}_{n-n_{1}}^{\top}\end{array}\right], \bar{x}_{2}^{\top}=\left[\begin{array}{lll}\mathbf{0}_{n_{1}}^{\top} & b_{2}^{\top} & \mathbf{0}_{n-n_{1}-n_{2}}^{\top}\end{array}\right]$, and so on. Let $x_{i}:=Q \bar{x}_{i}$ for $i \in[r]$, then

$$
X=\sum_{i=1}^{r} x_{i} x_{i}^{\top}
$$

where $x_{i} \in\{0, \pm 1\}^{n}$. This construction yields the following decomposition of PSD $\{0, \pm 1\}$ matrices.

Theorem 6.18. Let $X \in\{0, \pm 1\}^{n \times n}$ be a symmetric matrix. Then $X \succeq \mathbf{0}$ if and only if $X=\sum_{j=1}^{r} x_{j} x_{j}^{\top}$ for some $x_{j} \in\{0, \pm 1\}^{n}, j \in[r]$.

The previous result is an extension of a similar result that is derived for $\operatorname{PSD}\{0,1\}-$ matrices, see Theorem 6.1. We now consider an equivalence between a PSD $\{0, \pm 1\}$-matrix of rank one and an extended linear matrix inequality, i.e., the analogue of Theorem 6.7.

Proposition 6.19. Let $Y=\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right) \in \mathcal{S}^{n+1}$ be a symmetric matrix with $\operatorname{supp}(\operatorname{diag}(X))=$ $\operatorname{supp}(x)$. Then $Y \in\{0, \pm 1\}^{(n+1) \times(n+1)}, Y \succeq \mathbf{0}$ if and only if $X=x x^{\top}$ for some $x \in\{0, \pm 1\}^{n}$.

Proof. Let $Y_{i j} \in\{0, \pm 1\}$ for all $i, j \in[n+1]$ and $Y \succeq \mathbf{0}$. Then $x \in\{0, \pm 1\}^{n}$ and it follows from the Schur complement lemma that $X-x x^{\top} \succeq \mathbf{0}$. If $X_{i i}=0$ then $x_{i}=0$, and if $X_{i i}=1$ then $x_{i}=1$ or $x_{i}=-1$. Thus $\operatorname{diag}\left(X-x x^{\top}\right)=\mathbf{0}$, from where it follows that $X=x x^{\top}$. The converse direction is trivial.

Clearly, the condition $\operatorname{supp}(\operatorname{diag}(X))=\operatorname{supp}(x)$ can be replaced by $\operatorname{diag}(X)_{i i}=\left|x_{i}\right|$ for all $i \in[n]$, where $|\cdot|$ denotes the absolute value. Finally, we prove the following result.

Proposition 6.20. Let $X \in\{0, \pm 1\}^{n \times n}$ be a symmetric matrix that can be written in the block form $X=\mathbf{B}_{n_{1}} \oplus \cdots \oplus \mathbf{B}_{n_{r}} \oplus \mathbf{0}_{n_{z} \times n_{z}}$ where $\mathbf{B}_{n_{i}}=b_{i} b_{i}^{\top}, b_{i} \in\{ \pm 1\}^{n_{i}}$ for $i \in[r]$ with $n=n_{1}+\cdots+n_{r}+n_{z}$. Then,

$$
t X-x x^{\top} \succeq 0 \text { if and only if } t \geq r
$$

where $x^{\top}=\left[\begin{array}{llll}b_{1}^{\top} & \ldots & b_{r}^{\top} & \mathbf{0}_{n_{z}}^{\top}\end{array}\right], b_{i} \in\{ \pm 1\}^{n_{i}}, i \in[r]$.

Proof. Suppose that $X$ is the given block matrix. We introduce the rank-one matrices

$$
X_{i}:=\left(\begin{array}{c}
1 \\
\mathbf{0}_{n_{1}+\cdots+n_{i-1}} \\
b_{i} \\
\mathbf{0}_{n_{i+1}+\cdots+n_{z}}
\end{array}\right)\left(\begin{array}{c}
1 \\
\mathbf{0}_{n_{1}+\cdots+n_{i-1}} \\
b_{i} \\
\mathbf{0}_{n_{i+1}+\cdots+n_{z}}
\end{array}\right)^{\top} \in\{0, \pm 1\}^{n \times n} \quad \text { for all } i \in[r] .
$$

After summing them, we obtain:

$$
\sum_{i=1}^{r} X_{i}=\left(\begin{array}{cc}
r & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}, \quad \text { or equivalently, } \quad r X-x x^{\top} \succeq \mathbf{0},
$$

where $x^{\top}=\left[\begin{array}{llll}b_{1}^{\top} & \ldots & b_{r}^{\top} & \mathbf{0}_{n_{z}}^{\top}\end{array}\right], b_{i} \in\{ \pm 1\}^{n_{i}}, i \in[r]$. Now, the statement clearly follows when we replace $r$ by $t$ such that $t \geq r$. To verify that the linear matrix inequality is not valid for $t<r$, it suffices to consider principal submatrices of $t X-x x^{\top}$ indexed by elements from different blocks except for the zero block. That submatrix is of the form $t \mathbf{I}_{k}-Z$ where $k \leq r$ and $Z \in\{ \pm 1\}^{k \times k}, Z \succeq \mathbf{0}, \operatorname{diag}(Z)=\mathbf{1}$. Since the nonzero eigenvalue of such rank-one matrix $Z$ is equal to $k$ where $k \leq r$, we have that $t \mathbf{I}_{k}-Z \succeq \mathbf{0}$ if and only if $t \geq r$.

### 6.3 Binary quadratic optimization problems

In this section we exploit the theoretic results on discrete PSD matrices from the previous section to derive exact reformulations of binary quadratic programs (BQPs) as binary semidefinite programs. In Section 6.3 .1 we consider the general class of quadratically constrained quadratic programs. In Section 6.3.2 we consider a subclass of these programs that allow for a formulation as a quadratic matrix program.

### 6.3.1 Binary quadratically constrained quadratic programs

A quadratically constrained quadratic program (QCQP) is an optimization problem with a quadratic objective function under the presence of quadratic constraints. Many combinatorial optimization problems can be formulated as QCQPs.

Let $Q_{0}, Q_{i} \in \mathcal{S}^{n}, c_{0}, c_{i} \in \mathbb{R}^{n}$ for all $i \in[m]$, and $a_{i} \in \mathbb{R}^{n}, b_{i} \in \mathbb{R}$ for all $i \in[p]$, where $m, p \in \mathbb{N}$. We consider binary programs of the following form:

$$
\begin{align*}
\min & x^{\top} Q_{0} x+c_{0}^{\top} x \\
\text { s.t. } & x^{\top} Q_{i} x+c_{i}^{\top} x \leq d_{i} \quad \forall i \in[m] \\
& a_{i}^{\top} x=b_{i} \quad \forall i \in[p]  \tag{QCQP}\\
& x \in\{0,1\}^{n} .
\end{align*}
$$

The quadratic terms in $(Q C Q P)$ can be written as $\left\langle Q_{i}, X\right\rangle+c_{i}^{\top} x$ for all $i$, where we substitute $X$ for $x x^{\top}$. This yields the following exact formulation of $(Q C Q P)$ :

$$
\begin{array}{cl}
\min & \left\langle Q_{0}, X\right\rangle+c_{0}^{\top} x \\
\text { s.t. } & \left\langle Q_{i}, X\right\rangle+c_{i}^{\top} x \leq d_{i} \quad \forall i \in[m] \\
& a_{i}^{\top} x=b_{i} \quad \forall i \in[p] \\
& \operatorname{diag}(X)=x \\
& Y=\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}, \quad \operatorname{rank}(Y)=1 .
\end{array}
$$

Here we used the conventional notion of exactness, i.e., the nonconvex constraint $\operatorname{rank}(Y)=1$. However, one can utilize an alternative notion of exactness in terms of integrality, namely by exploiting Theorem 6.7. This leads to the following BSDP:

$$
\begin{array}{cl}
\min & \left\langle Q_{0}, X\right\rangle+c_{0}^{\top} x \\
\text { s.t. } & \left\langle Q_{i}, X\right\rangle+c_{i}^{\top} x \leq d_{i} \quad \forall i \in[m] \\
& a_{i}^{\top} x=b_{i} \quad \forall i \in[p] \\
& \operatorname{diag}(X)=x \\
& \left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}, x \in\{0,1\}^{n} .
\end{array}
$$

Observe that it is sufficient to impose integrality on the diagonal of $X$. Namely, it follows
from the determinants of the $3 \times 3$ principal submatrices of the matrix $Y$ that $X_{i j} \in\{0,1\}$ whenever $X_{i i}, X_{j j} \in\{0,1\}$ for all $i$ and $j$, see e.g., [201, Section 3.2]. Hence, a matrix $X$ that is feasible for $\left(B S D P_{Q C Q P}\right)$ is an element of $\mathcal{D}_{1}^{n}$, see (6.3). The next result follows directly from the previous discussion and Theorem 6.7.

Theorem 6.21. ( $B S D P_{Q C Q P}$ ) is equivalent to $(Q C Q P)$.
To provide a more compact BSDP formulation of ( $Q C Q P$ ), we prove the following result.

Lemma 6.22. Let $S=\sum_{i=1}^{p}\binom{-b_{i}}{a_{i}}\binom{-b_{i}}{a_{i}}^{\top}$ and $Y=\left(\begin{array}{ll}1 & x^{\top} \\ x & X\end{array}\right) \succeq \mathbf{0}$, where $\operatorname{diag}(X)=x$ and $X \in\{0,1\}^{n \times n}$. Then, the following statements are equivalent:
(i) $a_{i}^{\top} x=b_{i}$ for all $i \in[p]$;
(ii) $\langle S, Y\rangle=0$.

Proof. It follows from Theorem 6.7 that $Y=\binom{1}{x}\binom{1}{x}^{\top}$. Assume that $a_{i}^{\top} x=b_{i}$ for all $i \in[p]$. Then it is not difficult to verify that $\langle S, Y\rangle=0$. Conversely, let $\langle S, Y\rangle=0$. Then,

$$
0=\sum_{i=1}^{p}\left\langle\binom{-b_{i}}{a_{i}}\binom{-b_{i}}{a_{i}}^{\top},\binom{1}{x}\binom{1}{x}^{\top}\right\rangle=\sum_{i=1}^{p}\left(b_{i}-a_{i}^{\top} x\right)^{2},
$$

from where it follows that $a_{i}^{\top} x=b_{i}$ for all $i \in[p]$.

Lemma 6.22 induces the following compact BSDP that is equivalent to $(Q C Q P)$ :

$$
\begin{aligned}
\min & \left\langle Q_{0}, X\right\rangle+c^{\top} x \\
\text { s.t. } & \left\langle Q_{i}, X\right\rangle+c_{i}^{\top} x \leq d_{i} \quad \forall i \in[m] \\
& \sum_{i=1}^{p}\left\langle\binom{-b_{i}}{a_{i}}\binom{-b_{i}}{a_{i}}^{\top},\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right)\right\rangle=0 \\
& \operatorname{diag}(X)=x \\
& \left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}, x \in\{0,1\}^{n} .
\end{aligned}
$$

There are various equivalent formulations of the binary quadratic program ( $Q C Q P$ ) in the literature. We finalize this subsection by mentioning below only those that are closely related to our approach.

Assume that $Q_{i}=\mathbf{0}, c_{i}=\mathbf{0}$, and $d_{i}=0$ for all $i \in[m]$ in $(Q C Q P)$. Burer [60] proved that the resulting optimization problem with a quadratic objective and linear constraints is
equivalent to the following completely positive program:

$$
\begin{array}{cl}
\min & \left\langle Q_{0}, X\right\rangle+c^{\top} x \\
\text { s.t. } & a_{i}{ }^{\top} x=b_{i} \quad \forall i \in[p] \\
& \left\langle a_{i} a_{i}^{\top}, X\right\rangle=b_{i}^{2} \quad \forall i \in[p] \\
& \operatorname{diag}(X)=x \\
& \left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \in \mathcal{C P}^{n+1},
\end{array}
$$

provided that the inequalities $0 \leq x_{i} \leq 1$ for $i \in[n]$ are implied by the constraints of the original problem. Here $\mathcal{C} \mathcal{P}^{n+1}$ is the cone of completely positive matrices, see (6.7).

On the other hand, Lieder et al. [254] proved the following equivalent formulation of the BQP:

$$
\begin{array}{cl}
\min & \left\langle Q_{0}, X\right\rangle+c^{\top} x \\
\text { s.t. } & a_{i}^{\top} x=b_{i} \quad \forall i \in[p] \\
& \operatorname{diag}(X)=x \\
& \left(\begin{array}{ll}
1 & x^{\top} \\
x & X
\end{array}\right) \in \mathcal{S C} P^{n+1},
\end{array}
$$

where the cone $\mathcal{S C} \mathcal{P}^{n+1}$ is defined in (6.8). The authors of [254] also proved that, under mild assumptions, the binary quadratic problem ( $Q C Q P$ ) with also quadratic constraints can be equivalently reformulated as an optimization problem over the set-completely positive matrix cone $\mathcal{S C} \mathcal{P}^{n+1}$.

Example 6.23 (The stable set problem). Let $G=(V, E)$ be a simple graph on $n$ vertices. A stable set in $G$ is a subset $S \subseteq V$ such that no two vertices in $S$ are adjacent in $G$. The stable set problem (SSP) asks for the largest size of a stable set in $G$. To model this problem, let $x \in\{0,1\}^{n}$ be such that $x_{i}=1$ if $i \in S$ and $x_{i}=0$ otherwise. Then, $x$ is the characteristic vector of a stable set in $G$ if $x^{\top}\left(\mathbf{E}_{\mathbf{i j}}+\mathbf{E}_{\mathbf{i j}}{ }^{\top}\right) x=0$ for all $\{i, j\} \in E$. The cardinality of the stable set equals $x^{\top} x$, hence the SSP is of the form $Q C Q P$. Applying Theorem 6.21, the following binary SDP models the SSP:

$$
\begin{align*}
\alpha(G):=\max & \left\langle\mathbf{I}_{n}, X\right\rangle \\
\text { s.t. } & X_{i j}=0 \quad \forall\{i, j\} \in E \\
& \operatorname{diag}(X)=x  \tag{6.17}\\
& \left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}, x \in\{0,1\}^{n} .
\end{align*}
$$

The doubly nonnegative relaxation of the SSP obtained after replacing $x \in\{0,1\}^{n}$ in (6.17) by $X \geq \mathbf{0}$, is well-studied in the literature, see e.g., [184]. It is equivalent to a strengthened version of the Lovász theta number, known as the Schrijver $\vartheta^{\prime}$-number [328].

### 6.3.2 Binary quadratic matrix programs

A quadratic matrix program (QMP) [37] is a programming formulation where the objective and constraints are quadratic functions of the form

$$
\begin{equation*}
\operatorname{tr}\left(P^{\top} Q_{i} P\right)+2 \operatorname{tr}\left(B_{i}^{\top} P\right)+d_{i} \tag{6.18}
\end{equation*}
$$

for some $Q_{i} \in \mathcal{S}^{n}, B_{i} \in \mathbb{R}^{n \times k}$ and $c_{i} \in \mathbb{R}$, where $P$ is an $n \times k$ matrix variable. QMPs are a special case of QCQPs and are particularly useful to model optimization problems where the matrix $P$ has entries in $\{0,1\}$ and represents a classification of $n$ objects over $k$ classes, i.e., $P_{i j}=1$ if and only if object $i$ is assigned to class $j$. For example, if each object needs to be assigned in exactly (resp. at most) one class, we call $P$ a partition (resp. packing) matrix.

In this section we consider two different binary QMPs of increasing generality and show how these can be reformulated as BSDPs. For both QMPs, we consider some well-known problems that fit in the framework.

Our first QMP incorporates a specific objective and constraint structure, while optimizing over the packing or partition matrices. Let $Q_{0}, Q_{i} \in \mathcal{S}^{n}, d_{i} \in \mathbb{R}$ for all $i \in[m], a_{i} \in \mathbb{R}^{n}$ and $b_{i} \in \mathbb{R}_{+}$for all $i \in[p]$. We consider the binary quadratic matrix program

$$
\begin{array}{ll}
\min & \operatorname{tr}\left(P^{\top} Q_{0} P\right) \\
\text { s.t. } & \operatorname{tr}\left(P^{\top} Q_{i} P\right)+d_{i} \leq 0 \quad \forall i \in[m] \\
& P^{\top} a_{i} \leq b_{i} \mathbf{1}_{k} \quad \forall i \in[p]  \tag{1}\\
& P \mathbf{1}_{k} \leq \mathbf{1}_{n} \\
& P \in\{0,1\}^{n \times k} .
\end{array}
$$

Observe that $P \mathbf{1}_{k} \leq \mathbf{1}_{n}$ implies that $P$ is a packing matrix. This constraint is replaced by $P \mathbf{1}_{k}=\mathbf{1}_{n}$ in case we deal with partition matrices. The constraints $\operatorname{tr}\left(P^{\top} Q_{i} P\right)+d_{i} \leq 0$ and $P^{\top} a_{i} \leq b_{i} \mathbf{1}_{k}$ might follow from the structure of the problem under consideration. Observe that these constraints differ from the general form (6.18) in the sense that the linear part $\operatorname{tr}\left(B_{i}^{\top} P\right)$ is only included in a very specific form.

A possible way to deal with the quadratic terms in $\left(Q M P_{1}\right)$ is by lifting the variables in a higher-dimensional space. By vectorizing the matrix $P$, the problem $\left(Q M P_{1}\right)$ can be written in the form $(Q C Q P)$, after which we can follow the approach of Section 6.3.1. This results in a BSDP where the matrix variable is of order $n k+1$. Since the resulting program is obtained from a lifting of the vectorization of $P$, we say that we applied a vector-lifting approach. To obtain a more compact problem formulation where the matrix variable is of lower order, we here consider a matrix-lifting approach. In particular, the objective function can be written as follows:

$$
\operatorname{tr}\left(P^{\top} Q_{0} P\right)=\operatorname{tr}\left(Q_{0} P P^{\top}\right)=\operatorname{tr}\left(Q_{0} X\right),
$$

where $X=P P^{\top}$. By doing so, we obtain the following BSDP:

$$
\begin{aligned}
\min & \left\langle Q_{0}, X\right\rangle \\
\text { s.t. } & \left\langle Q_{i}, X\right\rangle+d_{i} \leq 0 \quad \forall i \in[m] \\
& X a_{i} \leq b_{i} x \quad \forall i \in[p] \\
& \operatorname{diag}(X)=x \\
& \left(\begin{array}{cc}
k & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}, \quad X \in\{0,1\}^{n \times n} .
\end{aligned}
$$

If a QMP is defined over the partition matrices, then $P \mathbf{1}_{k} \leq \mathbf{1}_{n}$ is replaced by $P \mathbf{1}_{k}=\mathbf{1}_{n}$ in $\left(Q M P_{1}\right)$, and consequently $\operatorname{diag}(X)=x$ is replaced by $\operatorname{diag}(X)=\mathbf{1}_{n}$ in $\left(B S D P_{Q M P 1}\right)$. By exploiting theory from Section 6.2.1, we show the following equivalence.

Theorem 6.24. $\left(B S D P_{Q M P 1}\right)$ is equivalent to $\left(Q M P_{1}\right)$.

Proof. Let $P$ be feasible for $\left(Q M P_{1}\right)$ and define $X=P P^{\top}$ and $x=P \mathbf{1}_{k}$. Since $P$ represents a packing matrix, we have $X \in\{0,1\}^{n \times n}$, where $x$ is a $\{0,1\}$-vector indicating whether object $i$ is packed in one of the classes or not. Then, $\left\langle Q_{i}, X\right\rangle+d_{i}=\left\langle Q_{i}, P P^{\top}\right\rangle+d_{i}=$ $\operatorname{tr}\left(P^{\top} Q_{i} P\right)+d_{i} \leq 0$ for all $i \in[m]$. Moreover, we have $X a_{i}=P P^{\top} a_{i} \leq b_{i} P \mathbf{1}_{k}=b_{i} x$. To show that $\operatorname{diag}(X)=x$, observe that

$$
X_{i i}=\sum_{j=1}^{k} P_{i j}^{2}=\sum_{j=1}^{k} P_{i j}=\mathbf{e}_{\mathbf{i}}^{\top} P \mathbf{1}_{k}=x_{i} .
$$

Finally, we can decompose the matrix $\left(\begin{array}{ll}k & x^{\top} \\ x & X\end{array}\right)$ into

$$
\left(\begin{array}{cc}
k & x^{\top} \\
x & X
\end{array}\right)=\binom{\mathbf{1}_{k}^{\top}}{P}\binom{\mathbf{1}_{k}^{\top}}{P}^{\top},
$$

showing that it is PSD. We conclude that $X$ and $x$ are feasible for $\left(B S D P_{Q M P 1}\right)$.
To show the converse inclusion, let $X$ and $x=\operatorname{diag}(X)$ be feasible for $\left(B S D P_{Q M P 1}\right)$. It follows from Corollary 6.4 that $X$ can be decomposed as the sum of at most $k$ rank-one symmetric $\{0,1\}$-matrices. By adding copies of the zero matrix in case $\operatorname{rank}(X)<k$, we may assume that there exist $x_{1}, \ldots, x_{k} \in\{0,1\}^{n}$ such that

$$
X=\sum_{j=1}^{k} x_{j} x_{j}^{\top} .
$$

Now, let $P=\left[\begin{array}{lll}x_{1} & \ldots & x_{k}\end{array}\right]$. Then, $P \in\{0,1\}^{n \times k}$ with $P \mathbf{1}_{k}=\sum_{j=1}^{k} x_{j}=\operatorname{diag}(X) \leq \mathbf{1}_{n}$. To prove that $P^{\top} a_{i} \leq b_{i} \mathbf{1}_{k}$, consider column $j^{*}$ of $P$. Either all entries in $P \mathbf{e}_{j^{*}}\left(=x_{j^{*}}\right)$ are zero, implying that $\mathbf{e}_{j^{*}}^{\top} P^{\top} a_{i}=0 \leq b_{i}$, since $b_{i} \in \mathbb{R}_{+}$. Otherwise, there exists a row $i^{*}$ such
that $P_{i^{*} j^{*}}=1$. For the $i^{*}$ th row of $X$, we know

$$
\mathbf{e}_{i^{*}}^{\top} X=\sum_{j=1}^{k}\left(x_{j}\right)_{i^{*}} x_{j}^{\top}=x_{j^{*}}^{\top} .
$$

The $i^{*}$ th row of the system $X a_{i} \leq b_{i} x$ then reads $x_{j^{*}}^{\top} a_{i} \leq b_{i} x_{i^{*}}=b_{i}$. Hence, $P^{\top} a_{i} \leq b_{i} \mathbf{1}_{k}$. Finally, the constraint $\operatorname{tr}\left(P^{\top} Q_{i} P\right)+d_{i} \leq 0$ follows immediately from $\left\langle Q_{i}, X\right\rangle+d_{i} \leq 0$ for all $i \in[m]$. Thus, $P$ is feasible for $\left(Q M P_{1}\right)$.

As the objective functions of $\left(Q M P_{1}\right)$ and $\left(B S D P_{Q M P 1}\right)$ clearly coincide with respect to the given mapping between $P$ and $X$, we conclude that the two programs are equivalent.

The matrix $P$ does no longer appear explicitly in $\left(B S D P_{Q M P 1}\right)$, and therefore we will not be able to write all quadratic problems over the packing or partition matrices in this form. The typical problems that can be modeled this way, are the ones that are symmetric over the classes [ $k$ ], i.e., we do not add constraints for one specific class. Below we discuss two examples from the literature that fit in the framework of $\left(Q M P_{1}\right)$.

Example 6.25 (The maximum $k$-colorable subgraph problem). Let $G=(V, E)$ be a simple graph with $n:=|V|$ and $m:=|E|$. Given a positive integer $k$, a graph is called $k$-colorable if it is possible to assign to each vertex in $V$ a color in $[k]$ such that any two adjacent vertices get assigned a different color. The maximum $k$-colorable subgraph ( $\mathrm{M} k \mathrm{CS}$ ) problem, see e.g., [241, 287], asks to find an induced subgraph $G^{\prime}=\left(V^{\prime}, E^{\prime}\right)$ of $G$ that is $k$-colorable such that $\left|V^{\prime}\right|$ is maximized.

The $\mathrm{M} k \mathrm{CS}$ problem can be modeled as $\left(Q M P_{1}\right)$ where $P \in\{0,1\}^{n \times k}$ is such that $P_{i j}=1$ if and only if vertex $i \in[V]$ is in color class $j \in[k]$. In order to model that $P$ induces a coloring in $G$, we include the constraints $\operatorname{tr}\left(P^{\top}\left(\mathbf{E}_{\mathbf{i j}}+\mathbf{E}_{\mathbf{j} \mathbf{i}}\right) P\right)=0$ for all $\{i, j\} \in E$, which takes care that if two adjacent vertices $i$ and $j$ are included in the subgraph, then $i$ and $j$ must be in different color classes. The objective function can be modeled as $\operatorname{tr}\left(P^{\top} \mathbf{I}_{n} P\right)$. Additional constraints of the form $P^{\top} a_{i} \leq b_{i} \mathbf{1}_{k}$ do not appear in the formulation.

Now, it follows from Theorem 6.24 that the $\mathrm{M} k \mathrm{CS}$ problem can be modeled as the following integer SDP:

$$
\begin{align*}
\max & \left\langle\mathbf{I}_{n}, X\right\rangle \\
\text { s.t. } & X_{i j}=0 \quad \forall\{i, j\} \in E \\
& \operatorname{diag}(X)=x  \tag{6.19}\\
& \left(\begin{array}{ll}
k & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}, X \in\{0,1\}^{n \times n} .
\end{align*}
$$

The induced doubly nonnegative relaxation of (6.19) obtained after replacing $X \in\{0,1\}^{n \times n}$ by $\mathbf{0} \leq X \leq \mathbf{J}$, equals the formulation $\theta_{k}^{3}(G)$ derived in [241].

The next example shows that the parameter $k$ in $\left(B S D P_{Q M P 1}\right)$ can also be used as a variable in order to quantify the number of classes in the solution.

Example 6.26 (The quadratic bin packing problem). Let a set of $n$ items be given, each with a positive weight $w_{i} \in \mathbb{R}_{+}, i \in[n]$. Assume an unbounded number of bins is available, each with total capacity $W \in \mathbb{R}_{+}$and cost $c \in \mathbb{R}_{+}$. Moreover, let $D \in \mathcal{S}^{n}$ denote a dissimilarity matrix, where $d_{i j}$ equals the cost of packing item $i$ and $j$ in the same bin. The goal of the quadratic bin packing problem (QBPP), see e.g., [72], is to assign each item to exactly one bin, under the condition that the total sum of weights for each bin does not exceed $W$, such that the sum of the total dissimilarity and the cost of the used bins is minimized.

Let us first consider the related problem where the number of available bins equals $k$. This problem can be modeled in the form $\left(Q M P_{1}\right)$, where $P \in\{0,1\}^{n \times k}$ is a matrix with $P_{i j}=1$ if and only if item $i$ is contained in bin $j$. Since all items need to be packed, we require $P$ to be a partition matrix, i.e., $P \mathbf{1}_{k}=\mathbf{1}_{n}$. Moreover, the capacity constraints can be modeled as $P^{\top} w \leq W \mathbf{1}_{k}$. Theorem 6.24 shows that this problem can be modeled as a binary SDP where the number of bins $k$ appears as a parameter. If we replace $k$ by a variable $z$, we obtain the following formulation of the QBPP:

$$
\begin{array}{ll}
\min & \left\langle\left(\begin{array}{cc}
z & \mathbf{1}_{n}^{\top} \\
\mathbf{1}_{n} & X
\end{array}\right), c \oplus D\right\rangle \\
\text { s.t. } & X w \leq W \mathbf{1}_{n} \\
& \operatorname{diag}(X)=\mathbf{1}_{n}  \tag{6.20}\\
& \left(\begin{array}{cc}
z & \mathbf{1}_{n}^{\top} \\
\mathbf{1}_{n} & X
\end{array}\right) \succeq \mathbf{0}, X \in\{0,1\}^{n \times n}, z \in \mathbb{R} .
\end{array}
$$

The variable $z$ is not explicitly restricted to be integer, since at an optimum solution it will always be equal to $\operatorname{rank}(X)$.

The quadratic matrix program $\left(Q M P_{1}\right)$ only includes specific types of constraints of the form (6.18). We now consider a generalization of ( $Q M P_{1}$ ). Let $Q_{0}, Q_{i} \in \mathcal{S}^{n}, B_{0}, B_{i} \in \mathbb{R}^{n \times k}$ and $d_{0}, d_{i} \in \mathbb{R}$ for all $i \in[m]$ and consider the quadratic matrix program

$$
\begin{array}{ll}
\text { min } & \operatorname{tr}\left(P^{\top} Q_{0} P\right)+2 \operatorname{tr}\left(B_{0}^{\top} P\right)+d_{0} \\
\text { s.t. } & \operatorname{tr}\left(P^{\top} Q_{i} P\right)+2 \operatorname{tr}\left(B_{i}^{\top} P\right)+d_{i} \leq 0 \quad \forall i \in[m]  \tag{2}\\
& P \mathbf{1}_{k} \leq \mathbf{1}_{n} \\
& P \in\{0,1\}^{n \times k} .
\end{array}
$$

Again, the constraint $P \mathbf{1}_{k} \leq \mathbf{1}_{n}$ can be replaced by $P \mathbf{1}_{k}=\mathbf{1}_{n}$ when optimizing over partition
matrices. Now, let us consider the binary SDP

$$
\begin{aligned}
\min & \left\langle\left(\begin{array}{cc}
\frac{d_{0}}{k} \mathbf{I}_{k} & B_{0}^{\top} \\
B_{0}^{\top} & Q_{0}
\end{array}\right),\left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right)\right\rangle \\
\text { s.t. } & \left\langle\left(\begin{array}{cc}
\frac{d_{i}}{k} \mathbf{I}_{k} & B_{i}^{\top} \\
B_{i}^{\top} & Q_{i}
\end{array}\right),\left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right)\right\rangle \leq 0 \quad \forall i \in[m] \\
& \operatorname{diag}(X)=P \mathbf{1}_{k} \\
& \left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right) \succeq \mathbf{0}, X \in\{0,1\}^{n \times n}, P \in\{0,1\}^{n \times k}
\end{aligned}
$$

which is equivalent to $\left(Q M P_{2}\right)$, as shown below.
Theorem 6.27. $\left(B S D P_{Q M P 2}\right)$ is equivalent to $\left(Q M P_{2}\right)$.
Proof. Let $P$ be feasible for $\left(Q M P_{2}\right)$ and define $Y \in\{0,1\}^{(n+k) \times(n+k)}$ as

$$
Y=\binom{\mathbf{I}_{k}}{P}\binom{\mathbf{I}_{k}}{P}^{\top}=\left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right)
$$

where $X:=P P^{\top}$. Clearly, we have $Y \succeq \mathbf{0}$ and $X_{i i}=\sum_{j=1}^{k} P_{i j}^{2}=\sum_{j=1}^{k} P_{i j}=\mathbf{e}_{i}^{\top} P \mathbf{1}_{k}$ for all $i \in[n]$, showing that $\operatorname{diag}(X)=P \mathbf{1}_{k}$. Moreover, we have

$$
\begin{aligned}
\operatorname{tr}\left(P^{\top} Q_{i} P\right)+2 \operatorname{tr}\left(B_{i}^{\top} P\right)+d_{i} & =\operatorname{tr}\left(Q_{i} X\right)+2 \operatorname{tr}\left(B_{i}^{\top} P\right)+d_{i} \\
& =\left\langle\left(\begin{array}{cc}
\frac{d_{i}}{k} \mathbf{I}_{k} & B_{i}^{\top} \\
B_{i}^{\top} & Q_{i}
\end{array}\right),\left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right)\right\rangle
\end{aligned}
$$

for all $i \in[m]$ and $i=0$. Hence, $X$ and $P$ are feasible for $\left(B S D P_{Q M P 2}\right)$ and the objective functions coincide.

Conversely, let $P \in\{0,1\}^{n \times k}$ and $X \in\{0,1\}^{n \times n}$ be feasible for $\left(B S D P_{Q M P 2}\right)$. Following the proof of Theorem 6.5 , it follows that there exist $x_{1}, \ldots, x_{k^{\prime}} \in\{0,1\}^{n}$ with $k^{\prime} \geq k$ such that

$$
P=\left[\begin{array}{lll}
x_{1} & \ldots & x_{k}
\end{array}\right] \quad \text { and } \quad X=\sum_{j=1}^{k^{\prime}} x_{j} x_{j}^{\top}
$$

Since $\operatorname{diag}(X)=P \mathbf{1}_{k}$, it follows that for all $i \in[n]$ we have

$$
X_{i i}=\mathbf{e}_{i}^{\top} P \mathbf{1}_{k} \quad \text { implying that } \quad \sum_{j=1}^{k^{\prime}}\left(x_{j}\right)_{i}^{2}=\sum_{j=1}^{k}\left(x_{j}\right)_{i}
$$

Since $\left(x_{j}\right)_{i} \in\{0,1\}$, the equality above only holds if $\left(x_{j}\right)_{i}=0$ for all $j=k+1, \ldots, k^{\prime}$. As this is true for all $i \in[n]$, we have $x_{j}=\mathbf{0}_{n}$ for all $j=k+1, \ldots, k^{\prime}$, implying that $X=\sum_{j=1}^{k} x_{j} x_{j}^{\top}=P P^{\top}$. We can now follow the derivation of the first part of the proof in the converse order to conclude that $P$ is feasible for $\left(Q M P_{2}\right)$.

Typical problems that fit in the framework of $\left(Q M P_{2}\right)$ and $\left(B S D P_{Q M P 2}\right)$ are quadratic matrix programs over the packing or partition matrices that require constraints for specific classes, see e.g., Example 6.28. Another important feature of $\left(B S D P_{Q M P 2}\right)$ is that it is possible to impose a condition on the rank of $X$. Corollary 6.6 implies that if we add the constraint $P^{\top} \mathbf{1}_{n} \geq \mathbf{1}_{k}$ to ( $B S D P_{Q M P 2}$ ), the resulting matrix $X$ has rank exactly $k$. This makes this formulation suitable for quadratic classification problems that require an exact number of classes, see e.g., Example 6.29.

Example 6.28 (The quadratic multiple knapsack problem). Let a set of $n$ items be given, each with a weight $w_{i} \in \mathbb{R}_{+}$and a profit $p_{i} \in \mathbb{R}_{+}, i \in[n]$. We are also given a set of $k$ knapsacks, each with a capacity $c_{j} \in \mathbb{R}_{+}, j \in[k]$. Finally, let $R=\left(r_{i \ell}\right)$ denote a revenue matrix, where $r_{i \ell}$ denotes the revenue of including items $i$ and $\ell$ in the same knapsack. The quadratic multiple knapsack problem (QMKP) aims at allocating each item to at most one knapsack such that we maximize the total profit of the included items and their interaction revenues. The QMKP is introduced in [206] and has applications in manufacturing, scheduling and resource allocation.

Let $P \in\{0,1\}^{n \times k}$ be a packing matrix where $P_{i j}=1$ if and only if item $i$ is allocated to knapsack $j$. The capacity constraint can be modeled as $P^{\top} w \leq c$, where $w \in \mathbb{R}_{+}^{n}$ and $c \in \mathbb{R}_{+}^{k}$ denote the vector of item weights and knapsack capacities, respectively. The total profit can be computed as $\left\langle R, P P^{\top}\right\rangle+p^{\top} P \mathbf{1}_{k}$, where $p \in \mathbb{R}^{n}$ denotes the vector of item profits. Both the capacity constraints and the objective fit in the framework of ( $Q M P_{2}$ ). It then follows from Theorem 6.27 that we can model the QMKP as the following integer SDP:

$$
\begin{array}{ll}
\max & \left\langle\left(\begin{array}{cc}
\mathbf{0} & \frac{1}{2} \mathbf{1}_{k} p^{\top} \\
\frac{1}{2} p \mathbf{1}_{k}^{\top} & R
\end{array}\right),\left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right)\right\rangle \\
\text { s.t. } & P^{\top} w \leq c, \operatorname{diag}(X)=P \mathbf{1}_{k}  \tag{6.21}\\
& \left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right) \succeq \mathbf{0}, X \in\{0,1\}^{n \times n}, P \in\{0,1\}^{n \times k} .
\end{array}
$$

Example 6.29 (The capacitated max- $k$-cut problem). Let $G=(V, E)$ be an undirected simple graph on $n:=|V|$ vertices with edge weights $w_{i j}$ for all $\{i, j\} \in E$ and let $c_{1}, \ldots, c_{k}$ denote $k$ positive integers such that $c_{1}+\cdots+c_{k} \geq n$. The capacitated max- $k$-cut problem (CMkC), see e.g., [167], asks for a partition of the vertex set into $k$ nonempty subsets, each subset $j$ of cardinality at most $c_{j}$, such that the total weight of edges with both endpoints in different sets is maximized. The $\mathrm{CM} k \mathrm{C}$ is related to the graph partition problem and has applications in, e.g., transportation.

Let $P \in\{0,1\}^{n \times k}$ denote the packing matrix where $P_{i j}=1$ if and only if vertex $i$ is included in subset $j$. In order to make sure that all vertices are covered and each subset contains at least one vertex, we add the constraints $P \mathbf{1}_{k}=\mathbf{1}_{n}$ and $P^{\top} \mathbf{1}_{n} \geq \mathbf{1}_{k}$. Moreover, the capacity constraints can be modeled as $P^{\top} \mathbf{1}_{n} \leq c$, where $c \in \mathbb{R}^{k}$ denotes the vector of capacities. Finally, since one would like to add up all the edge weights that are cut by the partition, the objective function reads

$$
\frac{1}{2}\left\langle W, \mathbf{J}_{n}-P P^{\top}\right\rangle=\frac{1}{2}\left\langle\operatorname{Diag}\left(W \mathbf{1}_{n}\right), P P^{\top}\right\rangle-\frac{1}{2}\left\langle W, P P^{\top}\right\rangle=\frac{1}{2}\left\langle L, P P^{\top}\right\rangle
$$

where $W \in \mathbb{R}^{n \times n}$ is the matrix of edge weights and $L:=\operatorname{Diag}\left(W \mathbf{1}_{n}\right)-W$ denotes the weighted Laplacian matrix of $G$. Applying Theorem 6.27, the CM $k$ C problem can be modeled as follows:

$$
\begin{array}{cl}
\max & \frac{1}{2}\langle L, X\rangle \\
\text { s.t. } & \mathbf{1}_{k} \leq P^{\top} \mathbf{1}_{n} \leq c, \operatorname{diag}(X)=\mathbf{1}_{n} \\
& \left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right) \succeq \mathbf{0}, X \in\{0,1\}^{n \times n}, P \in\{0,1\}^{n \times k} . \tag{6.22}
\end{array}
$$

### 6.4 Problem-specific formulations

In this section we consider MISDP formulations of problems that do not belong to the class of binary quadratic problems or for which the reformulation technique differs from the ones discussed in Section 6.3.

### 6.4.1 The QAP as a MISDP

We present a MISDP formulation of the quadratic assignment problem (QAP) that is derived by a matrix-lifting approach. To the best of our knowledge, our QAP formulation provides the most compact convex mixed-integer formulation of the problem in the literature. The formulation is motivated by the matrix-lifting SDP relaxations of the QAP derived in [101].

The quadratic assignment problem is an optimization problem of the following form:

$$
\begin{equation*}
\min _{X \in \Pi_{n}} \operatorname{tr}\left(A X B X^{\top}\right)+\operatorname{tr}\left(C X^{\top}\right) \tag{6.23}
\end{equation*}
$$

where $A, B \in \mathcal{S}^{n}, C \in \mathbb{R}^{n \times n}$ and $\Pi_{n}$ is the set of $n \times n$ permutation matrices. The QAP is among the most difficult $\mathcal{N} \mathcal{P}$-hard combinatorial optimization problems to solve in practice. The QAP is introduced in 1957 by Koopmans and Beckmann [238] as a model for location problems. Nowadays, the QAP is known as a generic model for various (real-life) problems.

By exploiting properties of the Kronecker product and Theorem 6.7, one can lift the QAP into the space of $\left(n^{2}+1\right) \times\left(n^{2}+1\right)\{0,1\}$-matrix variables and obtain a BSDP formulation of the QAP, see Section 6.3.1. Since this vector-lifting approach results in a problem formulation with a large matrix variable, we consider here a matrix-lifting approach for the QAP. Ding and Wolkowicz [101] introduce several matrix-lifting SDP relaxations of the QAP with matrix variables of order $3 n$. By imposing integrality on the matrix variable $X$ in one of these SDP relaxations, i.e., the relaxation $M S D R_{0}$ in [101], we obtain:

$$
\begin{align*}
\min & \langle A, Y\rangle+\langle C, X\rangle \\
\text { s.t. } & \left(\begin{array}{ccc}
\mathbf{I}_{n} & X^{\top} & R^{\top} \\
X & \mathbf{I}_{n} & Y \\
R & Y & Z
\end{array}\right) \succeq \mathbf{0}  \tag{6.24}\\
& R=X B, X \in \Pi_{n}, R \in \mathbb{R}^{n \times n}, Y, Z \in \mathcal{S}^{n} .
\end{align*}
$$

Note that if $B$ is an integer matrix, then $R$ is also an integer matrix. However, we do not have to impose integrality on $R$ explicitly.

The Schur complement lemma implies that the linear matrix inequality in (6.24) is equivalent to

$$
\left(\begin{array}{cc}
\mathbf{I}_{n} & Y  \tag{6.25}\\
Y & Z
\end{array}\right)-\left(\begin{array}{ll}
X X^{\top} & X R^{\top} \\
R X^{\top} & R R^{\top}
\end{array}\right) \succeq \mathbf{0} .
$$

Now, we are ready to prove the following result.
Proposition 6.30. The MISDP (6.24) is equivalent to (6.23).
Proof. Let ( $X, Y, Z, R$ ) be feasible for (6.24). Then $X X^{\top}=\mathbf{I}_{n}$ and

$$
\left(\begin{array}{cc}
\mathbf{I}_{n}-X X^{\top} & Y-X R^{\top} \\
Y-R X^{\top} & Z-R R^{\top}
\end{array}\right) \succeq \mathbf{0}
$$

imply that $Y=X R^{\top}$. Thus, $Y=X B^{\top} X^{\top}=X B X^{\top}$, meaning that the two objectives coincide.

Conversely, let $X$ be feasible for (6.23). Define $R:=X B, Y:=X R^{\top}$ and $Z:=R R^{\top}$. It trivially follows that the constraints in (6.24) are satisfied and that the two objective functions coincide.

Many combinatorial optimization problems can be formulated as the QAP, see e.g., [62]. We provide an example below.

Example 6.31 (The traveling salesman problem). We are given a complete undirected graph $K_{n}=(V, E)$ with $n:=|V|$ vertices and a nonnegative matrix $D=\left(d_{i j}\right) \in \mathcal{S}^{n}$, where $d_{i j}$ is the cost of edge $\{i, j\} \in E$. The goal of the traveling salesman problem (TSP) is to find a Hamiltonian cycle of minimum cost in $K_{n}$.

Let $B$ be the adjacency matrix of the standard circuit on $n$ vertices, i.e., $B$ is a symmetric Toeplitz matrix whose first row is [0 $\left.1 \mathbf{0}_{n-3}^{\top} 1\right]$. It is well-known that (6.23) with this matrix $B$ and $A=D$ is a formulation for the TSP. Following Proposition 6.30, a MISDP formulation of the TSP is:

$$
\begin{array}{ll}
\min & \frac{1}{2}\langle D, Y\rangle \\
\text { s.t. } & R=X B \\
& \left(\begin{array}{ccc}
\mathbf{I}_{n} & X^{\top} & R^{\top} \\
X & \mathbf{I}_{n} & Y \\
R & Y & Z
\end{array}\right) \succeq \mathbf{0} \\
& X \in \Pi_{n}, R \in \mathbb{R}^{n \times n}, Y, Z \in \mathcal{S}^{n} .
\end{array}
$$

Another MISDP formulation of the TSP is given in Section 6.4.3, see also (6.33). The latter formulation is, to the best of our knowledge, the most compact formulation of the TSP.

### 6.4.2 MISDP formulations of the graph partition problem

We present here various MISDP formulations of the graph partition problem (GPP). Several of the here derived formulations cannot be obtained by using results from Section 6.3.1 and Section 6.3.2.

The GPP is the problem of partitioning the vertex set of a graph into a fixed number of sets, say $k$, of given sizes such that the sum of weights of edges joining different sets is optimized. If all sets are of equal size, then the corresponding problem is known as the $k$ equipartition problem ( $k$-EP). The case of the GPP with $k=2$ is known as the graph bisection problem (GBP). To formalize, let $G=(V, E)$ be an undirected graph on $n:=|V|$ vertices and let $W:=\left(w_{i j}\right) \in \mathcal{S}^{n}$ denote a weight matrix with $w_{i j}=0$ if $\{i, j\} \notin E$. The graph partition problem aims to partition the vertices of $G$ into $k(2 \leq k \leq n-1)$ disjoint sets $S_{1}, \ldots, S_{k}$ of specified sizes $m_{1} \geq \cdots \geq m_{k} \geq 1, \sum_{j=1}^{k} m_{j}=\bar{n}$ such that the total weight of edges joining different sets $S_{j}$ is minimized.

For a given partition of $V$ into $k$ subsets, let $P=\left(P_{i j}\right) \in\{0,1\}^{n \times k}$ be the partition matrix, where $P_{i j}=1$ if and only if $i \in S_{j}$ for $i \in[n]$ and $j \in[k]$. The total weight of the partition equals:

$$
\begin{equation*}
\frac{1}{2} \operatorname{tr}\left(W\left(\mathbf{J}_{n}-P P^{\top}\right)\right)=\frac{1}{2} \operatorname{tr}\left(L P P^{\top}\right) \tag{6.26}
\end{equation*}
$$

where $L:=\operatorname{Diag}\left(W \mathbf{1}_{n}\right)-W$ is the weighted Laplacian matrix of $G$. The GPP can be formulated as the following quadratic matrix program:

$$
\begin{array}{ll}
\min & \frac{1}{2}\left\langle L, P P^{\top}\right\rangle \\
\text { s.t. } & P \mathbf{1}_{k}=\mathbf{1}_{n}  \tag{6.27}\\
& P^{\top} \mathbf{1}_{n}=\mathbf{m} \\
& P \in\{0,1\}^{n \times k}
\end{array}
$$

where $\mathbf{m}=\left[\begin{array}{lll}m_{1} & \ldots & m_{k}\end{array}\right]^{\top}$. The formulation (6.27) is a special case of the quadratic matrix program $\left(Q M P_{2}\right)$. Therefore, applying Theorem 6.27 , the GPP can be modeled as follows:

$$
\begin{array}{ll}
\min & \frac{1}{2}\langle L, X\rangle \\
\text { s.t. } & P \mathbf{1}_{k}=\mathbf{1}_{n} \\
& P^{\top} \mathbf{1}_{n}=\mathbf{m} \\
& \operatorname{diag}(X)=\mathbf{1}_{n} \\
& \left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X
\end{array}\right) \succeq \mathbf{0}, X \in\{0,1\}^{n \times n}, P \in\{0,1\}^{n \times k}
\end{array}
$$

For the $k$-EP and the GBP, we can derive more simple MISDP formulations by removing $P$ from the model.

In the case of the $k$-EP, the QMP (6.27) is a special case of $\left(Q M P_{1}\right)$, and therefore $k$-EP
can be modeled as follows:

$$
\begin{array}{ll}
\min & \frac{1}{2}\langle L, X\rangle \\
\text { s.t. } & \operatorname{diag}(X)=\mathbf{1}_{n} \\
& X \mathbf{1}_{n}=\frac{n}{k} \mathbf{1}_{n}  \tag{6.29}\\
& k X-\mathbf{J}_{n} \succeq \mathbf{0}, \quad X \in \mathcal{S}^{n}, X \in\{0,1\}^{n \times n} .
\end{array}
$$

This result follows from Theorem 6.24. An alternative proof is provided below.
Proposition 6.32. Let $\mathbf{m}=\frac{n}{k} \mathbf{1}_{k}$. Then, the $Q M P$ (6.27) for the $k-E P$ is equivalent to the BSDP (6.29).

Proof. Let $P$ be feasible for (6.27) where $\mathbf{m}=\frac{n}{k} \mathbf{1}_{k}$. We define $X:=P P^{\top}$. The first and second constraint in (6.29), as well as $X \in\{0,1\}^{n \times n}$ follow by direct verification. Let $p_{i}$ be the $i$ th column of $P$ for $i \in[k]$, then
$k X-\mathbf{J}_{n}=k P P^{\top}-\mathbf{1}_{n} \mathbf{1}_{n}^{\top}=k \sum_{i=1}^{k} p_{i} p_{i}^{\top}-\left(\sum_{i=1}^{k} p_{i}\right)\left(\sum_{i=1}^{k} p_{i}\right)^{\top}=\sum_{i<j}\left(p_{i}-p_{j}\right)\left(p_{i}-p_{j}\right)^{\top} \succeq \mathbf{0}$.
Conversely, let $X$ be feasible for (6.29). Then it follows from Theorem 6.1 and Proposition 6.2 that there exist $x_{i} \in\{0,1\}^{n}, i \in[r], k \geq r$ such that $X=\sum_{i=1}^{r} x_{i} x_{i}^{\top}$ where $\sum_{i=1}^{r} x_{i}=\mathbf{1}_{n}$. Since the constraint $X \mathbf{1}_{n}=\frac{n}{k} \mathbf{1}_{n}$ is invariant under permutation of rows and columns of $X$, we have that the sum of the elements in each row and column of the block matrix $\mathbf{J}_{n_{1}} \oplus \cdots \oplus \mathbf{J}_{n_{r}}$ equals $n / k$. From this it follows that $r=k$ and $\mathbf{1}_{n}^{\top} x_{i}=n / k$ for $i \in[k]$. It is not difficult to verify that $P:=\left[x_{1} \ldots x_{k}\right] \in\{0,1\}^{n \times k}$ is feasible for (6.27). Since the two objectives coincide, the result follows.

Next result shows that the MISDP (6.28) also simplifies for the GBP. It has to be noted, however, that the GBP is not a special case of $\left(Q M P_{1}\right)$.

Proposition 6.33. Let $\mathbf{m}=\left[m_{1} n-m_{1}\right]^{\top}, 1 \leq m_{1} \leq n / 2$. Then, the $Q M P$ (6.27) for the $G B P$ is equivalent to the following BSDP:

$$
\begin{array}{cl}
\min & \frac{1}{2}\langle L, X\rangle \\
\text { s.t. } & \operatorname{diag}(X)=\mathbf{1}_{n}  \tag{6.30}\\
& \left\langle\mathbf{J}_{n}, X\right\rangle=m_{1}^{2}+\left(n-m_{1}\right)^{2} \\
& 2 X-\mathbf{J}_{n} \succeq \mathbf{0}, X \in \mathcal{S}^{n}, X \in\{0,1\}^{n \times n} .
\end{array}
$$

Proof. Let $P$ be feasible for (6.27). We define $X:=P P^{\top}$. The first and second constraint in (6.30) follow by direct verification. Let $p_{i}$ be the $i$ th column of $P$ for $i \in[2]$, then

$$
2 X-\mathbf{J}_{n}=2 P P^{\top}-\mathbf{1}_{n} \mathbf{1}_{n}^{\top}=2 \sum_{i=1}^{2} p_{i} p_{i}^{\top}-\left(\sum_{i=1}^{2} p_{i}\right)\left(\sum_{i=1}^{2} p_{i}\right)^{\top}=\left(p_{1}-p_{2}\right)\left(p_{1}-p_{2}\right)^{\top} \succeq \mathbf{0}
$$

Conversely, let $X$ be feasible for (6.30). Then, it follows from Theorem 6.1 and Proposition 6.2 that there exist $x_{1}, x_{2} \in\{0,1\}^{n}$ such that $X=x_{1} x_{1}^{\top}+x_{2} x_{2}{ }^{\top}$ where $x_{1}+x_{2}=\mathbf{1}_{n}$. Note that $X$ cannot have rank one or zero for $1 \leq m_{1}<n$. From $\left\langle\mathbf{J}_{n}, X\right\rangle=m_{1}^{2}+\left(n-m_{1}\right)^{2}$
it follows that $\mathbf{1}_{n}^{\top} x_{1}=m_{1}$ or $\mathbf{1}_{n}^{\top} x_{1}=n-m_{1}$. Without loss of generality we assume that $\mathbf{1}^{\top} x_{1}=m_{1}$. Clearly, $P:=\left[\begin{array}{ll}x_{1} & x_{2}\end{array}\right] \in\{0,1\}^{n \times 2}$ is feasible for (6.27). Moreover, the two objective functions coincide.

In the remainder of this section, we derive yet another alternative MISDP formulation of the GPP, different from (6.28). For that purpose we notice that the GPP can also be formulated as a QMP of the following form:

$$
\begin{array}{lll}
\text { min } & \operatorname{tr}\left(P^{\top} Q_{0} P\right)+\operatorname{tr}\left(P C_{0} P^{\top}\right)+2 \operatorname{tr}\left(B_{0}^{\top} P\right)+d_{0} & \\
\text { s.t. } & \operatorname{tr}\left(P^{\top} Q_{i} P\right)+\operatorname{tr}\left(P C_{i} P^{\top}\right)+2 \operatorname{tr}\left(B_{i}^{\top} P\right)+d_{i} \leq 0 \quad \forall i \in[m] \quad\left(Q M P_{3}\right) \\
& P \in \mathbb{R}^{n \times k},
\end{array}
$$

where $Q_{i} \in \mathcal{S}^{n}, C_{i} \in \mathcal{S}^{k}, B_{i} \in \mathbb{R}^{n \times k}, d_{i} \in \mathbb{R}$ for $i=0,1, \ldots, m$. Note that $\left(Q M P_{2}\right)$ is a special case of $\left(Q M P_{3}\right)$. Examples of problems that are of this form are quadratic problems with orthogonality constraints, see e.g., [17]. The GPP can be formulated as follows, see e.g., [100]:

$$
\begin{array}{ll}
\min & \frac{1}{2}\left\langle L, P P^{\top}\right\rangle \\
\text { s.t. } & P^{\top} \mathbf{1}_{n}=\mathbf{m} \\
& P^{\top} P=\operatorname{Diag}(\mathbf{m})  \tag{6.31}\\
& \operatorname{diag}\left(P P^{\top}\right)=\mathbf{1}_{n} \\
& P \geq \mathbf{0}, P \in \mathbb{R}^{n \times k} .
\end{array}
$$

To reformulate (6.31) as a MISDP we first introduce a matrix $X_{1} \in \mathcal{S}^{n}$ such that $X_{1}=P P^{\top}$ Now, we relax $X_{1}=P P^{\top}$ and $\operatorname{Diag}(\mathbf{m})=P^{\top} P$ to the following linear matrix inequalities (LMIs):

$$
X_{1}-P P^{\top} \succeq \mathbf{0}, \quad \text { or equivalently, } \quad\left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X_{1}
\end{array}\right) \succeq \mathbf{0}
$$

and

$$
\operatorname{Diag}(\mathbf{m})-P^{\top} P \succeq \mathbf{0}, \quad \text { or equivalently, } \quad\left(\begin{array}{cc}
\mathbf{I}_{n} & P \\
P^{\top} & \operatorname{Diag}(\mathbf{m})
\end{array}\right) \succeq \mathbf{0} .
$$

After introducing the constraint $\operatorname{diag}\left(X_{1}\right)=\mathbf{1}_{n}$, we obtain the following MISDP:

$$
\begin{array}{ll}
\min & \frac{1}{2}\left\langle L, X_{1}\right\rangle \\
\text { s.t. } & P \mathbf{1}_{k}=\mathbf{1}_{n} \\
& \operatorname{diag}\left(X_{1}\right)=\mathbf{1}_{n} \\
& \left(\begin{array}{cc}
\mathbf{I}_{k} & P^{\top} \\
P & X_{1}
\end{array}\right) \succeq \mathbf{0},\left(\begin{array}{cc}
\mathbf{I}_{n} & P \\
P^{\top} & \operatorname{Diag}(\mathbf{m})
\end{array}\right) \succeq \mathbf{0}  \tag{6.32}\\
& X_{1} \in \mathcal{S}^{n}, P \in\{0,1\}^{n \times k} .
\end{array}
$$

We prove below that (6.32) is an exact formulation of the GPP.

Proposition 6.34. The MISDP (6.32) is an exact formulation of the GPP.
Proof. We prove the result by showing the equivalence between (6.31) and (6.32).
Let $P \in \mathbb{R}^{n \times k}$ be feasible for (6.31). Then, it follows from $\operatorname{diag}\left(P P^{\top}\right)=\mathbf{1}_{n}$ that we have $\left(P P^{\top}\right)_{i i}=\sum_{j=1}^{k} P_{i j}^{2}=1$ for $i \in[n]$. From this and $P \geq \mathbf{0}$, we obtain $0 \leq P_{i j} \leq 1$ for all $i \in[n], j \in[k]$. From $P^{\top} \mathbf{1}_{n}=\mathbf{m}$ it follows $\sum_{i, j} P_{i j}=n$ and from $P^{\top} P=\operatorname{Diag}(\mathbf{m})$ that $\operatorname{tr}\left(P^{\top} P\right)=n$, and thus $\sum_{i, j} P_{i j}^{2}=n$. Therefore, $P_{i j} \in\{0,1\}$ for all $i \in[n], j \in[k]$. The equality $\operatorname{diag}\left(P P^{\top}\right)=\mathbf{1}_{n}$ then implies that $P \mathbf{1}_{k}=\mathbf{1}_{n}$. It follows from the discussion prior to the proposition that $P$ and $X_{1}:=P P^{\top}$ are feasible for (6.32).

Conversely, let $X_{1}$ and $P$ be feasible for (6.32). From $P \in\{0,1\}^{n \times k}$ and $P \mathbf{1}_{k}=\mathbf{1}_{n}$ it follows that $\operatorname{diag}\left(P P^{\top}\right)=\mathbf{1}_{n}$. From $\operatorname{Diag}(\mathbf{m})-P^{\top} P \succeq \mathbf{0}$ and $\mathbf{1}_{k}^{\top}\left(\operatorname{Diag}(\mathbf{m})-P^{\top} P\right) \mathbf{1}_{k}=0$ it follows that $\left(\operatorname{Diag}(\mathbf{m})-P^{\top} P\right) \mathbf{1}_{k}=\mathbf{0}$ and thus $P^{\top} \mathbf{1}_{n}=\mathbf{m}$. Moreover, we have $\left(P^{\top} P\right)_{i i}=$ $\sum_{j=1}^{n} P_{j i}^{2}=\sum_{j=1}^{n} P_{j i}=m_{i}$ for $i \in[k]$, implying that $\operatorname{diag}\left(P^{\top} P\right)=\mathbf{m}$. Finally, we have $\operatorname{Diag}(\mathbf{m})-P^{\top} P \succeq \mathbf{0}$, where it follows from above that the latter matrix has a diagonal of zeros. Thus, we must have $P^{\top} P=\operatorname{Diag}(\mathbf{m})$, which concludes the proof.

The MISDP (6.32) has two LMIs and requires integrality constraints only on a matrix of size $n \times k$, while (6.28) has only one LMI and asks for integrality on a matrix of size of $n \times n$ and a matrix of size $n \times k$.

### 6.4.3 MISDP formulations based on algebraic connectivity

In this section we show how the concept of algebraic connectivity of graphs can be exploited to obtain compact MISDP formulations of several graph optimization problems.

Let $G=(V, E)$ be an undirected graph and define $L:=\operatorname{diag}\left(X \mathbf{1}_{n}\right)-X$ as its (unweighted) Laplacian matrix. Fiedler [132] defined the algebraic connectivity of a graph as the second smallest eigenvalue of $L$, i.e., $a(G):=\lambda_{2}(L) \geq 0$. Fiedler [132] showed that $a(G)>0$ if and only if $G$ is connected.

A lower bound on the algebraic connectivity of a graph can be established by the use of a linear matrix inequality, as stated by the following result.

Proposition 6.35 ([273]). Let $G$ be a simple graph on $n \geq 3$ vertices and let $L$ be the Laplacian matrix of $G$. Then $\lambda_{2}(L) \geq \beta$ if and only if $L+\frac{\beta}{n} \mathbf{J}_{n}-\beta \mathbf{I}_{n} \succeq \mathbf{0}$.

Proof. The eigenvalues of $L+\frac{\beta}{n} \mathbf{J}_{n}$ are $\beta$ and $\lambda_{2}(L) \leq \cdots \leq \lambda_{n}(L)$. If $\lambda_{2}(L) \geq \beta$, then all eigenvalues of $L+\frac{\beta}{n} \mathbf{J}_{n}$ are greater or equal than $\beta$ and therefore $L+\frac{\beta}{n} \mathbf{J}_{n}-\beta \mathbf{I}_{n} \succeq \mathbf{0}$. Conversely, if $L+\frac{\beta}{n} \mathbf{J}_{n}-\beta \mathbf{I}_{n} \succeq \mathbf{0}$, then all eigenvalues of $L+\frac{\beta}{n} \mathbf{J}_{n}$ greater or equal to $\beta$ and therefore $\lambda_{2}(L) \geq \beta$.

In some graph optimization problems, algebraic connectivity in combination with Proposition 6.35 can be usefully applied. To model that a graph is connected, we typically require that for any partition $(S, V \backslash S)$ of the vertex set, there must exist an edge crossing the cut. For example, this can be done using the cut-set constraints, see Section 1.4.1. Since there exist exponentially many of these cuts, we need to include a large number of (linear) constraints to model connectivity of the graph. Algebraic connectivity serves as a compact alternative to this approach.

Assume we consider an optimization problem on a graph $G$ where the constraints include connectivity constraints, e.g., cut-set or subtour elimination constraints, see Section 1.4.1.

More precisely, the feasible set of the problem consist of subgraphs of $G$ that are all connected, e.g., Hamiltonian cycles, trees or connected regular subgraphs. Depending on the structure of the problem, let $\hat{\beta}>0$ denote a lower bound on the algebraic connectivity of all feasible subgraphs. In order to model the connectivity constraints, we apply Proposition 6.35 with $\beta=\hat{\beta}$.

Below, we follow this approach for two well-known combinatorial problems and show how this results in a MISDP formulation of the corresponding problems.

Example 6.36 (The traveling salesman problem). Let us reconsider the traveling salesman problem (TSP), see Example 6.31. A tour can be modeled by an adjacency matrix $X \in\{0,1\}^{n \times n}$, where $X_{i j}=1$ if edge $\{i, j\}$ is in the tour and $X_{i j}=0$ otherwise. One easily verifies that in order for $X$ to be feasible for the TSP, $X$ should be an element of the following matrix set:

$$
\bar{\Pi}_{n}(G):=\left\{X \in\{0,1\}^{n \times n} \cap \mathcal{S}^{n}: X \mathbf{1}_{n}=2 \cdot \mathbf{1}_{n}, X_{i j}=0 \text { if }\{i, j\} \notin E\right\},
$$

which is the undirected equivalent of the set $\Pi_{n}(G)$ defined in Section 5.4.1, see also Appendix A.4. Indeed, each $X \in \bar{\Pi}_{n}(G)$ is the adjacency matrix of a vertex-disjoint cycle cover of $G$, i.e., a set of cycles such that each vertex is on exactly one cycle. In order to manage that $X$ corresponds to a single cycle, the induced subgraph should be connected. It is well-known that the algebraic connectivity of the cycle graph $C_{n}$ on $n$ vertices equals $a\left(C_{n}\right)=2-2 \cos \left(\frac{2 \pi}{n}\right)$. Applying Proposition 6.35 with $\beta=2-2 \cos \left(\frac{2 \pi}{n}\right)$ yields the following ISDP formulation of the TSP:

$$
\begin{align*}
\min & \frac{1}{2}\langle D, X\rangle \\
\text { s.t } & X \mathbf{1}_{n}=2 \cdot \mathbf{1}_{n} \\
& X_{i j}=0 \quad \forall\{i, j\} \notin E  \tag{6.33}\\
& 2 \cos \left(\frac{2 \pi}{n}\right) \mathbf{I}_{n}-X+\frac{2}{n}\left(1-\cos \left(\frac{2 \pi}{n}\right)\right) \mathbf{J}_{n} \succeq \mathbf{0} \\
& X \in\{0,1\}^{n \times n}, X \in \mathcal{S}^{n} .
\end{align*}
$$

This compact ISDP formulation of the TSP is introduced by Cvetković et al. [86]. A similar procedure for the quadratric traveling salesman problem has been followed in [276].

Example 6.37 (The quadratic minimum spanning tree problem). Let $G=(V, E)$ be an undirected graph with $n:=|V|$ and $m:=|E|$ and let $Q=\left(q_{e f}\right) \in \mathbb{R}^{m \times m}$ denote a symmetric cost matrix on the set of edges. The quadratic minimum spanning tree problem (QMST) asks for a spanning tree $T$ in $G$ such that the total quadratic costs among the edges used in $T$, i.e., $\sum_{e, f \in T, e \neq f} q_{e f}$, are minimized.

To model the problem, we fix an ordering of $E=\left\{e_{1}, \ldots, e_{m}\right\}$ and let $Y=\left(Y_{e f}\right) \in \mathcal{S}^{m}$ denote a matrix such that $Y_{e f}=1$ if edges $e$ and $f$ are both in $T$ and $Y_{e f}=0$ otherwise. Similarly, let $X=\left(X_{i j}\right) \in \mathcal{S}^{n}$ be such that $X_{i j}=1$ if edge $\{i, j\}$ is in $T$ and $X_{i j}=0$ otherwise. We also define the linear mapping $\mathcal{T}: \mathcal{S}^{n} \rightarrow \mathbb{R}^{m}$, where $\mathcal{T}(X)$ maps $X$ to a column vector containing the entries of $X$ corresponding to $E$ with respect to the fixed ordering of the edge set, i.e., $\mathcal{T}(X)_{i}:=X_{j k}$, where $e_{i}=\{j, k\}$.

Assad and Xu [22] derive the following formulation of the quadratic minimum spanning
tree problem:

$$
\begin{array}{ll}
\min & \frac{1}{2}\langle Q, Y\rangle \\
\text { s.t. } & Y \mathbf{1}_{m}=(n-1) \mathcal{T}(X) \\
& \operatorname{diag}(Y)=\mathcal{T}(X)  \tag{6.34}\\
& \mathbf{0} \leq Y \leq \mathbf{J}_{n} \\
& Y \in \mathcal{S}^{m}, \quad X \in \bar{\Psi}_{n}(G),
\end{array}
$$

where $\bar{\Psi}_{n}(G)$ denotes the set of adjacency matrices of spanning trees in $G$. One can check that the constraints in combination with the binarity of $X$ are sufficient to obtain the coupling between the matrices $Y$ and $X$.

In order for the subgraph induced by $X$ to be a spanning tree, it suffices to require that it has exactly $n-1$ edges and that it is connected. It is well-known that the algebraic connectivity of a tree $T_{n}$ on $n \geq 3$ vertices is always within the bounds $2\left(1-\cos \left(\frac{\pi}{n}\right)\right) \leq a\left(T_{n}\right) \leq 1$, see e.g., [181]. Therefore, we can explicitly model the set $\bar{\Psi}_{n}(G)$ as follows:

$$
\bar{\Psi}_{n}(G):=\left\{X \in\{0,1\}^{n \times n} \cap \mathcal{S}^{n}: \begin{array}{r}
\left\langle X, \mathbf{J}_{n}\right\rangle=2(n-1), X_{i j}=0 \text { if }\{i, j\} \notin E \\
\operatorname{Diag}\left(X \mathbf{1}_{n}\right)-X+\frac{\beta}{n} \mathbf{J}_{n}-\beta \mathbf{I}_{n} \succeq \mathbf{0}
\end{array}\right\},
$$

where $\beta=2\left(1-\cos \left(\frac{\pi}{n}\right)\right)$. This leads to the following ISDP formulation of the QMSTP:

$$
\begin{array}{ll}
\min & \frac{1}{2}\langle Q, Y\rangle \\
\text { s.t. } & Y \mathbf{1}_{m}=(n-1) \mathcal{T}(X) \\
& \operatorname{diag}(Y)=\mathcal{T}(X) \\
& \mathbf{0} \leq Y \leq \mathbf{J}_{n} \\
& \left\langle X, \mathbf{J}_{n}\right\rangle=2(n-1)  \tag{6.35}\\
& X_{i j}=0 \quad \forall\{i, j\} \notin E \\
& \operatorname{Diag}\left(X \mathbf{1}_{n}\right)-X+\frac{\beta}{n} \mathbf{J}_{n}-\beta \mathbf{I}_{n} \succeq \mathbf{0} \\
& Y \in \mathcal{S}^{m}, X \in\{0,1\}^{n \times n}, X \in \mathcal{S}^{n} .
\end{array}
$$

This formulation is derived and analyzed in [273].

### 6.4.4 MISDP formulations beyond binarity

Almost all problem formulations that have been discussed before involve matrix variables with entries in $\{0,1\}$. In this section we consider several problems that allow for semidefinite formulations where (some of) the variables are integers, but not necessarily restricted to $\{0,1\}$.

Example 6.38 (The integer matrix completion problem). A well-known problem in data analysis is the problem of low-rank matrix completion. Suppose a partially observed data matrix is given, i.e., let $\Omega \subseteq[n] \times[m]$ denote the set of observed entries and let $D \in \mathbb{R}^{n \times m}$ denote a given data matrix that has its support on $\Omega$. The goal of the low-rank matrix completion problem is to find a minimum rank matrix $X \in \mathbb{R}^{n \times m}$ such that $X$ coincides
with $D$ on the set $\Omega$. The intuition behind this problem is that the observed entries in $D$ are outcomes of a multiplicative model $X=A B^{\top}$, where $A \in \mathbb{R}^{n \times k}, B \in \mathbb{R}^{m \times k}$ represent exogenous features that explain the dependent matrix variable $X$. As the number of features is preferred to be small for the sake of simplicity, we would like to fit a model with a low value of $k$.

Since minimizing $\operatorname{rank}(X)$ leads to a nonconvex and therefore hard problem, a related but tractable alternative is given by minimizing the nuclear norm of $X$, i.e., $\|X\|_{*}:=\sum_{i=1}^{n} \sigma_{i}(X)$, where $\sigma_{i}$ denotes the $i$ th singular value of $X$. Hence, a near-optimal solution to the low-rank matrix completion problem can be obtained by solving the following program:

$$
\begin{aligned}
\min _{X \in \mathbb{R}^{n \times m}} & \|X\|_{*} \\
\text { s.t. } & X_{i j}=D_{i j} \quad \text { for all }(i, j) \in \Omega
\end{aligned}
$$

As shown by Recht et al. [316], the optimization problem above is equivalent to the following semidefinite programming program

$$
\begin{array}{ll}
\min & \left\langle\mathbf{I}_{n}, Z_{1}\right\rangle+\left\langle\mathbf{I}_{m}, Z_{2}\right\rangle \\
\text { s.t. } & X_{i j}=D_{i j} \text { for all }(i, j) \in \Omega \\
& \left(\begin{array}{cc}
Z_{1} & X \\
X^{\top} & Z_{2}
\end{array}\right) \succeq \mathbf{0}
\end{array}
$$

Now, suppose additionally that the model $X=A B^{\top}$ is such that the values of the dependent variables in $X$ are always integers. For instance, this is the case in the famous Netflix problem [338], where $X_{i j}$ represents the score that movie $i$ receives from user $j$ on an integer scale from 1 to 5 . The underlying model reads $X_{i j}=\sum_{\ell=1}^{k} A_{i \ell} B_{\ell j}$, where $A_{i \ell}$ and $B_{\ell j}$ represent movie- and user-specific values of feature $\ell$, respectively. Since not all movies are watched by all users, only a part of the given scores are available, which are represented in $D$. The goal of the problem is to learn from the data a low-rank model $X=A B^{\top}$ such that the values in $X$ are predictions of the movie-user scores that are missing in $\Omega$. Since the scores in $X$ are restricted to be integer, one can solve the SDP above and round its values to integers afterwards. Alternatively, a more precise approach adds integrality in the problem itself. This yields the following integer matrix completion problem:

$$
\begin{array}{cl}
\min & \left\langle\mathbf{I}_{n}, Z_{1}\right\rangle+\left\langle\mathbf{I}_{m}, Z_{2}\right\rangle \\
\text { s.t. } & X_{i j}=D_{i j} \quad \text { for all }(i, j) \in \Omega \\
& X_{i j} \in S \quad \text { for all }(i, j) \notin \Omega \\
& \left(\begin{array}{cc}
Z_{1} & X \\
X^{\top} & Z_{2}
\end{array}\right) \succeq \mathbf{0} .
\end{array}
$$

where $S \subseteq \mathbb{Z}$ is a discrete set, e.g., $S=\{1, \ldots, 5\}$ in the case of the Netflix problem.

Example 6.39 (The chromatic number of a graph). Let $G=(V, E)$ be a simple graph with $n:=|V|$. The chromatic number of $G$, denoted by $\chi(G)$, is the minimum number of colors needed to color the vertices of $G$ such that adjacent vertices receive distinct colors.

Meurdesoif [279] derived the following formulation for $\chi(G)$ :

$$
\begin{aligned}
\chi(G)=\min & k \\
\text { s.t. } & \operatorname{diag}(X)=(k-1) \mathbf{1}_{n} \\
& X_{i j}=-1 \quad \forall\{i, j\} \in E \\
& X \succeq 0, X_{i j} \in\{-1, k-1\} \quad \forall i, j \in[n] .
\end{aligned}
$$

As the domain of the entries in $X$ depend on the variable $k$, the problem formulation above does not immediately fit in our framework. We can resolve this by introducing an auxiliary variable $Y_{i j}$ for each $\{i, j\} \notin E$, which has entries in $\{-1,0\}$ and is completely entangled with $X_{i j}$. Let $M$ denote a given upper bound on $\chi(G)$, e.g., $M=n$. Now, we consider

$$
\begin{aligned}
\chi(G)=\min & k \\
\text { s.t. } & \operatorname{diag}(X)=(k-1) \mathbf{1}_{n} \\
& X_{i j}=-1 \quad \forall\{i, j\} \in E \\
& X_{i j} \geq M Y_{i j}+k-1 \\
& X_{i j} \geq-1 \\
& \left.X_{i j} \leq M+(M+1) Y_{i j}\right\} \quad \forall\{i, j\} \notin E \\
& X_{i j} \leq k-1 \\
& X \succeq 0, X \in \mathcal{S}^{n}, Y_{i j} \in\{-1,0\} \quad \forall\{i, j\} \notin E .
\end{aligned}
$$

One can verify that the inequalities imply that if $Y_{i j}=-1$, then $X_{i j}=-1$ and if $Y_{i j}=0$, then $X_{i j}=k-1$. Note that the integrality of $X$ follows from the integrality of $Y$.

Example 6.40 (The sparse integer least squares problem). In the integer least squares problem we are given a matrix $M \in \mathbb{R}^{n \times k}$ and a column $b \in \mathbb{R}^{n}$ and we seek the closest point to $b$ in the lattice spanned by the columns of $M$. Del Pia and Zhou [303] consider the related sparse integer least squares problem (SILS), where the aimed vector of integer weights $x$ is required to have entries in $\{0, \pm 1\}$ and the number of nonzero elements in $x$ may not exceed a given positive integer $K$. Thus, the SILS problem can be formulated as

$$
\begin{array}{cl}
\min & \frac{1}{n}\|M x-b\|_{2}^{2}  \tag{6.36}\\
\text { s.t. } & x \in\{0, \pm 1\}^{k},\|x\|_{0} \leq K .
\end{array}
$$

The SILS problem has applications in, among other, multiuser detection and sensor networks, see [303] and the references therein. Now, consider the following ternary SDP:

$$
\begin{array}{ll}
\min & \frac{1}{n}\left\langle\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right),\left(\begin{array}{cc}
b^{\top} b & -b^{\top} M \\
-M^{\top} b & M^{\top} M
\end{array}\right)\right\rangle \\
\text { s.t. } & \operatorname{tr}(X) \leq K, \operatorname{diag}(X)=y_{1}+y_{2}, x=y_{1}-y_{2}, y_{1}, y_{2} \in \mathbb{R}_{+}^{n}  \tag{6.37}\\
& \left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \succeq \mathbf{0}, \quad\left(\begin{array}{cc}
1 & x^{\top} \\
x & X
\end{array}\right) \in\{0, \pm 1\}^{(k+1) \times(k+1)} .
\end{array}
$$

It is easy to verify that if $x$ is a solution to (6.36), then $x, X=x x^{\top}, y_{1}=\max (x, \mathbf{0})$ and $y_{2}=\max (-x, \mathbf{0})$ is feasible for (6.37) with the same objective value. Conversely, let the tuple $\left(x, X, y_{1}, y_{2}\right)$ be feasible for (6.37). Since $x=y_{1}-y_{2}$ and $y_{1}, y_{2} \in \mathbb{R}_{+}^{n}$, it follows that $\operatorname{diag}(X)=y_{1}+y_{2}=|x|$. Hence, $\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right)$ is PSD, has entries in $\{0, \pm 1\}$ and is such that $\operatorname{supp}(\operatorname{diag}(X))=\operatorname{supp}(x)$. It then follows from Proposition 6.19 that $\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right)$ is of the form $\binom{1}{x}\binom{1}{x}^{\top}$, from where the result follows. We finally mention that it is sufficient to impose the integrality constraints only on $x$. Namely, the nonnegative determinants of the $3 \times 3$ principal submatrices of $\left(\begin{array}{cc}1 & x^{\top} \\ x & X\end{array}\right)$ then imply that $X \in\{0, \pm 1\}^{(k+1) \times(k+1)}$.

### 6.5 Bounds by integer semidefinite programming duality

The ISDP formulations derived in Section 6.3 and 6.4 can be used to derive exact solutions for the considered problems using generic MISDP solvers, e.g., branch-and-bound algorithms [156, 270] or branch-and-cut algorithms [234, 276]. In this section we show that we can also exploit the MISDP formulations in order to obtain bounds for the problems that have the potential to be tighter than the SDP relaxations obtained by dropping the integrality constraints. These bounds are based on Lagrangian duality. We show how such bounds can be computed for general binary SDPs and, in particular, for the max-cut problem. The resulting bounds are related to the so-called exact subgraph approach considered in [1, 146, 147, 148].

We first extend the Lagrangian duality theory from integer linear programming to the case of integer semidefinite programming. After that, we derive a projected subgradient algorithm that can be used to compute Lagrangian dual bounds. We finalize the section by a preliminary experimental study of our algorithm on the ISDP formulation of the max-cut problem.

### 6.5.1 Lagrangian duality theory for mixed-integer semidefinite programming

Let $C \in \mathcal{S}^{n}, b \in \mathbb{R}^{m}$ and let $\mathcal{A}: \mathcal{S}^{n} \rightarrow \mathbb{R}^{m}$ be a linear operator defined by $\mathcal{A}(X)_{i}:=\left\langle A_{i}, X\right\rangle$ with $A_{i} \in \mathcal{S}^{n}$ for all $i \in[m]$. We define $\mathcal{A}^{*}: \mathbb{R}^{m} \rightarrow \mathcal{S}^{n}$ to be its adjoint. Moreover, we let $\mathcal{J} \subseteq[n] \times[n]$ be a symmetric index set of the discrete variables in the program. For each $(i, j) \in \mathcal{J}$, the set $B_{i j} \subseteq \mathbb{Z}$ denotes the discrete solution space of the variable indexed by $(i, j)$. Finally, for all $(i, j) \in \mathcal{J}$, let $l_{i j}, u_{i j} \in \mathbb{Z}$ denote the lower and upper bound, respectively, with respect to the set $B_{i j}$. We assume these lower and upper bounds to be finite, implying that the sets $B_{i j}$ are finite. We consider a MISDP in the following general form:

$$
\begin{align*}
z_{M I S D P}:=\min & \langle C, X\rangle \\
\text { s.t. } & \mathcal{A}(X)=b, X \succeq \mathbf{0}  \tag{6.38}\\
& X_{i j} \in B_{i j} \quad \text { for all }(i, j) \in \mathcal{J} .
\end{align*}
$$

By relaxing the integrality constraints, we obtain the continuous SDP relaxation of (6.38):

$$
\begin{align*}
z_{S D P}:=\min & \langle C, X\rangle \\
\text { s.t. } & \mathcal{A}(X)=b, X \succeq \mathbf{0}  \tag{6.39}\\
& l_{i j} \leq X_{i j} \leq u_{i j} \quad \text { for all }(i, j) \in \mathcal{J} .
\end{align*}
$$

The inequality $z_{S D P} \leq z_{M I S D P}$ clearly holds. Throughout this section, we make the following assumption.

Assumption 6.41. The feasible set of (6.38) is bounded.
Assumption 6.41 is natural for the MISDPs that follow from combinatorial optimization problems and that we have derived in Section 6.3 and 6.4. Since all integer variables are bounded by $l_{i j}$ and $u_{i j}$, respectively, Assumption 6.41 furthermore implies that bounds on the continuous variables are enforced by the constraints of (6.38). Observe that the boundedness assumption on (6.38) also implies that a solution to the continuous SDP (6.39) is attained. Namely, if the feasible set of (6.39) would be unbounded, then there exists a ray $R \in \mathcal{S}^{n}$ with $\mathcal{A}(R)=\mathbf{0}$ and $R \succeq \mathbf{0}$. Since all bounds on variables in $\mathcal{J}$ are finite, $R_{i j}=0$ for all $(i, j) \in \mathcal{J}$. Therefore, $R$ would also be a ray of the feasible set of (6.38), contradicting Assumption 6.41.

The Lagrangian dual of (6.38) is obtained by dualizing (a part of) the constraints that are intractable in combination with the integrality constraints. In our setting, it is natural to dualize the constraint $X \succeq \mathbf{0}$. Moreover, we can distinguish between the equalities in $\mathcal{A}(X)=b$ that are tractable with the integrality constraints and the equalities that are not. Consequently, we split $\mathcal{A}(X)=b$ into $\mathcal{A}_{1}(X)=b_{1}$ and $\mathcal{A}_{2}(X)=b_{2}$ where $b_{1} \in \mathbb{R}^{m_{1}}$ and $b_{2} \in \mathbb{R}^{m_{2}}$ with $m_{1}+m_{2}=m$. Here we assume that the equalities $\mathcal{A}_{1}(X)=b_{1}$ are not tractable in combination with the integrality constraints. We define the Lagrangian $\mathcal{L}$ after dualizing the constraints $X \succeq \mathbf{0}$ and $\mathcal{A}_{1}(X)=b_{1}$ as

$$
\mathcal{L}(X, S, \lambda):=\langle C, X\rangle-\langle S, X\rangle+\lambda^{\top}\left(\mathcal{A}_{1}(X)-b_{1}\right),
$$

where $S \succeq \mathbf{0}$ and $\lambda \in \mathbb{R}^{m_{1}}$ are the corresponding Lagrange multipliers. Moreover, we let $P$ denote the mixed-integer set of matrices induced by the remaining constraints, i.e.,

$$
\begin{equation*}
P:=\left\{X \in \mathcal{S}^{n}: \mathcal{A}_{2}(X)=b_{2}, X_{i j} \in B_{i j} \text { for all }(i, j) \in \mathcal{J}\right\} . \tag{6.40}
\end{equation*}
$$

Without loss of generality, we may assume that $P$ is bounded. Namely, if not, we can add to $\mathcal{A}_{2}(X)=b_{2}$ the variable bounds on the continuous variables (which exist due to Assumption 6.41). Now, we define the Lagrangian dual function $g: \mathcal{S}_{+}^{n} \times \mathbb{R}^{m_{1}} \rightarrow \mathbb{R}$ as follows:

$$
\begin{equation*}
g(S, \lambda):=\min \{\mathcal{L}(X, S, \lambda): X \in P\} \tag{6.41}
\end{equation*}
$$

Obviously, for all $S \in \mathcal{S}_{+}^{n}, \lambda \in \mathbb{R}^{m_{1}}$ we have $g(S, \lambda) \leq \mathcal{L}\left(X^{*}, S, \lambda\right) \leq\left\langle C, X^{*}\right\rangle=z_{M I S D P}$, where $X^{*}$ is an optimal solution to (6.38). To obtain the best lower bound for $z_{M I S D P}$, we take the supremum of $g(S, \lambda)$ with respect to the dual variables $S$ and $\lambda$. This leads to the Lagrangian dual problem of (6.38):

$$
\begin{align*}
& z_{L D}:= \sup \quad g(S, \lambda) \\
& \text { s.t. }  \tag{6.42}\\
& S \succeq \mathbf{0}, \lambda \in \mathbb{R}^{m_{1}} .
\end{align*}
$$

The following result follows from construction.
Theorem 6.42 (Weak duality). $z_{L D} \leq z_{M I S D P}$
To increase understanding of the Lagrangian dual, we show that it is possible to ob-
tain $z_{L D}$ as the solution of a continuous semidefinite programming problem. We consider

$$
\begin{array}{cl}
\min & \langle C, X\rangle \\
\text { s.t. } & X \in \operatorname{conv}(P)  \tag{6.43}\\
& \mathcal{A}_{1}(X)=b_{1} \\
& X \succeq \mathbf{0} .
\end{array}
$$

Since the feasible set of (6.43) is contained in the feasible set of (6.39) and the latter one is compact, it follows that an optimal solution to (6.43) is attained. We now show that the problem (6.43) is equivalent to the Lagrangian dual under Assumption 6.41, based on a similar result for MILP by Geoffrion [169].

Theorem 6.43. Let $\hat{z}$ denote the optimal objective value to (6.43), then $z_{L D}=\hat{z}$.

Proof. Since $\mathcal{L}(\cdot, S, \lambda)$ is linear on $\mathcal{S}^{n}$ for all fixed $S \in \mathcal{S}_{+}^{n}$ and $\lambda \in \mathbb{R}^{m_{1}}$, we have

$$
\begin{aligned}
z_{L D} & =\sup _{S \succeq \mathbf{0}, \lambda} \min _{X}\left\{\langle C, X\rangle-\langle S, X\rangle+\lambda^{\top}\left(\mathcal{A}_{1}(X)-b_{1}\right): X \in P\right\} \\
& =\sup _{S \succeq \mathbf{0}, \lambda} \min _{X}\left\{\langle C, X\rangle-\langle S, X\rangle+\lambda^{\top}\left(\mathcal{A}_{1}(X)-b_{1}\right): X \in \operatorname{conv}(P)\right\} .
\end{aligned}
$$

Since $\mathcal{L}(X, \cdot, \cdot)$ is also linear on $\mathcal{S}_{+}^{n} \times \mathbb{R}^{m_{1}}$ for all fixed $X \in \operatorname{conv}(P)$, and $\operatorname{conv}(P)$ is compact, Sion's minimax theorem [340] implies that we are allowed to interchange the order of taking the minimum and the supremum, yielding

$$
\begin{aligned}
z_{L D} & =\min _{X} \sup _{S \succeq 0, \lambda}\left\{\langle C, X\rangle-\langle S, X\rangle+\lambda^{\top}\left(\mathcal{A}_{1}(X)-b_{1}\right): X \in \operatorname{conv}(P)\right\} \\
& =\min _{X}\left\{\langle C, X\rangle: X \in \operatorname{conv}(P), \mathcal{A}_{1}(X)=b_{1}, X \succeq \mathbf{0}\right\} .
\end{aligned}
$$

The final equality follows from the fact that if $\mathcal{A}_{1}(X) \neq b_{1}$ or $X \nsucceq \mathbf{0}$, then the inner supremum is unbounded.

Combining the results from Theorem 6.42 and 6.43 leads to the following sandwich result, that we include for future reference.

Corollary 6.44 (Sandwich theorem). Under Assumption 6.41, $z_{S D P} \leq z_{L D} \leq z_{M I S D P}$.
We briefly describe conditions under which equality holds in the chain of Corollary 6.44. Let $\mathcal{F}_{M I S D P}, \mathcal{F}_{S D P}$ and $\mathcal{F}_{L D}$ denote the feasible sets of (6.38), (6.39) and (6.43), respectively. Moreover, for any convex set $K \subseteq \mathcal{S}^{n}$, its normal cone at $X \in K$ is defined as

$$
\mathcal{N}_{K}(X):=\left\{G \in \mathcal{S}^{n}:\langle G, X\rangle \geq\langle G, Y\rangle \text { for all } Y \in K\right\} .
$$

Theorem 6.45. Let $\chi_{M I S D P}^{*}$ and $\chi_{L D}^{*}$ denote the set of optimizers to (6.38) and (6.43), respectively. Then,
(i) $z_{L D}=z_{M I S D P}$ if and only if $-C \in \mathcal{N}_{\mathcal{F}_{L D}}\left(X^{*}\right)$ for all $X^{*} \in \chi_{M I S D P}^{*}$;
(ii) $z_{S D P}=z_{L D}$ if and only if $-C \in \mathcal{N}_{\mathcal{F}_{S D P}}\left(X^{*}\right)$ for all $X^{*} \in \chi_{L D}^{*}$.

Proof. (i) Let $X^{*} \in \chi_{M I S D P}^{*}$. Since $X^{*} \in \mathcal{F}_{L D}, z_{M I S D P}=z_{L D}$ if and only if $X^{*}$ is also an optimizer to (6.43). The latter holds if and only if $X^{*}$ is such that $\left\langle C, X^{*}\right\rangle \leq\langle C, Y\rangle$ for all $Y \in \mathcal{F}_{L D}$, which is equivalent to $-C \in \mathcal{N}_{\mathcal{F}_{L D}}\left(X^{*}\right)$.
(ii) The proof of the second statement is very similar, replacing $\chi_{M I S D P}^{*}$ by $\chi_{L D}^{*}$ and $\mathcal{F}_{L D}$ by $\mathcal{F}_{S D P}$.

It follows from Theorem 6.45 that a sufficient condition for $z_{L D}=z_{M I S D P}$ is that $\operatorname{conv}\left(P \cap\left\{X \in \mathcal{S}^{n}: \mathcal{A}_{1}(X)=b_{1}, X \succeq \mathbf{0}\right\}\right)=\operatorname{conv}(P) \cap\left\{X \in \mathcal{S}^{n}: \mathcal{A}_{1}(X)=b_{1}, X \succeq \mathbf{0}\right\}$, whereas a sufficient condition for $z_{S D P}=z_{L D}$ is

$$
\operatorname{conv}(P)=\left\{X \in \mathcal{S}^{n}: \mathcal{A}_{2}(X)=b_{2}, l_{i j} \leq X_{i j} \leq u_{i j} \text { for all }(i, j) \in \mathcal{J}\right\}
$$

### 6.5.2 A projected subgradient method for solving the Lagrangian dual

Although the optimum to the Lagrangian dual (6.42) is theoretically the same as the solution to (6.43), an explicit description of $\operatorname{conv}(P)$ is in most cases unavailable. Therefore, we need other techniques to obtain the optimum to the Lagrangian dual problem. The problem (6.42) boils down to solving a nonsmooth convex optimization problem. Although many approaches for these problems have been proposed [289], a natural first candidate to be considered is the classical projected subgradient algorithm.

Since the function $g(S, \lambda)$ is a piecewise-linear concave function, its first-order conditions rely on the use of subgradients. A subgradient of $g(\cdot, \cdot)$ at a point $\left(S^{*}, \lambda^{*}\right) \in \mathcal{S}_{+}^{n} \times \mathbb{R}^{m_{1}}$ is a pair $(\Gamma, \gamma) \in \mathcal{S}^{n} \times \mathbb{R}^{m_{1}}$ such that

$$
\begin{equation*}
g(S, \lambda) \leq g\left(S^{*}, \lambda^{*}\right)+\left\langle\Gamma, S-S^{*}\right\rangle+\gamma^{\top}\left(\lambda-\lambda^{*}\right) \quad \text { for all } S, \lambda \in \mathcal{S}_{+}^{n} \times \mathbb{R}^{m_{1}} \tag{6.44}
\end{equation*}
$$

Now, let $X^{*}:=\arg \min \left\{\mathcal{L}\left(X, S^{*}, \lambda^{*}\right): X \in P\right\}$. Then,

$$
\begin{aligned}
g(S, \lambda)= & \min _{X \in P}\left\{\langle C, X\rangle-\langle S, X\rangle+\lambda^{\top}\left(\mathcal{A}_{1}(X)-b_{1}\right)\right\} \\
\leq & \left\langle C, X^{*}\right\rangle-\left\langle S, X^{*}\right\rangle+\lambda^{\top}\left(\mathcal{A}_{1}\left(X^{*}\right)-b_{1}\right) \\
= & \left\langle C, X^{*}\right\rangle-\left\langle S^{*}, X^{*}\right\rangle+\left(\lambda^{*}\right)^{\top}\left(\mathcal{A}_{1}\left(X^{*}\right)-b_{1}\right) \\
& +\left\langle-X^{*}, S-S^{*}\right\rangle+\left(\mathcal{A}_{1}\left(X^{*}\right)-b_{1}\right)^{\top}\left(\lambda-\lambda^{*}\right) \\
= & g\left(S^{*}, \lambda^{*}\right)+\left\langle-X^{*}, S-S^{*}\right\rangle+\left(\mathcal{A}_{1}\left(X^{*}\right)-b_{1}\right)^{\top}\left(\lambda-\lambda^{*}\right) .
\end{aligned}
$$

This shows that $(\Gamma, \gamma)=\left(-X^{*}, \mathcal{A}_{1}\left(X^{*}\right)-b_{1}\right)$ is a subgradient of $g(\cdot, \cdot)$ at $\left(S^{*}, \lambda^{*}\right)$.
Based on this description of the subgradient, we can now solve the Lagrangian dual (6.42) as follows. We start with an initial set of dual multipliers $\left(S^{0}, \lambda^{0}\right) \in \mathcal{S}_{+}^{n} \times \mathbb{R}^{m_{1}}$. Then, we iteratively obtain $g\left(S^{k}, \lambda^{k}\right)$ by minimizing $\mathcal{L}\left(X, S^{k}, \lambda^{k}\right)$ over $X \in P$, yielding an optimal solution $X^{k}$. By the choice of incorporating only the constraints $\mathcal{A}_{2}(X)=b_{2}$ in $P$, we assume that this minimization is tractable. We now compute the subgradient vector $\left(\Gamma^{k}, \gamma^{k}\right):=\left(-X^{k}, \mathcal{A}_{1}\left(X^{k}\right)-b_{1}\right)$. The dual multipliers are updated by a step update in the direction $\left(D^{k}, d^{k}\right) \in \mathcal{S}^{n} \times \mathbb{R}^{m_{1}}$, where this direction is based on the subgradient vector $\left(\Gamma^{k}, \gamma^{k}\right)$. This can be done in several ways, as we discuss below. Then, we set

$$
\begin{equation*}
S^{k+1} \leftarrow \mathcal{P}_{\mathcal{S}_{+}^{n}}\left(S^{k}+\alpha^{k} D^{k}\right) \quad \text { and } \quad \lambda^{k+1} \leftarrow \lambda^{k-1}+\beta^{k} d^{k} \tag{6.45}
\end{equation*}
$$

where $\alpha^{k}$ and $\beta^{k}$ are appropriate stepsize parameters. The resulting dual matrix $S^{k}+\alpha^{k} D^{k}$ is projected onto the PSD cone in order to stay dual feasible. A pseudo-code of the projected subgradient algorithm is given in Algorithm 6.1.

```
Algorithm 6.1 Projected subgradient algorithm for solving (6.42)
Input: \(C, \mathcal{A}_{1}, \mathcal{A}_{2}, b_{1}, b_{2}, B_{i j}\) for all \((i, j) \in \mathcal{J}\)
    Initialize dual pair \(\left(S^{0}, \lambda^{0}\right) \in \mathcal{S}_{+}^{n} \times \mathbb{R}^{m_{1}}\) and set \(k=0, \hat{z}=-\infty\)
    while stopping criteria are not met do
        Compute \(g\left(S^{k}, \lambda^{k}\right)\) and obtain \(X^{k}:=\arg \min \left\{\mathcal{L}\left(X, S^{k}, \lambda^{k}\right): X \in P\right\}\).
        if \(\hat{z}<g\left(S^{k}, \lambda^{k}\right)\) then
            \(\hat{z} \leftarrow g\left(S^{k}, \lambda^{k}\right)\).
        end if
        Update subgradient \(\Gamma^{k}:=-X^{k}, \gamma^{k}=\mathcal{A}_{1}\left(X^{k}\right)-b_{1}\).
        Update stepsize parameters \(\alpha^{k}, \beta^{k}\) and dual updates \(D^{k}, d^{k}\). \(\triangleright\) Dual updates can be based
                                    on pure subgradient, deflected
                                    subgradient or conditional sub-
                                    gradient.
        Update \(S^{k+1} \leftarrow \mathcal{P}_{\mathcal{S}_{+}^{n}}\left(S^{k-1}+\alpha^{k} D^{k}\right), \lambda^{k+1} \leftarrow \lambda^{k-1}+\beta^{k} d^{k}\).
        \(k \leftarrow k+1\)
    end while
Output: \(\hat{z}\)
```

Different strategies for the choice of the dual updates $\left(D^{k}, d^{k}\right)$ and the stepsize parameters $\alpha^{k}$ and $\beta^{k}$ are proposed in the literature for the case of integer linear programming. For an overview of such strategies, we refer the reader to Guta [190]. Below we discuss several different update strategies, which we alter to the case of mixed-integer semidefinite programming. These strategies are compared in the computational experiments in Section 6.5.4.

1. Subgradient: In the (pure) subgradient update step we set $D^{k}=\Gamma^{k}$ and $d^{k}=\gamma^{k}$. For the stepsize parameters, the most commonly used update scheme is due to Polyak [307], which reads

$$
\alpha^{k}:=\frac{\mu_{1}^{k}\left(U B-g\left(S^{k}, \lambda^{k}\right)\right)}{\left\|\Gamma^{k}\right\|_{F}^{2}} \quad \text { and } \quad \beta^{k}:=\frac{\mu_{2}^{k}\left(U B-g\left(S^{k}, \lambda^{k}\right)\right)}{\left\|\gamma^{k}\right\|^{2}}
$$

Here $U B$ is an upper bound on the optimal Lagrangian dual value $z_{L D}$, which can be obtained by a heuristic on the primal problem, and $0<\mu_{1}, \mu_{2} \leq 2$ are stepsize parameters. The justification for this step length is its theoretical convergence when $U B$ is set equal to $z_{L D}$ [307]. A possible choice for $\mu_{1}^{k}$ and $\mu_{2}^{k}$ proposed by Held and Karp [198] is to choose initial parameters $\mu_{1}^{0}, \mu_{2}^{0}$ and halve its value whenever the dual function $g\left(S^{k}, \lambda^{k}\right)$ did not increase for $N_{\text {step }}$ subsequent iterations.
2. Deflected Subgradient: A common issue in the practical use of subgradient methods (for integer linear programming) is the possibility of poor convergence. One typical cause is the fact that two consecutive update directions form an obtuse angle [64, 335], i.e., $\left\langle D^{k}, D^{k+1}\right\rangle<0$ and/or $\left(d^{k}\right)^{\top} d^{k+1}<0$. As a result, the dual multipliers form a zigzagging pattern, which force the next dual multiplier to be very close to the current one. To prevent this behaviour from happening, the deflected subgradient algorithm constructs the dual update vector as a linear combination between the subgradient vector and the previous dual update vector, i.e.,

$$
D^{k}=\Gamma^{k}+\delta_{1}^{k} D^{k-1} \quad \text { and } \quad d^{k}=\gamma^{k}+\delta_{2}^{k} d^{k-1}
$$

where $\delta_{1}^{k}, \delta_{2}^{k} \geq 0$ are deflection parameters. With respect to the choice of these parameters, the following is proposed:

- Based on an approach by Camerini et al. [64], we set

$$
\begin{aligned}
& \delta_{1}^{k}= \begin{cases}-\tau^{k} \frac{\left\langle\Gamma^{k}, D^{k-1}\right\rangle}{\left\|D^{k-1}\right\|_{F}^{2}} & \text { if }\left\langle\Gamma^{k}, D^{k-1}\right\rangle<0, \\
0 & \text { otherwise }\end{cases} \\
& \delta_{2}^{k}= \begin{cases}-\tau^{k} \frac{\left(\gamma^{k}\right)^{\top} d^{k-1}}{\left\|d^{k-1}\right\|^{2}} & \text { if }\left(\gamma^{k}\right)^{\top} d^{k-1}<0 \\
0 & \text { otherwise }\end{cases}
\end{aligned}
$$

Here $\tau^{k} \geq 1$ is a fixed parameter to be determined by preliminary testing.

- Sherali and Ulular [335] propose to take

$$
\delta_{1}^{k}=\frac{\left\|\Gamma^{k}\right\|_{F}}{\left\|D^{k-1}\right\|_{F}} \quad \text { and } \quad \delta_{2}^{k}=\frac{\left\|\gamma^{k}\right\|}{\left\|d^{k-1}\right\|}
$$

in order to let the new dual update vector bisect the angle between the current subgradient $\Gamma^{k}$ (resp. $\gamma^{k}$ ) and the previous dual update $D^{k-1}$ (resp. $d^{k-1}$ ).

The stepsize parameters $\alpha^{k}$ and $\beta^{k}$ in these implementations can be chosen according to the Polyak update [307].
3. Conditional Subgradient: Another practical burden in applying the subgradient algorithm in integer linear programming can be caused by the subgradient vector being almost orthogonal to the face of the dual feasible region, in our case $\mathcal{S}_{+}^{n}$, containing the current point $S^{k}$. In that case the projection of $S^{k-1}+\alpha^{k} D^{k}$ onto $\mathcal{S}_{+}^{n}$ is a point very close to $S^{k}$, leading to a poor convergence. To overcome this, the conditional subgradient algorithm is proposed in the integer linear programming literature, see e.g., [243]. In our conditional subgradient algorithm, we let

$$
D^{k}=\Gamma^{k}-V^{k}
$$

where $V^{k} \in \mathcal{N}_{\mathcal{S}_{+}^{n}}\left(S^{k}\right)$ is a point in the normal cone of $\mathcal{S}_{+}^{n}$ at $S^{k}$. For instance, one can take the orthogonal projection of $\Gamma^{k}$ onto $\mathcal{N}_{\mathcal{S}_{+}^{n}}\left(S^{k}\right)$ as the point $V^{k}$. Since we do not project the update of $\lambda^{k}$, we apply a pure subgradient step for this dual multiplier. For the stepsize parameters, we can again apply the Polyak update [307].
As stopping criteria, we use the subgradient vector $\left(\Gamma^{k}, \gamma^{k}\right)$ having only zero entries and the difference between consecutive dual multipliers to be small, i.e., $\left\|S^{k}-S^{k-1}\right\|_{F}<\varepsilon_{1}$ and $\left\|\lambda^{k}-\lambda^{k-1}\right\|<\varepsilon_{2}$ for some predefined parameters $\varepsilon_{1}, \varepsilon_{2}>0$. Moreover, we implement a stagnation criterion: we stop the algorithm if there has been no improvement for the last $N_{\text {stag }}$ iterations.

### 6.5.3 Lagrangian dual bounds for binary SDPs and the max-cut problem

In this section we apply Algorithm 6.1 to two particular cases: general binary SDPs and the max-cut problem. In these implementations, we particularly focus on the construction of $P$, see (6.40). On the one hand, the set $P$ should be chosen such that $\operatorname{conv}(P)$ is a proper subset of $\left\{X: \mathcal{A}_{2}(X)=b_{2}, l_{i j} \leq X_{i j} \leq u_{i j}\right.$ for all $\left.(i, j) \in \mathcal{J}\right\}$ in order for the Lagrangian
dual to be stronger than its continuous SDP relaxation, while its structure should be simple enough to optimize over it.

### 6.5.3.1 Binary semidefinite programs

Consider the problem (6.38) with $B_{i j}=\{0,1\}$ for all $(i, j) \in[n] \times[n]$. Without prior knowledge about the problem, it is difficult to induce a splitting of $\mathcal{A}(X)=b$ based on its complexity. When dualizing all constraints $\mathcal{A}(X)=b$, the set $P$ becomes $\{0,1\}^{n \times n}$. Although optimizing $\mathcal{L}(X, S, \lambda)$ over this set is tractable, we have $\operatorname{conv}(P)=[0,1]^{n \times n}$. Therefore, the program (6.43) is equivalent to the continuous SDP relaxation (6.39), leading to $z_{L D}=z_{S D P}$.

In order to obtain bounds that are stronger than (6.39), one would like to tighten the set $P$. The only generic option is to exploit the fact that $X \in \mathcal{S}_{+}^{n}$ is a constraint in the original problem. Taking $P=\mathcal{S}_{+}^{n} \cap\{0,1\}^{n \times n}=\mathcal{D}_{n}^{n}$ boils down to solving a binary SDP in the subproblem itself, and is therefore practically infeasible. However, it is inexpensive to optimize over $\mathcal{D}_{r}^{r}$ for a small value of $r$, e.g., by a complete enumeration, see Section 6.2.1. Therefore, instead of defining $P$ to be the full set of $n \times n \operatorname{PSD}\{0,1\}$-matrices, we can impose a condition on certain submatrices to be an $r \times r \operatorname{PSD}\{0,1\}$-matrix. To that end, assume that $\mathcal{K}=\left\{K_{1}, \ldots, K_{N}\right\}$ denotes a packing on the set $[n]$ and define $r_{\ell}:=\left|K_{\ell}\right|$ for all $K_{\ell} \in \mathcal{K}$. We restrict $r_{\ell} \leq p$ for all $K_{\ell} \in \mathcal{K}$ for some given positive integer $p$. Now, we consider the following equivalent binary SDP:

$$
\begin{array}{cl}
\min & \langle C, X\rangle \\
\text { s.t. } & \mathcal{A}(X)=b \\
& X\left[K_{\ell}\right] \in \mathcal{D}_{r_{\ell}}^{r_{\ell}} \quad \forall K_{\ell} \in \mathcal{K}  \tag{6.46}\\
& X \succeq \mathbf{0}, X \in\{0,1\}^{n \times n}
\end{array}
$$

After dualizing $X \succeq \mathbf{0}$ and $\mathcal{A}(X)=b$, we obtain the following feasible set of remaining constraints, which we denote by $P(\mathcal{K})$ :

$$
\begin{equation*}
P(\mathcal{K}):=\left\{X \in \mathcal{S}^{n} \cap\{0,1\}^{n \times n}: X\left[K_{\ell}\right] \in \mathcal{D}_{r_{\ell}}^{r_{\ell}}, \text { for all } K_{\ell} \in \mathcal{K}\right\} \tag{6.47}
\end{equation*}
$$

Let us now check whether this set $P(\mathcal{K})$ can be effectively applied as the feasible set of the dual function (6.41). Let $X^{*}$ denote the optimal solution of minimizing $\mathcal{L}\left(X, S^{k}, \lambda^{k}\right)$ over $X \in P(\mathcal{K})$. Since the index sets in $\mathcal{K}$ are mutually disjoint, the submatrices $\left\{X\left[K_{\ell}\right]\right.$ : $\left.K_{\ell} \in \mathcal{K}\right\}$ do not intersect. Hence, the optimal submatrices $X^{*}\left[K_{\ell}\right]$ can be obtained via a complete enumeration over $\mathcal{D}_{r_{\ell}}^{r_{\ell}}$ for all $K_{\ell} \in \mathcal{K}$ independently. It was shown in Section 6.2.1 that this set contains $B_{r+1}$ elements, see Theorem 6.10. Moreover, any element $X_{i j}^{*}$ where $i$ and $j$ do not belong to the same set in $\mathcal{K}$ is set to 1 if $\left(C-S^{k}+\mathcal{A}_{1}^{*}\left(\lambda^{k}\right)\right)_{i j}<0$ and to 0 otherwise. We conclude that the optimization over $P(\mathcal{K})$ is feasible.

Next, let us consider the strength of the corresponding bound. It follows from Theorem 6.43 that the optimal Lagrangian dual value of (6.46) equals

$$
\begin{array}{cl}
\min & \langle C, X\rangle \\
\mathrm{s.t.} & \mathcal{A}(X)=b \\
& X\left[K_{\ell}\right] \in \mathcal{P}_{r_{\ell}}^{r_{\ell}} \quad \forall K_{\ell} \in \mathcal{K}  \tag{6.48}\\
& X \succeq \mathbf{0}, \mathbf{0} \leq X \leq \mathbf{J}
\end{array}
$$

where $\mathcal{P}_{r_{\ell}}^{r_{\ell}}=\operatorname{conv}\left(\mathcal{D}_{r_{\ell}}^{r_{\ell}}\right)$, see (6.4). Semidefinite relaxations including constraints of the form $X\left[K_{\ell}\right] \in \operatorname{conv}\left(\mathcal{D}_{r_{\ell}}^{r}\right)$ are studied by Adams et al. [1] and Gaar and Rendl [146, 147], in which these are referred to as exact subgraph constraints (ESC). The latter works have shown the quality of including ESCs in SDP relaxations of the max-cut, stable set and coloring problem. To the best of our knowledge, we are the first to realize that such bounds are in fact Lagrangian dual bounds resulting from an appropriate binary SDP. This observation leads to new computational perspectives by solving it as an ISDP, e.g., using the projected subgradient algorithm introduced in Section 6.5.2. The relaxations in [146, 147, 148] are solved by interior-point methods and bundle approaches and overall require lots of computation time.

The value of $p$, i.e., the maximum size of the sets in $\mathcal{K}$, has an impact on the quality of (6.48). If $p$ increases, the proportion of entries that is required to be binary and positive semidefinite in the subproblems becomes larger, leading to improved bounds. However, this comes at the cost of computation time, as the size of $\mathcal{D}_{r_{\ell}}^{r_{\ell}}$ grows exponentially at the rate of the Bell number. For a given value of $p$, we denote the Lagrangian dual bound obtained by the procedure above as a level- $p$ Lagrangian dual bound.

Another ingredient that has a significant impact on the Lagrangian dual bound is the choice of the packing $\mathcal{K}$. As a measure of the strength of set $K_{\ell}$ in $\mathcal{K}$, we can consider the violation of the constraint $X\left[K_{\ell}\right] \in \mathcal{P}_{r_{\ell}}^{r_{\ell}}$ when this constraint is left out of (6.48). This would, however, ask for a total description of the facets of $\mathcal{P}_{r_{\ell}}^{r_{\ell}}$. A relaxation of this measure can be obtained by considering the following description of $\mathcal{D}_{r_{\ell}}^{r}$, following from Corollary 6.3:

$$
\mathcal{D}_{r_{\ell}}^{r_{\ell}}=\left\{X \in \mathcal{S}^{r_{\ell}} \cap\{0,1\}^{r_{\ell} \times r_{\ell}}: \quad \begin{array}{r}
X_{i j} \leq X_{i i} \quad \forall i \neq j  \tag{6.49}\\
X_{i j}+X_{i k} \leq X_{i i}+X_{j k} \quad \forall j<k, i \neq j, k
\end{array}\right\} .
$$

The polytope induced by these inequalities is a relaxation of $\mathcal{P}_{r_{\ell}}^{r_{\ell}}$ and contains $O\left(r_{\ell}^{3}\right)$ facets. Now, let us solve the continuous SDP relaxation (6.48) with $\mathcal{K}=\emptyset$ and let $X^{*}$ denote its optimal solution. Then, we introduce the following violation measure for each tuple of distinct elements in $[n]$ :

$$
\begin{equation*}
c(i, j, k):=\left(X_{i j}^{*}-X_{i i}^{*}\right)^{+}+\left(X_{i j}^{*}-X_{j j}^{*}\right)^{+}+\cdots+\left(X_{i j}^{*}+X_{i k}^{*}-X_{i i}^{*}-X_{j k}^{*}\right)^{+}, \tag{6.50}
\end{equation*}
$$

where $(\cdot)^{+}:=\max (\cdot, 0)$. Hence, we measure the total violation of the nine inequalities in (6.49) with $K_{\ell}=\{i, j, k\}$. Moreover, for each subset $K \subseteq[n],|K| \geq 3$, we define the total violation measure as

$$
\begin{equation*}
c(K):=\sum_{\substack{i, j, k \in K \\|\{i, j, k\}|=3}} c(i, j, k) . \tag{6.51}
\end{equation*}
$$

If $c(K)$ is large, the addition of $K$ to $\mathcal{K}$ is likely to contribute to the strength of the resulting bound. Since the sets in $\mathcal{K}$ must be mutually disjoint, we cannot just enumerate over all sets $K$ of size at most $p$ and take the most violated ones. Instead, we apply a greedy heuristic for this selection. The idea behind this heuristic is to first enumerate all subsets of size 3 and take the subset $K$ that maximizes $c(K)$. If $p>3$, we iteratively check whether there exist an index $i \notin K$ such that $c(K \cup\{i\})-c(K)>0$. If so, we add $i$ to $K$ and repeat this procedure until $|K|=p$ or until no such index can be found. We add $K$ to $\mathcal{K}$ and repeat the entire procedure on the remaining indices.

### 6.5.3.2 The max-cut problem

We now apply the procedure followed in the previous section to the max-cut problem. Given an undirected graph $G=(V, E)$ with $n:=|V|$ and a weight matrix $W:=\left(w_{i j}\right) \in \mathcal{S}^{n}$ with $w_{i j}=0$ if $\{i, j\} \notin E$, the aim is to cut $V$ into two subsets such that the total weight of the edges in the cut is maximized. Let $L:=\operatorname{Diag}\left(W \mathbf{1}_{n}\right)-W$ denote the weighted Laplacian of $G$. Then, the max-cut problem can be modeled as the following ISDP on $\{ \pm 1\}$-matrices:

$$
\begin{array}{ll}
\max & \frac{1}{4}\langle L, X\rangle \\
\text { s.t. } & \operatorname{diag}(X)=\mathbf{1}_{n}  \tag{6.52}\\
& X \succeq \mathbf{0}, X \in\{ \pm 1\}^{n \times n}
\end{array}
$$

After relaxing the integrality constraints in (6.52), the resulting problem is equivalent to the basic SDP relaxation for the max-cut problem, see [95, 306]. Lagrangian duality can again be used to obtain strengthened bounds for (6.52). Similar to the approach followed in Section 6.5.3.1, we consider the following ISDP that is equivalent to (6.52):

$$
\begin{array}{ll}
\max & \frac{1}{4}\langle L, X\rangle \\
\text { s.t. } & \operatorname{diag}(X)=\mathbf{1}_{n}  \tag{6.53}\\
& X\left[K_{\ell}\right] \in \widehat{\mathcal{D}}^{r_{\ell}} \quad \forall K_{\ell} \in \mathcal{K} \\
& X \succeq \mathbf{0}, X \in\{ \pm 1\}^{n \times n}
\end{array}
$$

where $\mathcal{K}$ is a collection of subsets of $[n]$, each of size at most $p$. Observe that we now require the subgraphs to be in the discrete set of PSD $\{ \pm 1\}$-matrices, see (6.13).

We construct the Lagrangian dual by dualizing the constraints $X \succeq \mathbf{0}$. The subproblem can be solved efficiently for moderate sizes of $r$. With respect to the structure of $\mathcal{K}$, there is a subtle difference compared to the case of general binary SDPs. Since we additionally know that $\operatorname{diag}(X)=\mathbf{1}_{n}$, we always fix the diagonal entries of $X$ to one in the subproblem. Therefore, the exact submatrix constraint $X\left[K_{\ell}\right]$ for some set $K_{\ell}$ only concerns the offdiagonal elements of $X\left[K_{\ell}\right]$. Consequently, the submatrix constraints for two sets $K_{1}$ and $K_{2}$ that intersect on a single index, i.e., $\left|K_{1} \cap K_{2}\right|=1$, can be evaluated independently. Thus, we no longer require $\mathcal{K}$ to consist of mutually disjoint sets, it is sufficient to require that the elements in $\mathcal{K}$ do not pairwise intersect in more than one index. We call a collection of subsets satisfying this property an edge-packing of [ $n$ ].

Due to the different structure of $\mathcal{K}$, we finalize this section by explaining how we can greedily construct an edge-packing $\mathcal{K}$ that is likely to provide a strong Lagrangian dual bound. In order to measure the violation of a certain submatrix constraint, let us consider the following description of the set $\widehat{\mathcal{D}}^{r_{\ell}}$ :

$$
\widehat{\mathcal{D}}^{r_{\ell}}=\left\{X \in \mathcal{S}^{r_{\ell}} \cap\{ \pm 1\}^{r_{\ell} \times r_{\ell}}: \begin{array}{rl}
X_{i i} & =1  \tag{6.54}\\
\quad X_{i j}-X_{i k}-X_{j k} \geq-1 & \forall i \\
& \forall i<j, k \neq i, j \\
X_{i j}+X_{i k}+X_{j k} \geq-1 & \forall i<j<k
\end{array}\right\}
$$

It is easy to check that the inequalities in (6.54) for $X \in \mathcal{S}^{r_{\ell}} \cap\{ \pm 1\}^{r_{\ell} \times r_{\ell}}$ imply that $X=x x^{\top}$ for some $x \in\{ \pm 1\}^{r}$. In fact, these inequalities, known as the triangle inequalities, are wellstudied, see e.g., [318], and are related to the metric polytope [244].

Let $X^{*}$ denote an optimal solution of the SDP relaxation of the max-cut problem, i.e., (6.52) after dropping the integrality constraints. For each tuple of distinct $i, j, k \in[n]$, we now define the following violation measure:

$$
\begin{align*}
c(i, j, k):= & \left(X_{i k}^{*}+X_{j k}^{*}-X_{i j}^{*}-1\right)^{+}+\left(X_{i j}^{*}+X_{j k}^{*}-X_{i k}^{*}-1\right)^{+} \\
& +\left(X_{i j}^{*}+X_{i k}^{*}-X_{j k}^{*}-1\right)^{+}+\left(-X_{i j}^{*}-X_{i k}^{*}-X_{j k}^{*}-1\right)^{+} . \tag{6.55}
\end{align*}
$$

Moreover, for each set $K \subseteq[n]$ we define $c(K):=\sum_{i, j, k \in K,|\{i, j, k\}|=3} c(i, j, k)$. Now, similar to the approach described in Section 6.5.3.1, we construct an edge-packing $\mathcal{K}$ in a greedy way. The only difference is that, instead of requiring that an index $i$ may not be included in more than one set of $\mathcal{K}$, we require that a pair of distinct indices $\{i, j\}$ may not be included in more than one set.

### 6.5.4 Preliminary computational results for the max-cut problem

In this section we provide a preliminary computational study on the strength of the Lagrangian dual bounds obtained from the ISDP formulation of the max-cut problem discussed in Section 6.5.3.2.

### 6.5.4.1 Design of computational experiments

In Section 6.5.2 we introduced several variants of the classical projected subgradient algorithm to obtain the Lagrangian dual for MISDPs. We compare these Lagrangian dual bounds with two semidefinite programming bounds for the max-cut problem. This leads to the following approaches:

- SDP: This refers to solving the basic SDP relaxation of the max-cut problem, i.e., (6.52) after dropping the integrality constraints.
- LD: This refers to the Lagrangian dual approach presented in Section 6.5.3.2, i.e., the Lagrangian dual of (6.53) after dualizing $X \succeq \mathbf{0}$. The edge-packing $\mathcal{K}$ that is used in the computations is derived after obtaining a solution $X^{*}$ to the basic SDP relaxation, constructing the violation values $c(i, j, k)$, see (6.55), and applying the greedy heuristic described at the end of Section 6.5.3.2. Lagrangian dual bounds for $p \in\{3, \ldots, 10\}$ are computed. To obtain these bounds, we consider the following implementations of our projected subgradient algorithm, see Section 6.5.2:
- SG: The standard subgradient algorithm using Polyak's step update [307]. We take $\mu_{1}^{0}=1$ and $N_{\text {step }}=80$. Observe that $\mu_{2}^{0}$ is not used in the method, as we only dualize the PSD constraint.
- DSG1: The deflected subgradient algorithm using the update strategy from Camerini et al. [64]. Here, we take $\tau^{k}=1.99$ for all $k$ and use Polyak's stepsize with the same parameters as mentioned above.
- DSG2: The deflected subgradient algorithm using the update scheme from Sherali and Ulular [335], again using the same stepsize update parameters.
- CSG: The conditional subgradient algorithm. It is well-known that the normal cone $\mathcal{N}_{\mathcal{S}_{+}^{n}}\left(S^{k}\right)$ is the set of negative semidefinite matrices $V$ with $S^{k} V=\mathbf{0}$. In each iteration, we construct $V^{k}=\lambda_{1} u_{1} u_{1}^{\top}+\cdots+\lambda_{k} u_{k} u_{k}^{\top}$, where $\left\{u_{1}, \ldots, u_{k}\right\}$ denotes an orthonormal basis for $\operatorname{Nul}\left(S^{k}\right)$ and the weights $\lambda_{1}, \ldots, \lambda_{k}$ are chosen uniformly
at random from the interval $[-1,0]$. After this construction, we normalize $V^{k}$. The stepsize update parameters are the same as in the other settings.

For the stopping criteria of the projected subgradient algorithm, we use $\varepsilon_{1}=10^{-4}$ and $N_{\text {stag }}=150$.

- SDP + Cuts: We know that the optimal Lagrangian dual bound is equivalent to the basic SDP relaxation including the constraints $X\left[K_{\ell}\right] \in \operatorname{conv}\left(\widehat{\mathcal{D}}^{r_{\ell}}\right), K_{\ell} \in \mathcal{K}$, see Theorem 6.43. Since the polytope enclosed by the triangle inequalities in (6.54) forms a relaxation of $\operatorname{conv}\left(\widehat{\mathcal{D}}^{r_{\ell}}\right)$, it is natural to compare our approach to solving the following SDP relaxation:

$$
\begin{align*}
\max & \frac{1}{4}\langle L, X\rangle \\
\text { s.t. } & \operatorname{diag}(X)=\mathbf{1}_{n} \\
& X_{i j}-X_{i k}-X_{j k} \geq-1 \quad \forall i, j, k \in K_{\ell}, i<j, k \neq i, j, K_{\ell} \in \mathcal{K}  \tag{6.56}\\
& X_{i j}+X_{i k}+X_{j k} \geq-1 \quad \forall i, j, k \in K_{\ell}, i<j<k, K_{\ell} \in \mathcal{K} \\
& X \succeq \mathbf{0} .
\end{align*}
$$

The values of the parameters are derived based on preliminary testing.
We compare our approaches on instances from the Biqmac Library [364], which contain a large number of max-cut benchmark instances. In our computational experiments, we include the following instance classes:

- Rudy instances: These instances are randomly generated using the graph generator rudy [322]. The instances 'g05_n' are unweighted graphs on $n$ vertices, where each edge is included with probability $\frac{1}{2}$. The instance classes 'pm1d_n' and 'pm1s_n' contain graphs on $n$ vertices with edge densities 0.9 and 0.1 , respectively, having edge weights chosen uniformly at random from $\{0, \pm 1\}$. The graphs 'pw $d \_n$ ' are defined on $n$ vertices with edge density $d \in\{0.1,0.5,0.9\}$, where the weights are integers from $\{0, \ldots, 10\}$ chosen uniformly at random. Finally, the class ' $w d \_n$ ' is defined similarly, except for the weights being chosen as integers from $\{-10, \ldots, 10\}$. Each instance class consists of 10 randomly generated instances, resulting in a test set of 130 instances. Optimal values of these max-cut instances are given in [364].
- Beasley instances: These instances are introduced in [36]. The graphs 'beasley_n' contain $n$ vertices, have edge density 0.1 and integer edge weights chosen uniformly at random from $\{-100, \ldots, 100\}$. Originally, these instances correspond to unconstrained binary quadratic minimization problems. However, we view them directly as max-cut instances, meaning that we view each data matrix directly as a weighted adjacency matrix of a graph. Consequently, we cannot compare our results to the optima given in [364]. Each instance class consists of 10 randomly generated instances, leading to a total of 30 instances.

We implemented the projected subgradient algorithms in Matlab [221]. The SDP bounds on graphs with $n \leq 100$ are computed via the interior-point solver of Mosek [284] in the default settings, where Yalmip [257] is used to define the models. For the larger instances with $n=250$, we exploit a simple implementation of the Alternating Direction Method of Multipliers (ADMM) tailor-made for the max-cut problem. Experiments are run on an $\operatorname{Intel}(\mathrm{R}) \operatorname{Core}(\mathrm{TM}) \mathrm{i} 5-8500 \mathrm{CPU}, 3.00 \mathrm{GHz}$ with 8 GB memory.

### 6.5.4.2 Discussion of computational results

The results on the rudy instances are displayed in Table 6.1, where the Lagrangian dual bounds are computed for $p=5$. Each row corresponds to the average results over 10 randomly generated instances of a certain class. The second and third column correspond to the average bound values of the semidefinite programming bound ('SDP') and the Lagrangian dual value ('LD'), respectively. Since all implementations of the subgradient algorithms converge to the same bound value, only the best found value is presented in the column 'LD'. Moreover, the value of the semidefinite program including cuts implied by $\mathcal{K}$, i.e., (6.56), is often very close or equal to (but not better than) the Lagrangian dual value, and is therefore omitted from the table. The column 'Imp (\%)' shows the relative improvement of the level-5 Lagrangian dual bound compared to the SDP bound, while 'Rel gap closed (\%)' denotes the average percentage of the original gap between the SDP bound and the optimum that is closed by the Lagrangian dual. The column 'OPT' corresponds to the average optimal values reported in the Biqmac Library [364]. The remaining columns correspond to the total clocktime needed to obtain the semidefinite programming and Lagrangian dual bounds. Among the different implementations of the projected subgradient algorithm, the most efficient one per instance class is presented in bold.

We observe from Table 6.1 that the relative improvement of the LD values compared to the SDP value is on average between $0.5 \%$ and $7.5 \%$. The largest improvements correspond to the classes 'pm1d_n', 'pm1s_n' and 'wd_n', which are exactly the classes where negative edge weights are included. Moreover, considering the evolvement of the improvements over $n$ for a fixed instance class, it can be seen that the improvements diminish when the number of vertices increases. This can be explained by the lower percentage of entries that is included in a submatrix constraint. The average percentage of the gap between the SDP bound and the optimum closed by the Lagrangian dual bound is considerable, and ranges from $41.67 \%$ to $55.71 \%$.

As one would expect, the simple pure subgradient method often takes the longest time to converge, shortly thereafter followed by the conditional subgradient method. The deflected methods are significantly faster, where the approach DSG2 provides the most efficient implementation for all the instance classes. The method is often a couple of seconds faster than the alternative approaches. Moreover, although the semidefinite programming bound (6.56) provides bounds that are very close to the Lagrangian dual value, the computation times are clearly outperformed by our best subgradient algorithm. This might not be very surprising, as interior-point methods have difficulties with the handling of a large number of polyhedral cutting planes, see e.g., [124]. Since the Lagrangian dual bound is equal to the optimal value of (6.53) and thus theoretically stronger than (6.56), the deflected subgradient algorithm DSG2 is both preferred in terms of strength and efficiency.

Similar experiments as summarized in Table 6.1 are performed for $p \in\{3, \ldots, 10\}$. Figure 6.2 shows the main characteristics of the analysis with respect to $p$. Figure 6.2a shows the evaluation of the Lagrangian dual bound over $p$ for the 13 instance classes. In order to compare these bounds, we normalized all values with respect to the level-3 Lagrangian dual bound. As one would expect, the bounds become stronger for larger values of $p$, although the improvement is diminishing when $p$ gets larger. Indeed, the largest marginal improvement is obtained for $p=3$.

Figure 6.2 b and 6.2 c show the average computation times of the approach DSG2 and the SDP bound (6.56) with respect to $p$, respectively. Here we observe a somewhat unexpected effect. When $p$ increases, the computation time of the Lagrangian dual reduces. This

| Instance Class | Average bound values |  |  |  |  | Average computation times (s) |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | SDP | LD | Imp <br> (\%) | Rel gap closed (\%) | OPT | SDP | SG | DSG1 | DSG2 | CG | $\begin{gathered} \text { SDP } \\ +\quad \text { Cuts } \end{gathered}$ |
| g05_60 | 545.79 | 538.33 | 1.37 | 55.71 | 532.4 | 1.05 | 6.60 | 5.91 | 4.91 | 6.50 | 6.61 |
| g05_80 | 950.30 | 939.88 | 1.10 | 49.15 | 929.1 | 2.83 | 13.33 | 11.06 | 10.61 | 13.46 | 14.26 |
| g05_100 | 1463.39 | 1449.79 | 0.93 | 42.51 | 1431.4 | 7.45 | 21.17 | 17.06 | 15.42 | 21.29 | 28.22 |
| pm1d_80 | 297.15 | 275.68 | 7.22 | 50.34 | 254.5 | 2.89 | 13.34 | 11.45 | 9.72 | 13.69 | 14.28 |
| pm1d_100 | 441.76 | 414.59 | 6.15 | 42.28 | 377.5 | 7.27 | 22.99 | 19.14 | 16.77 | 23.15 | 28.83 |
| pm1s_80 | 91.20 | 84.53 | 7.31 | 55.12 | 79.1 | 2.86 | 13.54 | 11.28 | 9.76 | 13.75 | 14.67 |
| pm1s_100 | 139.45 | 131.73 | 5.54 | 45.82 | 122.6 | 7.45 | 22.81 | 19.65 | 15.66 | 23.27 | 28.90 |
| pw01_100 | 2143.68 | 2095.65 | 2.24 | 49.07 | 2045.8 | 7.95 | 26.43 | 21.82 | 17.79 | 26.28 | 28.95 |
| pw05_100 | 8364.64 | 8273.99 | 1.08 | 41.67 | 8147.1 | 8.63 | 24.26 | 21.33 | 17.55 | 25.66 | 29.07 |
| pw09_100 | 13765.73 | 13677.68 | 0.64 | 42.78 | 13559.9 | 9.08 | 24.52 | 20.24 | 18.54 | 24.87 | 28.66 |
| w01_100 | 790.63 | 742.20 | 6.13 | 53.38 | 699.9 | 7.45 | 32.51 | 27.37 | 22.79 | 31.54 | 28.80 |
| w05_100 | 1917.26 | 1796.25 | 6.31 | 44.24 | 1643.7 | 8.19 | 25.37 | 20.95 | 18.33 | 25.71 | 28.92 |
| w09_100 | 2544.81 | 2382.73 | 6.37 | 44.05 | 2176.9 | 8.30 | 26.67 | 22.51 | 19.30 | 26.89 | 28.58 |

Table 6.1: Average Lagrangian dual bounds and average computation times of various implementations of the projected subgradient algorithm for the rudy instances of the Biqmac Library [364]. Lagrangian dual bounds are shown for $p=5$. Each instance class consists of 10 randomly generated instances of the given type.
behaviour can be explained after a more thorough analysis into the deflected subgradient algorithm. As $p$ increases, the sets $\widehat{D}^{r_{\ell}}$ grow exponentially, leading to more exhausting enumerations. At the same time, since the elements in $\mathcal{K}$ are overlapping in at most one index, the number of subsets in $\mathcal{K}$ decrease over $p$. Since the enumerations in the subproblem are performed one after another, a larger number of subsets will increase the computation time. Hence, when $p$ increases, two opposite effects are taking place, where apparently the reduction in the number of subsets has a larger effect. When we would hypothetically increase $p$ further, the total computation time should go up at some point, since for $p=n$ we would enumerate all possible cuts in the graph. Figure 6.2 c clearly shows that the complexity of solving (6.56) increases with $p$, due to the larger number of cutting planes.

Finally, we consider the Beasley instances in Table 6.2. The structure of this table is similar to that of Table 6.1, except for the fact that the optimal values for these instances are unknown (recall that we interpret these instances differently than in [364]). Moreover, for each instance class we show the average results for $p \in\{3, \ldots, 6\}$. The improvements with respect to the basic SDP relaxation are between $5 \%$ and $6.5 \%$ for the instances with $n=50,100$. For the larger instances, we see that the improvements are smaller, due to the reasons explained above. Moreover, we again observe that the bound values increase with $p$, while the computation times for the subgradient algorithms in general diminishes.

We observe that DSG2 is in most cases again the favorable subgradient approach with respect to efficiency, although for some instances we observe that DSG1 and CG are also performing very well or even a bit better than DSG2. Especially for larger $n$, the update scheme of Camerini et al. [64] outperforms the one from Sherali and Ulular [335]. For the instances with $n=250$, we see that (6.56) cannot be computed anymore due to memory issues.

(a) Overview of average relative improvements of Lagrangian dual over $p$.

(b) Overview of average computation time of 'DSG2' over $p$.

(c) Overview of average computation time of 'SDP + Cuts' over $p$.

Figure 6.2: Overview of average characteristics of Lagrangian dual bounds for 13 rudy instance classes of Biqmac Library [364].

### 6.6 Conclusions

In this chapter we show that the class of mixed-integer semidefinite programs embodies a rich structure, allowing for compact formulations of many well-known combinatorial optimization problems. These formulations follow from generic matrix theoretical and algebraic notions. Due to the recent progress in computational methods for solving MISDPs [156, 211, 234, $270,276]$, these formulations can be exploited to obtain alternative methods for solving the problems to optimality.

As most problems are naturally encoded using binary or ternary variables, we start our research with a study on the general theory related to $\operatorname{PSD}\{0,1\}-,\{ \pm 1\}-$ and $\{0, \pm 1\}$ matrices. Section 6.2 provides a comprehensive overview on this matter, including known and new results. In particular, we provide a combinatorial, polyhedral, set-completely positive and integer hull description of the set of $\operatorname{PSD}\{0,1\}$-matrices bounded by a certain rank, see Section 6.2.1. Several of these results are extended to matrices having entries in $\{ \pm 1\}$ and $\{0, \pm 1\}$.

Based on these matrix results, in particular Theorem 6.1-6.7 and Corollary 6.6, we follow a generic approach to model binary quadratic problems as BSDPs. We provide a BSDP for the class of binary quadratically constrained quadratic programs, see $\left(B S D P_{Q C Q P}\right)$, and for

| Instance <br> Class | p | Average bound values |  |  | Average computation times (s) |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  | SDP | LD | $\operatorname{Imp}$ <br> (\%) | SDP | SG | DSG1 | DSG2 | CG | $\begin{aligned} & \text { SDP } \\ &+ \text { Cuts } \end{aligned}$ |
| beasley_50 | 3 | 2089.93 | 1979.83 | 5.27 | 2.11 | 24.20 | 22.24 | 18.50 | 19.69 | 4.24 |
|  | 4 |  | 1964.37 | 6.01 | 2.32 | 16.98 | 15.97 | 12.80 | 13.66 | 5.91 |
|  | 5 |  | 1956.08 | 6.40 | 2.48 | 14.21 | 14.33 | 12.68 | 12.09 | 7.01 |
|  | 6 |  | 1955.18 | 6.45 | 2.64 | 12.31 | 11.91 | 10.27 | 10.65 | 8.41 |
| beasley_100 | 3 | 6961.49 | 6589.04 | 5.35 | 9.61 | 56.17 | 46.50 | 33.26 | 53.26 | 18.67 |
|  | 4 |  | 6535.52 | 6.12 | 9.93 | 42.82 | 33.96 | 29.61 | 40.84 | 26.85 |
|  | 5 |  | 6527.45 | 6.23 | 9.97 | 34.93 | 29.23 | 23.68 | 36.01 | 31.96 |
|  | 6 |  | 6515.68 | 6.40 | 10.28 | 31.75 | 26.87 | 25.60 | 32.14 | 38.64 |
| beasley_250 | 3 | 31043.38 | 30405.21 | 2.06 | 16.63 | 311.16 | 206.33 | 253.70 | 286.59 | - |
|  | 4 |  | 30259.20 | 2.53 | 17.36 | 250.91 | 133.69 | 180.04 | 222.14 | - |
|  | 5 |  | 30138.47 | 2.91 | 17.41 | 220.23 | 129.78 | 127.11 | 203.49 | - |
|  | 6 |  | 30021.53 | 3.29 | 17.40 | 206.03 | 119.47 | 121.95 | 198.61 | - |

Table 6.2: Average Lagrangian dual bounds and average computation times of various implementations of the projected subgradient algorithm for the Beasley instances of the Biqmac Library [364]. Each instance class consists of 10 randomly generated instances of the given type.
two types of binary quadratic matrix programs, see $\left(B S D P_{Q M P 1}\right)$ and $\left(B S D P_{Q M P 2}\right)$. These results are widely applicable to a large number of combinatorial optimization problems, see also the examples in Section 6.3.

We moreover consider problem-specific MISDP formulations that are derived in a different way than through this generic approach. We provide compact MISDP formulations for the quadratic assignment problem, see (6.24), and various variants of the graph partition problem, see $(6.28),(6.29)$ and (6.30). We derive several MISDP formulations for combinatorial optimization problems that can be modeled using algebraic connectivity, see Section 6.4.3. We also consider problems that have matrix variables that are nonbinary, such as the integer matrix completion problem and the sparse integer least squares problem, see Example 6.38 and 6.40 , respectively.

The chapter is finalized by considering a computational application of exploiting a MISDP formulation of a problem in terms of Lagrangian duality. Inspired by similar work in integer linear programming, we show the potential of integer Lagrangian dual bounds for MISDPs. These bounds, wherein both positive semidefiniteness and integrality are exploited, are obtained by solving a nonsmooth convex optimization problem. As a first step towards studying the potential of this approach, we apply a standard projected subgradient algorithm to obtain these bounds, see Algorithm 6.1. We propose several implementations of this algorithm for the ISDP formulation of the max-cut problem. The resulting bounds are closely related to the approach followed in $[1,146,147,148]$. Preliminary computational results on benchmark instances show that our approach leads to an average improvement up to $7.5 \%$ compared to the standard SDP relaxation of the max-cut problem. The Lagrangian dual bound closes on average roughly $50 \%$ of the gap between the SDP bound and the optimum. We also compare our approach to the value of the SDP relaxation with additional cutting planes. While the bounds are competitive, the computation times of our best performing algorithm, i.e., a deflected subgradient algorithm inspired by [335], are substantially smaller than those
of a state-of-the-art interior-point method that solves the SDP with cutting planes. We expect further improvements are possible by considering more advanced approaches from nonsmooth convex optimization, see e.g., [289].

Given the wide range of combinatorial optimization problems for which we derived a MISDP formulation, we expect more problems to allow for such a formulation. It is also interesting to study the behaviour of MISDP solvers on the presented formulations to see whether this leads to competitive solution approaches for the considered problems. For several problems, we introduced more than one MISDP formulation. It is part of future research to investigate which MISDP formulations are practically more beneficial. Finally, given the attractive behaviour of the introduced Lagrangian dual bounds, we expect them to be beneficially applicable in a branching framework.

# Exploiting symmetries in optimal quantum circuit design 

## Chapter summary


#### Abstract

A physical limitation in quantum circuit design is the fact that gates in a quantum system can only act on qubits that are physically adjacent in the architecture. To overcome this problem, SWAP gates need to be inserted in the quantum circuit to make the circuit physically realizable. The nearest neighbour compliance problem (NNCP) asks for an optimal embedding of qubits in a given architecture such that the total number of SWAP gates to be inserted is minimized. In this chapter we study the NNCP on general quantum architectures. Building upon the linear programming formulation of Matsuo and Yamashita [269], we show how the model can be reduced by exploiting the symmetries of the graph underlying the formulation. Based on an in-depth analysis of the automorphism group of specific Cayley graphs, we derive a symmetry-reduced NNCP algorithm that involves solving a generalized network flow problem. As a byproduct of our approach, we show that the NNCP is polynomial time solvable for several classes of highly symmetric quantum architectures. Numerical tests on various architectures indicate that the reductions in the number of variables and constraints on average is at least $90 \%$. In particular, NNCP instances on the star architecture can be solved for quantum circuits up to 100 qubits and more than 1000 quantum gates within a very short computation time. These results are far beyond the computational capacity when solving the instances without the exploitation of symmetries.


### 7.1 Introduction

Quantum theory embraces the mathematical model that can describe quantum mechanisms: the behaviour of our universe at the scale of atoms and subatomic particles. These mechanisms can be exploited in specialized hardware to perform computations that we previously believed not to be possible. This realization provides the framework of quantum computing (QC), that since the 1980s has attracted lots of attention from both academic and corporate societies.

As the physical rules that guide quantum computers are completely different than their classical counterparts, QC has been able to provide algorithms with a complexity scaling that often significantly outperform the state-of-the-art algorithms. Among the most vibrant achievements of QC are more efficient algorithms for database search [187, 188], graph problems [114] and factoring integers into primes in the seminal work by Shor [337]. Given its recent advances, experts expect that QC can potentially play a groundbreaking role in many areas, such as optimization [8], finance [294] and molecular biology [296]. For a more extensive overview of the advances and applications of quantum computing, see e.g., [282].

The most commonly used model for quantum computation is that of the gated quantum computer, where a calculation is performed by executing so-called quantum circuits. A quantum circuit acts on multiple quantum bits, i.e., qubits, which are the physical particles embedded in a quantum system. Whereas classical bits exclusively take the Boolean values zero or one, qubits can be in a superposition state, which upon measurement are displayed as zero or one with a certain probability. A quantum circuit sequentially acts on the qubits via quantum gates, which are unitary transformations that sequentially adjust the state of one or more qubits to perform an operation. Quantum circuits extend on the gate model for classical computing, and hence, a quantum computer can perform any computation that a classical computer can perform [291]. However, based on quantum phenomena such as superposition and entanglement, a quantum system is able to perform a much broader spectrum of operations.

Given the current state of technology, most physical implementations of quantum gates operate on only one or two qubits at a time [192, 291, 312]. In this setting, gates that act on more than two qubits therefore need to be realized as a sequence of gates of size at most two, which, fortunately, is possible for any quantum gate [291]. For instance, the set of one-qubit gates and two-qubit controlled-NOT gates is universal [31], meaning that this set is sufficient to perform any quantum computation.

The qubits in a quantum system are physically embedded in a certain design, i.e., the quantum architecture. This architecture is commonly represented as a coupling graph, where the vertices represent the qubits and an edge is drawn between two qubits whenever the qubits can communicate in the quantum system. With "communicate", we refer to the possibility to apply a gate to the two qubits and consequently affect their simultaneous state. Among the special coupling graphs considered in the literature are the linear array, see e.g., $[45,73,207,235,286]$, the two-dimensional grid, see e.g., [9, 44, 75], the threedimensional grid [126], the IBM QX architecture, see e.g., [366], but also general coupling graphs [46, 83, 222, 251, 341, 358].

A physical limitation of the architecture is that two-qubit gates can only be applied when the qubits are physically adjacent to each other in the coupling graph. These restrictions are known as nearest neighbour constraints and have been subject of interest in the design of quantum realizations of specific circuits, see e.g., [142], or the design of quantum architectures itself, see [286] and the references therein. Instead of research on quantum realizations that
comply with the nearest neighbour constraints, we can also disregard these constraints at first and alter existing quantum circuits to make them feasible, which will be the followed approach in this chapter.

A quantum circuit can be made compliant with respect to the nearest neighbour constraints by the insertion of SWAP gates. A SWAP gate acts on two adjacent qubits by interchanging their location in the coupling graph ${ }^{1}$. If the coupling graph is connected, any quantum circuit can be made compliant by the insertion of a finite number of SWAP gates and there are often many ways to do so. However, due to a qubit's interaction with its environment [102], quantum systems currently still suffer from physical instability of qubit states after some period of time. This raises the desire for quantum circuits with as few gates as possible. We therefore prefer to add the minimum number of SWAP gates in order to make a circuit compliant.

Given a quantum circuit and a coupling graph, the nearest neighbour compliance problem (NNCP) asks for an optimal sequential allocation of the qubits over the quantum architecture such that the total number of SWAP gates to be inserted is minimized. With "sequential", we refer to the decision variables to not only concern the initial allocation, but also the actual SWAP operations that take place over time. The NNCP was proven to be $\mathcal{N} \mathcal{P}$-hard via a reduction from the token swapping problem [341].

Most research on the NNCP has been on heuristic methods, such as greedy methods [9, 207], harmony search [9], optimal linear arrangement [299] and receding horizon methods [207, 235, 332, 368]. Exact approaches to tackle the NNCP include exhaustive search [99, 207], explicit cost enumeration [369] and linear programming (LP) based methods on the adjacent transposition graph [269, 285]. All these methods embrace an implicit factorial scaling in the number of qubits, due to the inherited total number of possible assignments of the qubits. Recently, also polynomial sized models have been considered that are based on mixed-integer linear programming [286, 356]. The construction considered in [286] is based on the linear array coupling graph, while the models in [356] consider ordering problems for distributed quantum computing. Other research focuses on a related version of the NNCP, where an initial qubit ordering has to be realized that minimizes the (approximated) number of SWAP operations, without actually considering the exact insertions into the quantum circuit, see [236, 237, 333].

Building upon the LP formulation considered in [269, 285], a main feature of our approach concerns the exploitation of symmetries in the model. The literature on symmetry reduction methods in mathematical optimization is extensive, and we refer the reader to [253, 266] for comprehensive overviews in this direction. It is well-known that symmetries in integer linear programming (ILP) problems lead to poor behaviour of numerical algorithms, due to the costly duplication of computational effort in branching approaches. To reduce this negative effect, symmetries need to be broken, e.g., by perturbation, symmetry-breaking inequalities (e.g., [265]) or specialized branching techniques (e.g., [334]). The literature on symmetry reduction for integer linear programs (ILPs) can be distinguished between problem-based approaches, whose symmetry groups are known a priori (see e.g., [247]), or generic techniques. The latter class on one hand contains methods based on branching tree reductions, which was mainly pioneered by Margot [263, 264] in his work on isomorphism pruning. The idea behind isomorphism pruning is to detect whenever multiple child nodes in the branching tree are isomorphic, and hence, only one of them needs to be considered. In the

[^0]same vein, orbital branching [295] exploits the orbits of the symmetry group in the branching tree in order to partition the feasible region while taking care of symmetries. Alternative methods mainly consider symmetry-handling constraints to restrict the feasible region of an optimization problem by eliminating symmetric solutions. Two well-known streams in this direction are the utilization of so-called orbitopes [230], which is the convex hull of solutions that are lexicographically maximal in their orbit, and fundamental domains [145], where the feasible region is expressed as the image of an affine transformation induced by a finite group. Branching tree reductions and symmetry-handling constraints can also be combined, see e.g., [105].

When considering symmetry reduction methods for linear programs, a major research line considers the study of symmetric polyhedra, see [266, Section 6] and the references therein. Another research line considers the exploitation of symmetries in the simplex algorithm [353, 354]. Bödi et al. [49] consider the exploitation of symmetries in linear programs by restricting to the subspace of fixed points under a linear map induced by the symmetries in the program. This approach can be generalized to convex programs and is closely related to the invariant-based symmetry reduction approaches applied to conic and semidefinite programs, see e.g., Gatermann and Parrilo [166], to which our reduction method also belongs.

## Main results and outline

In this chapter we consider the nearest neighbour compliance problem on general coupling graphs. Following the linear programming (LP) formulation derived in [269], we analyse the group symmetry of the underlying graph, which is a sequence of connected Cayley graphs. By exploiting these symmetries, we reduce the LP model in the number of variables and constraints, leading to a symmetry-reduced algorithm for solving the NNCP. We show the theoretical and practical strength of our approach for several classes of symmetric coupling graphs for which the reduction is most significant, namely the graphs that embrace a large automorphism group.

The LP formulation of [269] can be viewed as a single-pair shortest path problem on a directed graph that we refer to as the graph $X=(V, A)$. As a first step in our approach, we consider the automorphism group of the subgraphs of $X$. Each subgraph is a Cayley graph of the symmetric group $\mathbb{S}_{n}$ generated by the edges in the coupling graph of the quantum architecture. We derive the full automorphism group of such Cayley graphs in the case that it is normal, and review some conditions on the coupling graph under which normality holds. Afterwards, we extend these automorphism results of the subgraphs to derive the automorphism group of $X$. In particular, we derive an explicit group description of a subgroup $G_{X}$ of the automorphism group of $X$, which is the full automorphism group of $X$ when normality holds. We also study the orbit and orbital structure of the group action of $G_{X}$ on $X$. The results on the group structure of these Cayley graphs are in itself interesting, as such graphs are of main importance in interconnection networks [164, 204].

By averaging over each orbital of the action of $G_{X}$ on $X$ via the Reynolds operator, we show how the LP formulation can be reduced following the approach of [49]. We show that the resulting reduced LP formulation is equivalent to a generalized network flow problem on an auxiliary graph following from our construction. For symmetric coupling graphs, this reduced LP formulation is significantly smaller in size. As a byproduct of our approach, we show that the NNCP is polynomial time solvable for coupling graph whose automorphism group scales factorially in the number of qubits, e.g., the star graph or complete bipartite
graphs with one of the sizes fixed. Our reduction leads to a symmetry-reduced NNCP algorithm, where all steps follow from the in-depth analysis of the algebraic structure of $X$ and do not rely on the use of any external algebraic software.

Although the ingredients of our approach are presented generally, we explicitly show how the reduced LP can be constructed for four special graph types: the cycle graph, the star graph, the biclique and the two-dimensional lattice graph. For each of these classes, we show how the orbital structure unfolds by analyzing a specific subgroup of the automorphism group of the coupling graphs.

Finally, we test our symmetry-reduced NNCP algorithm on real and randomly generated quantum circuits defined on the above-mentioned coupling graphs. Our numerical tests confirm that the effort spent in the algebraic analysis pays off, as computation times to solve an instance are several orders of magnitude smaller compared to the nonreduced model. Whereas the model from [269] can only solve instances up to 8 qubits, the largest instances we solve contain up to 100 (resp. 40) qubits and several hundreds of quantum gates on the star (resp. biclique) coupling graph. Observe that such instances are far out of reach for the nonreduced model, as this would require the use of at least $100!\approx 9.33 \cdot 10^{157}$ constraints and even more variables.

This chapter is structured as follows. Section 7.2 formally introduces the NNCP and reviews the shortest path formulation of [269]. In Section 7.3 we analyse the automorphism group of the graph underlying the formulation, as well as its orbit and orbital structure. These algebraic properties are exploited in Section 7.4, where we present our symmetryreduced NNCP algorithm. In Section 7.5 we apply our approach to several specific types of coupling graphs. Computational results are discussed in Section 7.6.

## Preliminaries on quantum computation

In this section we briefly provide some background on the basic ingredients of quantum computing. Since we consider the problem from a mathematical point of view, any knowledge that goes beyond this introduction is not needed to understand the contents of this chapter.

The state of a single qubit can be represented as a unit vector in $\mathbb{C}^{2}$, which is a linear combination between the computational basis states $|0\rangle:=\binom{1}{0} \in \mathbb{C}^{2}$ and $|1\rangle:=\binom{0}{1} \in$ $\mathbb{C}^{2}$ (using Dirac notation). These states can be viewed as the Boolean values of classical bits. Hence, the state of a qubit can be displayed as $|\phi\rangle=\alpha|0\rangle+\beta|1\rangle$ with $\alpha, \beta \in \mathbb{C}$ and $|\alpha|^{2}+|\beta|^{2}=1$. Whenever $|\phi\rangle$ is not in one of the basis states, we say that the qubit is in superposition. A qubit in superposition cannot be observed in that way, as the qubit upon measurement will always be in one of the basis states. The probability that $|\phi\rangle$ is in state $|0\rangle$ upon measurement is $|\alpha|^{2}$, whereas the state $|1\rangle$ is obtained with probability $|\beta|^{2}$.

A quantum computer typically contains multiple qubits. The state of the quantum system is determined by the combination of the individual qubit states. Extending on the notation above, the state of an $n$-qubit quantum system is displayed by a vector in $\mathbb{C}^{2^{n}}$. When two qubits are in states $\left|\phi_{1}\right\rangle=\alpha_{1}|0\rangle+\beta_{1}|1\rangle$ and $\left|\phi_{2}\right\rangle=\alpha_{2}|0\rangle+\beta_{2}|1\rangle$, respectively, then the combined state $|\omega\rangle$ is the tensor product these states, i.e.,

$$
|\omega\rangle=\left|\phi_{1}\right\rangle \otimes\left|\phi_{2}\right\rangle=\alpha_{1} \alpha_{2}|00\rangle+\alpha_{1} \beta_{2}|01\rangle+\beta_{1} \alpha_{2}|10\rangle+\beta_{1} \beta_{2}|11\rangle,
$$

where $|00\rangle,|01\rangle,|10\rangle$ and $|11\rangle$ denote the computational basis vectors of $\mathbb{C}^{4}$. Any unit vector in $\mathbb{C}^{4}$ denotes a state of a 2-qubit quantum system. If this state can be written as a single tensor product of two single-qubit states, e.g., the state $|\omega\rangle$ displayed above, we call the state
pure. Sometimes this is not possible, e.g.,

$$
|\omega\rangle=\frac{1}{\sqrt{2}}|00\rangle+\frac{1}{\sqrt{2}}|11\rangle
$$

cannot be written as a single tensor product of single-qubit states. Such a state is called entangled. Upon measurement, each of the states $|00\rangle$ and $|11\rangle$ is obtained with probability $\frac{1}{2}$. That implies that if we would measure the state of only one of the qubits, we have gained information about the other qubit's state instantly. This quantum phenomenon is a necessary ingredient for the success of many applications of quantum computing.

A quantum gate can adjust the state of a quantum system by applying a unitary transformation to its state vector. More precisely, a quantum gate that acts on system in state $|\phi\rangle$ is displayed as a complex unitary matrix $U$, and the state after applying the gate is given by the matrix vector product $U|\phi\rangle$. For instance, a simple elementary quantum gate is the Pauli- $X$-gate, also referred to as the classical NOT gate. Its matrix representation is

$$
X=\left(\begin{array}{ll}
0 & 1 \\
1 & 0
\end{array}\right)
$$

and applying the gate on a single qubit in state $|\phi\rangle=\binom{\alpha}{\beta}$ results in

$$
X|\phi\rangle=\left(\begin{array}{ll}
0 & 1 \\
1 & 0
\end{array}\right)\binom{\alpha}{\beta}=\binom{\beta}{\alpha}
$$

The only quantum gates that can be directly applied are the gates that act on at most two qubits. Gates that act on more than two qubits therefore need to be decomposed into oneor two-qubit gates in order to be physically applicable. In Appendix A. 6 we provide an overview of the most commonly used gates and their quantum realizations.

A quantum circuit can be represented in a common way by a set of parallel lines that represent the qubits, where the gates are drawn as vertical segments spanned between the qubits they act on. An example quantum circuit is provided in Figure 7.1, where each quantum gate is presented in a simplified way using empty squares. Typically, these gates are of different types and their functionality is presented by the symbols in the circuit. The quantum circuit in Figure 7.1a acts on four qubits $q_{1}, \ldots, q_{4}$ and has four quantum gates. Suppose the communication in the quantum system is given by the coupling graph $G$ given in Figure 7.1b. This coupling graph implies that gates can only be applied to pairs of qubits that are adjacent in $G$. The labels $q_{1}, \ldots, q_{4}$ displayed next to the horizontal lines in Figure 7.1a denote the initial allocation of the qubits in the coupling graph $G$. We see that the first gate can be applied, since $q_{2}$ and $q_{3}$ are adjacent in $G$. However, this is not the case for the second gate. In order to apply it, we need to bring $q_{1}$ and $q_{4}$ adjacent to one another in the architecture. This can be done by inserting a SWAP gate, which interchanges the location of two qubits in the architecture. Either a SWAP gate between $q_{1}$ and $q_{2}$ or a SWAP gate between $q_{2}$ and $q_{4}$ can be inserted to apply the second gate of the architecture. One can check that if we adopt the first option, we do not need any more SWAP gates to compile the circuit, whereas this is not the case for the other option. The resulting quantum circuit that can be physically compiled is displayed in Figure 7.1c. The functionality of the circuit remains unchanged by the insertion of the SWAP gate. Observe, however, that the allocation of the qubits over the architecture before and after applying the circuit is different.

(a) Quantum circuit

(b) Coupling graph $G$

(c) Feasible quantum circuit

Figure 7.1: Example of an (abstract) quantum circuit (left) with four gates that acts on four qubits. The quantum circuit cannot be compiled on the coupling graph $G$ (middle). By inserting an appropriate SWAP gate, the circuit can be compiled on $G$ (right).

The example of Figure 7.1 shows a very small instance of the nearest neighbour compliance problem (NNCP), which will be formally introduced in the next section.

### 7.2 Nearest neighbour compliance problem

A given quantum circuit can be made feasible with respect to the adjacent interaction constraints by inserting SWAP gates. Although these do not interfere with the functionality of the quantum circuit, the total number of gates is favoured to be as small as possible. The nearest neighbour compliance problem (NNCP) aims at finding an embedding of the qubits over a given architecture such that the number of SWAP gates needed to make the final circuit feasible with respect to the adjacent interaction constraints is minimized.

In this section we formally introduce the nearest neighbour compliance problem as a shortest path problem.

### 7.2.1 Mathematical formulation of the NNCP

We make two model assumptions about the quantum circuits under consideration. First, quantum gates that act on a single qubit always comply with the adjacent interaction constraints and are therefore not taken into consideration. Second, it only makes sense to talk about adjacency in the context of two-qubit quantum gates. If a quantum gate acts on more than two qubits, we first decompose it into two-qubit gates. This is always possible [291] and there exist a large variety of ways for doing this. Throughout this chapter, we assume without loss of generality that quantum circuits consist of a sequence of two-qubit gates.

Let $Q=[n]$ denote the set of qubits of the quantum system. The qubits need to be embedded in a certain topology, that we refer to as the architecture of the quantum system. This architecture is fixed and can be modeled as a graph $(L, E)$. Here $L=[n]$ denotes a set of physical locations and $E \subseteq L^{(2)}$ is the adjacency structure of the architecture. That is, if $\{i, j\} \in E$, then locations $i$ and $j$ are physically adjacent to each another and can therefore directly share information. The graph is denoted as the coupling graph of the quantum system and denoted by $\operatorname{Coup}(E):=(L, E)$. We assume that $(L, E)$ is connected, which implies that all pairs of locations can indirectly share information.

Each qubit in $Q$ needs to be assigned to a physical location in $L$. A bijection $\tau: L \rightarrow Q$ is called a qubit order. To present a qubit order, we use one-line notation with respect to
the images in $Q$. For example, the order

$$
\tau=(\tau(1), \tau(2), \tau(3), \tau(4))=(2,3,1,4)
$$

corresponds to the assignment where qubit 2 is on location 1 , qubit 3 on location 2 , qubit 1 on location 3 and qubit 4 on location 4. The set of all qubit orders on $n$ qubits is equal to $\mathbb{S}_{n}$.

A SWAP gate interchanges the qubits on two locations in the embedding. It can also be modeled as an element $\sigma \in \mathbb{S}_{n}$, where $\sigma$ is a transposition. Using cycle notation, the SWAP gate $\sigma=(i j)$ applied on the qubit order $\tau$ interchanges the qubits $\tau(i)$ and $\tau(j)$. Applying this SWAP gate can be seen as a right action of $\sigma$ on $\mathbb{S}_{n}$, i.e.,

$$
\begin{aligned}
\tau \circ \sigma & =(\tau(1), \tau(2), \ldots, \tau(i), \ldots, \tau(j), \ldots, \tau(n)) \circ(i j) \\
& =(\tau(1), \tau(2), \ldots, \tau(j), \ldots, \tau(i), \ldots, \tau(n))
\end{aligned}
$$

for all $\tau \in \mathbb{S}_{n}$. To simplify notation, we omit the $\circ$ in group actions and just write $\tau \sigma$ in the sequel.

Remark 7.1. Although both elements of $\mathbb{S}_{n}, \tau$ represents a qubit order, while $\sigma$ represents a SWAP gate. To discriminate between these objects, we always use one-line notation for qubit orders and cycle notation for SWAP gates throughout the chapter.

A SWAP gate can only be applied to qubits on locations that are adjacent in $\operatorname{Coup}(E)$. Whenever there is an edge $\{i, j\} \in E$, the SWAP gate $(i j)$ acts on adjacent locations. Let

$$
\begin{equation*}
T:=\left\{(i j) \in \mathbb{S}_{n}:\{i, j\} \in E\right\} \tag{7.1}
\end{equation*}
$$

denote the set of transpositions that correspond to a SWAP gate in the quantum system. Observe that $E$ and $T$ are related, although $E$ is a graph-theoretical and $T$ is an algebraic concept.

Given two qubit orders $\tau_{1}, \tau_{2} \in \mathbb{S}_{n}$, we are interested in the minimum number of SWAP gates that need to be applied to $\tau_{1}$ to obtain $\tau_{2}$ by only using SWAP gates from $T$. Let $J_{T}: \mathbb{S}_{n} \times \mathbb{S}_{n} \rightarrow \mathbb{Z}_{+}$be defined as

$$
J_{T}\left(\tau_{1}, \tau_{2}\right):=\min \left\{k: \tau_{2}=\tau_{1} \sigma_{1} \sigma_{2} \ldots \sigma_{k}, \sigma_{1}, \ldots, \sigma_{k} \in T\right\}
$$

which forms a metric on all qubit orders and depends on the quantum architecture $T$. Observe that this metric is left-invariant, i.e., $J_{T}\left(\tau_{1}, \tau_{2}\right)=J_{T}\left(\pi \tau_{1}, \pi \tau_{2}\right)$ for all $\pi \in \mathbb{S}_{n}$, implying that $J_{T}\left(\tau_{1}, \tau_{2}\right)$ equals the length of the shortest sequence of transpositions of $T$ needed to generate $\tau_{2}^{-1} \tau_{1}$. It is known that finding such minimum-length sequence is in general PSPACE-complete [226]. For special types of coupling graphs, however, the metric $J_{T}$ is computationally tractable, e.g., when $\operatorname{Coup}(E)$ is a path or the complete graph. For these cases, $J_{T}$ coincides with the Kendall tau distance and the Cayley distance, respectively.

Let $i, j \in Q$ be two qubits such that $i \neq j$. Then the unordered pair $g_{i j}=\{i, j\}$ is a twoqubit quantum gate that acts on qubits $i$ and $j$. Whenever the specific qubits on which the gate acts are irrelevant, we sometimes omit the subscripts. A finite sequence $C=\left(g^{1}, \ldots, g^{m}\right)$ of gates $g^{1}, \ldots, g^{m}$ is called a gate sequence of size $m$. Given a set of qubits $Q$ and a gate sequence $C$, the tuple $\Gamma=(Q, C)$ is called a quantum circuit.

We say that a qubit order $\tau$ complies with a gate $g_{i j}$ if qubits $i$ and $j$ are adjacent in $\tau$ with respect to the coupling graph $\operatorname{Coup}(E)$, i.e., if $\tau^{-1}\left(g_{i j}\right)=\left\{\tau^{-1}(i), \tau^{-1}(j)\right\} \in E$. We now formulate the NNCP.

Definition 7.2 (NNCP). Let $\Gamma=(Q, C)$ be a quantum circuit with $n$ qubits and $m$ gates, and let $\operatorname{Coup}(E)=(L, E)$ be the coupling graph of the underlying architecture. Then, the nearest neighbor compliance problem asks for a sequence of qubit orders $\tau^{k}, k \in[m]$, each one corresponding to an order prior to applying a gate of $C$, such that $\sum_{k=1}^{m-1} J_{T}\left(\tau^{k}, \tau^{k+1}\right)$ is minimized and such that $\tau^{k}$ complies with $g^{k}$ for all $k \in[m]$.

The NNCP as presented in Definition 7.2 is known to be $\mathcal{N} \mathcal{P}$-hard in general [341].
We end this section by introducing the notion of the so-called gate graph, which captures the underlying qubit dependencies imposed by the gates in the circuit.

Definition 7.3. Let $\Gamma=(Q, C)$ be a quantum circuit. The gate graph $(Q, U)$ is an undirected graph that has vertex set $Q$ and edge set $U=\{g: g \in C\}$.

The gate graph $(Q, U)$ will be exploited in Section 7.3.2.

### 7.2.2 The NNCP as a shortest path problem

In this section we show how the NNCP can be modeled as a shortest path problem in a directed graph following the construction of [269, 285].

Let an instance of the NNCP as defined in Section 7.2 .1 be given. Of key importance in the reduction to a shortest problem is the notion of a Cayley graph.

Definition 7.4 (Cayley graph). Let $G$ be a finite group and let $S$ be a subset of $G$ such that $\operatorname{id}_{G} \notin S$ and $S=S^{-1}:=\left\{s^{-1}: s \in S\right\}$. The Cayley graph Cay $(G, S)$ on $G$ with respect to $S$ is defined as the (directed) graph with vertex set $G$ and $\operatorname{arc}$ set $\{(g, g s): g \in G, s \in S\}$.

Observe that Cay $(G, S)$ as in Definition 7.4 contains an arc if and only if it also contains the reversed arc. Although this suggests that any $\operatorname{Cay}(G, S)$ is undirected, we stick to the setting of two reversed directed arcs, since we will employ the Cayley graphs as subgraphs of a larger directed graph.

Let $H:=\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$, where $T$ is given by (7.1). More precisely, the vertex and arc set of $H$ are given by

$$
V(H):=\mathbb{S}_{n} \quad \text { and } \quad A(H):=\left\{(\tau, \tau \sigma): \tau \in \mathbb{S}_{n}, \sigma \in T\right\}
$$

respectively. Each vertex in $V(H)$ represents a qubit order, while an arc in $A(H)$ represents a SWAP gate that translates a qubit order into another qubit order with respect to the coupling graph. Now, we define the subgraphs $H^{k}$ for $k \in[m]$ as disjoint copies of $H$, one for each gate in the circuit.

The $m$ subgraphs $H^{k}$ are merged to obtain a graph $X=(V, A)$. The vertex set $V$ of $X$ consists of the union of all $V^{k}, k \in[m]$, as well as a source $s$ and $\operatorname{sink} t$, i.e., $V=$ $\{s\} \cup V^{1} \cup \cdots \cup V^{m} \cup\{t\}$. Since the subgraphs $H^{1}, \ldots, H^{m}$ are identical, we use superscripts to indicate to which subgraph a vertex belongs. For example, $\tau^{k}$ and $\tau^{k+1}$ correspond to the same qubit order in subgraph $k$ and $k+1$, respectively.

The arc set $A$ of $X$ contains the union of all $A^{k}, k \in[m]$. Moreover, the arcs between different subgraphs are introduced by the following sets:

$$
\begin{align*}
D^{0} & :=\left\{\left(s, \tau^{1}\right): \tau^{1} \in V^{1}\right\} \\
D^{k} & :=\left\{\left(\tau^{k}, \tau^{k+1}\right): \tau^{k} \in V^{k}, \tau^{k+1} \in V^{k+1},\left(\tau^{k}\right)^{-1}\left(g^{k}\right) \in E\right\}, \quad k \in[m-1]  \tag{7.2}\\
D^{m} & :=\left\{\left(\tau^{m}, t\right): \tau^{m} \in V^{m},\left(\tau^{m}\right)^{-1}\left(g^{m}\right) \in E\right\} .
\end{align*}
$$

These sets can be interpreted as follows. The set $D^{0}$ contains an arc from $s$ to all nodes in $H^{1}$. For all $k \in[m-1], D^{k}$ contains the connecting arcs from $H^{k}$ to $H^{k+1}$. Suppose the gate $g^{k}$ acts on qubits $i$ and $j$. Then we include an arc from a qubit order $\tau^{k}$ in $H^{k}$ to the same qubit order $\tau^{k+1}$ in $H^{k+1}$ if and only if $i$ and $j$ are adjacent in $\tau^{k}$ with respect to $\operatorname{Coup}(E)$. That is, whenever $\left(\tau^{k}\right)^{-1}\left(g^{k}\right)=\left\{\left(\tau^{k}\right)^{-1}(i),\left(\tau^{k}\right)^{-1}(j)\right\} \in E$. Similarly, $D^{m}$ contains all arcs from $\tau^{m}$ with this property to the sink node $t$. Now, the $\operatorname{arc}$ set $A$ of $X$ is given by

$$
A=A^{1} \cup \cdots \cup A^{m} \cup D^{0} \cup D^{1} \cup \cdots \cup D^{m} .
$$

We set the cost of each arc in $A^{k}, k \in[m]$, equal to one, as traversing these arcs corresponds to applying one SWAP gate. The cost of the arcs in $D^{k}, k=0, \ldots, m$, is equal to zero, as no SWAP gates are applied when moving from a subgraph to the next.

We are now ready to state the main result of this section.
Theorem 7.5 ([285]). Any ( $s, t$ )-path in $X$ corresponds to a sequence $\left(\tau^{1}, \ldots, \tau^{m}\right)$ of qubit orders that all comply with the adjacent interaction constraints. A shortest $(s, t)$-path in $X$ corresponds to an optimal solution of the NNCP.

Proof. Follows immediately from the construction.
There are many algorithms in the literature for solving the shortest path instance, e.g., Dijkstra's algorithm with Fibonacci heaps [144]. Alternatively, we can solve it as a linear programming (LP) problem. For all $k \in[m]$ and $e \in A^{k}$, let $x_{e}$ denote a variable that is one if arc $e$ is used on a path, and zero otherwise. Similarly, for all $k \in\{0\} \cup[m]$ and $e \in D^{k}$, let $y_{e}$ denote a variable that is one if $\operatorname{arc} e$ is used on a path, and zero otherwise. Then the shortest ( $s, t$ )-path in $X$ can be found by solving the following LP:
(SPP)

$$
\begin{aligned}
\min & \sum_{k=1}^{m} \sum_{e \in A^{k}} x_{e} \\
\text { s.t. } & \sum_{e \in D^{0}} y_{e}=1, \quad \sum_{e \in D^{m}} y_{e}=1 \\
& \sum_{e \in \delta^{-}\left(\tau, D^{k-1}\right)} y_{e}+\sum_{e \in \delta^{-}\left(\tau, A^{k}\right)} x_{e}=\sum_{e \in \delta^{+}\left(\tau, D^{k}\right)} y_{e}+\sum_{e \in \delta^{+}\left(\tau, A^{k}\right)} x_{e} \forall \tau \in V^{k}, k \in[m] \\
& 0 \leq x_{e} \leq 1 \quad \forall e \in A^{k}, k \in[m], \\
& 0 \leq y_{e} \leq 1 \quad \forall e \in D^{k}, k \in\{0\} \cup[m] .
\end{aligned}
$$

### 7.3 Symmetries in $X=(V, A)$

The graph $X$ constructed in Section 7.2 .2 contains $\Theta(m n!)$ vertices and $\Theta(|E| m n!)$ arcs. The bottleneck in solving the NNCP to optimality is clearly the factorial scaling in the number of qubits. Fortunately, for many structured quantum system architectures, the problem can be reduced by exploiting the symmetries in $X$. In this section we study these symmetries in terms of its automorphism group.

In Section 7.3 .1 and 7.3 .2 we study the automorphism group of Cayley graphs generated by transpositions and the automorphism group of $X$, respectively. In Section 7.3 .3 we study the orbit and orbital structure induced by this group action on $X$. The results in this section are the key ingredients of the symmetry reduction explained in Section 7.4.

### 7.3.1 Automorphism group of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$

For a directed graph $X$ with vertex set $V$ and arc set $A$, a permutation $\rho \in \operatorname{Sym}(V)$ is called an automorphism of $X$ if $(\rho(i), \rho(j)) \in A$ if and only if $(i, j) \in A$. We also say that such $\rho$ acts on $X$. The automorphism group of $X$ is the group of all automorphisms of $X$ and is denoted by $\operatorname{Aut}(X)$.

In order to determine the automorphism group of the graph $X$ introduced in Section 7.2.2, we start by considering the automorphism group of the subgraphs $H^{k}, k \in[m]$. Recall that all $H^{k}$ are identical and equal to $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$, where $T$ is a set of transpositions, see (7.1). Hence, the goal of this subsection is to study $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$.

There exist several works in the literature on the automorphism group of Cayley graphs generated by transpositions. As indicated by Feng [130], we can show that $\mathbb{S}_{n}$ acts on Cay $\left(\mathbb{S}_{n}, T\right)$ by left multiplication. That is, for any $a \in \mathbb{S}_{n}$ the mapping $\tau \mapsto a \tau$ defines an automorphism of $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$. All such automorphisms form a subgroup of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$. We can also show that the group $\operatorname{Aut}(\operatorname{Coup}(E))$ acts on $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ by right multiplication via the mapping $\tau \mapsto \tau b^{-1}$, which is an automorphism of Cay $\left(\mathbb{S}_{n}, T\right)$ for all $b \in$ $\operatorname{Aut}(\operatorname{Coup}(E))$. To verify this, let $\left(\tau_{1}, \tau_{2}\right)$ be an arc in $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$. Then $\tau_{2}=\tau_{1} \sigma_{1}$ for some $\sigma_{1} \in T$. The image of this arc under the action of an element $b \in \operatorname{Aut}(\operatorname{Coup}(E))$ is

$$
\left(\tau_{1} b^{-1}, \tau_{2} b^{-1}\right)=\left(\tau_{1} b^{-1}, \tau_{1} \sigma_{1} b^{-1}\right)=\left(\tau_{1} b^{-1}, \tau_{1} b^{-1} b \sigma_{1} b^{-1}\right)
$$

It is well-known that if a permutation maps $i$ to $j$, then the conjugate of this permutation by $b$ maps $b(i)$ to $b(j)$. Therefore, if $\sigma_{1}=(i j)$, then $\sigma_{2}:=b \sigma_{1} b^{-1}=(b(i) b(j))$. Since $b$ is an automorphism of $\operatorname{Coup}(E), \sigma_{2} \in T$, which implies that $\left(\tau_{1} b^{-1}, \tau_{2} b^{-1}\right)$ is again an arc of $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$. Since $\tau \mapsto \tau b^{-1}$ is bijective, it follows that $\operatorname{Aut}(\operatorname{Coup}(E))$ indeed acts on $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ by right multiplication.

We now show how both group actions are combined in order to obtain a subgroup of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$. Let us define the mapping $\theta: \mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E)) \rightarrow \operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$ given by

$$
\begin{equation*}
\theta(a, b):=\left(\tau \mapsto a \tau b^{-1}\right) \tag{7.3}
\end{equation*}
$$

Indeed, $\theta(a, b)$ is the composition of an action by left multiplication by an element $a \in \mathbb{S}_{n}$ and a right multiplication by an element $b \in \operatorname{Aut}(\operatorname{Coup}(E))$ (in arbitrary order). So, for all $(a, b)$ in its domain, $\theta(a, b)$ is indeed an automorphism of Cay $\left(\mathbb{S}_{n}, T\right)$. We can show that the map $\theta$ is a group homomorphism that is injective.

Theorem 7.6. For $n \geq 3$, the mapping $\theta$ is a group homomorphism from $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$ to $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$ that is injective.

Proof. We start by showing that $\theta$ is indeed a group homomorphism. Let $\left(a_{1}, b_{1}\right),\left(a_{2}, b_{2}\right) \in$ $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$. Then, for all $\tau \in \mathbb{S}_{n}$ :

$$
\begin{aligned}
\theta\left(\left(a_{1}, b_{1}\right)\left(a_{2}, b_{2}\right)\right)(\tau) & =\theta\left(\left(a_{1} a_{2}, b_{1} b_{2}\right)\right)(\tau)=a_{1} a_{2} \tau\left(b_{1} b_{2}\right)^{-1}=a_{1} a_{2} \tau b_{2}^{-1} b_{1}^{-1} \\
\theta\left(\left(a_{1}, b_{1}\right)\right) \theta\left(\left(a_{2}, b_{2}\right)\right)(\tau) & =\theta\left(a_{1}, b_{1}\right)\left(a_{2} \tau b_{2}^{-1}\right)=a_{1} a_{2} \tau b_{2}^{-1} b_{1}^{-1} .
\end{aligned}
$$

Hence, $\theta$ is a group homomorphism. To prove injectivity, assume that $\left(a_{1}, b_{1}\right),\left(a_{2}, b_{2}\right) \in$ $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$ are such that $\theta\left(\left(a_{1}, b_{1}\right)\right)=\theta\left(\left(a_{2}, b_{2}\right)\right)$. Then, $a_{1} \tau b_{1}^{-1}=a_{2} \tau b_{2}^{-1}$ for all $\tau \in \mathbb{S}_{n}$. In particular, this must hold for $\tau=\mathrm{id}$, from which it follows that $a_{1} b_{1}^{-1}=a_{2} b_{2}^{-1}$,
and hence, $a_{2}=a_{1} b_{1}^{-1} b_{2}$. Substituting this into $a_{1} \tau b_{1}^{-1}=a_{2} \tau b_{2}^{-1}$, yields

$$
a_{1} \tau b_{1}^{-1}=a_{1} b_{1}^{-1} b_{2} \tau b_{2}^{-1} \quad \forall \tau \in \mathbb{S}_{n}, \quad \text { or equivalently, } \quad \tau b_{1}^{-1} b_{2}=b_{1}^{-1} b_{2} \tau \quad \forall \tau \in \mathbb{S}_{n}
$$

This implies that $b_{1}^{-1} b_{2} \in Z\left(\mathbb{S}_{n}\right):=\left\{g \in \mathbb{S}_{n}: g h=h g \quad \forall h \in \mathbb{S}_{n}\right\}$. It is well-known that the center $Z\left(\mathbb{S}_{n}\right)$ is trivial for $n \geq 3$, hence $b_{1}=b_{2}$. From this, it simply follows that also $a_{1}=a_{2}$, hence $\theta$ is injective.

Theorem 7.6 shows that the image of $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$ under $\theta$ is a subgroup of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$. It follows from the injectivity of $\theta$ that this subgroup is isomorphic to the direct product $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$. To simplify notation, we will from now on identify this subgroup as $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$, although in fact we mean its image under $\theta$.

The map $\theta$ is in general not a bijection, which means that $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$ is not the full automorphism group of $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$. However, in many of the cases that are interesting for our application, the subgroup turns out to be the full automorphism group. We now present a series of sufficient conditions for this to be true.

We call the Cayley graph $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ normal if the subgroup of all automorphisms by left multiplication by elements of $\mathbb{S}_{n}$, i.e., $\left\{(\tau \mapsto a \tau): a \in \mathbb{S}_{n}\right\}$, is a normal subgroup of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$.

Theorem 7.7 ([163]). The graph $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is normal if and only if $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right) \cong$ $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$.

The following theorem states some known sufficient conditions for $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ to be (non)normal. Recall that the girth of a graph is the length of its shortest cycle. Trees have infinite girth.

Theorem 7.8. The graph $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is normal if:

- $\operatorname{Coup}(E)$ is a tree;
- $\operatorname{Coup}(E)$ is a graph with girth at least 5.

The graph Cay $\left(\mathbb{S}_{n}, T\right)$ is nonnormal if:

- $\operatorname{Coup}(E)$ is the 4-cycle $C_{4}$;
- $\operatorname{Coup}(E)$ is the complete graph $K_{n}$.

Proof. The normality of $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ when $\operatorname{Coup}(E)$ is a tree is first shown by Feng [130]. Ganesan [161] shows the more general condition on the girth of $\operatorname{Coup}(E)$. The nonnormality results implied by $\operatorname{Coup}(E)$ to be $C_{4}$ or $K_{n}$ are obtained by Ganesan [161] and Ganesan [162], respectively.

In [164] it is conjectured that the two latter cases from Theorem 7.8 are the only connected coupling graphs for which its corresponding Cayley graph Cay $\left(\mathbb{S}_{n}, T\right)$ is nonnormal. If this conjecture is true, it follows from Theorem 7.7 that $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$ is the full automorphism group for almost all quantum architectures. In case $\operatorname{Coup}(E)$ is $C_{4}$ or $K_{n}$, the automorphism group of $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is known, see [161, Section 3] and [162, Theorem 1.1], respectively.

### 7.3.2 Automorphism group of $X$

Now that we established either the full automorphism group of Cay $\left(\mathbb{S}_{n}, T\right)$ or a subgroup of it, we focus on the automorphism group of the entire graph $X$. Indeed, we need to take the arc structure in-between the subgraphs $H^{k}$ into account. We start by showing how these arcs restrict the automorphism group of a single subgraph, after which we combine these results to obtain $\operatorname{Aut}(X)$.

Each $H^{k}$ corresponds to a gate $g^{k}$ acting on two qubits in $Q$. The set of outgoing arcs $D^{k}$ consists of arcs leaving qubit orders $\tau$ where $\tau^{-1}\left(g^{k}\right) \in E$, see (7.2). Since this arc structure needs to be preserved, the automorphisms of interest must setwise fix the qubit orders with this property. For all $k \in[m]$, let

$$
\begin{equation*}
F^{k}:=\left\{\tau \in \mathbb{S}_{n}: \tau^{-1}\left(g^{k}\right) \in E\right\} . \tag{7.4}
\end{equation*}
$$

Instead of the automorphism group of $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$, we are only interested in its subgroup that setwise fixes $F^{k}$. That is,

$$
\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right):=\left\{\rho \in \operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right): \rho\left(F^{k}\right)=F^{k}\right\}
$$

For each $S \subseteq[n]$, let $\mathbb{S}_{n}(S)=\left\{\tau \in \mathbb{S}_{n}: \tau(S)=S\right\}$, which is clearly a subgroup of $\mathbb{S}_{n}$. Now, if $\operatorname{Coup}(E)=K_{n}$, it follows that $F^{k}=\mathbb{S}_{n}$ and $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)=\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$. The following results establish a characterization of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$ when $\operatorname{Coup}(E) \neq K_{n}$.
Theorem 7.9. Let $\operatorname{Coup}(E)$ be connected. Aut $\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$ has a subgroup that is isomorphic to $\mathbb{S}_{n}\left(g^{k}\right) \times \operatorname{Aut}(\operatorname{Coup}(E))$. If $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is normal, then this subgroup equals $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$.
Proof. Let $\theta$ be the group homomorphism defined in (7.3). We now consider its restriction to the subgroup $\mathbb{S}_{n}\left(g^{k}\right) \times \operatorname{Aut}(\operatorname{Coup}(E))$, which we denote by $\theta_{r}$. Then its image $\theta_{r}\left(\mathbb{S}_{n}\left(g^{k}\right) \times\right.$ $\operatorname{Aut}(\operatorname{Coup}(E)))$ is clearly a subgroup of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$. Since $\theta$ is injective by Theorem 7.6, so is $\theta_{r}$, and thus $\theta_{r}\left(\mathbb{S}_{n}\left(g^{k}\right) \times \operatorname{Aut}(\operatorname{Coup}(E))\right)$ is isomorphic to $\mathbb{S}_{n}\left(g^{k}\right) \times \operatorname{Aut}(\operatorname{Coup}(E))$.

We now prove that $\theta_{r}\left(\mathbb{S}_{n}\left(g^{k}\right) \times \operatorname{Aut}(\operatorname{Coup}(E))\right)$ is a subgroup of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$. Let $a \in \mathbb{S}_{n}\left(g^{k}\right)$ and $b \in \operatorname{Aut}(\operatorname{Coup}(E))$. Then $\theta_{r}(a, b)$ is the mapping $\tau \mapsto a \tau b^{-1}$. Now, let $\tau \in F^{k}$, i.e., $\tau^{-1}\left(g^{k}\right) \in E$. Using the fact that $a\left(g^{k}\right)=g^{k}$ and $b$ maps pairs in $E$ to pairs in $E$, we obtain

$$
\left(a \tau b^{-1}\right)^{-1}\left(g^{k}\right)=\left(b \tau^{-1} a^{-1}\right)\left(g^{k}\right) \in E
$$

which implies $a \tau b^{-1} \in F^{k}$. So, $\theta_{r}(a, b) \in \operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$, from where it follows that $\theta_{r}\left(\mathbb{S}_{n}\left(g^{k}\right) \times \operatorname{Aut}(\operatorname{Coup}(E))\right)$ is a subgroup of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$.

Next, we show that if $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is normal, then it is actually the full automorphism group. It suffices to show that any element in $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$ is of the form $\theta_{r}(a, b)$ for some $a \in \mathbb{S}_{n}\left(g^{k}\right)$ and $b \in \operatorname{Aut}(\operatorname{Coup}(E))$. Let $\rho \in \operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$. By Theorem 7.6, we know that $\rho: \tau \mapsto a \tau b^{-1}$ for some $a \in \mathbb{S}_{n}, b \in \operatorname{Aut}(\operatorname{Coup}(E))$. Suppose $a \notin \mathbb{S}_{n}\left(g^{k}\right)$. Let $g^{k}$ be the pair $\{i, j\}$. Then there exist $k_{1}, k_{2}$ such that $a\left(k_{1}\right)=i$ and $a\left(k_{2}\right)=j$, with $\left\{k_{1}, k_{2}\right\} \neq\{i, j\}$. Now, we select two pairs of vertices $e \in E$ and $f \notin E$ as follows. If $\left|\left\{k_{1}, k_{2}, i, j\right\}\right|=3$, take $e$ and $f$ such that they share one vertex, otherwise take $e$ and $f$ disjoint. The only cases in which such selection is not possible, is when the subgraph induced by any three distinct vertices is a clique or for each edge in $E$ the graph resulting from deleting the edge is a clique. The only connected coupling graphs that satisfy either of these
properties are $C_{4}$ and $K_{n}$. However, by Theorem $7.8, \operatorname{Coup}(E)$ cannot be these graphs due to the normality of the Cayley graph.

Now, take any $\hat{\tau} \in \mathbb{S}_{n}$ such that

$$
\hat{\tau}(e)=\{i, j\} \quad \text { and } \quad \hat{\tau}(f)=\left\{k_{1}, k_{2}\right\} .
$$

As $\hat{\tau}^{-1}(\{i, j\})=e \in E$, it follows that $\hat{\tau} \in F^{k}$. However,

$$
\rho(\hat{\tau})^{-1}(\{i, j\})=\left(a \hat{\tau} b^{-1}\right)^{-1}(\{i, j\})=b \hat{\tau}^{-1} a^{-1}(\{i, j\})=b \hat{\tau}^{-1}\left(\left\{k_{1}, k_{2}\right\}\right)=b(f) \notin E,
$$

since $b$ maps non-edges to non-edges in $\operatorname{Coup}(E)$. We conclude that $\rho(\hat{\tau}) \notin F^{k}$, which implies that $\rho \notin \operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$. Since this is a contradiction, each automorphism in $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$ is in $\theta_{r}\left(\mathbb{S}_{n}\left(g^{k}\right) \times \operatorname{Aut}(\operatorname{Coup}(E))\right)$.

Let $G_{\text {sub }}^{k}$ denote the subgroup of $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right), F^{k}\right)$ that is isomorphic to $\mathbb{S}_{n}\left(g^{k}\right) \times$ $\operatorname{Aut}(\operatorname{Coup}(E))$. Suppose $X$ consists of one subgraph. Then, $X$ has vertex set $\{s\} \cup V^{1} \cup\{t\}$. One can verify that in that case $\operatorname{id}_{\{s\}} \times G_{\text {sub }}^{1} \times \operatorname{id}_{\{t\}}$ is a subgroup of $\operatorname{Aut}(X)$, which is the entire automorphism group in case Cay $\left(\mathbb{S}_{n}, T\right)$ is normal. Now, suppose $X$ has two subgraphs. Then, $H^{1}$ corresponds to gate $g^{1}$ and $H^{2}$ corresponds to a possibly different gate $g^{2}$. In the sequel, we study how this affects the automorphism group of $X$.

To that end, we need two intermediate results. For a set $S \subseteq[n]$, let $C\left(\mathbb{S}_{n}(S)\right)$ denote the centralizer subgroup of $\mathbb{S}_{n}(S)$ which is defined as

$$
\begin{equation*}
C\left(\mathbb{S}_{n}(S)\right)=\left\{\tau \in \mathbb{S}_{n}: \tau \pi=\pi \tau \text { for all } \pi \in \mathbb{S}_{n}(S)\right\} \tag{7.5}
\end{equation*}
$$

When $n \leq 2$, we know that $\mathbb{S}_{n}$ is abelian and thus $C\left(\mathbb{S}_{n}(S)\right)=\mathbb{S}_{n}$. Otherwise, we show that the centralizer subgroup is contained in $\mathbb{S}_{n}(S)$.

Lemma 7.10. Let $n \geq 3$. Then, we have $C\left(\mathbb{S}_{n}(S)\right) \subseteq \mathbb{S}_{n}(S)$ for all $S \subseteq[n]$.
Proof. Since $\mathbb{S}_{n}(S)=\mathbb{S}_{n}([n] \backslash S)$, we may assume that $|S| \geq 2$. Now, let $\tau \in C\left(\mathbb{S}_{n}(S)\right)$ and assume for the sake of contradiction that $\tau \notin \mathbb{S}_{n}(S)$. Then there exist distinct $i, j \in S$ such that $\tau(i) \notin S$. Now, consider the transposition $(i j)$. We have $(i j) \tau(i)=\tau(i)$, while $\tau(i j)(i)=\tau(j)$. Hence, $\tau$ and $(i j)$ do not commute, while $(i j) \in \mathbb{S}_{n}(S)$. Therefore, $\tau \notin C\left(\mathbb{S}_{n}(S)\right)$, which is a contradiction.

Exploiting Lemma 7.10, we can show the following result for general sets $F$ of the form (7.4).

Theorem 7.11. Let $i, j \in[n], n \geq 3$, and let $F=\left\{\tau \in \mathbb{S}_{n}:\left\{\tau^{-1}(i), \tau^{-1}(j)\right\} \in E\right\}$. Let $a, b \in \mathbb{S}_{n}$ and suppose that $a \tau b^{-1}=\tau$ for all $\tau \in F$. Then $a=b=\mathrm{id}$.

Proof. Observe that for all $\tau_{1}, \tau_{2} \in F$ we have:

$$
\tau_{1} b \tau_{1}^{-1}=a=\tau_{2} b \tau_{2}^{-1} .
$$

Now, let us fix an edge $e \in E$. We can write any element $\pi \in \mathbb{S}_{n}(e)$ in the form $\pi=\tau^{-1} \tau^{\prime}$ for some $\tau, \tau^{\prime} \in F$. To verify this, observe that since $e \in E$ there exist elements in $F$ that map $e$ to $\{i, j\}$. By combining two such elements $\tau$ and $\tau^{\prime}$, the composition $\tau^{-1} \tau^{\prime}$ always maps $e$ back to $e$. On the complement $[n] \backslash e$ we find all possible permutations in $F$, so we can always find $\tau, \tau^{\prime} \in F$ such that $\tau^{-1} \tau^{\prime}$ acts like $\pi$ on the set $[n] \backslash e$.

Let $\tau_{1}, \tau_{2} \in F$ be such that $\pi=\tau_{1}^{-1} \tau_{2}$. Then we know $\tau_{1} b \tau_{1}^{-1}=\tau_{2} b \tau_{2}^{-1}$, or equivalently, $\left(\tau_{1}^{-1} \tau_{2}\right)^{-1} b \tau_{1}^{-1} \tau_{2}=b$, which can be rewritten as $\pi^{-1} b \pi=b$. As $\pi \in \mathbb{S}_{n}(e)$ was chosen arbitrarily, it follows that $\pi^{-1} b \pi=b$ for all $\pi \in \mathbb{S}_{n}(e)$, and thus $b \in C\left(\mathbb{S}_{n}(e)\right)$. We now apply Lemma 7.10 with $S=e$. Since $n \geq 3$, it follows that $b \in \mathbb{S}_{n}(e)$.

By repeating this argument for all $e \in E$, it follows that $b \in \bigcap_{e \in E} \mathbb{S}_{n}(e)$. As $\operatorname{Coup}(E)$ is connected, we conclude that $b=\mathrm{id}$, from which it immediately follows that $a=\mathrm{id}$ as well.

Let $\rho \in \operatorname{Aut}(X)$ where $X$ consists of two subgraphs. The case $n \leq 2$ leads to a trivial NNCP instance. Therefore, we may assume that $n \geq 3$. Then the restriction of $\rho$ to $H^{1}$ is an element of $G_{\text {sub }}^{1}$. In particular, each $\tau^{1} \in F^{1}$ is mapped to $\rho\left(\tau^{1}\right)$ (here the superscript 1 is added to indicate that $\tau^{1}$ is a vertex of $H^{1}$ ). In order to maintain the arc structure of $D^{1}$, it follows that the restriction of $\rho$ to $H^{2}$ should not only be an element of $G_{\mathrm{sub}}^{2}$, it should also pointwise fix the elements $\rho\left(\tau^{2}\right)$ for all $\tau^{2} \in F^{1}$. Applying the result of Theorem 7.11, the restriction of $\rho$ to $H^{2}$ should be the same automorphism as the restriction to $H^{1}$. On top of that, this restriction must also be in $G_{\text {sub }}^{2}$. Thus, $\rho$ is of the form $\left(\mathrm{id}_{\{s\}}, \pi, \pi, \mathrm{id}_{\{t\}}\right)$ with $\pi \in G_{\text {sub }}^{1} \cap G_{\text {sub }}^{2}$. Extending this argument to larger $k$, let us define the following groups:

$$
\begin{align*}
G_{\text {sub }} & :=\bigcap_{k=1}^{m} G_{\text {sub }}^{k} \cong \bigcap_{k=1}^{m} \mathbb{S}_{n}\left(g^{k}\right) \times \operatorname{Aut}(\operatorname{Coup}(E))  \tag{7.6}\\
G_{X} & :=\left\{\left(\operatorname{id}_{\{s\}}, \rho, \ldots, \rho, \operatorname{id}_{\{t\}}\right) \in \operatorname{id}_{\{s\}} \times \prod_{k=1}^{m} \operatorname{Aut}\left(H^{k}\right) \times \operatorname{id}_{\{t\}}: \rho \in G_{\text {sub }}\right\} . \tag{7.7}
\end{align*}
$$

By construction, $G_{X}$ is a subgroup of $\operatorname{Aut}(X)$. If follows from the results above that it is the full automorphism group whenever $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is normal.

To get rid of of the intersection in the definition of $G_{\text {sub }}$, we exploit the notion of the gate graph $(Q, U)$ of a quantum circuit $\Gamma$, see Definition 7.3. Recall that the gate graph contains an edge between two qubits whenever there is a gate in $\Gamma$ acting on this pair of qubits. If $g^{k_{1}}$ is in $C$ with $g^{k_{1}}=\{i, j\}$, this implies that the set $\{i, j\}$ must be setwise fixed by all permutations in the group $\mathbb{S}_{n}\left(g^{k_{1}}\right)$. If also $g^{k_{2}} \in C$ with $g^{k_{2}}=\{j, \ell\}$, there is no other option than fixing $i, j$ and $\ell$ elementwise in the group intersection $\mathbb{S}_{n}\left(g^{k_{1}}\right) \cap \mathbb{S}_{n}\left(g^{k_{2}}\right)$. From this observation, we can partition all qubits in $Q$ based on whether they belong to a connected component of size one, two or at least three. This leads to the introduction of the fixing pattern of $\Gamma$.

Definition 7.12. Let $\Gamma=(Q, C)$ be a quantum circuit on $n$ qubits. We define the fixing pattern of $\Gamma$ as the partition $\mathcal{F}:=\left\{S_{1}, \ldots, S_{l}\right\}$ of $Q$ such that each $S_{i}$ is either:

- a single qubit contained in a connected component of the gate graph $(Q, U)$ of size at least 3;
- a pair of qubits $\{i, j\}$ that forms a connected component in the gate graph $(Q, U)$;
- the set of all singletons in the gate graph $(Q, U)$, which we denote by the free set in $\mathcal{F}$.

Moreover, we define $f$ to be size of the free set in $\mathcal{F}, p$ to be the number of pairs in $\mathcal{F}$ and $c(=n-2 p-f)$ to be the number of qubits in a connected component of size at least 3 in $(Q, U)$.

Observe that $\mathcal{F}$ can be easily constructed by a scan of the connected components of $(Q, U)$. The extreme cases are $\mathcal{F}=\{Q\}$ if $\Gamma$ contains no gates, whereas $\mathcal{F}=\{\{1\}, \ldots,\{n\}\}$ if $(Q, U)$ is connected.

The group $\cap_{k=1}^{m} \mathbb{S}_{n}\left(g^{k}\right)$ consists of all permutations that setwise fix the elements in $\mathcal{F}$. To simplify notation, we define

$$
\mathbb{S}_{n}(\mathcal{F}):=\left\{a \in \mathbb{S}_{n}: a\left(S_{i}\right)=S_{i} \text { for all } i \in[l]\right\} .
$$

It is not difficult to determine the order of $\mathbb{S}_{n}(\mathcal{F})$.
Theorem 7.13. $\left|\mathbb{S}_{n}(\mathcal{F})\right|=2^{p} f!$.
Proof. Each qubit belongs to exactly one set in $\mathcal{F}$. If this set is not a pair or the free set, the qubit must be mapped to itself by all permutations in $\mathbb{S}_{n}(\mathcal{F})$. For all pairs $\{i, j\} \in \mathcal{F}$, there are exactly two possibilities, namely mapping $i$ to $i$ and $j$ to $j$, or mapping $i$ to $j$ and $j$ to $i$. Since we have $p$ pairs, this leads to $2^{p}$ possibilities. Finally, the qubits in the free set can be permuted freely between themselves, leading to another $f$ ! possibilities. The total order then becomes $2^{p} f!$.

By construction, we know that $G_{\text {sub }} \cong \mathbb{S}_{n}(\mathcal{F}) \times \operatorname{Aut}(\operatorname{Coup}(E))$, from which it also follows that

$$
\begin{equation*}
G_{X} \cong \mathbb{S}_{n}(\mathcal{F}) \times \operatorname{Aut}(\operatorname{Coup}(E)) \tag{7.8}
\end{equation*}
$$

Recall that $G_{X}=\operatorname{Aut}(X)$ when $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is normal.

### 7.3.3 Orbit and orbital structure of group action on $X$

The elements of $G_{X}$ act on the vertices and arcs of $X$. In this section we study this group action in terms of its induced orbit and orbital structure, which will become of key importance in the symmetry reduction explained in Section 7.4.

Each automorphism in $G_{X}$ maps the vertex set of $X$ to itself. Given a vertex $\tau \in V$, the orbit of $\tau$ is the set of vertices to which $\tau$ is mapped by the elements in $G_{X}$, i.e., all vertices $\rho(\tau)$ with $\rho \in G_{X}$. The set of orbits forms a partition of $V$, which is written as the quotient $V / G_{X}$.

Similarly, $G_{X}$ acts on the arc set $A$ by $\rho\left(\left(\tau_{1}, \tau_{2}\right)\right)=\left(\rho\left(\tau_{1}\right), \rho\left(\tau_{2}\right)\right)$ for all $\rho \in G_{X}$. We denote the set of orbitals by $A / G_{X}$. Note that arcs in the same orbital have their initial vertices in the same orbit. It is therefore natural to first understand the orbit structure of the action of $G_{X}$ on $V$.

Let $\operatorname{Orb}(\tau)$ denote the orbit of vertex $\tau \in V$. It follows from the construction of $G_{X}$ that $\operatorname{Orb}(s)=\{s\}$ and $\operatorname{Orb}(t)=\{t\}$. Moreover, the subgraphs $H^{k}, k \in[m]$, are invariant under the action of $G_{X}$ on $X$. For that reason, we can restrict ourselves to identifying the orbits within each subgraph $H^{k}$ under the action of $G_{\text {sub }}$. Since all subgraphs are identical, this provides the orbit description for the entire graph $G_{X}$.

Similar as before, we use $\tau$ to denote a vertex, as each vertex represents a qubit order in $\mathbb{S}_{n}$. For all $k \in[m]$ and all $\tau \in V^{k}$, we obtain

$$
\begin{equation*}
\operatorname{Orb}(\tau)=\left\{\rho(\tau): \rho \in G_{\text {sub }}\right\}=\left\{a \tau b^{-1}: a \in \mathbb{S}_{n}(\mathcal{F}), b \in \operatorname{Aut}(\operatorname{Coup}(E))\right\} \tag{7.9}
\end{equation*}
$$

We also define the stabilizer subgroup with respect to $\tau$ under the action of $G_{\text {sub }}$ as

$$
\begin{align*}
\operatorname{Stab}(\tau) & :=\left\{\rho \in G_{\text {sub }}: \rho(\tau)=\tau\right\} \\
& \cong\left\{(a, b) \in \mathbb{S}_{n}(\mathcal{F}) \times \operatorname{Aut}(\operatorname{Coup}(E)): a \tau b^{-1}=\tau\right\} \tag{7.10}
\end{align*}
$$

The condition given in (7.10) for $(a, b)$ to act as a stabilizer can be rewritten as $a=\tau b \tau^{-1}$. Thus, a pair $(a, b) \in \mathbb{S}_{n}(\mathcal{F}) \times \operatorname{Aut}(\operatorname{Coup}(E))$ corresponds to an element in $\operatorname{Stab}(\tau)$ if and only if the permutation $\tau b \tau^{-1}$ is in $\mathbb{S}_{n}(\mathcal{F})$ and $a=\tau b \tau^{-1}$. This implies that for all $S_{i} \in \mathcal{F}$ we must have $\tau b \tau^{-1}\left(S_{i}\right)=S_{i}$, or equivalently, $b\left(\tau^{-1}\left(S_{i}\right)\right)=\tau^{-1}\left(S_{i}\right)$. Hence, $b$ setwise fixes the inverse fixing pattern in $\mathcal{F}$ with respect to $\tau$. Let us define the subgroup $B_{\tau}$ of $\operatorname{Aut}(\operatorname{Coup}(E))$ that consists of all such elements, i.e.,

$$
\begin{equation*}
B_{\tau}:=\left\{b \in \operatorname{Aut}(\operatorname{Coup}(E)): b\left(\tau^{-1}\left(S_{i}\right)\right)=\tau^{-1}\left(S_{i}\right) \quad \forall i \in[l]\right\} . \tag{7.11}
\end{equation*}
$$

Observe that for each $b \in B_{\tau}$, there exists exactly one element $a \in \mathbb{S}_{n}(\mathcal{F})$ such that $a \tau b^{-1}=\tau$, namely $a=\tau b \tau^{-1}$. Therefore,

$$
\begin{equation*}
\operatorname{Stab}(\tau) \cong\left\{(a, b): b \in B_{\tau}, a=\tau b \tau^{-1}\right\} \tag{7.12}
\end{equation*}
$$

in particular, we have $|\operatorname{Stab}(\tau)|=\left|B_{\tau}\right|$.
As $B_{\tau}$ is a subgroup of $\operatorname{Aut}(\operatorname{Coup}(E))$, it acts on the edge set of $\operatorname{Coup}(E)$. The orbital of an edge $\{i, j\} \in E$ under this group action is the set of all edges $\{b(i), b(j)\}$ with $b \in B_{\tau}$. We denote by the quotient $E / B_{\tau}$ the set of orbitals under this group action.

We can show that if $\tau_{1}$ and $\tau_{2}$ belong to the same orbit, then the subgroups $B_{\tau_{1}}$ and $B_{\tau_{2}}$ are conjugate subgroups. Moreover, the quotients of their actions on $E$ have the same cardinality.

Lemma 7.14. Let $\tau_{1}$ and $\tau_{2}$ be two qubit orders with $\tau_{2}=a \tau_{1} b^{-1}$ for some a $\in \mathbb{S}_{n}(\mathcal{F})$ and $b \in \operatorname{Aut}(\operatorname{Coup}(E))$. Then,
(i) $B_{\tau_{2}}=b B_{\tau_{1}} b^{-1}$;
(ii) there exists a bijection from $E / B_{\tau_{1}}$ to $E / B_{\tau_{2}}$ given by left multiplication with $b$.

Proof. (i) Exploiting the fact that $a^{-1}\left(S_{i}\right)=S_{i}$ for all $i \in[l]$, we obtain

$$
\begin{aligned}
B_{\tau_{2}} & =\left\{b_{2} \in \operatorname{Aut}(\operatorname{Coup}(E)): b_{2}\left(\tau_{2}^{-1}\left(S_{i}\right)\right)=\tau_{2}^{-1}\left(S_{i}\right) \quad \forall i \in[l]\right\} \\
& =\left\{b_{2} \in \operatorname{Aut}(\operatorname{Coup}(E)): b_{2}\left(\left(a \tau_{1} b^{-1}\right)^{-1}\left(S_{i}\right)\right)=\left(a \tau_{1} b^{-1}\right)^{-1}\left(S_{i}\right) \quad \forall i \in[l]\right\} \\
& =\left\{b_{2} \in \operatorname{Aut}(\operatorname{Coup}(E)): b_{2} b \tau_{1}^{-1} a^{-1}\left(S_{i}\right)=b \tau_{1}^{-1} a^{-1}\left(S_{i}\right) \quad \forall i \in[l]\right\} \\
& =\left\{b_{2} \in \operatorname{Aut}(\operatorname{Coup}(E)): b^{-1} b_{2} b\left(\tau_{1}^{-1}\left(S_{i}\right)\right)=\tau_{1}^{-1}\left(S_{i}\right) \quad \forall i \in[l]\right\} \\
& =\left\{b b_{1} b^{-1} \in \operatorname{Aut}(\operatorname{Coup}(E)): b_{1}\left(\tau_{1}^{-1}\left(S_{i}\right)\right)=\tau_{1}^{-1}\left(S_{i}\right) \quad \forall i \in[l]\right\} \\
& =b B_{\tau_{1}} b^{-1} .
\end{aligned}
$$

(ii) This fact follows directly from (i), by observing that

$$
b \operatorname{Orb}_{B_{\tau_{1}}}(i)=\left\{b b_{1} b^{-1}(b(i)): b_{1} \in B_{\tau_{1}}\right\}=\left\{b_{2}(b(i)): b_{2} \in B_{\tau_{2}}\right\}=\operatorname{Orb}_{B_{\tau_{2}}}(b(i))
$$

One easily verifies that left multiplication by $b$ gives a bijection.

As a consequence of the well-known orbit-stabilizer theorem, we establish the following relation between $\operatorname{Orb}(\tau)$ and $\operatorname{Stab}(\tau)$ :

$$
\begin{equation*}
|\operatorname{Orb}(\tau)|=\frac{\left|G_{\text {sub }}\right|}{|\operatorname{Stab}(\tau)|}=\frac{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}{\left|B_{\tau}\right|} . \tag{7.13}
\end{equation*}
$$

Of course, $\operatorname{Orb}(\tau)$ does not depend on the particular choice of the representative $\tau$ in the orbit. Indeed, it follows from Lemma 7.14 that the orders of the subgroups $B_{\tau}$ for all $\tau$ within an orbit are the same.

To increase our understanding of $\operatorname{Orb}(\tau)$, we rewrite (7.9) as follows:

$$
\begin{equation*}
\operatorname{Orb}(\tau)=\mathbb{S}_{n}(\mathcal{F}) \tau \operatorname{Aut}(\operatorname{Coup}(E))=\bigcup_{\tilde{\tau} \in \mathbb{S}_{n}(\mathcal{F}) \tau} \tilde{\tau} \operatorname{Aut}(\operatorname{Coup}(E)) \tag{7.14}
\end{equation*}
$$

In other words, if $\mathbb{S}_{n}(\mathcal{F})$ is trivial, then the orbit partition of $V^{k}$ is given by the left cosets of $\operatorname{Aut}(\operatorname{Coup}(E))$ in $G_{\text {sub }}$. Otherwise, each orbit is the union of several left cosets of $\operatorname{Aut}(\operatorname{Coup}(E))$ in $G_{\text {sub }}$, where the union is determined by the elements in the right cosets of $\mathbb{S}_{n}(\mathcal{F})$ in $G_{\text {sub }}$.

It follows from the discussion above that the size of an orbit containing $\tau$ increases when the subgroups $\mathbb{S}_{n}(\mathcal{F})$ and $\operatorname{Aut}(\operatorname{Coup}(E))$ increase, while it decreases with the order of $B_{\tau}$. Of particular importance in the symmetry reduction is the number of orbits in each subgraph. We let $V^{k} / G_{X}$ denote the set of orbits of vertices in $V^{k}$ under the action of $G_{X}$, although we formally refer to the action of $G_{X}$ restricted to $V^{k}$. We allow for this slight abuse of notation, in order to simplify the terminology in Section 7.4.
Theorem 7.15. The number of orbits of $V^{k}$ under $G_{X}$ is $\left|V^{k} / G_{X}\right|=\frac{\sum_{\tau \in \mathbb{S}_{n}}\left|B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}$.
Proof. Let $\left(V^{k}\right)^{\rho}$ denote the set of vertices in $V^{k}$ that are (pointwise) fixed by $\rho \in G_{X}$. Then, Burnside's lemma implies that

$$
\left|V^{k} / G_{X}\right|=\frac{\sum_{\rho \in G_{X}}\left|\left(V^{k}\right)^{\rho}\right|}{\left|G_{X}\right|}
$$

The sum in the numerator counts for every group element the number of vertices that are fixed. Alternatively, we can also sum over all vertices and count the number of group elements that stabilize the vertex. This leads to

$$
\left|V^{k} / G_{X}\right|=\frac{\sum_{\tau \in \mathbb{S}_{n}}|\operatorname{Stab}(\tau)|}{\left|\mathbb{S}_{n}(\mathcal{F}) \times \operatorname{Aut}(\operatorname{Coup}(E))\right|}=\frac{\sum_{\tau \in \mathbb{S}_{n}}\left|B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}
$$

We now shift our focus to the analysis of the orbital structure of the arcs of $X$ under the action of $G_{X}$. Recall that $A$ consists of two types of arcs: arcs within a subgraph (the sets $A^{k}, k \in[m]$ ) and the arcs between the subgraphs (the sets $D^{k}, k \in\{0\} \cup[m]$ ). Since the sets $A^{1}, \ldots, A^{k}$ are identical and each set is invariant under the group action $G_{X}$, we can restrict our focus to the action of $G_{\text {sub }}$ on a single subgraph. The orbital of an $\operatorname{arc}(\tau, \tau \sigma) \in A^{k}$ corresponding to transposition $\sigma=(i j) \in T$ is given by

$$
\operatorname{Orb}((\tau, \tau \sigma)):=\left\{(\rho(\tau), \rho(\tau \sigma)): \rho \in G_{\text {sub }}\right\}
$$



Figure 7.2: Graphical overview of orbital structure within a subgraph $H^{k}$. Each line type (solid, dotted, dashed and curled) corresponds to another orbital. Case I (left): $B_{\tau_{1}}$ is trivial. Case II (middle): $B_{\tau_{1}}$ is nontrivial and the orbital of $\sigma_{1}$ under $B_{\tau_{1}}$ contains $\sigma_{2}$. Case III (right): $B_{\tau_{1}}$ is nontrivial, but the orbital of $\sigma_{1}$ under $B_{\tau_{1}}$ only consists of $\sigma_{1}$.

$$
\begin{aligned}
& =\left\{\left(a \tau b^{-1}, a \tau \sigma b^{-1}: a \in \mathbb{S}_{n}(\mathcal{F}), b \in \operatorname{Aut}(\operatorname{Coup}(E))\right\}\right. \\
& =\left\{\left(a \tau b^{-1}, a \tau b^{-1}(b(i) b(j)): a \in \mathbb{S}_{n}(\mathcal{F}), b \in \operatorname{Aut}(\operatorname{Coup}(E))\right\}\right.
\end{aligned}
$$

where the last line follows from the fact that $b(i j) b^{-1}=(b(i) b(j))$. This expression of $\operatorname{Orb}((\tau, \tau \sigma))$ implies that all arcs within the same orbital start at vertices within the same orbit and end at vertices within the same orbit (where the start- and end-orbits can differ). Moreover, the transpositions to which the arcs in $\operatorname{Orb}((\tau, \tau \sigma))$ correspond are related via the elements of $\operatorname{Aut}(\operatorname{Coup}(E))$. It follows from above that all such transpositions in $\operatorname{Orb}((\tau, \tau \sigma))$ belong to the same orbital in the coupling graph $\operatorname{Coup}(E)$, namely the orbital of the edge corresponding to $\sigma$.

When the group $B_{\tau}$ is trivial, the stabilizer subgroup of $\tau$ in $G_{\text {sub }}$ is trivial. This implies that the arcs within the same orbital all start at different vertices, and consequently, the orbital has cardinality $\left|\mathbb{S}_{n}(\mathcal{F})\right| \cdot|\operatorname{Aut}(\operatorname{Coup}(E))|$. When $B_{\tau}$ is not trivial, there exist nontrivial $(a, b)$ in $\operatorname{Stab}(\tau)$, see (7.12). If such $b$ maps the edge corresponding to $\sigma$ in $\operatorname{Coup}(E)$ to a different edge, then there exist multiple distinct arcs in $\operatorname{Orb}((\tau, \tau \sigma))$ that start from the same vertex. However, if $b$ maps this edge to itself, then the stabilizer subgroup of $(\tau, \tau \sigma)$ is nontrivial and the orbital has a smaller cardinality. Figure 7.2 provides a graphical overview of these three cases.

The observations above lead to the following result regarding the cardinality of the set of orbitals of $A^{k}$ under the action of $G_{X}$ restricted to $A^{k}$. By slight abuse of notation, we again denote this set by the quotient $A^{k} / G_{X}$.
Theorem 7.16. The number of orbitals of $A^{k}$ under $G_{X}$ is $\left|A^{k} / G_{X}\right|=\frac{\sum_{\tau \in s_{n}}\left|B_{\tau}\right| \cdot\left|E / B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}$.
Proof. Since the arcs belonging to an orbital all start from vertices in the same orbit, we restrict ourselves to the orbitals starting from a single orbit. We then enumerate over all orbits and count the number of orbitals starting from vertices belonging to each orbit.

Let $\tau \in \mathbb{S}_{n}$ and let $\delta^{+}\left(\tau, A^{k}\right)=\{(\tau, \tau \sigma): \sigma \in T\}$. Each automorphism in $G_{X}$ maps the arcs in $\delta^{+}\left(\tau, A^{k}\right)$ to the arcs in $\delta^{+}\left(\tau^{\prime}, A^{k}\right)$ for some $\tau^{\prime} \in \operatorname{Orb}(\tau)$. Therefore, to count the number of orbitals starting from $\operatorname{Orb}(\tau)$, it suffices to consider to how many distinct orbitals the arcs in $\delta^{+}\left(\tau, A^{k}\right)$ belong. In the extreme case these arcs all belong to distinct orbitals, leading to a total number of $|E|$ orbitals. Two $\operatorname{arcs}$ in $\delta^{+}\left(\tau, A^{k}\right)$, say $\left(\tau, \tau \sigma_{1}\right)$
and $\left(\tau, \tau \sigma_{2}\right)$ with $\sigma_{1}(:=(i j)), \sigma_{2} \in T$, belong to the same orbital if and only if there exists a mapping $\rho \in G_{X}$ such that $\tau=\rho(\tau)$ and $\tau \sigma_{2}=\rho\left(\tau \sigma_{1}\right)$. Equivalently, if there exist $a \in \mathbb{S}_{n}(\mathcal{F})$ and $b \in \operatorname{Aut}(\operatorname{Coup}(E))$ such that
$\left\{\begin{array}{l}\tau=a \tau b^{-1} \\ \tau \sigma_{2}=a \tau \sigma_{1} b^{-1}\end{array} \Longleftrightarrow\left\{\begin{array}{l}b \in B_{\tau} \text { and } a=\tau b \tau^{-1} \\ \tau \sigma_{2}=a \tau b^{-1}(b(i) b(j))\end{array} \Longleftrightarrow\left\{\begin{array}{l}b \in B_{\tau} \text { and } a=\tau b \tau^{-1} \\ \sigma_{2}=(b(i) b(j))\end{array}\right.\right.\right.$
where we used that $b(i j) b^{-1}=(b(i) b(j))$. Thus, any arc starting from $\tau$ that is induced by a transposition of the form $(b(i) b(j))$ with $b \in B_{\tau}$ is in the same orbital as arc $(\tau, \tau(i j))$. The orbital partition of $\delta^{+}\left(\tau, A^{k}\right)$ is therefore corresponding to the orbit partition implied by the group action of $B_{\tau}$ on the edge set $E$ of the coupling graph. The number of distinct orbitals in $\delta^{+}\left(\tau, A^{k}\right)$ equals $\left|E / B_{\tau}\right|$. Let $\tau^{\prime} \in \operatorname{Orb}(\tau)$ be any other vertex in $\operatorname{Orb}(\tau)$. Since the arcs in $\delta^{+}\left(\tau, A^{k}\right)$ are mapped to arcs in $\delta^{+}\left(\tau^{\prime}, A^{k}\right)$, it then follows that the number of distinct orbitals in $\delta^{+}\left(\tau^{\prime}, A^{k}\right)$ is at least $\left|E / B_{\tau}\right|$. From the same analysis, it follows that this cardinality must be $\left|E / B_{\tau^{\prime}}\right|$, which equals $\left|E / B_{\tau}\right|$ by Lemma 7.14. Thus, the number of distinct orbitals starting from vertices in $\operatorname{Orb}(\tau)$ is $\left|E / B_{\tau}\right|$, where the choice of $\tau$ in the orbit is arbitrary.

We now obtain the total number of orbitals by adding the quantities $\left|E / B_{\tau}\right|$ over all $\operatorname{Orb}(\tau) \in V^{k} / G_{X}:$

$$
\begin{aligned}
\left|A^{k} / G_{X}\right| & =\sum_{\operatorname{Orb}(\tau) \in V^{k} / G_{X}}\left|E / B_{\tau}\right| \\
& =\sum_{\operatorname{Orb}(\tau) \in V^{k} / G_{X}} \frac{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}{\left|B_{\tau}\right|} \cdot \frac{\left|B_{\tau}\right| \cdot\left|E / B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|} \\
& =\sum_{\operatorname{Orb}(\tau) \in V^{k} / G_{X}}|\operatorname{Orb}(\tau)| \cdot \frac{\left|B_{\tau}\right| \cdot\left|E / B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|} \\
& =\frac{\sum_{\tau \in \mathbb{S}_{n}}\left|B_{\tau}\right| \cdot\left|E / B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|} .
\end{aligned}
$$

In the third equality we used (7.13), as well as the fact that the sum of $|\operatorname{Orb}(\tau)| \cdot\left|B_{\tau}\right| \cdot\left|E / B_{\tau}\right|$ over all orbits equals the sum of $\left|B_{\tau}\right| \cdot\left|E / B_{\tau}\right|$ over all vertices, since $\left|B_{\tau}\right| \cdot\left|E / B_{\tau}\right|$ is constant for all $\tau$ within an orbit, see Lemma 7.14.

To study the orbital representation of $D^{k}$ under the action of $G_{X}$, we distinguish between the case $k=0$ and $k \in[m]$. For $k=0, D^{k}$ contains all arcs between $s$ and $V^{1}$. Therefore, each orbital of $D^{0}$ under $G_{X}$ consists of all arcs starting from $s$ and ending at vertices in an orbit of $V^{1}$. The arcs in $D^{k}, k \in[m]$, correspond to ordered pairs $\left(\tau^{k}, \tau^{k+1}\right)$, where $\tau$ represents the same qubit order in $H^{k}$ and $H^{k+1}$. Such an arc exists in $D^{k}$ whenever $\tau^{k} \in F^{k}$, see (7.4). The orbital of $\left(\tau^{k}, \tau^{k+1}\right)$ is the set

$$
\begin{aligned}
\operatorname{Orb}\left(\left(\tau^{k}, \tau^{k+1}\right)\right) & =\left\{\left(\rho\left(\tau^{k}\right), \rho\left(\tau^{k+1}\right): \rho \in G_{\text {sub }}\right\}\right. \\
& =\left\{\left(a \tau^{k} b^{-1}, a \tau^{k+1} b^{-1}\right): a \in \mathbb{S}_{n}(\mathcal{F}), b \in \operatorname{Aut}(\operatorname{Coup}(E))\right\}
\end{aligned}
$$

Let $D^{k} / G_{X}$ denote the set of orbitals of the group action of $G_{X}$ restricted to $D^{k}$. Since $\tau^{k}$ and $\tau^{k+1}$ represent the same qubit orders in $H^{k}$ and $H^{k+1}$, respectively, all arcs within $\operatorname{Orb}\left(\left(\tau^{k}, \tau^{k+1}\right)\right)$ start and end at vertices in the same orbit. This leads to the following result.

Theorem 7.17. The number of orbitals of $D^{0}$ under $G_{X}$ is $\left|D^{0} / G_{X}\right|=\frac{\sum_{\tau \in s_{n}}\left|B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}$. For $k \neq 0$, the number of orbitals of $D^{k}$ under $G_{X}$ is $\left|D^{k} / G_{X}\right|=\frac{\sum_{\tau \in F^{k}}\left|B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}$.

Proof. The first part follows directly from Theorem 7.15. For the second part, observe that we have $D^{k}=\left\{\left(\tau^{k}, \tau^{k+1}\right): \tau^{k} \in V^{k}, \tau^{k+1} \in V^{k+1}, \tau^{k} \in F^{k}\right\}$, where $F^{k}$ is defined in (7.4). The cardinality of $D^{k} / G_{X}$ is equal to the number of orbits of $F^{k}$ under the action of $G_{X}$ restricted to the vertices in $F^{k}$. Observe that this restriction is well-defined, since each vertex in $F^{k}$ is always mapped to another vertex in $F^{k}$. The cardinality of the quotient $F^{k} / G_{X}$ can be derived in the same way as done in the proof of Theorem 7.15 , which leads to

$$
\left|D^{k} / G_{X}\right|=\left|F^{k} / G_{X}\right|=\frac{\sum_{\tau \in F^{k}}\left|B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}
$$

The results of Theorems 7.15, 7.16 and 7.17 are summarized in Table 7.1. Moreover, we simplify the cardinalities of the quotients for the special case where $B_{\tau}$ is trivial for all $\tau \in \mathbb{S}_{n}$.

| Quotient | Order | Order when $B_{\tau}$ is <br> trivial for all $\tau \in \mathbb{S}_{n}$ |
| :---: | :---: | :---: |
| $V^{k} / G_{X}, k \in[m]$ | $\frac{\sum_{\tau \in \mathrm{s}_{n}}\left\|B_{\tau}\right\|}{2^{p} f!\cdot\|\operatorname{Aut}(\operatorname{Coup}(E))\|}$ | $\frac{n!}{2^{p} f!\cdot\|\operatorname{Aut}(\operatorname{Coup}(E))\|}$ |
| $A^{k} / G_{X}, k \in[m]$ | $\frac{\sum_{\tau \in \mathrm{S}_{n}}\left\|B_{\tau}\right\| \cdot\left\|E / B_{\tau}\right\|}{2^{p} f!\cdot\|\operatorname{Aut}(\operatorname{Coup}(E))\|}$ | $\frac{n!\cdot\|E\|}{2^{p} f!\cdot\|\operatorname{Aut}(\operatorname{Coup}(E))\|}$ |
| $D^{0} / G_{X}$ | $\frac{\sum_{\tau \in \mathrm{S}_{n}}\left\|B_{\tau}\right\|}{2^{p} f!\cdot\|\operatorname{Aut}(\operatorname{Coup}(E))\|}$ | $\frac{n!}{2^{p} f!\cdot\|\operatorname{Aut}(\operatorname{Coup}(E))\|}$ |
| $D^{k} / G_{X}, k \in[m]$ | $\frac{\sum_{\tau \in F^{k}}\left\|B_{\tau}\right\|}{2^{p} f!\cdot\|\operatorname{Aut}(\operatorname{Coup}(E))\|}$ | $\frac{2\|E\|(n-2)!}{2^{p} f!\cdot\|\operatorname{Aut}(\operatorname{Coup}(E))\|}$ |

Table 7.1: Overview of the orders of quotients $V^{k} / G_{X}, A^{k} / G_{X}$ and $D^{k} / G_{X}$ in terms of the cardinality of $B_{\tau}$.

In practical situations, it is often appropriate to possess an orbit (resp. orbital) representation of some set under a group action. Such representation contains exactly one element from each orbit (resp. orbital). In the sequel, we let $\mathcal{R}\left(V^{k} / G_{X}\right) \subseteq V^{k}$ denote an orbit representation of $\mathbb{S}_{n}$ under the group action $G_{\text {sub }}$. We can obtain $\mathcal{R}\left(V^{k} / G_{X}\right)$ by exploiting (7.14). First, one can efficiently obtain a representation of left cosets of $\operatorname{Aut}(\operatorname{Coup}(E))$ in $\mathbb{S}_{n}$, see e.g., Dixon and Majeed [103]. This coset representation can be compressed to an orbit representation by a merge operation of multiple left cosets. For each representative $\tau$, we enumerate the elements of the right coset of $\mathbb{S}_{n}(\mathcal{F})$ containing $\tau$. This provides the representatives of left cosets that belong to the same orbit.

An orbital representation $\mathcal{R}\left(A^{k} / G_{X}\right)$ can be obtained by exploiting the proof of Theorem 7.16. We know that each orbital can be represented by the orbit from where the arcs in the orbital start, combined with a representative element from the quotient $E / B_{\tau}$, where $\tau$
belongs to the orbit. Hence, $\mathcal{R}\left(A^{k} / G_{X}\right)=\left\{(\tau, \sigma): \tau \in \mathcal{R}\left(V^{k} / G_{X}\right), \sigma \in \mathcal{R}\left(E / B_{\tau}\right)\right\}$, where $\mathcal{R}\left(E / B_{\tau}\right)$ is an orbital representation of the edges in $E$ under the action of $B_{\tau}$. As the coupling graph is typically small, $\mathcal{R}\left(E / B_{\tau}\right)$ can be obtained by enumeration.

Finally the orbital representation $\mathcal{R}\left(D^{k} / G_{X}\right)$ follows from the subset of $\mathcal{R}\left(V^{k} / G_{X}\right)$ associated with the orbits in $F^{k}$, i.e., $\mathcal{R}\left(D^{k} / G_{X}\right)=\left\{\left(\tau^{k}, \tau^{k+1}\right): \tau^{k} \in \mathcal{R}\left(V^{k} / G_{X}\right), \tau^{k} \in F^{k}\right\}$.

The orbit and orbital representations can be found more easily when the underlying coupling graph is known, see Section 7.5.

### 7.4 Symmetry reduction for the NNCP

In this section we show how the automorphism results derived in Section 7.3 can be exploited to reduce the size of the NNCP introduced in Section 7.2.2.

In Section 7.4.1 we exploit the subgroup $G_{X}$, see (7.8), in order to reduce the linear programming formulation (SPP) in terms of the number of variables and constraints. In Section 7.4.2 we show how this reduced LP can be rewritten as a new and equivalent shortest path instance.

### 7.4.1 Reduced LP formulation

The elements in $G_{X}$ act on the vertex set $V$ of $G$. In a similar way, $G_{X}$ also acts on the arc set $A$ of $G$. For any arc $e \in A$ and any $\rho \in G_{X}$, let $\rho(e)$ denote the ordered pair to which $e$ is mapped to by $\rho$, which is again in $A$ since $\rho$ is an automorphism. Now, let $x \in \prod_{k=1}^{m} \mathbb{R}^{A^{k}}$ and $y \in \prod_{k=0}^{m} \mathbb{R}^{D^{k}}$ be feasible for (SPP). We define the Reynolds operator $\psi$ that maps $x$ (resp. $y$ ) to the average of the images of $x$ (resp. $y$ ) under the action of $G_{X}$ on $A$. That is,

$$
\begin{equation*}
\psi(x):=\frac{1}{\left|G_{X}\right|} \sum_{\rho \in G_{X}} x^{\rho} \quad \text { and } \quad \psi(y):=\frac{1}{\left|G_{X}\right|} \sum_{\rho \in G_{X}} y^{\rho}, \tag{7.15}
\end{equation*}
$$

where $x^{\rho}$ and $y^{\rho}$ are defined as $x_{e}^{\rho}=x_{\rho(e)}$ and $y_{e}^{\rho}=y_{\rho(e)}$ for all arcs $e$. As $A^{k}$ for all $k \in[m]$ and $D^{k}$ for all $k \in\{0\} \cup[m]$ are invariant under the action of $G_{X}$ on $A$, it follows that $\psi(x) \in \prod_{k=1}^{m} \mathbb{R}^{A^{k}}$ and $\psi(y) \in \prod_{k=0}^{m} \mathbb{R}^{D^{k}}$. We can now prove the following result, which was proven for general linear programs by Bödi et al. [49].

Theorem 7.18. Let $(x, y) \in \prod_{k=1}^{m} \mathbb{R}^{A^{k}} \times \prod_{k=0}^{m} \mathbb{R}^{D^{k}}$ be feasible (resp. optimal) for (SPP). Then, $(\psi(x), \psi(y))$ is also feasible (resp. optimal) for (SPP).

Proof. As the flow conservation constraints hold for $(x, y)$ and $\rho$ preserves the arc structure of $X$, the pair $\left(x^{\rho}, y^{\rho}\right)$ also satisfies these constraints for all $\rho \in G_{X}$. It follows that $\left(x^{\rho}, y^{\rho}\right)$ is feasible for (SPP) for all $\rho \in G_{X}$. Observe that the pair $(\psi(x), \psi(y))$ is a convex combination of $\left(x^{\rho}, y^{\rho}\right)$ over the elements of $G_{X}$. Because the feasible set of (SPP) is convex, it follows that $(\psi(x), \psi(y))$ is also feasible for (SPP).

The objective function of (SPP) can be written as $f(x, y):=\sum_{e \in A} x_{e}$. Since arcs are mapped to arcs by all $\rho \in G_{X}$, we have $f\left(x^{\rho}, y^{\rho}\right)=f(x, y)$. We then obtain:

$$
f(\psi(x), \psi(y))=\sum_{e \in A} \psi(x)_{e}=\frac{1}{\left|G_{X}\right|} \sum_{\rho \in G_{X}} \sum_{e \in A} x_{e}^{\rho}=\frac{1}{\left|G_{X}\right|}\left|G_{X}\right| \sum_{e \in A} x_{e}=f(x, y) .
$$

Thus, if $(x, y)$ is optimal for (SPP), then so is $(\psi(x), \psi(y))$.

An implication of Theorem 7.18 is that we may restrict the feasible set of (SPP) to the subspace

$$
\begin{equation*}
\mathcal{H}_{G_{X}}:=\left\{(\psi(x), \psi(y)):(x, y) \in \prod_{k=1}^{m} \mathbb{R}^{A^{k}} \times \prod_{k=0}^{m} \mathbb{R}^{D^{k}}\right\} \tag{7.16}
\end{equation*}
$$

which is also denoted as the fixed point subspace in [49]. By construction of the Reynolds operator (7.15), the entries in $\psi(x)$ belonging to the same orbital are equal. Therefore, the subspace $\mathcal{H}_{G_{X}}$ is spanned by the incidence vectors of orbitals of $X$. In Section 7.3.3 we derived the orbital structure of the action of $G_{X}$ on $X$. Recall that $A^{k} / G_{X}$ denotes (the index set of) the collection of orbitals of $A^{k}$ under the action of $G_{X}$. Now, if we denote the $i$ th orbital of $A^{k}$ by $W_{i}^{k}$, we obtain

$$
\begin{equation*}
A^{k}=\bigsqcup_{i \in A^{k} / G_{X}} W_{i}^{k} \quad \text { for all } k \in[m] \tag{7.17}
\end{equation*}
$$

In a similar fashion, the arc sets $D^{k}, k \in\{0\} \cup[m]$ can be partitioned into its collection of orbitals. If $Z_{i}^{k}$ denotes the $i$ th orbital of $D^{k}$, then

$$
\begin{equation*}
D^{k}=\bigsqcup_{i \in D^{k} / G_{X}} Z_{i}^{k} \quad \text { for all } k \in\{0\} \cup[m] . \tag{7.18}
\end{equation*}
$$

Now, the subspace $\mathcal{H}_{G_{X}}$ can be rewritten as:

$$
\begin{equation*}
\mathcal{H}_{G_{X}}=\prod_{k=1}^{m}\left(\operatorname{Span}\left\{\mathbb{1}_{W_{i}^{k}}: i \in A^{k} / G_{X}\right\}\right) \times \prod_{k=0}^{m}\left(\operatorname{Span}\left\{\mathbb{1}_{Z_{i}^{k}}: i \in D^{k} / G_{X}\right\}\right) \tag{7.19}
\end{equation*}
$$

which implies that the characteristic vectors of the orbitals form a basis for $\mathcal{H}_{G_{X}}$.
Also the orbits of each of the vertex sets $V^{k}$ under the action of $G_{X}$ induce a partition of $V^{k}$. Let $V^{k} / G_{X}$ denote (the index set of) the collection of orbits of $V^{k}$ under $G_{X}$. The $u$ th orbit of $V^{k}$ is denoted by $O_{u}^{k}$, with $u \in V^{k} / G_{X}$. Then,

$$
\begin{equation*}
V^{k}=\bigsqcup_{u \in V^{k} / G_{X}} O_{u}^{k} \quad \forall k \in[m] . \tag{7.20}
\end{equation*}
$$

To write the symmetry-reduced equivalent of (SPP) explicitly, we need some further terminology. Let the out-degree $d^{+}\left(\tau, W_{i}^{k}\right)$ (resp. in-degree $\left.d^{-}\left(\tau, W_{i}^{k}\right)\right)$ denote the number of arcs in orbital $W_{i}^{k}$ that start (resp. end) at vertex $\tau$, i.e.,

$$
d^{+}\left(\tau, W_{i}^{k}\right):=\left|\left\{(\tau, \tau \sigma) \in W_{i}^{k}: \sigma \in T\right\}\right| \quad \text { and } \quad d^{-}\left(\tau, W_{i}^{k}\right):=\left|\left\{(\tau \sigma, \tau) \in W_{i}^{k}: \sigma \in T\right\}\right|,
$$

for all $i \in A^{k} / G_{X}$ and $k \in[m]$. Since $d^{+}\left(\tau_{1}, W_{i}^{k}\right)=d^{+}\left(\tau_{2}, W_{i}^{k}\right)$ for all orbitals $i$ when $\tau_{1}$ and $\tau_{2}$ belong to the same orbit, it makes sense to define $d^{+}\left(W_{i}^{k}\right)\left(:=d^{+}\left(\tau, W_{i}^{k}\right)\right.$ for any $\left.(\tau, \tau \sigma) \in W_{k}^{i}\right)$ as the orbital out-degree in $W_{i}^{k}$. In a similar fashion we define $d^{-}\left(W_{i}^{k}\right)$.

From the discussion prior to Theorem 7.16 we know that there is a single case in which $d^{+}\left(\tau, W_{i}^{k}\right)>1$. Namely, two distinct arcs $\left(\tau, \tau \sigma_{1}\right)$ and $\left(\tau, \tau \sigma_{2}\right)$ with $\sigma_{1}=(i j)$ are both in the same orbital $W_{i}^{k}$ if and only if there exists a $b \in B_{\tau}$ such that $\sigma_{2}=(b(i) b(j))$.

This corresponds to case II in Figure 7.2. Hence, we have

$$
\begin{aligned}
& d^{+}\left(\tau, W_{i}^{k}\right)=\left|\left\{b(\{i, j\}): b \in B_{\tau}\right\}\right| \text { for some }(\tau, \tau(i j)) \in W_{i}^{k}, \\
& d^{-}\left(\tau, W_{i}^{k}\right)=\left|\left\{b(\{i, j\}): b \in B_{\tau}\right\}\right| \text { for some }(\tau(i j), \tau) \in W_{i}^{k} .
\end{aligned}
$$

Indeed, these equal the number of elements in an orbital of $\operatorname{Coup}(E)$ under the action of $B_{\tau}$. Moreover, we also define $d^{+}\left(Z_{i}^{0}\right)$ (resp. $d^{-}\left(Z_{i}^{m}\right)$ ) as the number of arcs in orbital $Z_{i}^{0}$ (resp. $Z_{i}^{m}$ ) starting from $s$ (resp. ending at $t$ ). For these degrees one can verify that $d^{+}\left(Z_{i}^{0}\right)=\left|Z_{i}^{0}\right|$ and $d^{-}\left(Z_{i}^{m}\right)=\left|Z_{i}^{m}\right|$.

For any vertex $\tau$, we let $\delta^{+}\left(\tau, A^{k} / G_{X}\right)$ (resp. $\delta^{-}\left(\tau, A^{k} / G_{X}\right)$ ) denote the set of orbitals that contain an arc starting (resp. ending) at vertex $\tau$. That is,

$$
\begin{aligned}
& \delta^{+}\left(\tau, A^{k} / G_{X}\right):=\left\{i \in A^{k} / G_{X}:(\tau, \tau \sigma) \in W_{i}^{k} \text { for some } \sigma \in T\right\} \\
& \delta^{-}\left(\tau, A^{k} / G_{X}\right):=\left\{i \in A^{k} / G_{X}:(\tau \sigma, \tau) \in W_{i}^{k} \text { for some } \sigma \in T\right\}
\end{aligned}
$$

Similar definitions hold for $\delta^{+}\left(\tau, D^{k} / G_{X}\right)$ and $\delta^{-}\left(\tau, D^{k} / G_{X}\right)$. Again, observe that if $\tau_{1}$ and $\tau_{2}$ belong to the same orbit $O_{u}^{k}$, then $\delta^{+}\left(\tau_{1}, A^{k} / G_{X}\right)=\delta^{+}\left(\tau_{2}, A^{k} / G_{X}\right)$. For that reason, it makes sense to define $\delta^{+}\left(O_{u}^{k}, A^{k} / G_{X}\right)$, which is equal to $\delta^{+}\left(\tau, A^{k} / G_{X}\right)$ for any $\tau \in O_{u}^{k}$. Stated differently, the set $\delta^{+}\left(O_{u}^{k}, A^{k} / G_{X}\right)$ contains all orbitals whose arcs start at a vertex belonging to orbit $O_{u}^{k}$. In a similar fashion, we define $\delta^{-}\left(O_{u}^{k}, A^{k} / G_{X}\right), \delta^{+}\left(O_{u}^{k}, D^{k} / G_{X}\right)$ and $\delta^{-}\left(O_{u}^{k}, D^{k} / G_{X}\right)$ for all $u \in V^{k} / G_{X}$ and $k \in[m]$.

The symmetry reduced equivalent formulation of (SPP) is obtained by replacing every variable $x_{e}$ in $H^{k}$ by a variable $\lambda_{i}^{k}$ corresponding to the orbital $W_{i}^{k}$ to which arc $e$ belongs. Similarly, we replace every variable $y_{e}$ in $D^{k}$ by a variable $\theta_{i}^{k}$ corresponding to the orbital $Z_{i}^{k}$ to which arc $e$ belongs. As a consequence, the flow conservation constraint corresponding to vertices that belong to the same orbit becomes equivalent, hence we only keep one per orbit. The remaining linear programming problem we denote by (RSPP) and is given by
(RSPP)

$$
\begin{array}{ll}
\min & \sum_{k=1}^{m} \sum_{i \in A^{k} / G_{X}}\left|W_{i}^{k}\right| \lambda_{i}^{k} \\
\text { s.t. } & \sum_{i \in D^{0} / G_{X}} d^{+}\left(Z_{i}^{0}\right) \theta_{i}^{0}=1, \sum_{i \in D^{m} / G_{X}} d^{-}\left(Z_{i}^{m}\right) \theta_{i}^{m}=1 \\
& \sum_{\substack{i \in \delta^{-}\left(O_{u}^{k}, D^{k-1} / G_{X}\right)}} \theta_{i}^{k-1}+\sum_{\substack{i \in \delta^{-}\left(O_{u}^{k}, A^{k} / G_{X}\right)}} d^{-}\left(W_{i}^{k}\right) \lambda_{i}^{k}= \\
& \sum_{\substack{i \in \delta^{+}\left(O_{u}^{k}, D_{i}^{k} / G_{X}\right)}} \sum_{\substack{i \in \delta^{+}\left(O_{u}^{k}, A^{k} / G_{X}\right)}} d^{+}\left(W_{i}^{k}\right) \lambda_{i}^{k} \quad \forall u \in V^{k} / G_{X}, k \in[m] \\
& 0 \leq \lambda_{i}^{k} \leq 1 \quad \forall i \in A^{k} / G_{X}, k \in[m] \\
& 0 \leq \theta_{i}^{k} \leq 1 \quad \forall i \in D^{k} / G_{X}, k \in\{0\} \cup[m] .
\end{array}
$$

Observe that $\left|W_{i}^{k}\right|, d^{+}\left(Z_{i}^{0}\right)$ and $d^{-}\left(Z_{i}^{m}\right)$ for all appropriate $k$ and $i$ are proportional to the size of an orbit in one of the subgraphs, which is in turn proportional to $|\operatorname{Aut}(\operatorname{Coup}(E))|$, see (7.13). For highly symmetric coupling graphs, the size of this automorphism group becomes very large, leading to enormous coefficient values in (RSPP). Since all known

LP-solvers run in weakly polynomial time (as opposed to strongly polynomial time), their running times depend on the binary encoding of the numerical values in the program. Besides, since other coefficients do not grow with $|\operatorname{Aut}(\operatorname{Coup}(E))|$, the condition number of the constraint matrix of (RSPP) can become very large, leading to numerical instability when solving the program.

To improve practical performance, we apply a scaling operation prior to solving the program. We first multiply both sides of the flow conservation constraints by $|\operatorname{Aut}(\operatorname{Coup}(E))|$ for all $u \in V^{k} / G_{X}$ and $k \in[m]$. After that, we apply the following substitution:

$$
\begin{aligned}
\bar{\lambda}_{i}^{k} & :=|\operatorname{Aut}(\operatorname{Coup}(E))| \lambda_{i}^{k} \\
\bar{\theta}_{i}^{k} & :=|\operatorname{Aut}(\operatorname{Coup}(E))| \theta_{i}^{k}
\end{aligned} \quad \text { for all } i \in A^{k} / G_{X}, k \in[m], ~ 子 D^{k} / G_{X}, k \in\{0\} \cup[m] .
$$

This leads to the equivalent linear program ( $\mathrm{RSPP}^{\prime}$ ). Observe that the new upper bound of $|\operatorname{Aut}(\operatorname{Coup}(E))|$ on $\bar{\lambda}_{i}^{k}$ and $\bar{\theta}_{i}^{k}$ can be omitted in this program. To verify this, observe that the out-degree of $s$ and in-degree of $t$ needs to be 1 , which implicitly enforces an upper bound of 1 on all $\bar{\theta}_{i}^{0}$ and $\bar{\theta}_{i}^{m}$. Since all variables are nonnegative and flow conservation holds for all orbits in the construction, any increase of a variable needs to be accounted for by a cycle over the orbits. As all objective coefficients are nonnegative, this leads to a possible increase of the objective value. Because we are minimizing, we can without loss of generality omit the upper bounds on the variables. Hence, the coefficients of this program no longer depend on $|\operatorname{Aut}(\operatorname{Coup}(E))|$.
( $\mathrm{RSPP}^{\prime}$ )

$$
\begin{array}{ll}
\min & \sum_{k=1}^{m} \sum_{i \in A^{k} / G_{X}} \frac{\left|W_{i}^{k}\right|}{|\operatorname{Aut}(\operatorname{Coup}(E))|} \bar{\lambda}_{i}^{k} \\
\text { s.t. } & \sum_{i \in D^{0} / G_{X}} \frac{d^{+}\left(Z_{i}^{0}\right)}{|\operatorname{Aut}(\operatorname{Coup}(E))|} \bar{\theta}_{i}^{0}=1, \sum_{i \in D^{m} / G_{X}} \frac{d^{-}\left(Z_{i}^{m}\right)}{|\operatorname{Aut}(\operatorname{Coup}(E))|} \bar{\theta}_{i}^{m}=1 \\
& \sum_{\substack{i \in \delta^{-}\left(O_{u}^{k}, D^{k-1} / G_{X}\right)}} \bar{\theta}_{i}^{k-1}+\sum_{\substack{i \in \delta^{-}\left(O_{u}^{k}, A^{k} / G_{X}\right)}} d^{-}\left(W_{i}^{k}\right) \bar{\lambda}_{i}^{k}= \\
& \sum_{\substack{i \in \delta^{+}\left(O_{u}^{k}, D^{k} / G_{X}\right)}}^{\bar{\theta}_{i}^{k}+\sum_{\substack{i \in \delta^{+}\left(O_{u}^{k}, A^{k} / G_{X}\right)}} d^{+}\left(W_{i}^{k}\right) \bar{\lambda}_{i}^{k} \quad \forall u \in V^{k} / G_{X}, k \in[m]} \\
& 0 \leq \bar{\lambda}_{i}^{k} \quad \forall i \in A^{k} / G_{X}, k \in[m] \\
& 0 \leq \bar{\theta}_{i}^{k} \quad \forall i \in D^{k} / G_{X}, k \in\{0\} \cup[m]
\end{array}
$$

Recall that the NNCP is in general $\mathcal{N} \mathcal{P}$-hard [341]. Based on the LP formulation ( $\mathrm{RSPP}^{\prime}$ ), we are able to unfold some special cases where the problem turns out to be polynomial time solvable. The condition that provides the key to this complexity result is the order of the automorphism group of the coupling graph.

Since all permutations in $B_{\tau}$ should setwise stabilize the sets $\tau^{-1}\left(S_{i}\right)$ for all $i \in[l]$, it follows that $B_{\tau}$ is a subgroup of $\mathbb{S}_{n}(\mathcal{G})$, where $\mathcal{G}:=\left\{\tau^{-1}\left(S_{1}\right), \ldots, \tau^{-1}\left(S_{l}\right)\right\}$. The order of $\mathbb{S}_{n}(\mathcal{G})$ is $2^{p} f$ !, which implies that $\left|B_{\tau}\right| \leq 2^{p} f$ !. This leads to the following complexity result.

Theorem 7.19. The NNCP is polynomial time solvable on coupling graphs with automorphism groups of order $\Omega((n-b)!)$, where $n$ is the number of vertices in the coupling graph and
$b$ is $a$ constant independent of $n$.
Proof. The number of variables in (RSPP) equals $m\left|A^{1} / G_{X}\right|+\left|D^{0} / G_{X}\right|+m\left|D^{1} / G_{X}\right|$. Based on Table 7.1 and the inequalities $\left|B_{\tau}\right| \leq 2^{p} f!,\left|E / B_{\tau}\right| \leq|E|$ and $\left|F^{k}\right| \leq 2|E|(n-2)$ ! for all $\tau \in \mathbb{S}_{n}$ and $k \in[m]$, we have

$$
\begin{aligned}
& \frac{m \sum_{\tau \in \mathbb{S}_{n}}\left|B_{\tau}\right| \cdot\left|E / B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}+\frac{\sum_{\tau \in \mathbb{S}_{n}}\left|B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}+\frac{m \sum_{\tau \in F^{1}}\left|B_{\tau}\right|}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|} \\
\leq & \frac{m 2^{p} f!|E| n!}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}+\frac{2^{p} f!n!}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|}+\frac{m 2|E|(n-2)!2^{p} f!}{2^{p} f!\cdot|\operatorname{Aut}(\operatorname{Coup}(E))|} \\
= & O\left(\frac{m|E| n!}{|\operatorname{Aut}(\operatorname{Coup}(E))|}\right) .
\end{aligned}
$$

Whenever $|\operatorname{Aut}(\operatorname{Coup}(E))|=\Omega((n-b)!)$, the number of variables in $(\operatorname{RSPP})$ is $O\left(\frac{m|E| n!}{(n-b)!}\right)=$ $O\left(m|E| n^{b}\right)$. Since $b$ does not depend on the input, the number of variables in the reduced instance is polynomial in $n, m$ and $|E|$.

The implication of Theorem 7.19 does not solely restrict to trivial NNCP classes, such as the ones with a coupling graph that is complete. An example of a less trivial class of coupling graphs having a sufficiently large automorphism group are the bicliques, i.e., the complete bipartite graphs.

Corollary 7.20. The $N N C P$ is polynomial time solvable on the biclique $K_{N, M}$ with $N$ of fixed size. In particular, the $N N C P$ on the star $K_{1, N}$ is polynomial time solvable.

### 7.4.2 Reduced combinatorial formulation

Similar to (SPP) being an LP formulation of a shortest path problem, we show in this section that (RSPP) and ( $\mathrm{RSPP}^{\prime}$ ) also have a combinatorial interpretation. Such combinatorial approaches often have the potential to induce efficient algorithms that are favoured over solving their LP formulation. In order to simplify notation, we work with (RSPP) in this section, although the construction for ( $\mathrm{RSPP}^{\prime}$ ) is similar.

To view (RSPP) as a combinatorial problem, we consider the so-called quotient graph of $X$ under the action of $G_{X}$. In its most general form, a quotient graph of a graph $X$ is induced by an equivalence relation on the vertices of $X$. We below provide the formal definition for the particular case where the equivalence relation is induced by an automorphism group of $X$.

Definition 7.21 (Quotient graph implied by automorphisms). Let $X=(V, A)$ be a directed graph and let $G$ be a subgroup of $\operatorname{Aut}(X)$. Then the quotient graph of $X$ under $G$ is the graph $\mathcal{X}=(\mathcal{V}, \mathcal{A})$ with $\mathcal{V}:=V / G$ and $\mathcal{A}:=A / G \subseteq \mathcal{V} \times \mathcal{V}$.

Since all arcs within an orbital of $X$ start at vertices in the same orbit and end at vertices in the same orbit, the quotient graph is well-defined. Observe that $\mathcal{X}$ can contain loops and multi-arcs, even if $X$ is simple.

Let $\mathcal{X}=(\mathcal{V}, \mathcal{A})$ be the quotient graph of $X$ under $G_{X}$. Since the source vertex $s$ and the sink vertex $t$ are in isolated orbits, the vertices $s$ and $t$ are again in $\mathcal{V}$. By abuse of notation, we again denote these vertices as $s, t \in \mathcal{V}$. Since the constraints and variables in (RSPP) correspond to orbits and orbitals of $X$ under $G_{X}$, respectively, the problem (RSPP) is an
optimization problem on the quotient graph $\mathcal{X}$. Now, for all $(j, \ell) \in \mathcal{A}$ we define the following flow variable $f_{j \ell}$ :

$$
f_{j \ell}:= \begin{cases}d^{+}\left(Z_{i}^{0}\right) \theta_{i}^{0} & \text { if }(j, \ell) \text { corresponds to } Z_{i}^{0},  \tag{7.21}\\ \theta_{i}^{k} & \text { if }(j, \ell) \text { corresponds to } Z_{i}^{k}, k \in[m], \\ d^{+}\left(W_{i}^{k}\right) \lambda_{i}^{k} & \text { if }(j, \ell) \text { corresponds to } W_{i}^{k}, k \in[m] .\end{cases}
$$

Moreover, we define for all $(j, \ell) \in \mathcal{A}$ a cost vector

$$
w_{j \ell}:= \begin{cases}\frac{\left|W_{i}^{k}\right|}{d^{+}\left(W_{i}^{k}\right)} & \text { if }(j, \ell) \text { corresponds to } W_{i}^{k}, k \in[m]  \tag{7.22}\\ 0 & \text { otherwise },\end{cases}
$$

and an upper bound vector

$$
u_{j \ell}:= \begin{cases}d^{+}\left(Z_{i}^{0}\right) & \text { if }(j, \ell) \text { corresponds to } Z_{i}^{0},  \tag{7.23}\\ 1 & \text { if }(j, \ell) \text { corresponds to } Z_{i}^{k}, k \in[m], \\ d^{+}\left(W_{i}^{k}\right) & \text { if }(j, \ell) \text { corresponds to } W_{i}^{k}, k \in[m]\end{cases}
$$

Finally, for all $(j, \ell) \in \mathcal{A}$ we define a multiplier $p_{j \ell}$ :

$$
p_{j \ell}:= \begin{cases}\frac{d^{-}\left(W_{i}^{k}\right)}{d^{+}\left(W_{i}^{k}\right)} & \text { if }(j, \ell) \text { corresponds to } W_{i}^{k}, k \in[m]  \tag{7.24}\\ d^{-}\left(Z_{i}^{m}\right) & \text { if }(j, \ell) \text { corresponds to } Z_{i}^{m} \\ 1 & \text { otherwise }\end{cases}
$$

We now substitute $f_{j \ell}, w_{j \ell}$ and $p_{j \ell}$ for all orbitals $(j, \ell) \in \mathcal{A}$ into (RSPP). This yields an equivalent linear programming problem that has the structure of a minimum cost generalized network flow problem:
(GNFP)

$$
\begin{array}{ll}
\min & \sum_{(j, \ell) \in \mathcal{A}} w_{j \ell} f_{j \ell} \\
\text { s.t. } & \sum_{(j, \ell) \in \delta^{+}(s)} f_{j \ell}=1, \sum_{(j, \ell) \in \delta^{-}(t)} p_{j \ell} f_{j \ell}=1 \\
& \sum_{(j, \ell) \in \delta^{+}(v)} f_{j \ell}= \\
0 \leq \sum_{j \ell} \leq u_{j \ell} \quad p_{j \ell} f_{j \ell} \quad \forall v \in \mathcal{V} \backslash\{s, t\} \\
& \forall(i, j) \in \mathcal{A} .
\end{array}
$$

A generalized flow is a flow starting from a sink $s$, conserving the flow at each vertex and ending at a source $t$, where along each arc $(j, \ell)$ only a fraction of $p_{j \ell}$ of flow is moved from $j$ to $\ell$. This fraction, called the multiplier, can also be larger than one, which means that the flow is increased along the arc. The problem (GNFP) aims to send a generalized flow of one from $s$ to $t$ that has a minimal cost with respect to the cost vector $w$.

The minimum cost generalized network flow problem is solvable in weakly polynomial time by the algorithm of Wayne [362]. This is the only known combinatorial algorithm for this problem in the literature. It is still an open problem whether the problem is also solvable in strongly polynomial time, see e.g., [208].

In the special case where $B_{\tau}$ is trivial for all $\tau \in \mathbb{S}_{n}$, the problem (GNFP) can be solved more efficiently. In that case we have $d^{+}\left(W_{i}^{k}\right)=d^{-}\left(W_{i}^{k}\right)=1$ for all orbitals $W_{i}^{k}$, hence $p_{j \ell}=1$ for all $W_{i}^{k}$. Now, for all $(j, \ell) \in \delta^{-}(t)$ we replace $p_{j \ell} f_{j \ell}$ by a new variable, say $g_{j \ell}$, that is upper-bounded by $d^{-}\left(Z_{i}^{m}\right)$. After these modifications, the resulting problem equals the LP formulation of a shortest path problem, for which strongly-polynomial time algorithms exist, e.g., Dijkstra's algorithm with Fibonacci heaps [144].

### 7.4.3 Symmetry-reduced NNCP algorithm

In this section we show how an optimal solution to (RSPP), ( $\mathrm{RSPP}^{\prime}$ ) or (GNFP) can be used to find an optimal sequence of qubit orders for the NNCP. Moreover, we briefly present an overview of the entire solution approach in terms of a pseudo-code.

By construction, solving (RSPP) or (GNFP) provides the optimal cost of a shortest path in $X$. However, because of the reduction, the solutions of (RSPP) or (GNFP) do no longer correspond itself to paths. Let $(\lambda, \theta)$ be an optimal solution to (RSPP) (in case of solving (GNFP), we can obtain ( $\lambda, \theta$ ) from the flow variable $f$ by (7.21)). Now, we define $(x, y) \in \prod_{k=1}^{m} \mathbb{R}^{A^{k}} \times \prod_{k=0}^{m} \mathbb{R}^{D^{k}}$ as follows:

$$
\begin{equation*}
x:=\left(\sum_{i \in A^{k} / G_{X}} \lambda_{i}^{k} \mathbb{1}_{W_{i}^{k}}\right)_{k=1}^{m} \quad \text { and } \quad y:=\left(\sum_{i \in D^{k} / G_{X}} \theta_{i}^{k} \mathbb{1}_{Z_{i}^{k}}\right)_{k=1}^{m} . \tag{7.25}
\end{equation*}
$$

It follows from the construction that the pair $(x, y)$ corresponds to an optimal solution of (SPP). Hence, it is a convex combination of characteristic vectors of ( $s, t$ )-paths in $X$. Let $X^{\text {sup }}$ denote the subgraph of $X$ induced by the support of $(x, y)$. Then, $X^{\text {sup }}$ is an acyclic graph. Namely, if there would exist a cycle in $X^{\text {sup }}$, due to the orientation of the arcs in $X$, it can only consist of arcs within one subgraph. Since these arcs all have a positive cost, the solution $(x, y)$ can be improved by excluding the cycle from it. By a similar argument, it follows that any $(s, t)$-path in $X^{\text {sup }}$ must be optimal. Namely, if there is an $(s, t)$-path in the support with a larger cost than the optimum, we can improve the solution $(x, y)$ by excluding this $(s, t)$-path and update the weights in the convex combination.

These observations make the identification of an optimal $(s, t)$-path simple: just find any $(s, t)$-path in $X^{\text {sup }}$. This can be done without actually constructing $X^{\text {sup }}$. Starting from $s$, we select an arbitrary arc from an orbital in $D^{0} / G_{X}$ that is in the support of $\theta^{0}$. This arc leads to a new vertex $\tau$. From the orbit where $\tau$ belongs to, we again select an orbital leaving this orbit that has a support in the optimal solution $(\lambda, \theta)$. Within this orbital, there is at least one arc starting from $\tau$ and we select such an arc arbitrarily if there are multiple. We continue doing this, which will eventually lead to the sink vertex $t$. It follows from the discussion above that this ( $s, t$ )-path provides an optimal qubit ordering for the NNCP.

We end this section by giving an overview of the symmetry-reduced NNCP algorithm. The approach is given in pseudo-code in Algorithm 7.1.

### 7.5 Special coupling graphs

Of key importance in the algorithm discussed in Section 7.4 are the orbit and orbital representation of the subgraphs, which rely on the subgroups $B_{\tau}$. These objects heavily depend on the specific coupling graph. In this section we demonstrate how these objects are obtained

```
Algorithm 7.1 Symmetry-reduced NNCP algorithm
Input: NNCP instance \(\Gamma=(Q, C)\) and coupling graph \(\operatorname{Coup}(E)=(L, E)\).
    Construct gate graph \((Q, U)\) and fixing pattern \(\mathcal{F}\).
    Construct orbit representation \(\mathcal{R}\) of \(\mathbb{S}_{n}\) under the action \(G_{\text {sub }}\).
    Initialize quotient subgraph \(\mathcal{H}^{\text {sub }}=\left(\mathcal{V}^{\text {sub }}, \mathcal{A}^{\text {sub }}\right)\) with \(\mathcal{V}^{\text {sub }}\) indexed by \(\mathcal{R}\) and \(\mathcal{A}^{\text {sub }}=\emptyset\).
    for \(\tau \in \mathcal{R}\) do
        Determine \(B_{\tau}\) and \(\mathcal{R}\left(E / B_{\tau}\right)\)
        for \(\{i, j\} \in \mathcal{R}\left(E / B_{\tau}\right)\) do
            Construct an arc in \(\mathcal{H}\) between \(\tau\) and the orbit to which \(\tau(i j)\) belongs, and add it to \(\mathcal{A}^{\text {sub }}\).
            Compute the size of \(\operatorname{Orb}((\tau, \tau(i j)))\) and its out- and in-degree in, resp., \(\operatorname{Orb}(\tau)\) and \(\operatorname{Orb}(\tau(i j))\).
        end for
    end for
    Initialize the quotient graph \(\mathcal{X}=(\mathcal{V}, \mathcal{A})\) where \(\mathcal{V}\) consists of \(m\) copies of \(\mathcal{V}^{\text {sub }}\), a source \(s\) and a sink
    \(t\), and \(\mathcal{A}\) consists of all arcs within the subgraphs \(\mathcal{A}^{\text {sub }}\). Add to \(\mathcal{A}\) all arcs between \(s\) and the first
    subgraph.
    for \(g^{k} \in C\) do
        Determine \(\mathcal{R}\left(D^{k} / G_{X}\right)\).
        if \(g\) is the \(m\) th quantum gate then
            For all orbit representatives \(\tau \in \mathcal{R}\left(D^{k} / G_{X}\right)\), add arcs from \(\operatorname{Orb}(\tau)\) to sink vertex \(t\).
        else
            For all orbit representatives \(\tau \in \mathcal{R}\left(D^{k} / G_{X}\right)\), add \(\operatorname{arcs}\) from \(\operatorname{Orb}(\tau)\) to same orbit in \(H^{k+1}\).
        end if
    end for
    Obtain optimal \((\lambda, \theta)\) pair via solving either (RSPP), ( \(\mathrm{RSPP}^{\prime}\) ) or the generalized network flow prob-
    lem (GNFP).
21: Find an optimal sequence of qubit orders \(\tau^{k}, k \in[m]\) by identifying any ( \(s, t\) )-path in the support of
    \((x, y)\), where \((x, y)\) are defined as in (7.25).
Output: \(\tau^{k}, k \in[m]\)
```

for four specific structured coupling graphs: the cycle graph, the biclique graph, the star graph and the square lattice graph.

Table 7.2 provides an overview of certain important characteristics of each of the considered coupling graphs. Details are provided in the subsections below.

### 7.5.1 Cycle graph $C_{N}$

Let $C_{N}=(L, E)$ denote the undirected cycle on $N$ vertices, i.e., $L=[N]$ and $E=\{\{i, i+1\}$ : $i \in[N-1]\} \cup\{N, 1\}$. Then $n=|L|=N$. It is well-known that the automorphism group of $C_{N}$ is given by $\mathcal{D}_{2 n}$, the dihedral group of order $2 n$, see e.g., Godsil and Royle [175]. This group consists of all reflections and rotations of the regular polygon of order $n$. It follows from Theorem 7.8 that $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is normal when $N \geq 5$ and, as a consequence, its full automorphism group is isomorphic to $\mathbb{S}_{n} \times \mathcal{D}_{2 n}$. The Cayley graph $\operatorname{Aut}\left(\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)\right)$ with $T$ induced by $C_{N}$ is in the literature known as the modified bubble-sort graph, see e.g., [242].

The first step in studying the orbit and orbital structure of $X$ under $G_{X}$ is the identification of $B_{\tau}$. It can be proven that $B_{\tau}$ is trivial under a very mild condition. Recall that $c$ is the number of qubits in a connected component of size at least three in $(Q, U)$, see Definition 7.12.

Theorem 7.22. Suppose $c \geq 3$. Then $B_{\tau}$ is trivial for all $\tau \in \mathbb{S}_{n}$.
Proof. Let $\tau \in \mathbb{S}_{n}$. If the gate graph $(Q, U)$ contains a connected component of size at least three, then the fixing pattern $\mathcal{F}$ contains at least three single-element sets, say $\{i\},\{j\}$ and $\{\ell\}$. Since $B_{\tau}$ is the subgroup of $\mathcal{D}_{2 n}$ that setwise stabilizes the sets $\tau^{-1}\left(S_{1}\right), \ldots, \tau^{-1}\left(S_{l}\right)$, it follows that any $b \in B_{\tau}$ must pointwise fix $\tau^{-1}(i), \tau^{-1}(j)$ and $\tau^{-1}(\ell)$. However, the only


Table 7.2: Summary of NNCP symmetry reduction characteristics for a set of special coupling graphs.
element in $\mathcal{D}_{2 n}$ that fixes more than two elements is the identity element. Thus, $B_{\tau}$ is trivial.

Observe that the condition of Theorem 7.22 is not restrictive. Namely, when $c<3$, the quantum circuit does not have overlapping quantum gates. This implies that a trivial qubit assignment is possible without the need of any inserted SWAP gates, making the NNCP instance trivial.

### 7.5.2 Biclique graph $K_{M, N}$ and star graph $K_{1, N}$

The biclique graph (or complete bipartite graph) $K_{M, N}$ is defined on $L=[M] \sqcup[N]$ and $E=$ $\{\{i, j\}: i \in[M], j \in[N]\}$. The induced partition of the vertex set $L$ we denote by the sets $L_{M}$ and $L_{N}$. We assume here that $M<N$. Any independent setwise permutation of vertices in $L_{M}$ and $L_{N}$ forms an automorphism of the graph, hence $\operatorname{Aut}(\operatorname{Coup}(E)) \cong \mathbb{S}_{M} \times \mathbb{S}_{N}$. The corresponding Cayley graph $\operatorname{Cay}\left(\mathbb{S}_{n}, T\right)$ is in the literature known as the generalized star graph, see e.g., [164]. With respect to the structure of the subgroups $B_{\tau}$, we prove the following result.

Theorem 7.23. Let $\tau \in \mathbb{S}_{n}$. Let $\mathcal{F}^{\prime}$ denote the fixing pattern obtained from $\mathcal{F}$ by replacing any $S \in \mathcal{F}$ with $|S| \geq 2$ by

$$
S_{1}=\left\{i \in S: \tau^{-1}(i) \in L_{M}\right\} \quad \text { and } \quad S_{2}=\left\{i \in S: \tau^{-1}(i) \in L_{N}\right\} .
$$

Then $B_{\tau} \cong \mathbb{S}_{n}\left(\mathcal{F}^{\prime}\right)$. Moreover, let $\hat{p}$ denote the number of pairs $\{i, j\}$ for which $\tau^{-1}(i) \in L_{M}$ and $\tau^{-1}(j) \in L_{N}$, let $f_{1}$ denote the number of elements in the free set that are mapped to $L_{M}$ by $\tau^{-1}$, and let $f_{2}=f-f_{1}$. Then,

$$
\left|B_{\tau}\right|=2^{p-\hat{p}} f_{1}!f_{2}!
$$

Proof. Let $\mathcal{G}:=\left\{\tau^{-1}\left(S_{1}\right), \ldots, \tau^{-1}\left(S_{l}\right)\right\}$ be the partition of $[n]$ defined by $\mathcal{F}$ shifted over $\tau^{-1}$. Then, $B_{\tau}$ is the subgroup of $\mathbb{S}_{n}(\mathcal{G})$ which are also automorphisms of $K_{M, N}$. Since any automorphism of $K_{M, N}$ setwise fix the vertices in $L_{M}$ and $L_{N}$, we obtain $B_{\tau}$ by splitting each set of $\mathcal{G}$ into its subset in $L_{M}$ and its subset in $L_{N}$, leading to the partition $\mathcal{G}^{\prime}$. The partition $\mathcal{F}^{\prime}$ is exactly $\mathcal{G}^{\prime}$ shifted over $\tau$, leading to $B_{\tau} \cong \mathbb{S}_{n}\left(\mathcal{F}^{\prime}\right)$. The second part of the statement follows from counting the number of elements in $\mathbb{S}_{n}\left(\mathcal{F}^{\prime}\right)$.

The special case where $M=1$ is commonly known as the star graph $K_{1, N}$. Its induced Cayley graph is studied in [242]. Since we consider this coupling graph extensively in the numerical results of Section 7.6 , we add this case explicitly to Table 7.2.

### 7.5.3 Lattice graph $G_{N, N}$

Let $G_{N, N}=(L, E)$ denote the lattice graph on $N^{2}$ vertices, i.e., $L=[N] \times[N]$ and $E=$ $\{\{(i, j),(i+1, j)\}: \quad i \in[N-1], j \in[N]\} \cup\{\{(i, j),(i, j+1)\}: i \in[N], j \in[N-1]\}$. Then, $n=N^{2}$. Its automorphisms are equal to the automorphisms of the square, which means that $\operatorname{Aut}(\operatorname{Coup}(E))=\mathcal{D}_{8}$, the dihedral group of order 8. Although this group is typically small, the case of the lattice is interesting from a practical point of view.

An important quantity for studying $B_{\tau}$ is again $c$. As discussed before, all instances with $c<3$ are trivial. When $c \geq 3$, the elements in $B_{\tau}$ should at least fix 3 elements. This is not possible for any rotation in $\mathcal{D}_{8}$, so the only possible elements in $B_{\tau}$ are reflections. Figure 7.3 shows the four reflection axes in $G_{N, N}$, which we denote by $z_{1}, \ldots, z_{4}$. We denote by $r_{j} \in \operatorname{Aut}\left(G_{N, N}\right)$ the corresponding reflection in $z_{j}$ as element in the automorphism group of $G_{N, N}$. We now introduce the notion of symmetrical parity.
Definition 7.24. A qubit ordering $\tau$ on the lattice $G_{N, N}$ is called symmetrically paired around the $z_{j}$-axis if for all singletons $S_{i} \in \mathcal{F}$ the preimages $\tau^{-1}\left(S_{i}\right)$ are on the $z_{j}$-axis and for all


Figure 7.3: Four axes of reflection in $G_{N, N}$. pairs $S_{i}=\{q, \ell\} \in \mathcal{F}$ we have $q=r_{j}(\ell)$, where $r_{j}$ is the reflection in the $z_{j}$-axis.

If $\tau$ is symmetrically paired around $z_{j}$, then $\tau^{-1}\left(S_{i}\right)$ is also automatically symmetrized with respect to $z_{j}$ for the free set $S_{i}$. Hence, informally speaking, $\tau$ is symmetrically paired around $z_{j}$ if the preimage of the fixing pattern is fully symmetrized w.r.t. the $z_{j}$-axis.

We can now prove the following result on $B_{\tau}$ for all relevant NNCP instances.

Theorem 7.25. If $c \geq N+1$, then $B_{\tau}$ is trivial for all $\tau \in \mathbb{S}_{n}$. If $3 \leq c \leq N$, then $B_{\tau}=\left\{\mathrm{id}, r_{j}\right\}$ if $\tau$ is symmetrically paired around the $z_{j}$-axis and $B_{\tau}=\{\mathrm{id}\}$ otherwise.

Proof. As stated above, besides the identity element, $B_{\tau}$ can only consist of reflections in the axes $z_{1}, \ldots, z_{4}$. Each of these reflections fixes either no or $N$ positions. If $c \geq N+1$, at least $N+1$ positions should be fixed by elements in $B_{\tau}$, so $B_{\tau}$ is trivial for all $\tau \in \mathbb{S}_{n}$. If $3 \leq c \leq N, B_{\tau}$ possibly contains reflections $r_{j}$, where the condition $r_{j}\left(\tau^{-1}\left(S_{i}\right)\right)=\tau^{-1}\left(S_{i}\right)$ for all $S_{i} \in \mathcal{F}$ is equivalent to $\tau$ being symmetrically paired around $z_{j}$. Observe that $B_{\tau}$ can never contain two or more reflections in $\left\{r_{1}, \ldots, r_{4}\right\}$, since its composition fixes at most one position. Therefore, $B_{\tau}=\left\{\mathrm{id}, r_{j}\right\}$ for some $j$ or $B_{\tau}$ is trivial.

### 7.6 Computational results

In this section we test the symmetry-reduced NNCP algorithm, see Algorithm 7.1, on a set of instances for several of the coupling graphs discussed in Section 7.5. We compare the result against the nonreduced shortest path formulation (SPP).

We first describe the design of our numerical tests in Section 7.6.1, after which we discuss the results on real and random instances in Section 7.6.2 and 7.6.3, respectively.

### 7.6.1 Design of computational experiments

For our experiments we consider both realistic as well as randomly generated quantum circuits on different coupling graphs. As described in Section 7.2, we are justified to make two assumptions on the quantum circuits under consideration, imposing a preprocessing strategy in case these assumptions are not met:

1. Single-qubit gates can be ignored for the NNCP, since these do always comply with the adjacent interaction constraints. Without loss of generality, we therefore remove the single-qubit gates from the circuits in the preprocessing phase.
2. All gates that act on more than two qubits are decomposed into gates that act on one or two qubits. Nielsen and Chuang [291] have shown that these gates are universal, and that any quantum gate can therefore be decomposed into one- or two-qubit gates. There exists a large number of different decomposition strategies, leading to possibly different quantum gates (with the same functionality, however). As the choice of the optimal decomposition strategy is outside the scope of our research, we always choose the same strategy, namely the method considered in [285, 286].

The quantum circuits that we consider in this chapter consist of general one- or two-qubit gates, multiple-control Toffoli gates up to size five, Peres gates and multiple-control Fredkin gates up to size four. In Appendix A. 6 we consider the decomposition of these gates into one- or two qubit gates, following the approach from [285, 286]. After that, we remove all single-qubit gates from the circuit. The preprocessed circuit that remains, will be the quantum circuit $\Gamma=(Q, C)$ that we take as an input to our algorithm.

We consider the following two instance classes:

- Real data: Realistic quantum circuits that we consider are obtained from the RevLib library [367]. This dataset consists of quantum gates of (well-known) reversible functions considered in the quantum literature. Due to the assumptions of the preprocessing phase, we only consider instances consisting of the above-mentioned gates, see

Appendix A. 6 for an overview. This leads to a set of 84 instances with $n \in\{5, \ldots, 17\}$ and $m \in\{7, \ldots, 112\}$.

- Random data: We also consider synthetic quantum gates in order to also test our approach on circuits consisting of more qubits and gates. We apply two strategies:
- Random Class I: Given $n$ and $m$, we create a random circuit on $n$ qubits consisting of $m$ two-qubit gates. Each gate acts on two qubits that are chosen uniformly at random from $[n]$ without replacement, independently from the other gates.
For each combination of $n \in\{20,30, \ldots, 100\}$ and $m=\{2 n, 4 n\}$, we consider 5 randomly generated instances of this type. This leads to a test set of 90 instances.
- Random Class II: Given $n$ and $m$, we first create a random circuit on $n$ qubits consisting of $m$ gates selected from: Toffoli gate (on 3, 4 or 5 qubits), Fredkin gate (on 3 or 4 qubits), Peres gate, or a general two-qubit gate. The latter class includes the CNOT, SWAP and controlled- $V$ or $-V^{\dagger}$ gates. Each gate type is selected with equal probability, and the qubits on which each gate acts, is chosen uniformly at random from $[n]$ without replacement. After that, we apply the preprocessing approach explained above to convert each circuit to an equivalent circuit of twoqubit gates. This leads to quantum gates with possibly more realistic patterns than Random Class I.
For each combination of $n \in\{20,30, \ldots, 100\}$ and $m \in\{n, 2 n\}$, we consider 5 randomly generated instances of this type, leading to a test set of 90 instances. After the preprocessing step, the values of $m$ increase and are within $117 \leq m \leq 1872$.

We solve the NNCP for each quantum circuit on the following coupling graphs:

- Cycle graph: The undirected cycle $C_{N}$ on $N=n$ qubits, see Section 7.5.1.
- Star graph: The star graph $K_{1, N}$ with $N=n-1$, see Section 7.5.2.
- Biclique graph: The biclique graph $K_{M, N}$ with $M=2$ and $N=n-2$, see Section 7.5.2.

Observe that we do not consider the biclique graphs with $M>2$ or the Lattice graphs. The reason is that the automorphism groups of such graphs are rather small. Although the resulting reductions in the number of variables and constraints are still substantial, preliminary tests have shown that the difference in performance between the unreduced and reduced variant is less significant than for the other coupling graphs.

For each combination of quantum circuit and coupling graph, we solve the unreduced LPformulation (SPP) and the reduced scaled formulation ( $\mathrm{RSPP}^{\prime}$ ). The unreduced formulation is implemented by a full construction of the graph $X=(V, A)$. The reduced formulation is implemented as described in Algorithm 7.1. We emphasize that our symmetry reduction entirely follows from the analysis in Sections 7.3, 7.4 and 7.5, and that it does not rely on the use of algebraic software, nor does it require a construction of the full graph $X$. Preliminary experiments have shown that the performance between the nonscaled and scaled formulations, (RSPP) and ( $\mathrm{RSPP}^{\prime}$ ), respectively, is very similar. However, as the size of the coefficients in (RSPP) grows with the order of the automorphism group of $\operatorname{Coup}(E)$, the LP formulation becomes unstable for the star and biclique graphs when $n \geq 11$ or $n \geq 12$, respectively. Therefore, we only use the more robust scaled version ( $\mathrm{RSPP}^{\prime}$ ) in our tests.

Experiments are carried out on a PC with an $\operatorname{Intel}(\mathrm{R})$ Core(TM) i7-8700 CPU, 3.20 GHz and 8 GB RAM. Our algorithms are implemented in Julia 1.8.4 using JuMP v1.6.0 [113] to model the mathematical optimization problems. We use the LP solver of Mosek 10.0 [18] to solve our models in the default configuration. The maximum computation time (including the construction time of the program) is set to 2 hours.

### 7.6.2 Results on RevLib instances

Table 7.3, 7.4 and 7.5 show the results for the RevLib instances on the cycle, star and biclique graph, respectively. The columns ' $n$ ' and ' $m$ ' show the number of qubits and quantum gates in the preprocessed circuit. The column 'OPT' shows the optimal value of the NNCP instance, i.e., the minimum number of inserted SWAP gates in order to make the quantum circuit compliant. The columns 'time ( $R S P P^{\prime}$ )' and 'time ( $S P P$ )' show the computation time (i.e., clocktimes) in seconds to solve the reduced model (RSPP') and the base model (SPP), respectively. The values are rounded to three decimals. The columns ' $\#$ var $\left(R S P P^{\prime}\right.$ ') and '\#const ( $R S P P^{\prime}$ )' denote the total number of variables and constraints after the symmetry reduction. Finally, the column 'reduction \#var (\%)' shows the relative reduction in the number of variables compared to the base model, i.e., $\frac{\# \operatorname{var}(S P P)-\# \mathrm{var}\left(R S P P^{\prime}\right)}{\# \operatorname{var}(S P P)} \cdot 100 \%$, rounded to two decimal places. The final column shows the same relative reduction for the number of constraints. Whenever a given instance is not solvable (including construction) within the time limit of 2 hours, or whenever an instance leads to a shortage of memory, we report a ' - ' in the tables.

It turned out that the 62 instances with $n=5$ are very easy to compute for both models $(S P P)$ and $\left(R S P P^{\prime}\right)$. For that reason, results on these instances are not depicted in Tables 7.3, 7.4 and 7.5 . The total relative reduction in the number of variables and constraints on the instances with $n=5$ turns out to be at least $90 \%$ and $89.8 \%$, respectively.

For the cycle graph, one can clearly see that the bottleneck in the computational limit is the number of qubits $n$. It follows from Table 7.3 that our algorithm is able to solve instances up to roughly 8 qubits, while the base model can only solve instances up to 7 qubits. The total computation time of ( $\mathrm{RSPP}^{\prime}$ ) is often negligible and below 30 seconds for the instances that can be solved. For the base model the total computation times are significantly higher, with a maximum difference of about a factor 100. This can be explained by the large reduction in the total number of variables and constraints, which are both above $91 \%$ for all instances.

For the star graph, we conclude from Table 7.4 that the reduced model can easily handle the full set of RevLib instances. The computation times are negligible for almost all instances and always below 0.2 seconds. This can be explained by the order of $\operatorname{Aut}(\operatorname{Coup}(E))$ being factorial in $n$, implying that the model ( $\mathrm{RSPP}^{\prime}$ ) scales linearly in both $m$ and $n$. The relative reductions with the base model are enormous, i.e., above $99 \%$ in terms of the number of variables and constraints on all instances. For the unreduced model, the largest instance we can solve has $n=8$ and $m=36$, which could not be solved on the cycle coupling graph. This can be explained by the fact that the star graph on $n$ vertices has one edge less than the cycle graph on $n$ vertices, resulting in the Cayley graph containing significantly fewer edges. The computational frontier, however, is reached already at the next instance, for which the base model runs into memory issues.

Finally, the results on the biclique coupling graph look very similar to the results of the star graph, see Table 7.5. The total relative reduction between the models is extremely large, leading to all instances to be solvable within 0.25 seconds using (RSPP'). The computation
times are slightly larger than in the case of the star graph, which can be explained by the smaller size of the automorphism group of the biclique. For the unreduced formulation we can only solve up to $n=7$, while the reduction in computation times for the largest instance that can be solved using (SPP) is about a factor 4700 .

| Benchmark | $\boldsymbol{n}$ | $\boldsymbol{m}$ | OPT | time <br> $\left(\boldsymbol{R S P} \boldsymbol{P}^{\prime}\right)$ | time <br> $(\boldsymbol{S P P})$ | \#var <br> $\left(\boldsymbol{R S} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | \#const <br> $\left(\boldsymbol{R S P} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | reduction <br> \#var (\%) | reduction <br> \#const (\%) |
| :--- | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| graycode6_47 | 6 | 5 | 0 | 0.000 | 0.172 | 1980 | 3602 | 91.67 | 91.62 |
| graycode6_48 | 6 | 5 | 0 | 0.016 | 0.172 | 1980 | 3602 | 91.67 | 91.62 |
| decod24-enable_124 | 6 | 21 | 5 | 0.047 | 0.937 | 8124 | 15122 | 91.67 | 91.65 |
| decod24-enable_125 | 6 | 21 | 4 | 0.047 | 0.906 | 8124 | 15122 | 91.67 | 91.65 |
| decod24-bdd_294 | 6 | 24 | 8 | 0.062 | 1.203 | 9276 | 17282 | 91.67 | 91.66 |
| mod5adder_129 | 6 | 71 | 27 | 0.157 | 4.563 | 27324 | 51122 | 91.67 | 91.66 |
| mod5adder_128 | 6 | 77 | 32 | 0.172 | 4.250 | 29628 | 55442 | 91.67 | 91.66 |
| decod24-enable_126 | 6 | 86 | 34 | 0.188 | 5.500 | 33084 | 61922 | 91.67 | 91.66 |
| xor5_254 | 6 | 5 | 3 | 0.016 | 0.188 | 1980 | 3602 | 91.67 | 91.62 |
| ex1_226 | 6 | 5 | 3 | 0.016 | 0.187 | 1980 | 3602 | 91.67 | 91.62 |
| 4mod5-bdd_287 | 7 | 23 | 8 | 0.469 | 36.203 | 61080 | 115922 | 92.86 | 92.86 |
| alu-bdd_288 | 7 | 28 | 7 | 0.641 | 51.031 | 74280 | 141122 | 92.86 | 92.86 |
| ham7_106 | 7 | 49 | 20 | 1.172 | 91.672 | 129720 | 246962 | 92.86 | 92.86 |
| ham7_105 | 7 | 65 | 32 | 1.485 | 135.625 | 171960 | 327602 | 92.86 | 92.86 |
| ham7_104 | 7 | 83 | 38 | 1.984 | 181.734 | 219480 | 418322 | 92.86 | 92.86 |
| rd53_137 | 7 | 66 | 33 | 3.750 | 146.811 | 174600 | 23762 | 92.24 | 92.86 |
| rd53_139 | 8 | 36 | 14 | 22.672 | - | 754200 | 90722 | 93.75 | 93.75 |
| rd53_138 | 8 | 44 | 20 | 26.266 | - | 921240 | 110882 | 93.75 | 93.75 |
| mini_alu_305 | 10 | 57 | - | - | - | - | - | - | - |
| sys6-v0_144 | 10 | 62 | - | - | - | - | - | - | - |
| rd73_141 | 10 | 64 | - | - | - | - | - | - | - |
| parity_247 | 17 | 16 | - | - | - | - | - | - | - |

Table 7.3: Results on the 'RevLib' instances on the cyclic coupling graph. We compare the performance of the base model $(S P P)$ with the reduced model $(R S P P)$. Times are clocktimes given in seconds.

### 7.6.3 Results on random instances

From Table 7.4 and 7.5 we observe that the RevLib instances can be easily solved by our symmetry reduced formulation. To test the performance on larger instances, we consider the random data set, consisting of quantum circuits with up to 100 qubits and 1837 quantum gates. For the cycle coupling graph, we have seen that we could only solve instances up to $n=8$. Therefore, we do not include the cycle coupling graph anymore for the random data set. For the same reason, we do no longer consider the base model (SPP).

Table 7.6 and 7.7 show the performance of our symmetry-reduced NNCP algorithm on Random Class I and Random Class II for both the star and biclique coupling graph. Next to the total solution time, which is given in the column 'time ( $R S P P$ )', we show in the column 'time constr.' the time that is required to construct the LP-instance, i.e., the time spent in lines $1-21$ of Algorithm 7.1. Each row in the tables corresponds to the average value over 5 randomly generated instances of that type. In Figure 7.4 we plot the averaged total computation time, i.e., construction and solution time, compared to $n$ and $m$ for both coupling graphs and random classes.

For the star coupling graph, we see that we can easily solve all instances from Random Class I within on average 25 seconds, while at most 90 seconds are needed to construct the model. For Random Class II, we can solve up to $n=100$, however, when $m$ is too large, the PC runs out of memory. For the biclique coupling graph on Random Class I, we can solve instances up to $n=40$ within the time span of 2 hours, whereas for Random Class II the instances with large $m$ cannot be solved anymore.

The sum of solution and construction times on the biclique graphs is significantly higher than on the star graphs, see Figure 7.4. The tables reveal that the solution times on the former are an order of magnitude 2 higher. This can be explained by the difference in the order of $\operatorname{Aut}(\operatorname{Coup}(E))$, as explained in Section 7.6.2. The construction times, however, heavily deviate among the instances on the star and the biclique coupling graph. Indeed, the smaller automorphism group increases the number of orbits, and thus, the number of orbit representatives in $\mathcal{R}$. For each of these representatives, one needs to enumerate over the orbits of the group action of $B_{\tau}$ on $E$. Moreover, in line 7 of Algorithm 7.1 a subroutine needs to identify to which orbit $\tau(i j)$ belongs. Since the number of orbits is larger, an enumerative search along the orbit list takes more time. Hence, the negative effects of having a smaller number of symmetries and a larger number of edges, strengthen one another and result in large construction times when $n$ and $m$ increase.

| Benchmark | $\boldsymbol{n}$ | $\boldsymbol{m}$ | OPT | time <br> $\left(\boldsymbol{R S} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | time <br> $(\boldsymbol{S P P} \boldsymbol{P}$ | \#var <br> $\left(\boldsymbol{R S} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | \#const <br> $\left(\boldsymbol{R S} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | reduction <br> \#var (\%) | reduction <br> \#const (\%) |
| :--- | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| graycode6_47 | 6 | 5 | 2 | 0.000 | 0.125 | 166 | 32 | 99.17 | 99.11 |
| graycode6_48 | 6 | 5 | 2 | 0.000 | 0.125 | 166 | 32 | 99.17 | 99.11 |
| decod24-enable_124 | 6 | 21 | 4 | 0.016 | 0.609 | 678 | 128 | 99.17 | 99.15 |
| decod24-enable_125 | 6 | 21 | 5 | 0.000 | 0.594 | 678 | 128 | 99.17 | 99.15 |
| decod24-bdd_294 | 6 | 24 | 8 | 0.000 | 0.672 | 774 | 146 | 99.17 | 99.16 |
| mod5adder_129 | 6 | 71 | 19 | 0.000 | 2.343 | 2278 | 428 | 99.17 | 99.16 |
| mod5adder_128 | 6 | 77 | 18 | 0.000 | 2.953 | 2470 | 464 | 99.17 | 99.16 |
| decod24-enable_126 | 6 | 86 | 19 | 0.016 | 2.718 | 2758 | 518 | 99.17 | 99.16 |
| xor5_254 | 6 | 5 | 0 | 0.000 | 0.109 | 166 | 32 | 99.17 | 99.11 |
| ex1_226 | 6 | 5 | 0 | 0.000 | 0.125 | 166 | 32 | 99.17 | 99.11 |
| 4mod5-bdd_287 | 7 | 23 | 5 | 0.000 | 14.875 | 1019 | 163 | 99.86 | 99.86 |
| alu-bdd_288 | 7 | 28 | 11 | 0.000 | 16.141 | 1239 | 198 | 99.86 | 99.86 |
| ham7_106 | 7 | 49 | 20 | 0.015 | 28.328 | 2163 | 345 | 99.86 | 99.86 |
| ham7_105 | 7 | 65 | 18 | 0.000 | 36.157 | 2867 | 457 | 99.86 | 99.86 |
| ham7_104 | 7 | 83 | 18 | 0.015 | 57.516 | 3659 | 583 | 99.86 | 99.86 |
| rd53_137 | 7 | 66 | 10 | 0.000 | 38.521 | 2911 | 464 | 99.86 | 99.86 |
| rd53_139 | 8 | 36 | 15 | 0.047 | 7031.828 | 2096 | 290 | 99.98 | 99.98 |
| rd53_138 | 8 | 44 | 12 | 0.000 | - | 2560 | 354 | 99.98 | 99.98 |
| mini_alu_305 | 10 | 57 | 16 | 0.016 | - | 5254 | 572 | 100.00 | 100.00 |
| sys6-v0_144 | 10 | 62 | 26 | 0.015 | - | 5714 | 622 | 100.00 | 100.00 |
| rd73_141 | 10 | 64 | 27 | 0.000 | - | 5898 | 642 | 100.00 | 100.00 |
| parity_247 | 17 | 16 | 0 | 0.000 | - | 4401 | 274 | 100.00 | 100.00 |
|  |  |  |  |  |  |  |  |  |  |

Table 7.4: Results on the 'RevLib' instances on the star coupling graph. We compare the performance of the base model $(S P P)$ with the reduced model $(R S P P)$. Times are clocktimes given in seconds.

| Benchmark | $\boldsymbol{n}$ | $\boldsymbol{m}$ | OPT | time <br> $\left(\boldsymbol{R S P} \boldsymbol{P}^{\prime}\right)$ | time <br> $(\boldsymbol{S P P} \boldsymbol{P})$ | \#var <br> $\left(\boldsymbol{R S} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | \#const <br> $\left(\boldsymbol{R S} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | reduction <br> \#var (\%) | reduction <br> \#const (\%) |
| :--- | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| graycode6_47 | 6 | 5 | 1 | 0.015 | 0.250 | 655 | 77 | 97.92 | 97.86 |
| graycode6_48 | 6 | 5 | 1 | 0.000 | 0.235 | 655 | 77 | 97.92 | 97.86 |
| decod24-enable_124 | 6 | 21 | 4 | 0.016 | 1.500 | 2703 | 317 | 97.92 | 97.90 |
| decod24-enable_125 | 6 | 21 | 4 | 0.000 | 1.297 | 2703 | 317 | 97.92 | 97.90 |
| decod24-bdd_294 | 6 | 24 | 5 | 0.015 | 1.485 | 3087 | 362 | 97.92 | 97.91 |
| mod5adder_129 | 6 | 71 | 15 | 0.032 | 5.031 | 9103 | 1067 | 97.92 | 97.91 |
| mod5adder_128 | 6 | 77 | 14 | 0.031 | 5.016 | 9871 | 1157 | 97.92 | 97.91 |
| decod24-enable_126 | 6 | 86 | 16 | 0.031 | 5.844 | 11023 | 1292 | 97.92 | 97.91 |
| xor5_254 | 6 | 5 | 1 | 0.000 | 0.266 | 655 | 77 | 97.92 | 97.86 |
| ex1_226 | 6 | 5 | 1 | 0.000 | 0.265 | 655 | 77 | 97.92 | 97.86 |
| 4mod5-bdd_287 | 7 | 23 | 4 | 0.016 | 72.110 | 5081 | 485 | 99.58 | 99.58 |
| alu-bdd_288 | 7 | 28 | 5 | 0.016 | 83.844 | 6181 | 590 | 99.58 | 99.58 |
| ham7_106 | 7 | 49 | 8 | 0.031 | 143.265 | 10801 | 1031 | 99.58 | 99.58 |
| ham7_105 | 7 | 65 | 14 | 0.031 | 217.359 | 14321 | 1367 | 99.58 | 99.58 |
| ham7_104 | 7 | 83 | 8 | 0.078 | 282.453 | 18281 | 1745 | 99.58 | 99.58 |
| rd53_137 | 7 | 66 | 10 | 0.047 | 223.981 | 14541 | 1388 | 98.14 | 99.58 |
| rd53_139 | 8 | 36 | 8 | 0.063 | - | 12556 | 1010 | 99.93 | 99.93 |
| rd53_138 | 8 | 44 | 10 | 0.062 | - | 15340 | 1234 | 99.94 | 99.94 |
| mini_alu_305 | 10 | 57 | 14 | 0.218 | - | 41997 | 2567 | 100.00 | 100.00 |
| sys6-v0_144 | 10 | 62 | 13 | 0.141 | - | 45677 | 2792 | 100.00 | 100.00 |
| rd3_141 | 10 | 64 | 14 | 0.095 | - | 47149 | 2882 | 100.00 | 100.00 |
| parity_247 | 17 | 16 | 1 | 0.108 | - | 65896 | 2178 | 100.00 | 100.00 |

Table 7.5: Results on the 'RevLib' instances on the biclique coupling graph. We compare the performance of the base model $(S P P)$ with the reduced model $(R S P P)$. Times are clocktimes given in seconds.


Figure 7.4: Overview of total average computation times (construction + solution time) of random instances with respect to $n$ and $m$. Each data point displays the average over 5 randomly generated instances of that type.

When comparing Random Class I and II, we do not observe significant structural differences. It seems to be primarily the magnitude of $n$ and $m$ that influences the complexity
of the instance. Due to the construction, $m$ grows more rapidly with respect to $n$ for Random Class II than for Random Class I. This effect can be observed from Figure 7.4, where we observe that for fixed $n$, an instance from Random Class II on average requires more computation time.

The largest quantum circuit that we can successfully solve contains 100 qubits and 1047 quantum gates. Observe that the unreduced model of this instance would embrace subgraphs of 100 ! vertices, hence solving this model is infeasible.

| Random Class I |  |  |  |  | Random Class II |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| $n$ | $m$ | OPT | $\begin{gathered} \text { time } \\ \left(R S P P^{\prime}\right) \end{gathered}$ | time constr. | $n$ | $m$ | OPT | $\begin{gathered} \text { time } \\ \left(R S P P^{\prime}\right) \end{gathered}$ | $\begin{aligned} & \text { time } \\ & \text { constr. } \end{aligned}$ |
| 20 | 40 | 29.6 | 0.031 | 0.088 | 20 | 125.6 | 35.0 | 0.119 | 1.425 |
| 20 | 80 | 65.2 | 0.056 | 0.134 | 20 | 365.6 | 80.8 | 0.334 | 0.712 |
| 30 | 60 | 52.4 | 0.106 | 0.274 | 30 | 255 | 59.6 | 0.544 | 1.189 |
| 30 | 120 | 101.2 | 0.243 | 0.551 | 30 | 564.6 | 126.4 | 1.350 | 2.940 |
| 40 | 80 | 72.4 | 0.282 | 0.820 | 40 | 302.2 | 76.6 | 1.150 | 2.482 |
| 40 | 160 | 144.4 | 0.631 | 1.556 | 40 | 652.4 | 159.8 | 3.150 | 8.081 |
| 50 | 100 | 91.2 | 0.569 | 1.971 | 50 | 441.4 | 104.6 | 3.272 | 7.223 |
| 50 | 200 | 184.6 | 1.312 | 3.336 | 50 | 854.8 | 203.8 | 8.091 | 16.474 |
| 60 | 120 | 112.8 | 1.025 | 3.349 | 60 | 471 | 122.8 | 5.737 | 13.121 |
| 60 | 240 | 222.4 | 2.447 | 6.162 | 60 | 1027.4 | 247.6 | 16.903 | 44.061 |
| 70 | 140 | 132.0 | 1.769 | 6.135 | 70 | 589.4 | 145.2 | 10.838 | 26.888 |
| 70 | 280 | 264.6 | 4.662 | 14.194 | 70 | 1223 | 292.8 | 31.875 | 149.451 |
| 80 | 160 | 151.0 | 3.313 | 10.704 | 80 | 722.4 | 171.6 | 23.634 | 97.156 |
| 80 | 320 | 304.2 | 7.775 | 21.305 | 80 | 1372.8 | 333.6 | 32.600 | 307.844 |
| 90 | 180 | 172.0 | 4.809 | 24.524 | 90 | 750 | 184.8 | 22.312 | 162.915 |
| 90 | 360 | 343.8 | 14.681 | 42.293 | 90 | 1602.8 | - | - | - |
| 100 | 200 | 191.8 | 9.106 | 31.802 | 100 | 921.2 | 218.8 | 36.966 | 363.073 |
| 100 | 400 | 385.0 | 21.066 | 85.295 | 100 | 1709.6 | - | - | - |

Table 7.6: Results on the random instances on the star coupling graph. Each row shows the average values over 5 randomly generated instances. Times are clocktimes given in seconds.

| Random Type I |  |  |  |  |
| :--- | ---: | :---: | :---: | ---: |
| $\boldsymbol{n}$ | $\boldsymbol{m}$ | OPT | time <br> $\left(\boldsymbol{R S} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | time <br> constr. |
| 20 | 40 | 20.6 | 1.588 | 12.029 |
| 20 | 80 | 43.4 | 2.590 | 14.217 |
| 30 | 60 | 38.4 | 9.675 | 374.035 |
| 30 | 120 | 71.2 | 18.322 | 390.226 |
| 40 | 80 | 54.2 | 44.053 | 3872.272 |
| 40 | 160 | 109.4 | 66.884 | 3989.450 |


| Random Type II |  |  |  |  |
| :--- | ---: | :---: | ---: | ---: |
| $\boldsymbol{n}$ | $\boldsymbol{m}$ | OPT | time <br> $\left(\boldsymbol{R S} \boldsymbol{P} \boldsymbol{P}^{\prime}\right)$ | time <br> constr. |
| 20 | 125.6 | 27.6 | 4.188 | 17.270 |
| 20 | 365.6 | 66.8 | 15.113 | 30.584 |
| 30 | 255 | 50.0 | 49.175 | 413.975 |
| 30 | 564.6 | 107.2 | 115.350 | 829.710 |
| 40 | 302.2 | 61.7 | 168.276 | 2800.738 |
| 40 | 652.4 | - | - | - |

Table 7.7: Results on the random instances on the biclique coupling graph. Each row shows the average values over 5 randomly generated instances. Times are clocktimes given in seconds.

### 7.7 Conclusions

In this chapter we study an exact method for solving the NNCP in the gated quantum computing model by exploiting symmetries in the underlying formulation.

Starting from the shortest path formulation introduced by [269], see (SPP), we study the algebraic structure of the underlying graph in Section 7.3. This graph is composed of a series of Cayley graphs of the symmetric group $\mathbb{S}_{n}$ generated by the transpositions in the coupling graph of the quantum system. We show that $\mathbb{S}_{n} \times \operatorname{Aut}(\operatorname{Coup}(E))$ is a subgroup of the automorphism group of such Cayley graph, which turns out to be the full automorphism group in case the Cayley graph is normal as shown by [163]. Although the automorphism groups of specific Cayley graphs generated by transpositions has been studied before in the literature, we do not make any assumption on the underlying coupling graph apart from being connected. Next, we show how these subgroups are merged into a subgroup $G_{X}$ of the automorphism group of the entire graph, see (7.8). One component of this subgroup is determined by the algebraic structure of the coupling graph, while the other component relies on a so-called fixing pattern $\mathcal{F}$ following from the quantum gates in the circuit, see Definition 7.12. The orbit and orbital structures of the action of this group on the graph are also studied, leading in particular to an overview of the cardinalities of the corresponding quotients, see Table 7.1.

By exploiting the convexity of (SPP), we reduce the symmetries in the formulation by averaging over all symmetric solutions using the Reynolds operator, see (7.15). This leads to a more compact equivalent formulation (RSPP) and its scaled variant ( $\mathrm{RSPP}^{\prime}$ ). We show that this formulation is equivalent to a generalized network flow problem (GNFP). Due to the in-depth analysis on the orbit and orbital structure, these formulations can be explicitly constructed from scratch without the need to first construct the exponentially large Cayley graphs. This leads to our symmetry-reduced NNCP algorithm, see Algorithm 7.1. A direct theoretical implication of our approach are the complexity results of Theorem 7.19 and Corollary 7.20 , which reveal a class of polynomial time solvable special cases of the NNCP.

The gain of using our approach compared to the base model (SPP) is most vibrant in case the fixing pattern is less restrictive and the coupling graph is (highly) symmetric. We test our approach on four types of coupling graphs, for which we explicitly derive the key ingredients of our algorithm, see Table 7.2. Our numerical results show that the gain in efficiency due to the exploitation of symmetries is very large. For each of the 84 real and 180 random instances, the total reduction in the number of variables and constraints is at least $90 \%$ and $89.8 \%$, respectively, and this number grows with $n$ and $m$. The computation times are significantly reduced compared to the unreduced model, resulting in solving NNCP instances that are much larger than the ones considered so far in the literature. The largest instance we can solve contains 100 qubits and 1047 quantum gates.

Given that we are only at the beginning of the quantum era, related optimization problems such as the NNCP are likely to remain important in the near future. Based on the successful implementation of our symmetry-reduced NNCP algorithm, it would be interesting to consider the NNCP on other quantum architectures having a large symmetry group.

## A <br> Appendices

## A. 1 Dykstra's parallel projection algorithm

In Section 3.5.4 Dykstra's cyclic algorithm is presented to iteratively project onto the polyhedra induced by the BQP cuts. Instead of projecting on each polyhedron one after another, it is also possible to project on all polyhedra simultaneously. This method is refered to as parallel Dykstra. Gaffke and Mathar [150] were the first who proposed this fully simultaneous method. The convergence of this algorithm in Euclidean spaces was shown by Iusem and De Pierro [223] using a construction by Pierra [304]. The approach was later generalized to Hilbert spaces, see e.g., [34].

The idea of the parallel Dykstra algorithm is to project onto each set simultaneously and monitor the sequence of weighted averages of these projections. We present here a tailor-made version of this approach by giving each triangle inequality an equal weight. Let $\theta \in(0,1)$. At the start, we set $X_{\mathcal{Y}}^{0}=X_{e_{i} f_{i} g_{i}}^{0}=M$ for all $\left(e_{i}, f_{i}, g_{i}\right) \in \mathcal{T}, R_{\mathcal{Y}}^{0}=\mathbf{0}$ and $R_{e_{i} f_{i} g_{i}}^{0}=\mathbf{0}$. Moreover, we set $\bar{X}^{0}=M$. Now, for each $k \geq 1$ we iterate:

$$
\left.\begin{array}{rl}
X_{\mathcal{Y}}^{k} & :=\mathcal{P}_{\mathcal{Y}}\left(\bar{X}^{k-1}+R_{\mathcal{Y}}^{k-1}\right) \\
R_{\mathcal{Y}}^{k} & :=\bar{X}^{k-1}+R_{\mathcal{Y}}^{k-1}-X_{\mathcal{Y}}^{k} \\
X_{e_{i} f_{i} g_{i}}^{k} & :=\mathcal{P}_{\mathcal{H}_{e_{i}} f_{i} g_{i}}\left(\bar{X}^{k-1}+R_{e_{i} f_{i} g_{i}}^{k-1}\right) \\
R_{e_{i} f_{i} g_{i}}^{k} & :=\bar{X}^{k-1}+R_{e_{i} f_{i} g_{i}}^{k-1}-X_{e_{i} f_{i} g_{i}}^{k} \\
\bar{X}^{k} & :=\theta X_{\mathcal{Y}}^{k}+(1-\theta) \frac{1}{T} \sum_{\left(e_{i}, f_{i}, g_{i}\right) \in \mathcal{T}} X_{e_{i} f_{i} g_{i}}^{k}
\end{array}\right\} \quad \text { for all }\left(e_{i}, f_{i}, g_{i}\right) \in \mathcal{T} \quad \text { (ParDyk) }
$$

Note that the projections in (ParDyk) can be performed simultaneously, as each projection solely uses information resulting from the previous iterate. Under some regularity conditions, the sequence $\left(\bar{X}^{k}\right)_{k \geq 1}$ in (ParDyk) converges strongly to the solution of the best approximation problem, see $[34,223]$. One of the sufficient conditions for convergence is that $\mathcal{Y}_{\mathcal{T}} \neq \emptyset$,
which always holds in our setting.
Based on a construction by Pierra [304], it follows that the algorithm (ParDyk) is equivalent to the cyclic Dykstra algorithm performed to the following two convex sets in the higher dimensional space $\left(\mathcal{S}^{m+1}\right)^{T+1}:=\mathcal{S}^{m+1} \times \cdots \times \mathcal{S}^{m+1}$ :

$$
\mathbf{s}_{1}:=\mathcal{Y} \times \prod_{(e, f, g) \in \mathcal{T}} \mathcal{H}_{e f g} \quad \text { and } \quad \mathbf{s}_{2}:=\left\{(X, X, \ldots, X) \in\left(\mathcal{S}^{m+1}\right)^{T+1}: X \in \mathcal{S}^{m+1}\right\}
$$

using the inner product $\langle\cdot, \cdot\rangle_{\theta}$ defined as

$$
\left\langle\left(X_{0}, X_{1}, \ldots, X_{T}\right),\left(Y_{0}, Y_{1}, \ldots, Y_{T}\right)\right\rangle_{\theta}:=\theta\left\langle X_{0}, Y_{0}\right\rangle+(1-\theta) \frac{1}{T} \sum_{i=1}^{T}\left\langle X_{i}, Y_{i}\right\rangle .
$$

Preliminary experiments show that the convergence of (ParDyk) in general takes more iterations than the convergence of (CycDyk), where we use the semi-parallel implementation of the latter. This is what one might expect, since in the cyclic version each iterate directly builds on the output of the previous iterates. However, since the projections can be performed simultaneously, the total computation time can still be smaller when implemented on parallel machines. Table A. 1 shows a comparison of both methods within the CP-ALM on a test set of Erdős-Rényi instances implemented on nonparallel machines. Results are presented for different values of $\theta$. In all cases the lower bound obtained by the CP-ALM using (ParDyk) in the subproblem at the moment the iteration limit is reached is weaker than the lower bound obtained from using (CycDyk) in the subproblem. Moreover, since the parallel version takes more iterations to converge, the computation times are significantly larger. We conclude that the use of (CycDyk) is favoured above the use of (ParDyk) within the CP-ALM in both quality and computation time. For that reason, we only use (CycDyk) in the numerical experiments of Section 3.7.

| $p$ | $n$ | $m$ | PRSM |  | CP-ALM using cyclic Dykstra |  | CP-ALM using parallel Dykstra |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  |  | $\theta=0.5$ | $\theta=0.85$ |  | $\theta=0.95$ |  |
|  |  |  | value | times |  |  | value | times | value | times | value | times | value | times |
| 0.3 | 20 | 119 | 319 | 0.331 | 319 | 0.384 | 319 | 0.415 | 319 | 0.378 | 319 | 0.389 |
|  | 25 | 177 | 386 | 1.822 | 386 | 5.437 | 386 | 26.61 | 386 | 24.35 | 386 | 24.01 |
|  | 30 | 280 | 333 | 20.75 | 339 | 96.27 | 335 | 7426 | 333 | 7036 | 333 | 1433 |
| 0.5 | 20 | 195 | 227 | 10.15 | 234 | 92.89 | 229 | 4203 | 227 | 2923 | 227 | 733.7 |
|  | 25 | 327 | 169 | 35.68 | 173 | 92.13 | 170 | 6640 | 169 | 5623 | 169 | 1852 |
|  | 30 | 442 | 198 | 91.71 | 202 | 130.8 | 199 | 12437 | 198 | 10815 | 198 | 3677 |

Table A.1: Performance of CP-ALM using cyclic and parallel Dykstra on a test set of 6 Erdős-Rényi instances with maxIter $=1000$, maxTotalIter $=5000$, numCuts $=150$ and all other parameters the same as given in Section 3.7.

In order to reduce the number of iterations to converge, we can perform a preprocessing step before the $Y$-subproblem is solved using (ParDyk). Suppose this subproblem involves the projection of a matrix $M$ onto $\mathcal{Y}_{\mathcal{T}}$. Since this projection is done iteratively, the length of the sequence before convergence depends on the initial distance between $M$ and $\mathcal{Y}_{\mathcal{T}}$. This distance can be shortened using a simple preprocessing step. This step involves the
projection onto all affine constraints of $\mathcal{Y}_{\mathcal{T}}$. We define:

$$
\mathcal{Y}^{\text {aff }}:=\left\{Y \in \mathcal{S}^{m+1}: Y_{11}=1, \operatorname{diag}(Y)=Y \mathbf{e}_{1}, \operatorname{tr}(Y)=n+1, Y_{e f}=0 \forall(e, f) \in \mathcal{Z}\right\} .
$$

Since $\mathcal{Y}^{\text {aff }}$ is an affine subspace, the projection $\mathcal{P}_{\mathcal{y}^{\text {aff }}}(\cdot)$ onto $\mathcal{Y}^{\text {aff }}$ can be found explicitly. Now, instead of projecting $M$ onto $\mathcal{Y}_{\mathcal{T}}$, we can equivalently project the 'closer' matrix $\mathcal{P}_{\mathcal{Y} \text { aff }}(M)$ onto $\mathcal{Y}_{\mathcal{T}}$, as shown by the following lemma.

Lemma A.1. $\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}(M)=\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}\left(\mathcal{P}_{\mathcal{Y}_{\text {aff }}}(M)\right)$.
Proof. Let $\bar{M}:=\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}(M)$ and $\hat{M}:=\mathcal{P}_{\mathcal{Y}_{\text {aff }}}(M)$. We have to show that $\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}(\hat{M})=\bar{M}$. Using the Kolmogorov conditions, the projection of $\hat{M}$ onto $\mathcal{Y}_{\mathcal{T}}$ is the unique solution s.t.:

$$
\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}(\hat{M}) \in \mathcal{Y}_{\mathcal{T}} \quad \text { and } \quad\left\langle Y-\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}(\hat{M}), \hat{M}-\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}(\hat{M})\right\rangle \leq 0 \quad \text { for all } \quad Y \in \mathcal{Y}_{\mathcal{T}}
$$

Clearly, $\bar{M}$ satisfies the first condition. Moreover,

$$
\langle Y-\bar{M}, \hat{M}-\bar{M}\rangle=\underbrace{\langle Y-\bar{M}, M-\bar{M}\rangle}_{\leq 0}+\underbrace{\langle Y-\bar{M}, \hat{M}-M\rangle}_{=0} \leq 0,
$$

for all $Y \in \mathcal{Y}_{\mathcal{T}}$. Here $\langle Y-\bar{M}, M-\bar{M}\rangle \leq 0$ follows from the Kolmogorov conditions for the projection of $M$ onto $\mathcal{Y}_{\mathcal{T}}$ and the equality $\langle Y-\bar{M}, \hat{M}-M\rangle=0$ follows from the fact that $Y, \bar{M} \in \mathcal{Y}^{\text {aff }}$ and $\hat{M}-M$ is orthogonal to the affine space $\mathcal{Y}^{\text {aff. }}$. We conclude that $\bar{M}=\mathcal{P}_{\mathcal{Y}_{\mathcal{T}}}(\hat{M})$.

Observe that the projection onto the unconstrained simplex $\bar{\Delta}(a):=\left\{x \in \mathbb{R}^{m}: \mathbf{1}^{\top} x=a\right\}$ is given by $\mathcal{P}_{\bar{\Delta}(a)}(x)=x-\frac{1^{\top} x-a}{1^{\top} 1} \mathbf{1}$. Thus the projection of $M$ onto $\mathcal{Y}^{\text {aff }}$ is explicitly given by:

$$
\mathcal{P}_{y^{\text {aff }}}(M)=\mathbf{E}_{11}+T_{\text {inner }}(M)+T_{\text {arrow }}^{*}\left(3 \cdot \mathcal{P}_{\bar{\Delta}(n)}\left(T_{\text {arrow }}(M)\right)\right) .
$$

Solving the $Y$-subproblem is now equivalent to performing the projection onto $\mathcal{Y}^{\text {aff }}$ once and apply (ParDyk) to project $\mathcal{P}_{\mathcal{Y} \text { aff }}(M)$ onto $\mathcal{Y}_{\mathcal{T}}$. Further experiments show that this step indeed reduces the number of iterations, but this reduction is not enough to exceed the performance of (CycDyk).

## A. 2 Additional computational results for Chapter 4

In this section we report additional numerical results. Table A. 2 serves to evaluate a quality of the DNN relaxation (4.11) with additional cuts for large graphs obtained after adding at most $3 n$ cuts in each outer while-loop of Algorithm 4.1. In Table 4.4, Section 4.5 we report the results when the number of added cuts in each outer while-loop is at most $5 n$. Our numerical results show that lower bounds might significantly improve when adding more cuts. Therefore, our final choice for adding cuts for large graphs is $5 n$.

We furthermore give in Table A. 3 additional numerical results for the relaxation (4.11) with additional cuts and $k=2$ for (rather small) instances from the literature. All these instances have been considered in [193]. The first group of instances are grid graphs of Brunetta et al. [57]. These graphs are as follows:

- Planar grid instances: To represent instances of equicut on planar grid graphs we assign a weight from 1 to 10, drawn from a uniform distribution, to the edges of a $h \times k$ planar grid, and a 0 weight to the other edges. The names of those graphs are formed by the size followed by the letter ' g '.
- Toroidal grid instances: Same as planar grid instances but for toroidal grids. The names of those graphs are formed by the size followed by the letter ' $t$ '.
- Mixed grid instances: These are dense instances with all edges having a nonzero weight. The edges of a planar grid receive weights from 10 to 100 uniformly generated and all the other edges a weight from 1 to 10 , also uniformly generated. The names of those graphs are formed by the size followed by the letter ' $m$ '.

The second group of instances are randomly generated graphs from [193]: for a fixed density, the edges are assigned integer weights uniformly drawn from [1,10]. Graphs' names begin with ' $v$ ', ' $t$ ', ' $q$ ', ' $c$ ', and ' $s$ '.

Table A. 3 also includes results on instances constructed with de Bruijn networks by [193], of which the data arise in applications related to parallel computer architecture [79, 129]. Graphs' names begin with 'db'. Finally, we test some instances from finite element meshes from [193], graphs' names begin with ' $m$ '.

There are 64 instances in Table A.3, and we prove optimality for 53 instances. The longest time required to compute a lower bound is 2.5 minutes, and computation of upper bounds is negligible.

| graph | $n$ | $k$ | UB | DNN |  | DNN+cuts |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { imp. } \\ \% \end{gathered}$ | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | \#tri. cuts | \#ind. set cuts | \#iter. | \#outer loops |
| G500.005 | 500 | 2 | 49 | 25 | 45.83 | 41 | 1094.09 | 63.96 | 66.67 | 14593 | 407 | 1560 | 10 |
| G500.01 | 500 | 2 | 218 | 156 | 22.60 | 196 | 222.60 | 25.64 | 64.52 | 14597 | 403 | 1116 | 10 |
| G500.02 | 500 | 2 | 626 | 513 | 15.90 | 554 | 91.75 | 7.99 | 36.28 | 10989 | 464 | 743 | 9 |
| G500.04 | 500 | 2 | 1744 | 1566 | 14.78 | 1613 | 35.03 | 3.00 | 26.40 | 8075 | 743 | 547 | 7 |
| G1000.0025 | 1000 | 2 | 102 | 45 | 421.80 | 72 | 7206.50 | 60.00 | 47.37 | 29796 | 204 | 1979 | 10 |
| G1000.005 | 1000 | 2 | 451 | 307 | 204.65 | 378 | 1400.91 | 23.13 | 49.31 | 29432 | 568 | 1975 | 10 |
| G1000.01 | 1000 | 2 | 1367 | 1113 | 152.35 | 1179 | 495.80 | 5.93 | 25.98 | 21285 | 799 | 1329 | 9 |
| G1000.02 | 1000 | 2 | 3389 | 3007 | 140.69 | 3079 | 374.94 | 2.39 | 18.85 | 15471 | 1242 | 1208 | 6 |
| U500.05 | 500 | 2 | 2 | 1 | 74.43 | 2 | 1391.38 | 100.00 | 100.00 | 15000 | 0 | 3136 | 10 |
| U500.10 | 500 | 2 | 26 | 8 | 39.52 | 19 | 393.38 | 137.50 | 61.11 | 9921 | 5079 | 1770 | 10 |
| U500. 20 | 500 | 2 | 178 | 56 | 66.28 | 135 | 565.41 | 141.07 | 64.75 | 10439 | 4561 | 2782 | 10 |
| U500.40 | 500 | 2 | 412 | 163 | 97.47 | 339 | 1004.65 | 107.98 | 70.68 | 8739 | 6261 | 3326 | 10 |
| U1000.05 | 1000 | 2 | 1 | 0 | 725.58 | 0 | 4498.22 | 0.00 | 0.00 | 30000 | 0 | 6792 | 10 |
| U1000.10 | 1000 | 2 | 39 | 8 | 674.90 | 22 | 2284.10 | 175.00 | 45.16 | 25014 | 4986 | 3583 | 10 |
| U1000.20 | 1000 | 2 | 222 | 51 | 213.67 | 133 | 1840.46 | 160.78 | 47.95 | 21093 | 8907 | 2827 | 10 |
| U1000.40 | 1000 | 2 | 737 | 240 | 561.08 | 571 | 3685.70 | 137.92 | 66.60 | 21429 | 8571 | 3310 | 10 |

Table A.2: Large $G$ and $U$ graphs from [228] (Maxineq=3n).

| graph | $n$ | $k$ | UB | DNN |  | DNN+cuts |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  |  | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | $\begin{gathered} \text { LB } \\ \text { (rounded) } \end{gathered}$ | clocktime <br> (s) | imp. $\%$ | $\begin{gathered} \text { gap closed } \\ \% \end{gathered}$ | \#tri. <br> cuts | \#ind. set cuts | \#iter. | \#outer <br> loops |
| 10x2g | 20 | 2 | 6 | 3 | 0.13 | 6 | 1.09 | 124.48 | 100.00 | 23 | 97 | 367 | 2 |
| $5 \times 6 \mathrm{~g}$ | 30 | 2 | 19 | 12 | 0.04 | 19 | 4.34 | 72.20 | 100.00 | 281 | 169 | 581 | 5 |
| 2x16g | 32 | 2 | 8 | 3 | 0.10 | 8 | 1.91 | 171.05 | 100.00 | 193 | 191 | 341 | 4 |
| $18 \times 2 \mathrm{~g}$ | 36 | 2 | 6 | 3 | 0.19 | 6 | 1.35 | 149.36 | 100.00 | 146 | 178 | 251 | 3 |
| $2 \times 19 \mathrm{~g}$ | 38 | 2 | 6 | 2 | 0.16 | 4 | 29.45 | 194.02 | 50.00 | 692 | 993 | 1184 | 16 |
















| 2 | 1658 | 1593 | 0.05 | 1655 | 4.88 | 3.92 | 95.38 | 725 | 325 | 529 | 7 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 2 | 603 | 556 | 0.06 | 600 | 7.11 | 7.99 | 93.62 | 817 | 315 | 694 | 9 |
| 2 | 368 | 328 | 0.08 | 367 | 22.48 | 11.94 | 97.50 | 967 | 353 | 1124 | 9 |
| 2 | 122 | 102 | 0.08 | 122 | 1.45 | 19.69 | 100.00 | 421 | 179 | 348 | 4 |
| 2 | 123 | 97 | 0.06 | 123 | 2.71 | 27.58 | 100.00 | 543 | 237 | 381 | 5 |
| 2 | 160 | 135 | 0.07 | 160 | 2.28 | 18.54 | 100.00 | 577 | 233 | 371 | 5 |
| 2 | 177 | 150 | 0.11 | 177 | 2.79 | 18.60 | 100.00 | 623 | 217 | 415 | 5 |
| 2 | 227 | 186 | 0.13 | 223 | 13.69 | 20.32 | 90.24 | 1219 | 347 | 861 | 9 |
| 2 | 238 | 201 | 0.09 | 238 | 5.98 | 18.68 | 100.00 | 886 | 374 | 539 | 7 |
| 2 | 4 | 4 | 0.00 | 4 | 0.00 | 4.65 | 0.00 | 0 | 0 | 69 | 0 |
| 2 | 10 | 7 | 0.02 | 10 | 1.41 | 45.26 | 100.00 | 194 | 94 | 349 | 3 |
| 2 | 18 | 11 | 0.10 | 18 | 5.52 | 76.04 | 100.00 | 1020 | 324 | 518 | 7 |
| 2 | 30 | 16 | 0.40 | 30 | 149.44 | 98.16 | 100.00 | 3650 | 958 | 910 | 12 |
| 2 | 6 | 4 | 0.04 | 6 | 0.51 | 63.21 | 100.00 | 138 | 54 | 163 | 2 |
| 2 | 2 | 1 | 0.11 | 2 | 0.26 | 169.34 | 100.00 | 132 | 30 | 181 | 1 |
| 2 | 3 | 2 | 0.14 | 3 | 0.40 | 149.76 | 100.00 | 90 | 90 | 171 | 1 |
| 2 | 7 | 3 | 0.19 | 7 | 2.39 | 140.55 | 100.00 | 475 | 365 | 351 | 4 |
| 2 | 4 | 2 | 0.23 | 4 | 5.17 | 264.83 | 100.00 | 626 | 706 | 446 | 6 |
| 2 | 6 | 3 | 0.16 | 6 | 2.22 | 163.69 | 100.00 | 420 | 468 | 342 | 4 |
| 2 | 4 | 2 | 0.24 | 4 | 4.79 | 263.41 | 100.00 | 567 | 873 | 448 | 6 |
| 2 | 4 | 2 | 0.31 | 4 | 5.15 | 287.21 | 100.00 | 915 | 705 | 437 | 6 |
| 2 | 4 | 1 | 0.78 | 4 | 47.39 | 351.32 | 100.00 | 2461 | 2639 | 922 | 17 |
| 2 | 9 | 3 | 2.33 | 7 | 49.85 | 160.59 | 66.67 | 2639 | 3146 | 864 | 14 |



Table A.3: Graphs considered in the paper of Hager, Phan and Zhang [193] (Maxineq=3n).

## A. 3 Derivation of subtour elimination constraints as CG cuts

In this appendix we elaborate on the construction of the five types of subtour elimination constraints given in Table 5.1 as $(S$-)CG cuts.

## A.3.1 Ordinary subtour elimination constraint

Let $S \subseteq N$ with $|S|<n$. The well-known subtour elimination constraint corresponding to $S$ can be obtained as a CG cut, see also [68]. Let $\mathbb{1}_{\mathbf{S}}$ be the indicator vector of the set of nodes $S$. Then,

$$
\left\langle\mathbb{1}_{\mathbf{S}} \mathbb{1}_{\mathbf{S}}^{\top}, \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right)\right\rangle \geq 0
$$

is a valid cut. Applying the CG procedure to this cut, yields

$$
\left\langle\mathbb{1}_{S} \mathbb{1}_{S}^{\top}, \frac{1}{2}\left(X+X^{\top}\right)\right\rangle \leq\left\lfloor\left\langle\mathbb{1}_{S} \mathbb{1}_{S}^{\top}, \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}\right\rangle\right\rfloor
$$

or equivalently,

$$
\sum_{i \in S, j \in S} x_{i j} \leq\lfloor|S|(\beta+\alpha|S|)\rfloor
$$

If $\beta=k_{n}$ and $\alpha=h_{n} / n$, then for all $S$ with $|S|<n$ we have $\beta+\alpha|S|<1$. Hence, the CG cut above implies

$$
\begin{equation*}
\sum_{i \in S, j \in S} x_{i j} \leq|S|-1 \tag{A.1}
\end{equation*}
$$

The cut (A.1) is the common subtour elimination constraint introduced by Dantzig et al. [91].

## A.3.2 Cut-set subtour elimination constraints

The cut-set subtour elimination constraints are known to be equivalent to the ordinary subtour elimination constraints of [91]. It is therefore no surprise that these cuts can be obtained similarly as the ordinary subtour elimination constraints.

Let $U=\mathbb{1}_{S} \mathbb{1}_{S}{ }^{\top}$ be the dual multiplier of the LMI $\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right) \succeq \mathbf{0}$ and let $\mathbb{1}_{S}$ be the dual multiplier of the constraints $-X \mathbf{1}=\mathbf{- 1}$. The sum of these constraints yields

$$
\left\langle\mathbb{1}_{S} \mathbb{1}_{S}{ }^{\top}, \frac{1}{2}\left(X+X^{\top}\right)\right\rangle-\mathbb{1}_{S}^{\top} X \mathbf{1} \leq\left\lfloor\left\langle\mathbb{1}_{S} \mathbb{1}_{S}^{\top}, \beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}\right\rangle-\mathbb{1}_{S}^{\top} \mathbf{1}\right\rfloor,
$$

or equivalently,

$$
-\sum_{i \in S, j \notin S} x_{i j} \leq\lfloor|S|(\beta+\alpha|S|)\rfloor-|S|
$$

If $\beta=k_{n}$ and $\alpha=h_{n} / n$, then the right-hand side becomes $|S|-1-|S|=-1$, which yields the desired cut.

## A.3.3 Merged subtour elimination constraint

Let $\left(S_{1}, \ldots, S_{k}\right)$ be a partition of the node set of $G$, i.e., $\bigcup_{l=1}^{k} S_{l}=N$ and $S_{l} \cap S_{p}=\emptyset$ for all $l \neq p$. We can obtain a merged subtour elimination constraint via the CG procedure in the following way.

Let $U=2 \sum_{l=1}^{k} \mathbb{1}_{S_{l}} \mathbb{1}_{S_{l}}{ }^{\top}$ be the dual multiplier for $\beta \mathbf{I}_{n}+\alpha \mathbf{J}_{n}-\frac{1}{2}\left(X+X^{\top}\right)$. Since each dual multiplier $\mathbb{1}_{S_{l}} \mathbb{1}_{S_{l}}{ }^{\top}$ leads to a CG cut of Type I in Table 5.1, its weighted sum also belongs to the elementary closure and looks as follows:

$$
2 \sum_{l=1}^{k} \sum_{\substack{i \in S_{l} \\ j \in S_{l}}} x_{i j} \leq 2 \sum_{l=1}^{k}\left(\left|S_{l}\right|-1\right)=2(n-k) .
$$

Now we add to this cut the equality $-X \mathbf{1}=-\mathbf{1}$ with dual multiplier $\mathbf{1}$, which yields the desired merged cut

$$
2 \sum_{l=1}^{k} \sum_{\substack{i \in S_{l} \\ j \in S_{l}}} x_{i j}-\mathbf{1}^{\top} X \mathbf{1} \leq 2(n-k)-\mathbf{1}^{\top} \mathbf{1}
$$

which is equivalent to

$$
\sum_{l=1}^{k} \sum_{\substack{i \in S_{l} \\ j \in S_{l}}} x_{i j}-\sum_{l \neq p} \sum_{\substack{i \in S_{l} \\ j \in S_{p}}} x_{i j} \leq n-2 k
$$

## A.3.4 Strengthened subtour elimination constraints of size two

Let $i \neq j$ and define $U=\mathbb{1}_{\{i, j\}} \mathbb{1}_{\{i, j\}}{ }^{\top}$. Taking $U$ as the dual multiplier with respect to $\beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}-\frac{1}{2}\left(\left(X+X^{(2)}\right)+\left(X+X^{(2)}\right)^{\top}\right) \succeq \mathbf{0}$, provides the following valid cut:

$$
\left\langle\mathbb{1}_{\{i, j\}} \mathbb{1}_{\{i, j\}}^{\top}, \beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}-\frac{1}{2}\left(\left(X+X^{(2)}\right)+\left(X+X^{(2)}\right)^{\top}\right)\right\rangle \geq 0 .
$$

By adding the coupling constraints $\sum_{k \in N:(i, k, j) \in \mathcal{A}} y_{i k j}-x_{i j}^{(2)}=0$ and $\sum_{k \in N:(j, k, i) \in \mathcal{A}} y_{j k i}-$ $x_{j i}^{(2)}=0$, each with dual multiplier 1, and the constraints $-x_{i i}=0,-x_{j j}=0,-x_{i i}^{(2)}=0$ and $-x_{j j}^{(2)}=0$, also each with dual mulitplier 1 , we obtain

$$
x_{i j}+x_{j i}+\sum_{\substack{k \in N: \\(i, k, j) \in \mathcal{A}}} y_{i k j}+\sum_{\substack{k \in N: \\(j, k, i) \in \mathcal{A}}} y_{j k i} \leq 2 \beta^{(2)}+4 \alpha^{(2)} .
$$

We now take $\beta^{(2)}=k_{n}^{(2)}$ and $\alpha^{(2)}=h_{n}^{(2)} / n$. Applying the standard CG procedure to this inequality results in the cut

$$
\begin{equation*}
x_{i j}+x_{j i}+\sum_{\substack{k \in N: \\(i, k, j) \in \mathcal{A}}} y_{i k j}+\sum_{\substack{k \in N: \\(j, k, i) \in \mathcal{A}}} y_{j k i} \leq\left\lfloor 2 k_{n}^{(2)}+4 \frac{h_{n}^{(2)}}{n}\right\rfloor . \tag{A.2}
\end{equation*}
$$

The right-hand side of this cut equals one if $5 \leq n \leq 7$, two if $8 \leq n \leq 12$ and three if $n \geq 13$.

For $n \geq 5$, we can strengthen this cut by applying the $S$-CG procedure as explained in Section 5.2.5. Since the cut (A.2) only involves variables $y$ and $X$, we can restrict the set $S$ to the space corresponding to these variables. Let $S=\mathcal{F}_{1} \cap\left(\{0,1\}^{\mathcal{A}} \times \mathcal{T}_{n}(G)\right)$ and let $c_{1}$ be the coefficient vector of the left-hand side in (A.2). Then the strengthened rounding looks as follows:

$$
\left\lfloor 2 k_{n}^{(2)}+\left.4 \frac{h_{n}^{(2)}}{n}\right|_{S, c_{1}}:=\max \left\{x_{i j}+x_{j i}+\sum_{\substack{k \in N: \\(i, k, j) \in \mathcal{A}}} y_{i k j}+\sum_{\substack{k \in N: \\(j, k, i) \in \mathcal{A}}} y_{j k i}:(\mathrm{A} .2),(y, X) \in S\right\} .\right.
$$

One can verify that the value of this maximization is equal to 1 for $n \geq 5$. Namely, if it would be greater than 1 , this implies a subtour of size two (if $x_{i j}=x_{j i}=1$ ), size three (e.g., if $x_{i j}=1$ and $y_{j k i}=1$ for some $k \in N \backslash\{i, j\}$ ) or size four (e.g., if $y_{i k j}=1$ and $y_{j l i}=1$ for some distinct $k, l \in N \backslash\{i, j\}$ ), which contradicts the fact that $X \in \mathcal{T}_{n}(G)$. We conclude that $\left\lfloor 2 k_{n}^{(2)}+4 \frac{h_{n}^{(2)}}{n}\right\rfloor_{S, c_{1}}=1$. Thus, we obtain the strengthened CG cut

$$
x_{i j}+x_{j i}+\sum_{\substack{k \in N: \\(i, k, j) \in \mathcal{A}}} y_{i k j}+\sum_{\substack{k \in N: \\(j, k, i) \in \mathcal{A}}} y_{j k i} \leq 1 .
$$

## A.3.5 Strenghtened subtour elimination constraints

Let $S \subset N$ with $2 \leq|S|<\frac{1}{2} n$ and define $U=\mathbb{1}_{S} \mathbb{1}_{S}{ }^{\top}$. Taking $U$ as the dual multiplier with respect to $\beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}-\frac{1}{2}\left(\left(X+X^{(2)}\right)+\left(X+X^{(2)}\right)^{\top}\right) \succeq \mathbf{0}$ provides the inequality

$$
\left\langle\mathbb{1}_{S} \mathbb{1}_{S}^{\top}, \beta^{(2)} \mathbf{I}_{n}+\alpha^{(2)} \mathbf{J}_{n}-\frac{1}{2}\left(\left(X+X^{(2)}\right)+\left(X+X^{(2)}\right)^{\top}\right)\right\rangle \geq 0 .
$$

For all $i, j \in S$ we now add the coupling constraints $\sum_{k \in N:(i, k, j) \in \mathcal{A}} y_{i k j}-x_{i j}^{(2)}=0$ with dual multiplier 1. Moreover, for all $(i, k, j) \in \mathcal{A}$ with $i, k, j \in S$ we add the constraint $-y_{i k j} \leq 0$ with multiplier 1 . This yields the following valid cut

$$
\sum_{\substack{i \in S \\ j \in S}} x_{i j}+\sum_{\substack{i \in S \\ j \in S}} \sum_{\substack{k \in N \backslash S: \\(i, k, j) \in \mathcal{A}}} y_{i k j} \leq|S| \beta^{(2)}+|S|^{2} \alpha^{(2)} .
$$

Again, we take $\beta^{(2)}=k_{n}^{(2)}$ and $\alpha^{(2)}=h_{n}^{(2)} / n$. The standard CG rounding step yields

$$
\begin{equation*}
\sum_{\substack{i \in S \\ j \in S}} x_{i j}+\sum_{\substack{i \in S \\ j \in S}} \sum_{\substack{k \in N \backslash S: \\(i, k, j) \in \mathcal{A}}} y_{i k j} \leq\left\lfloor|S|\left(k_{n}^{(2)}+|S| \frac{h_{n}^{(2)}}{n}\right)\right\rfloor . \tag{A.3}
\end{equation*}
$$

Since $|S|<\frac{1}{2} n$, we know

$$
\left\lfloor|S|\left(k_{n}^{(2)}+|S| \frac{h_{n}^{(2)}}{n}\right)\right\rfloor \leq\left\lfloor|S|\left(k_{n}^{(2)}+\frac{1}{2} n \frac{2-k_{n}^{(2)}}{n}\right)\right\rfloor=\left\lfloor|S|\left(1+\frac{1}{2} k_{n}^{(2)}\right)\right\rfloor \leq 2|S|-1 .
$$

However, similar to the approach in Appendix A.3.4, we obtain a tighter bound if we apply the strengthened CG procedure. Let $T=\mathcal{F}_{1} \cap\left(\{0,1\}^{\mathcal{A}} \times \mathcal{T}_{n}(G)\right)$ and let $c_{2}$ be the coefficient
vector of the left-hand side of (A.3). Then,

$$
\left||S|\left(k_{n}^{(2)}+|S| \frac{h_{n}^{(2)}}{n}\right)\right|_{T, c_{2}}:=\max \left\{\sum_{\substack{i \in S \\ j \in S}} x_{i j}+\sum_{\substack{i \in S \\ j \in S}} \sum_{\substack{k \in N \backslash, \backslash: \\(i, k, j) \in \mathcal{A}}} y_{i k j}:(\mathrm{A} .3),(y, X) \in T\right\} .
$$

It can be verified that this maximum is equal to $|S|-1$ for all $S$ with $|S|<\frac{1}{2} n$. Namely, if $(y, X) \in T$, we cannot have both $x_{i j}=1$ and $y_{i k j}=1$ for some $k \in N$. Therefore, $x_{i j}+\sum_{k \in N \backslash S:(i, k, j) \in \mathcal{A}} y_{i k j} \leq 1$ for all $i, j \in S$. If we now sum over all $i, j \in S$, the result must be at most $|S|-1$, otherwise a subtour would exist. The strengthened CG cut corresponding to (A.3) becomes

$$
\sum_{\substack{i \in S \\ j \in S}} x_{i j}+\sum_{\substack{i \in S \\ j \in S}} \sum_{\substack{k \in N \backslash S: \\(i, k, j) \in \mathcal{A}}} y_{i k j} \leq|S|-1 .
$$

## A. 4 The symmetric quadratic traveling salesman problem

In this appendix we briefly consider the symmetric quadratic traveling salesman problem (SQTSP). Although this problem is very related to the asymmetric version (that we continue to denote by QTSP), the underlying model is different. We show how to construct this model and how all cuts for the QTSP can be extended to the symmetric case.

Let $G=(V, E)$ be an undirected graph, where $E$ consists of undirected pairs of nodes $\{i, j\}$. We define $\mathcal{E}=\{\langle i, j, k\rangle=\langle k, j, i\rangle: i, j, k \in V,|\{i, j, k\}|=3\}$ to be the set of two-edges in $G$, where a two-edge is a sequence of three distinct nodes where the reverse sequence is regarded as identical. Given is a cost matrix $Q=\left(q_{i j k}\right)$, where a cost is zero if $\langle i, j, k\rangle \notin \mathcal{E}$.

The goal of the SQTSP is to find an undirected Hamiltonian cycle in $G$ such that the total quadratic cost is minimized. To model this problem, let $\bar{x} \in\{0,1\}^{E}$ and $\bar{y} \in\{0,1\}^{\varepsilon}$ denote indicator vectors that are 1 if and edge, respectively two-edge, is present in the solution and 0 otherwise. We aim to find a tuple $(\bar{x}, \bar{y})$ with $\bar{y}_{i j k}=\bar{x}_{i j} \bar{x}_{j k}$, representing a Hamiltonian cycle such that $\sum_{\langle i, j k\rangle \in \mathcal{E}} q_{i j k} y_{i j k}$ is minimized.

The symmetric equivalent of the set $\mathcal{F}_{1}$, see (5.34), is now given by:

$$
\mathcal{F}_{1}^{s}:=\left\{(\bar{y}, \bar{x}) \in\{0,1\}^{\mathcal{E}} \times\{0,1\}^{E}: \quad \bar{x}_{i j}=\sum_{\substack{k \in V \\\langle i, j, k\rangle \in \mathcal{E}}} \bar{y}_{i j k}=\sum_{\substack{k \in V\rangle \in \mathcal{L} \\\langle k, i, j\rangle \in \mathcal{E}}} \bar{y}_{k i j} \forall\{i, j\} \in E\right\}
$$

where $\delta(i) \in V$ denotes the set of edges that are incident to $i$. The formulation used in $\mathcal{F}_{1}^{s}$ is introduced by Fischer and Helmberg [137] where it is shown that the equation $\bar{y}_{i j k}=\bar{x}_{i j} \bar{x}_{j k}$ is indeed established for all $\langle i, j, k\rangle \in \mathcal{E}$. Moreover, similar to the asymmetric case, we can relax the integrality of $y$, since it is enforced by the integrality of $\bar{x}$ and the coupling constraints, see Remark 5.35. It follows that the tuples in $\mathcal{F}_{1}^{s}$ are characteristic vectors of node-disjoint cycle covers in $G$, where the smallest cycles have size 3 due to the definition of $\mathcal{E}$.

The B\&C algorithm presented in Section 5.3 can now be applied to the SQTSP, starting from optimizing over $\mathcal{F}_{1}^{s}$. In order to cut off solutions that do not correspond to a Hamilto-
nian cycle in $G$, we need separation routines for the symmetric version. Instead of providing symmetric equivalents to all QTSP cutting planes derived in Section 5.4.3, we present a transformation that maps any valid cut for the asymmetric version to a cut for the SQTSP. To that end, we introduce a directed graph $H=(V, A)$ that is defined on the same set of nodes as the undirected graph $G$, where $A$ is such that it contains both pairs $(i, j)$ and $(j, i)$ whenever the corresponding edge $\{i, j\}$ is contained in $G$. Moreover, we define the cost of each two-arc $(i, j, k)$ in $H$ to be equal to $q_{\langle i, j, k\rangle}$ for the corresponding two-edge $\langle i, j, k\rangle$ in $G$. Let $\mathcal{I}_{\text {S }}$ denote the original SQTSP instance and let $\mathcal{I}_{\mathrm{A}}$ denote the corresponding asymmetric instance.

The variables in the two programs can now be related as follows: Let $(y, X)$ be variables in $\mathcal{I}_{\mathrm{A}}$ and define the tuple $(\bar{y}, \bar{x})$ by

$$
\begin{aligned}
\bar{x}_{i j} & =x_{i j}+x_{j i} \quad \text { for all } \quad & \{i, j\} \in E \\
\bar{y}_{i j k} & =y_{i j k}+y_{k j i} \quad \text { for all } & \langle i, j, k\rangle \in \mathcal{E}
\end{aligned}
$$

From the constraints in $\mathcal{F}_{1}$ and $\mathcal{F}_{1}^{s}$, it follows that any solution $(y, X)$ in $\mathcal{I}_{\mathrm{A}}$ leads to a solution $(\bar{y}, \bar{x})$ in $\mathcal{I}_{\mathrm{S}}$ with the same objective value. Conversely, any solution $(\bar{y}, \bar{x})$ in $\mathcal{I}_{\mathrm{S}}$ leads to a solution (or actually two solutions, one for each direction) $(y, X)$ in $\mathcal{I}_{\mathrm{A}}$ with the same objective value. As a result, any valid cut for $\mathcal{I}_{\mathrm{A}}$ is also valid for $\mathcal{I}_{\mathrm{S}}$.

This implies that all cuts defined in Section 5.4 .3 can be converted to cuts for the SQTSP. Namely, given a cut for $\mathcal{I}_{\mathrm{A}}$, we define the coefficient on $\bar{x}_{i j}$ to be the sum of the coefficients on $x_{i j}$ and $x_{j i}$ for all edges $\{i, j\} \in E$. Similarly, we define the coefficient on $\bar{y}_{i j k}$ to be the sum of the coefficients on $y_{i j k}$ and $y_{k j i}$ for all two-edges $\langle i, j, k\rangle \in \mathcal{E}$. If no more violated cuts can be found in $\mathcal{I}_{\mathrm{A}}$, the corresponding solution in $\mathcal{I}_{\mathrm{S}}$ is also optimal. This proves the validity of the $\mathrm{B} \& \mathrm{C}$ algorithm for the symmetric version of the problem.

## A. 5 Additional computational results for Chapter 5

In this appendix we present a complete overview of the computational results from which the summarized results in Section 5.5 follow. We consider the instances from bioinformatics and the reload instances. No additional results are presented for the turn instances, since for these instances the complete overview is already given in Section 5.5.

In all tables showing computation times, the setting that provides the shortest time is presented in bold for each instance. Moreover, a '-' indicates that a given algorithm could not solve the instance within 8 hours.

The computation times and the number of branching nodes for the class of 'bma' instances from bioinformatics are given in Table A. 4 and A.5, respectively. Table A. 6 and A. 7 provide computation times and number of branching nodes for the 'map' instances, respectively. The computation times and the number of branching nodes for the ' ml ' instances are presented in Table A. 8 and A.9, respectively.

Finally, we present a more elaborate overview of the reload instances. For each of the two classes and for different values of $n, p$ and $c$, average results over 10 randomly generated instances are presented. Table A. 10 and A. 11 present the computation times and number of branching nodes, respectively. Table A. 12 shows the computation times on 72 additional reload instances on both reload classes with $n \in\{21, \ldots, 26\}, p \in\{0.5,0.8\}$ and $c \in\{5,10,20\}$ in order to further investigate the difference between the settings SEC-simple and SEC. As indicated in Section 5.5, the results in Table A. 12 are still not decisive on which of the two settings performs better.

| Instance | SCIP-SDP | KT | CG1 | CG2 | SEC-simple | SEC | SEC-CG |
| :--- | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| bma2_3 | $\mathbf{0 . 0 0 1}$ | 0.001 | 0.006 | 0.153 | $\mathbf{0 . 0 0 1}$ | $\mathbf{0 . 0 0 1}$ | $\mathbf{0 . 0 0 1}$ |
| bma2_4 | 0.075 | $\mathbf{0 . 0 0 1}$ | 0.002 | 0.002 | 0.002 | 0.002 | $\mathbf{0 . 0 0 1}$ |
| bma2_5 | 0.079 | $\mathbf{0 . 0 0 5}$ | 0.004 | 0.017 | $\mathbf{0 . 0 0 5}$ | $\mathbf{0 . 0 0 5}$ | 0.008 |
| bma2_6 | 0.221 | 0.034 | 0.034 | 0.047 | $\mathbf{0 . 0 2 9}$ | 0.555 | 0.101 |
| bma2_7 | 0.436 | 0.048 | 0.051 | 0.062 | 0.054 | 0.157 | $\mathbf{0 . 0 3 9}$ |
| bma2_8 | 2.76 | 0.083 | 0.073 | 0.081 | 0.090 | $\mathbf{0 . 0 4 9}$ | 0.102 |
| bma2_9 | 3.85 | 0.114 | 0.121 | 0.165 | 0.120 | $\mathbf{0 . 0 2 8}$ | 0.043 |
| bma2_10 | 17.11 | 0.179 | 0.205 | 0.374 | 0.185 | $\mathbf{0 . 0 5 6}$ | 0.29 |
| bma2_11 | 91.38 | 0.496 | 0.442 | 0.841 | 0.272 | 0.281 | $\mathbf{0 . 2 5 1}$ |
| bma2_12 | 440.6 | 0.667 | 0.8 | 1.428 | 0.333 | $\mathbf{0 . 2 0 3}$ | 0.303 |
| bma2_13 | 1327 | 2.005 | 1.006 | 2.954 | 0.501 | $\mathbf{0 . 1 5 4}$ | 0.671 |
| bma2_14 | 4794 | 2.76 | 2.524 | 4.396 | 2.190 | $\mathbf{0 . 5 8 2}$ | 0.82 |
| bma2_15 | 13075 | 7.118 | 4.136 | 7.313 | 2.811 | $\mathbf{0 . 4 8 8}$ | 1.27 |
| bma2_16 | - | 11.83 | 6.129 | 12.24 | 2.962 | $\mathbf{0 . 2 4 7}$ | 1.66 |
| bma2_17 | - | 24.19 | 6.214 | 23.87 | 4.509 | $\mathbf{1 . 4 5 7}$ | 3.105 |
| bma2_18 | - | 58.43 | 16.42 | 84.56 | 9.579 | $\mathbf{1 . 4 8 9}$ | 3.699 |
| bma2_19 | - | 49.98 | 20.66 | 54.24 | 13.97 | $\mathbf{1 . 3 8 6}$ | 3.043 |
| bma2_20 | - | 98.18 | 20.37 | 127.9 | 20.40 | $\mathbf{1 . 5 1 6}$ | 3.496 |
| bma2_21 | - | 1068 | 45.41 | 103.2 | 19.64 | 2.387 | $\mathbf{2 . 0 8 6}$ |
| bma2_22 | - | 2120 | 59.95 | 464.3 | 52.09 | $\mathbf{3 . 0 9 3}$ | 8.663 |
| bma2_23 | - | 3855 | 117 | 527.0 | 87.85 | 4.76 | 9.586 |
| bma2_24 | - | 2461 | 90.07 | 945.9 | 81.87 | $\mathbf{5 . 8 6 6}$ | 17.73 |
| bma2_25 | - | 26594 | 160 | 1129 | 166.3 | $\mathbf{7 . 5 3}$ | 7.765 |
| bma2_26 | - | - | 451.6 | 3777 | 269.2 | $\mathbf{6 . 7 7 9}$ | 38.962 |
| bma2_27 | - | - | 372.6 | 8718 | 350.0 | $\mathbf{3 . 1 0 8}$ | 46.95 |
| bma2_28 | - | - | 1628 | - | 440.7 | $\mathbf{2 2 . 4 7}$ | 46.5 |
| bma2_29 | - | - | 2095 | - | 2700 | $\mathbf{1 0 . 6 5}$ | 229 |
| bma2_30 | - | - | 2453 | - | 1614 | $\mathbf{2 3 . 5 9}$ | 42.8 |
| bma2_31 | - | - | 5997 | - | 3434 | $\mathbf{3 2 . 7 2}$ | 63.66 |
| bma2_32 | - | - | 11835 | - | 2227 | $\mathbf{4 1 . 7 9}$ | 168.2 |
| bma2_33 | - | - | - | - | 7985 | $\mathbf{4 2 . 9 7}$ | 150 |
| bma2_34 | - | - | - | - | 6673 | $\mathbf{6 7 . 8}$ | 845.3 |
| bma2_35 | - | - | - | - | - | 98.17 | $\mathbf{7 7 . 0 3}$ |
| bma2_36 | - | - | - | - | - | 173.3 | $\mathbf{1 5 . 2 6}$ |
| bma2_37 | - | - | - | - | - | 101.1 | $\mathbf{3 5 . 6 4}$ |
| bma2_38 | - | - | - | - | - | $\mathbf{9 6 . 3 3}$ | 2127 |
| bma2_39 | - | - | - | - | - | $\mathbf{8 6 . 4 6}$ | 1012 |
| bma2_40 | - | - | - | - | - | $\mathbf{2 3 6 . 2}$ | 1964 |
|  |  |  |  |  |  |  |  |

Table A.4: Computation times (s) for bioinformatics instances from the 'bma' class.

| Instance | SCIP-SDP | KT | CG1 | CG2 | SEC-simple | SEC | SEC-CG |
| :--- | ---: | ---: | ---: | ---: | ---: | ---: | ---: |
| map2_3 | 0 | 0.004 | O | 0.001 | 0.001 | 0.001 | o |
| map2_4 | 0.132 | $\mathbf{0 . 0 0 1}$ | $\mathbf{0 . 0 0 1}$ | $\mathbf{0 . 0 0 1}$ | 0.002 | $\mathbf{0 . 0 0 1}$ | $\mathbf{0 . 0 0 1}$ |
| map2_5 | 0.149 | $\mathbf{0 . 0 0 5}$ | 0.007 | 0.088 | $\mathbf{0 . 0 0 5}$ | 0.006 | 0.022 |
| map2_6 | 0.291 | 0.033 | 0.026 | 0.042 | 0.051 | $\mathbf{0 . 0 2 1}$ | 0.029 |
| map2_7 | 1.551 | 0.06 | 0.057 | 0.147 | 0.058 | $\mathbf{0 . 0 2 7}$ | 0.032 |
| map2_8 | 3.175 | 0.094 | 0.095 | 0.104 | 0.064 | $\mathbf{0 . 0 4 3}$ | 0.056 |
| map2_9 | 8.373 | 0.128 | 0.096 | 0.154 | 0.106 | $\mathbf{0 . 0 3}$ | 0.044 |
| map2_10 | 34.23 | 0.254 | 0.193 | 0.376 | 0.189 | $\mathbf{0 . 0 9 1}$ | 0.324 |
| map2_11 | 143.8 | 0.464 | 0.512 | 0.671 | 0.227 | 0.127 | $\mathbf{0 . 0 9 4}$ |
| map2_12 | 436.4 | 0.944 | 0.99 | 1.316 | 0.628 | $\mathbf{0 . 1 7 1}$ | 0.233 |
| map2_13 | 1393 | 1.751 | 1.268 | 3.007 | 0.507 | $\mathbf{0 . 1 4 9}$ | 0.189 |
| map2_14 | 5342 | 2.718 | 1.532 | 9.388 | 0.887 | $\mathbf{0 . 5 4 6}$ | 0.856 |
| map2_15 | 21851 | 4.57 | 2.691 | 8.088 | 1.778 | $\mathbf{0 . 7 4 2}$ | 1.17 |
| map2_16 | - | 11.95 | 4.292 | 16.69 | 2.826 | $\mathbf{0 . 9 8 5}$ | 1.627 |
| map2_17 | - | 14.2 | 6.987 | 17.79 | 9.434 | $\mathbf{1 . 6 5 4}$ | 2.622 |
| map2_18 | - | 29.77 | 10.03 | 49.97 | 12.14 | $\mathbf{1 . 2 4}$ | 4.798 |
| map2_19 | - | 106.1 | 15.15 | 62.60 | 16.11 | 1.913 | $\mathbf{1 . 2 6 6}$ |
| map2_20 | - | 70.09 | 22.55 | 120.70 | 18.61 | $\mathbf{2 . 6 2 1}$ | 4.935 |
| map2_21 | - | 3831 | 29.79 | 162.26 | 23.55 | 3.384 | $\mathbf{1 . 9 4 3}$ |
| map2_22 | - | 1891 | 57.3 | 533.68 | 53.73 | $\mathbf{3 . 4 9 5}$ | 6.949 |
| map2_23 | - | 7268 | 107.5 | 847.80 | 108.8 | $\mathbf{3 . 3 3 4}$ | 8.991 |
| map2_24 | - | 24633 | 214.3 | 1346 | 80.45 | $\mathbf{5 . 4 8 7}$ | 39.06 |
| map2_25 | - | - | 517.4 | 1159 | 187.9 | 14.19 | $\mathbf{1 2 . 1 8}$ |
| map2_26 | - | - | 1097 | 1874 | 280.3 | $\mathbf{1 2 . 6 7}$ | 16.07 |
| map2_27 | - | - | 675.5 | 13952 | 373.0 | $\mathbf{1 . 9 5 1}$ | 22.09 |
| map2_28 | - | - | 3809 | - | 415.5 | $\mathbf{1 2 . 5 8}$ | 98.8 |
| map2_29 | - | - | 1724 | - | 740.6 | $\mathbf{1 5 . 7 6}$ | 84.06 |
| map2_30 | - | - | 25983 | - | 1236 | $\mathbf{1 5 . 1 3}$ | 236.9 |
| map2_31 | - | - | 3090 | - | 1109 | $\mathbf{4 1 . 9 1}$ | 74.52 |
| map2_32 | - | - | 15660 | - | 8951 | $\mathbf{3 6 . 1 5}$ | 200.8 |
| map2_33 | - | - | - | - | 14630 | $\mathbf{5 1 . 8 8}$ | 239.5 |
| map2_34 | - | - | - | - | - | $\mathbf{8 0 . 5}$ | 452.2 |
| map2_35 | - | - | - | - | - | $\mathbf{1 2 . 6 8}$ | 35.15 |
| map2_36 | - | - | - | - | - | 154.6 | $\mathbf{1 6 . 8 7}$ |
| map2_37 | - | - | - | - | - | $\mathbf{6 6 . 7 7}$ | 990.2 |
| map2_38 | - | - | - | - | - | $\mathbf{2 0 0 . 2}$ | 2564 |
| map2_39 | - | - | - | - | - | $\mathbf{7 8 . 2 8}$ | 1333 |
|  | - | - | - | - | $\mathbf{1 6 0 . 1}$ | 1135 |  |
|  |  |  |  |  |  |  |  |

Table A.6: Computation times (s) for bioinformatics instances from the 'map' class.

| Instance | SCIP-SDP | KT | CG1 | CG2 | SEC-simple | SEC | SEC-CG |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| m12_3 | 0 | 0.001 | 0 | 0.001 | 0 | 0 | 0.001 |
| m12_4 | 0.056 | 0.002 | 0.001 | 0.002 | 0.001 | 0.001 | 0.001 |
| ml2_5 | 0.089 | 0.004 | 0.004 | 0.027 | 0.007 | 0.003 | 0.019 |
| ml2_6 | 0.182 | 0.032 | 0.034 | 0.035 | 0.044 | 0.015 | 0.015 |
| ml2_7 | 0.402 | 0.064 | 0.057 | 0.063 | 0.056 | 0.024 | 0.04 |
| ml2_8 | 1.769 | 0.075 | 0.075 | 0.100 | 0.077 | 0.039 | 0.089 |
| ml2_9 | 4.528 | 0.118 | 0.116 | 0.172 | 0.112 | 0.026 | 0.04 |
| ml2_10 | 11.19 | 0.196 | 0.225 | 0.419 | 0.151 | 0.109 | 0.302 |
| m12_11 | 115.9 | 0.56 | 0.407 | 0.753 | 0.243 | 0.193 | 0.19 |
| m12_12 | 734.2 | 1.094 | 1.019 | 1.530 | 0.331 | 0.208 | 0.176 |
| m12_13 | 2544 | 1.379 | 1.032 | 4.594 | 0.615 | 0.302 | 0.334 |
| m12_14 | 9235 | 3.404 | 2.218 | 7.348 | 1.117 | 0.455 | 0.647 |
| m12_15 | 24930 | 3.827 | 2.397 | 7.858 | 1.468 | 0.633 | 1.888 |
| ml2_16 | - | 7.03 | 7.557 | 14.35 | 1.817 | 0.993 | 1.684 |
| m12_17 | - | 11.08 | 8.268 | 33.36 | 5.035 | 1.282 | 2.008 |
| ml2_18 | - | 34.76 | 13.61 | 54.37 | 7.403 | 0.988 | 3.594 |
| m12_19 | - | 83.41 | 20.42 | 57.72 | 9.350 | 1.337 | 0.611 |
| m12_20 | - | 313.1 | 22.72 | 90.01 | 14.42 | 2.731 | 4.522 |
| ml2_21 | - | 820.5 | 26.01 | 139.9 | 27.21 | 2.82 | 1.226 |
| m12_22 | - | 2159 | 82.52 | 483.6 | 59.51 | 4.997 | 6.803 |
| ml2_23 | - | 15380 | 105.3 | 627.5 | 77.46 | 4.556 | 13.15 |
| m12_24 | - | 10106 | 170.2 | 871.8 | 106.2 | 6.188 | 28.29 |
| ml2_25 | - | - | 497.3 | 1496 | 257.5 | 8.106 | 11.83 |
| $\mathrm{ml2} 26$ | - | - | 339.2 | 1959 | 303.7 | 7.327 | 42.75 |
| m12_27 | - | - | 834.4 | 14290 | 332.8 | 4.965 | 39.99 |
| m12_28 | - | - | 799.3 | - | 1273 | 24.31 | 92.09 |
| m12_29 | - | - | 1905 | - | 1363 | 15.8 | 70.04 |
| ml2_30 | - | - | 2552 | - | 794.5 | 45.74 | 113.8 |
| m12_31 | - | - | 5976 | - | 894.0 | 27.43 | 48.51 |
| ml2_32 | - | - | - | - | 1281 | 29.85 | 394.9 |
| m12_33 | - | - | - | - | 9315 | 39.37 | 460.7 |
| m12_34 | - | - | - | - | - | 96.96 | 273.3 |
| m12_35 | - | - | - | - | - | 99.8 | 69.91 |
| ml2_36 | - | - | - | - | - | 53.87 | 21.26 |
| m12_37 | - | - | - | - | - | 110.2 | 866.7 |
| m12_38 | - | - | - | - | - | 83.92 | 921.9 |
| m12_39 | - | - | - | - | - | 124.6 | 688.3 |
| ml2_40 | - | - | - | - | - | 238.6 | 4244 |

Table A.8: Computation times (s) for bioinformatics instances from the ' ml ' class.

| Instance |  |  | Average computation times (s) |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| Class | $n$ | ( $p, \mathrm{c}$ ) | OPT | SCIP-SDP | KT | CG1 | CG2 | SEC-simple | SEC | SEC-CG |
| 1 | 10 | $(0.5,5)$ | 5.2 | 0.131 | 0.049 | 0.025 | 0.029 | 0.018 | 0.020 | 0.030 |
|  | 10 | $(0.5,10)$ | 5.8 | 0.135 | 0.034 | 0.038 | 0.042 | 0.029 | 0.020 | 0.031 |
|  | 10 | $(0.5,20)$ | 7.7 | 0.218 | 0.022 | 0.023 | 0.035 | 0.024 | 0.017 | 0.033 |
|  | 10 | $(1,5)$ | 2 | 2.135 | 0.177 | 0.160 | 0.210 | 0.163 | 0.152 | 0.236 |
|  | 10 | $(1,10)$ | 3.4 | 1.888 | 0.121 | 0.135 | 0.176 | 0.129 | 0.121 | 0.182 |
|  | 10 | $(1,20)$ | 4.5 | 0.857 | 0.100 | 0.086 | 0.108 | 0.092 | 0.065 | 0.095 |
|  |  | Average | 4.8 | 0.894 | 0.084 | 0.078 | 0.100 | 0.076 | 0.066 | 0.101 |
|  | 15 | $(0.5,5)$ | 4.1 | 3.187 | 0.211 | 0.202 | 0.386 | 0.185 | 0.185 | 0.280 |
|  | 15 | $(0.5,10)$ | 6.5 | 2.131 | 0.160 | 0.182 | 0.197 | 0.146 | 0.134 | 0.235 |
|  | 15 | $(0.5,20)$ | 8.5 | 1.450 | 0.102 | 0.097 | 0.171 | 0.095 | 0.098 | 0.153 |
|  | 15 | $(1,5)$ | 0.4 | 450.1 | 2.657 | 1.720 | 9.566 | 1.884 | 1.835 | 5.209 |
|  | 15 | $(1,10)$ | 2.9 | 204.4 | 1.065 | 1.064 | 2.697 | 0.887 | 0.788 | 2.243 |
|  | 15 | $(1,20)$ | 5.1 | 77.44 | 0.557 | 0.587 | 1.246 | 0.514 | 0.497 | 1.022 |
|  |  | Average | 4.583 | 123.1 | 0.792 | 0.642 | 2.377 | 0.619 | 0.589 | 1.524 |
|  | 20 | $(0.5,5)$ | 3.2 | 169.6 | 0.869 | 0.938 | 2.521 | 0.690 | 0.629 | 1.994 |
|  | 20 | $(0.5,10)$ | 6.1 | 19.67 | 0.543 | 0.541 | 1.111 | 0.424 | 0.473 | 0.957 |
|  | 20 | $(0.5,20)$ | 9.3 | 56.92 | 0.419 | 0.396 | 0.897 | 0.335 | 0.294 | 0.759 |
|  | 20 | $(1,5)$ | 0 | 1985 | 496.3 | 235.2 | 5464 | 458.0 | 102.1 | 9648 |
|  | 20 | $(1,10)$ | 2.143 | 6549 | 48.71 | 32.95 | 245.7 | 24.89 | 23.69 | 174.2 |
|  | 20 | $(1,20)$ | 4.8 | 3189 | 6.223 | 6.422 | 19.26 | 4.733 | 3.808 | 12.04 |
|  |  | Average | 4.257 | 1995 | 92.19 | 46.09 | 955.7 | 81.51 | 21.83 | 1639 |
|  | 25 | $(0.5,5)$ | 2.6 | - | 218.6 | 97.39 | 3390 | 42.47 | 41.41 | 727.6 |
|  | 25 | $(0.5,10)$ | 6.4 | - | 6.876 | 6.253 | 22.05 | 6.361 | 5.287 | 14.56 |
|  | 25 | $(0.5,20)$ | 10.8 | - | 3.029 | 3.356 | 11.725 | 2.544 | 2.127 | 5.156 |
|  |  | Average | 6.6 | - | 76.19 | 35.66 | 1141 | 17.12 | 16.27 | 249.1 |
| 2 | 10 | $(0.5,5)$ | 16.1 | 0.185 | 0.036 | 0.038 | 0.061 | 0.028 | 0.031 | 0.043 |
|  | 10 | $(0.5,10)$ | 22 | 0.117 | 0.031 | 0.038 | 0.043 | 0.025 | 0.023 | 0.040 |
|  | 10 | $(0.5,20)$ | 30.11 | 0.252 | 0.040 | 0.042 | 0.044 | 0.034 | 0.033 | 0.048 |
|  | 10 | $(1,5)$ | 4.6 | 1.316 | 0.260 | 0.227 | 0.166 | 0.110 | 0.201 | 0.200 |
|  | 10 | $(1,10)$ | 8.4 | 0.833 | 0.111 | 0.106 | 0.151 | 0.108 | 0.117 | 0.136 |
|  | 10 | $(1,20)$ | 11.6 | 0.736 | 0.120 | 0.110 | 0.141 | 0.131 | 0.098 | 0.136 |
|  |  | Average | 15.46 | 0.573 | 0.100 | 0.094 | 0.101 | 0.073 | 0.084 | 0.101 |
|  | 15 | $(0.5,5)$ | 17.7 | 1.967 | 0.160 | 0.182 | 0.256 | 0.164 | 0.141 | 0.263 |
|  | 15 | $(0.5,10)$ | 23.3 | 2.476 | 0.198 | 0.150 | 0.241 | 0.183 | 0.169 | 0.259 |
|  | 15 | $(0.5,20)$ | 27.2 | 4.525 | 0.221 | 0.191 | 0.269 | 0.173 | 0.206 | 0.260 |
|  | 15 | $(1,5)$ | 2.1 | 660.9 | 2.440 | 3.639 | 10.80 | 2.430 | 2.051 | 7.077 |
|  | 15 | $(1,10)$ | 6.5 | 118.2 | 0.925 | 0.911 | 1.697 | 0.829 | 0.855 | 1.590 |
|  | 15 | $(1,20)$ | 11.7 | 53.32 | 0.723 | 0.753 | 1.427 | 0.621 | 0.664 | 1.203 |
|  |  | Average | 14.75 | 140.2 | 0.778 | 0.971 | 2.449 | 0.733 | 0.681 | 1.775 |
|  | 20 | $(0.5,5)$ | 8.3 | 72.89 | 0.661 | 0.646 | 1.377 | 0.593 | 0.641 | 1.580 |
|  | 20 | $(0.5,10)$ | 19.2 | 68.31 | 0.544 | 0.528 | 1.057 | 0.545 | 0.517 | 1.289 |
|  | 20 | $(0.5,20)$ | 26.8 | 34.82 | 0.521 | 0.582 | 1.304 | 0.517 | 0.571 | 1.186 |
|  | 20 | $(1,5)$ | 0 | 1313 | 103.2 | 38.98 | 3452 | 16.93 | 33.99 | 2472 |
|  | 20 | $(1,10)$ | 4.125 | 5659 | 23.31 | 17.48 | 95.82 | 13.71 | 12.45 | 68.51 |
|  | 20 | $(1,20)$ | 10.11 | 1094 | 5.451 | 6.179 | 13.59 | 5.029 | 4.203 | 9.496 |
|  |  | Average | 11.42 | 1374 | 22.28 | 10.73 | 594.2 | 6.223 | 8.730 | 425.7 |
|  | 25 | $(0.5,5)$ | 8.3 | - | 3886 | 935.3 | 15452 | 276.2 | 219.9 | 5077 |
|  | 25 | $(0.5,10)$ | 17.7 | - | 5.649 | 5.672 | 17.37 | 4.408 | 4.231 | 13.89 |
|  | 25 | $(0.5,20)$ | 23.1 | - | 4.142 | 4.364 | 9.869 | 3.760 | 3.280 | 10.88 |
|  |  | Average | 16.4 | - | 1298 | 315.1 | 5159 | 94.80 | 75.81 | 1701 |

Table A.10: Computation times of the reload instances averaged over 10 generated instances for given values of $n, p$ and $c$.

| Class | Instance |  | Number of branching nodes |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | $n$ | ( $p, \mathrm{c}$ ) | SCIP-SDP | KT | CG1 | CG2 | SEC-simple | SEC | SEC-CG |
| 1 | 10 | $(0.5,5)$ | 19.7 | 1.1 | 1.7 | 1.4 | 0.7 | 0.7 | 0.8 |
|  | 10 | $(0.5,10)$ | 16.6 | 2.2 | 2.1 | 3.8 | 0.8 | 0.7 | 0.8 |
|  | 10 | $(0.5,20)$ | 43.4 | 4.3 | 3.9 | 0.7 | 0.7 | 1.8 | 2.6 |
|  | 10 | $(1,5)$ | 132 | 125.9 | 173.1 | 168.6 | 74.6 | 69.4 | 99.4 |
|  | 10 | $(1,10)$ | 115 | 79 | 82.4 | 72.3 | 43.2 | 32.5 | 43.4 |
|  | 10 | $(1,20)$ | 46.5 | 15.4 | 15.4 | 11.6 | 9.5 | 4.6 | 7.1 |
|  |  | Average | 62.2 | 37.98 | 46.433 | 43.06 | 21.58 | 18.28 | 25.68 |
|  | 15 | $(0.5,5)$ | 155.9 | 99.4 | 92.9 | 111 | 73.5 | 67 | 53.2 |
|  | 15 | $(0.5,10)$ | 91.8 | 70.9 | 105.1 | 76.9 | 45.7 | 28.4 | 46.8 |
|  | 15 | $(0.5,20)$ | 48.3 | 34.9 | 17.3 | 28.3 | 11.6 | 12.9 | 7.6 |
|  | 15 | $(1,5)$ | 3001 | 3606 | 2506 | 3607 | 2643 | 2215 | 2084 |
|  | 15 | $(1,10)$ | 1692 | 1223 | 1176 | 1293 | 921.2 | 533.8 | 618.6 |
|  | 15 | $(1,20)$ | 405.1 | 279.4 | 170.5 | 241.1 | 157.1 | 140.1 | 108.5 |
|  |  | Average | 899.3 | 885.8 | 678.1 | 893.2 | 642.0 | 499.5 | 486.5 |
|  | 20 | $(0.5,5)$ | 1929 | 1312.8 | 1522 | 1534 | 991 | 637.5 | 685.4 |
|  | 20 | $(0.5,10)$ | 293.6 | 336 | 369.1 | 266.7 | 159.9 | 148.5 | 152 |
|  | 20 | $(0.5,20)$ | 459.4 | 188.4 | 175.6 | 165.9 | 97.7 | 50.3 | 75.7 |
|  | 20 | $(1,5)$ | 13529 | 65622 | 54077 | 44724 | 125560 | 27878 | 67724 |
|  | 20 | $(1,10)$ | 18062 | 29287 | 21583 | 18798 | 15521 | 10308 | 12110 |
|  | 20 | $(1,20)$ | 4749 | 3114 | 3400 | 3312 | 2499 | 1525 | 1507 |
|  |  | Average | 6503 | 16643 | 13521 | 11467 | 24138 | 6758 | 13709 |
|  | 25 | $(0.5,5)$ | - | 131626 | 73671 | 106956 | 45480 | 34390 | 40735 |
|  | 25 | $(0.5,10)$ | - | 6051 | 5405 | 6423 | 5548 | 4605 | 3778 |
|  | 25 | $(0.5,20)$ | - | 3123 | 3280 | 3827 | 2352 | 1823 | 1500 |
|  |  | Average | - | 46933 | 27452 | 39069 | 17793 | 13606 | 15337 |
| 2 | 10 | $(0.5,5)$ | 19 | 5.6 | 5.6 | 0.9 | 3.7 | 4.2 | 0.9 |
|  | 10 | $(0.5,10)$ | 7.778 | 2.667 | 1.5556 | 2.333 | 1.556 | 0.778 | 2.444 |
|  | 10 | $(0.5,20)$ | 29.56 | 3.889 | 3.556 | 3.333 | 3.222 | 3.888 | 2.667 |
|  | 10 | $(1,5)$ | 88.6 | 63.4 | 68.1 | 76.6 | 46.9 | 22.2 | 68.1 |
|  | 10 | $(1,10)$ | 35.6 | 27.2 | 35 | 35.4 | 18 | 17 | 16.7 |
|  | 10 | $(1,20)$ | 28.5 | 27.3 | 21.6 | 19.5 | 16.2 | 18.1 | 20.3 |
|  |  | Average | 34.84 | 21.68 | 22.59 | 23.01 | 14.93 | 11.03 | 18.52 |
|  | 15 | $(0.5,5)$ | 81.2 | 73.7 | 105.1 | 75.3 | 58.2 | 43.1 | 46 |
|  | 15 | $(0.5,10)$ | 93.5 | 67.6 | 50.5 | 81.6 | 63.3 | 49.1 | 65.4 |
|  | 15 | $(0.5,20)$ | 188.1 | 67.7 | 65.8 | 65.1 | 48.7 | 38.6 | 50.3 |
|  | 15 | $(1,5)$ | 5588 | 3306 | 5257 | 4394 | 4016 | 2734 | 2594 |
|  | 15 | $(1,10)$ | 715.2 | 1052 | 861.3 | 778.3 | 669 | 619.9 | 502.4 |
|  | 15 | $(1,20)$ | $317.8$ | 442.3 | 393.3 | 451.3 | 349 | 307.7 | 315.8 |
|  |  | Average | 1164 | 834.9 | 1122 | 974.4 | 867.4 | 632.1 | 595.6 |
|  | 20 | $(0.5,5)$ | 664.2 | 689.6 | 596.8 | 469.2 | 335.9 | 335.2 | 426.1 |
|  | 20 | $(0.5,10)$ | 746.9 | 317 | 218.8 | 240.3 | 237.3 | 195.1 | 153.7 |
|  | 20 | $(0.5,20)$ | 372.8 | 374 | 474.4 | 320.6 | 258.9 | 340.2 | 328.6 |
|  | 20 | $(1,5)$ | 7213 | 23766 | 16086 | 44835 | 8717 | 9182 | 20919 |
|  | 20 | $(1,10)$ | 13725 | 15129 | 11001 | 10907 | 8124 | 6421 | 5870 |
|  | 20 | $(1,20)$ | 2091 | 2995 | 2789 | 3253 | 2564 | 1753 | 1667 |
|  |  | Average | 4136 | 7212 | 5195 | 10004 | 3373 | 3038 | 4894 |
|  | 25 | $(0.5,5)$ | - | 353522 | 222429 | 311712 | 100967 | 86978 | 107318 |
|  | 25 | $(0.5,10)$ | - | 4673 | 3975 | 4156 | 3335 | 2965 | 3421 |
|  | 25 | $(0.5,20)$ | - | 2766 | 2973 | 2779 | 2583 | 2537 | 2533 |
|  |  | Average | - | 120320 | 76459 | 106216 | 35628 | 30827 | 37757 |

Table A.11: Number of branching nodes for the reload instances averaged over 10 generated instances for given values of $n, p$ and $c$.

| n | p | c | Class | OPT | SEC-simple | SEC |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 21 | 0.5 | 5 | 1 | 2 | 0.787 | 0.771 |
|  |  |  | 2 | 4 | 0.738 | 0.751 |
|  |  | 10 | 1 | 8 | 2.767 | 2.342 |
|  |  |  | 2 | 13 | 0.521 | 0.543 |
|  |  | 20 | 1 | 9 | 0.501 | 0.841 |
|  |  |  | 2 | 31 | 0.395 | 0.407 |
|  | 0.8 | 5 | 1 | 2 | 169.1 | 231.1 |
|  |  |  | 2 | 4 | 198.8 | 249.6 |
|  |  | 10 | 1 | 4 | 4.973 | 7.479 |
|  |  |  | 2 | 8 | 6.993 | 9.556 |
|  |  | 20 | 1 | 6 | 2.479 | 1.680 |
|  |  |  | 2 | 13 | 2.157 | 3.265 |
| 22 | 0.5 | 5 | 1 | 2 | 2.469 | 8.384 |
|  |  |  | 2 | 9 | 0.681 | 0.851 |
|  |  | 10 | 1 | 8 | 1.530 | 1.143 |
|  |  |  | 2 | 19 | 0.481 | 0.782 |
|  |  | 20 | 1 | 10 | 0.890 | 0.808 |
|  |  |  | 2 | 27 | 1.289 | 1.193 |
|  | 0.8 | 5 | 1 | 2 | 399.4 | 764.4 |
|  |  |  | 2 | 0 | 17.18 | 11.65 |
|  |  | 10 | 1 | 3 | 8.797 | 7.626 |
|  |  |  | 2 | 8 | 62.52 | 24.11 |
|  |  | 20 | 1 | 6 | 6.435 | 7.146 |
|  |  |  | 2 | 17 | 13.23 | 8.686 |
| 23 | 0.5 | 5 | 1 | 2 | 8.852 | 8.528 |
|  |  |  | 2 | 10 | 27.23 | 20.34 |
|  |  | 10 | 1 | 7 | 3.754 | 2.791 |
|  |  |  | 2 | 10 | 7.896 | 4.142 |
|  |  | 20 | 1 | 11 | 0.460 | 0.395 |
|  |  |  | 2 | 19 | 1.128 | 0.870 |
| 23 | 0.8 | 5 | 1 | 0 | 84.41 | 27.89 |
|  |  |  | 2 | 4 | 3655 | 2366 |
|  |  | 10 | 1 | 3 | 52.55 | 59.89 |
|  |  |  | 2 | 8 | 153.8 | 63.12 |
|  |  | 20 | 1 | 7 | 19.18 | 10.07 |
|  |  |  | 2 | 14 | 12.80 | 12.12 |


| n | p | c | Class | OPT | SEC-simple | SEC |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 24 | 0.5 | 5 | 1 | 3 | 37.95 | 29.45 |
|  |  |  | 2 | 10 | 13.90 | 17.00 |
|  |  | 10 | 1 | 8 | 1.947 | 2.045 |
|  |  |  | 2 | 22 | 1.295 | 2.501 |
|  |  | 20 | 1 | 11 | 1.636 | 1.631 |
|  |  |  | 2 | 28 | 5.907 | 9.120 |
|  | 0.8 | 5 | 1 | - | - | - |
|  |  |  | 2 | 2 | 8249 | 10517 |
|  |  | 10 | 1 | 3 | 101.8 | 132.4 |
|  |  |  | 2 | 6 | 168.2 | 143.9 |
|  |  | 20 | 1 | 7 | 6.754 | 8.559 |
|  |  |  | 2 | 13 | 31.22 | 44.26 |
| 25 | 0.5 | 5 | 1 | 2 | 88.49 | 57.76 |
|  |  |  | 2 | 8 | 1.701 | 1.367 |
|  |  | 10 | 1 | 8 | 18.29 | 18.18 |
|  |  |  | 2 | 18 | 18.21 | 17.08 |
|  |  | 20 | 1 | 11 | 2.821 | 1.554 |
|  |  |  | 2 | 18 | 1.431 | 1.838 |
|  | 0.8 | 5 | 1 | 0 | 9479 | 273.1 |
|  |  |  | 2 | 4 | 14175 | 8042 |
|  |  | 10 | 1 | 2 | 68.52 | 94.96 |
|  |  |  | 2 | 8 | 30.53 | 55.61 |
|  |  | 20 | 1 | 6 | 35.54 | 49.12 |
|  |  |  | 2 | 13 | 68.26 | 67.23 |
| 26 | 0.5 | 5 | 1 | 2 | 491.6 | 562.8 |
|  |  |  | 2 | 6 | 13.74 | 11.77 |
|  |  | 10 | 1 | 6 | 18.41 | 16.05 |
|  |  |  | 2 | 18 | 6.667 | 8.945 |
|  |  | 20 | 1 | 9 | 1.877 | 1.800 |
|  |  |  | 2 | 26 | 8.740 | 8.587 |
| 26 | 0.8 | 5 | 1 | - | - | - |
|  |  |  | 2 | - | - | - |
|  |  | 10 | 1 | 3 | 127.6 | 159.1 |
|  |  |  | 2 | 8 | 3327 | 3104 |
|  |  | 20 | 1 | 7 | 183.3 | 86.4 |
|  |  |  | 2 | 12 | 115.7 | 92.15 |

Table A.12: Computation times of SEC-simple and SEC on 72 additional reload instances for given values of $n, p$ and $c$.

## A. 6 Quantum gates and their two-qubit decompositions

Since the NNCP is only well-defined when a quantum circuit consists solely of one- or two-qubit gates, we have to decompose all gates that act on more than two gates. As indicated in Section 7.6, this task can be completed in lots of ways and performing this decomposition optimally can be seen as a research problem in itself. In Chapter 7 we apply the decomposition method used in [286], although the authors of [286] already indicated that this method might be open for improvement.

The quantum circuits that we consider in our experiments consist of the following types of quantum gates: one-qubit gates, two-qubit gates, three-qubit Peres gates, three- and four-
qubit Fredkin gates and three-, four- and five-qubit Toffoli gates. Commonly used one-qubit gates are the Hadamard gate and the Pauli-gates, e.g., the Pauli- $X$-gate. When applying the Hadamard gate to a qubit in any state, it brings the qubit in a superposition state where it has an equal probability to be 0 or 1 upon measurement. The Hadamard gate in a quantum circuit is depicted as $-H-$. The Pauli- $X$-gate is also known as the NOT gate and can be seen as its quantum analog. The NOT-gate is depicted as $\triangle$.

The most commonly used two-qubit gates are depicted in Figure A.1. The controlledNOT gate, also known as CNOT or Feynman gate, acts on a control qubit and a target qubit. If the control qubit is in state $|1\rangle$, a NOT-gate is applied to the target qubit, otherwise nothing happens. The SWAP gate swaps the states of the two qubits where it acts on. The controlled $-V$ and controlled- $V^{\dagger}$ act similarly to the controlled-NOT gate, with the only difference that the unitary operation $V$ or $V^{\dagger}$ is applied to the target qubit. The operation $V$ and $V^{\dagger}$ are the square root of the NOT-gate and its Hermitian conjugate, respectively. That is, if two controlled- $V$ gates are placed in succession, the result is similar to a controlled-NOT gate, while the identity gate is obtained when applying a controlled- $V$ and a controlled- $V^{\dagger}$ gate in succession.

(a) Controlled-NOT

(b) SWAP

(c) Controlled- $V$

(d) Controlled- $V^{\dagger}$

Figure A.1: Overview of commonly used two-qubit quantum gates.

A Toffoli gate [352] is the multiple-control NOT gate. Acting on several control qubits and a single target qubit, a NOT gate is applied to the target qubit if all the control qubits are in state $|1\rangle$. The three-qubit Toffoli gate is depicted in Figure A.2, along with a possible decomposition into two-qubit gates, following the approach of [31].


Figure A.2: Decomposition of multiple-control Toffoli gate with two controls and a single target qubit.

The Peres gate [301] is obtained from a combination of a two-qubit controlled-NOT gate and standard controlled-NOT gate. Following the approach from [219], the Peres gate can be decomposed into four two-qubit gates, as shown in Figure A.3.


Figure A.3: Decomposition of Peres gate on three qubits.

The Fredkin gate [143] operates on three qubits as a controlled-SWAP gate. If the state of the control qubit is $|1\rangle$, then a SWAP gate on the two target qubits is performed. The decomposition into two-qubit gates that we adapt here is the same as the one considered in $[286,367]$, see Figure A. 4


Figure A.4: Decomposition of Fredkin gate (controlled swap gate) with one control qubit.
Finally, we consider the four- and five qubit variants of the Fredkin and Toffoli gate. The functionality of these gates is similar to their three-qubit implementation, only the number of control qubits is larger. The four-qubit Fredkin gate can be decomposed as shown by [10], see Figure A.5. Fredkin gates on a larger number of qubits do not appear in our experiments.


Figure A.5: Decomposition of Fredkin gate (controlled swap gate) with two control qubits.

Finally, the four- and five-qubit Toffoli gates are shown in Figure A. 6 and A.7. The decompositions shown here follow from the construction derived in [31]. Toffoli gates on more than five qubits do not appear in our experiments.

||

Figure A.6: Decomposition of multiple-control Toffoli gate with three controls and a single target qubit.


# Scientific summary 

Semidefinite programming is the subfield of mathematical optimization that deals with the optimization of a linear function over the cone of positive semidefinite matrices under the presence of affine constraints. Capturing linear programs as a subclass, semidefinite programming is regarded as a natural generalization to the well-established area of linear programming, allowing for a richer source of theory and applications. In particular, semidefinite programs have proven themselves useful in their ability to provide strong relaxations of hard combinatorial optimization problems.

The success of linear optimization in many real-world applications has initiated the study of integer linear programming, which allows to model decision problems under finitely many alternatives. A natural way to approach these problems is provided via so-called cuttingplane methods, which have become a fundamental tool in almost all modern optimization algorithms. For semidefinite programs, the concepts of integrality and cutting planes have not yet been investigated to their full extent. To render semidefinite programs suitable for practical use, further investigation and exploration of these conceptual foundations are imperative. This dissertation, consisting of six self-contained essays, contributes to the understanding of these concepts in the framework of SDPs.

In the first part of the thesis we focus on the incorporation of cutting planes in continuous semidefinite programs. It is well-known that polyhedral cuts can significantly strengthen the SDP relaxations of combinatorial problems. The current state-of-the-art SDP solvers, however, have difficulties solving models involving a large number of cutting planes. As an alternative, we introduce an advanced SDP-based cutting-plane method that is suitable for the addition of polyhedral cuts. Building on the framework of an augmented Lagrangian method, we propose to incorporate these cuts in a subroutine via Dykstra's projection algorithm. Our method exploits warm starts and can be (partially) parallelized by clustering the set of cuts into subsets of nonoverlapping cuts. These ingredients allow for a highly efficient implementation of the algorithm. Since the major components of our method are generic, our algorithm is suitable for solving general-purpose SDPs with a large number of polyhedral cuts. We apply and compare various flavours of our algorithm on several different problems in Chapter 3 and 4 of this thesis. Numerical results show substantial improvements in the strength of the bounds for these problems, while being able to handle thousands of cutting planes.

The second part of the thesis focuses on integer semidefinite programming. Although semidefinite programs are commonly used as an approach to obtain tight relaxations of discrete optimization problems, the incorporation of integrality constraints in semidefinite programs itself has been considered only recently. We study various theoretical notions related to integer semidefinite programming, which are mainly generalizations of similar concepts from integer linear programming. In particular, we study the Chvátal-Gomory closure for integer semidefinite programs and its connection to total dual integrality in Chapter 5. This leads to several characterizations and conditions for a linear matrix inequality to be totally dual integral. In Chapter 6 we extend the Lagrangian duality theory from integer linear
programming to the case of integer semidefinite programming. Moreover, by examining the interplay between positive semidefiniteness and integrality in matrix theory, we show that a wide range of combinatorial optimization problems can be modeled as integer semidefinite programs, including various types of binary quadratic problems and problems related to data science. To the best of our knowledge, this thesis provides the first generic approach to show the modeling power of integer semidefinite programming. Although the above-mentioned theoretical generalizations from integer linear programming are interesting on their own, we also demonstrate how these can be exploited to obtain improved solution strategies for the problems under consideration. This leads to the introduction of a Chvátal-Gomory-based cutting-plane method and a projected subgradient method for integer semidefinite programs in Chapter 5 and 6, respectively. Numerical results show the potential of the introduced approaches on several problem classes, providing algorithms that show more powerful or competitive behaviour compared to the state-of-the-art solvers for these problems.

Throughout the thesis, a particular emphasis is placed on the application of our approaches to problems in combinatorial optimization. As returning examples throughout this work, we consider three graph optimization problems: the graph partition problem (GPP), the quadratic cycle cover problem ( QCCP ) and the quadratic traveling salesman problem (QTSP). The GPP and its variations are mainly treated in Chapter 4, where we apply our SDP-based cutting-plane approach to obtain bounds for large-scale graph partition problems. These are currently the strongest GPP bounds in the literature. The QCCP and the QTSP are highly related optimization problems that have applications in, among others, bioinformatics, logistics and energy distribution networks. Our study on integrality and cutting planes in SDPs pays off in terms of novel and efficient methods for solving both problems. On top of these approaches, we also investigate alternative solution strategies. In Chapter 2 of this thesis we study the linearization problem of the QCCP, which asks whether a given instance of the QCCP can be solved as an equivalent linear cycle cover problem. We derive various sufficient conditions for the cost matrix to be linearizable and exploit these conditions to obtain bounds that are both strong and efficiently computable. Also, various novel LP- and SDP-based upper bounding techniques for the QCCP are introduced, such as randomized rounding methods and a reinforcement learning approach.

In the final chapter of the thesis, we consider yet another combinatorial optimization problem. Although the proposed solution strategy is not directly related to the other parts of the thesis, the problem itself is related to a variation of the QTSP, namely the generalized traveling salesman problem (GTSP). An interesting application of this setting can be found in quantum circuit design, where the objective is to find an assignment of qubits over a given architecture that minimizes the total overhead costs to compile the circuit. By exploiting the symmetries underlying the problem, we are able to significantly reduce the number of variables and constraints in the model. For certain special quantum architectures, this reduction empowers us to obtain optimal assignments for extremely large quantum circuits.

## Wetenschappelijke samenvatting

Semidefiniet programmeren is het deelgebied van wiskundige optimalisatie dat zich bezighoudt met de optimalisatie van een lineaire functie over de kegel van positief semidefiniete matrices onder de aanwezigheid van affiene randvoorwaarden. Omdat dit deelgebied ook lineaire programma's omvat, kan semidefiniete optimalisatie worden gezien als een natuurlijke generalisering van lineaire wiskundige optimalisatie. In het bijzonder hebben semidefiniete programma's (SDP's) hun nut bewezen in het vermogen om sterke relaxaties te bieden voor complexe combinatorische optimaliseringsproblemen.

Het succes van lineaire optimalisatie in veel praktische applicaties vormde de aanleiding voor het bestuderen van geheeltallige lineaire optimalisatie, waarmee beslissingsproblemen met een eindig aantal oplossingen kunnen worden gemodelleerd. Een natuurlijke manier om deze problemen te benaderen wordt geboden via zogenaamde snijvlakmethoden, die een fundamenteel onderdeel zijn geworden in bijna alle moderne optimaliseringsalgoritmen. Voor semidefiniete programma's zijn de theorie en het gebruik van geheeltalligheid en snijvlakken echter nog niet volledig onderzocht. Om semidefiniet programmeren geschikt te maken voor praktisch gebruik, is verder onderzoek en verkenning van deze conceptuele grondslagen noodzakelijk. Dit proefschrift draagt bij aan het begrip van deze concepten in het kader van semidefiniet programmeren, in de vorm van zes op zichzelf staande hoofdstukken.

In het eerste deel van dit proefschrift concentreren we ons op de integratie van snijvlakken in continue semidefiniete programma's. Het is bekend dat polyhedrale snijvlakken de semidefiniete relaxaties van combinatorische problemen aanzienlijk kunnen versterken. De huidige oplossingsalgoritmen voor semidefiniete programma's in de literatuur hebben echter moeite met het omgaan met deze snijvlakken. Als alternatief introduceren we een op SDP gebaseerde snijvlakmethode die geschikt is voor de toevoeging van een groot aantal snijvlakken. Voortbouwend op een implementatie van de uitgebreide methode van Lagrange, worden snijvlakken toegevoegd middels een toepassing van het projectie-algoritme van Dykstra. Onze methode maakt gebruik van een warme-start strategie en kan (gedeeltelijk) worden geparallelliseerd door de verzameling snijvlakken te clusteren in deelverzamelingen van nietoverlappende snijvlakken. Deze ingrediënten maken een zeer efficiënte implementatie van het algoritme mogelijk. Omdat de belangrijkste componenten van onze methode generiek zijn, is ons algoritme geschikt voor het oplossen van algemene SDP's met een groot aantal polyhedrale snijvlakken. We passen verschillende vormen van ons algoritme toe op diverse problemen in Hoofdstuk 3 en 4 van dit proefschrift. Numerieke resultaten laten substantiële verbeteringen zien in de sterkte van de berekende begrenzingen voor deze problemen, waarbij duizenden snijvlakken kunnen worden toegevoegd.

Het tweede deel van het proefschrift richt zich op geheeltallige semidefiniete optimalisatie. Hoewel semidefiniete programma's vaak worden gebruikt om sterke relaxaties voor discrete optimaliseringsproblemen te verkrijgen, is de integratie van geheeltalligheidsbeperkingen in de semidefiniete programma's zelf pas zeer recentelijk in overweging genomen. We bestuderen diverse theoretische concepten gerelateerd aan semidefiniet programmeren met gehele getallen, welke voornamelijk generalisaties zijn van soortgelijke concepten uit geheeltallige
lineaire programma's. In het bijzonder bestuderen we de Chvátal-Gomory afsluiting voor geheeltallig semidefiniet programmeren en de verbinding ervan met totale duale geheeltalligheid in Hoofdstuk 5. Dit leidt tot verschillende karakteriseringen en condities voor totale duale geheeltalligheid van een lineaire matrix ongelijkheid. In Hoofdstuk 6 breiden we de Lagrangiaanse dualiteitstheorie van geheeltallig lineair programmeren uit naar het geval van geheeltallig semidefiniet programmeren. Bovendien leidt een reeks resultaten met betrekking tot positief semidefiniete en geheeltallige matrices tot het modelleren van een breed scala aan combinatorische optimaliseringsproblemen als geheeltallige semidefiniete programma's, waaronder verschillende binaire kwadratische problemen en problemen gerelateerd aan datawetenschap. Voor zover bij ons bekend, biedt dit proefschrift de eerste generieke benadering om de modelleringskracht van geheeltallig semidefiniet programmeren te demonstreren. Hoewel de bovengenoemde generalisaties van concepten uit geheeltallige lineaire optimalisatie op zichzelf interessant zijn, laten we ook zien hoe deze praktisch kunnen worden benut. Dit leidt tot de introductie van een op Chvátal-Gomory gebaseerde snijvlakmethode en een geprojecteerde subgradiëntmethode voor geheeltallig semidefiniet programmeren in respectievelijk Hoofdstuk 5 en 6 . Numerieke resultaten laten de potentie van de geïntroduceerde aanpak voor verschillende probleemklassen zien, waarbij de verkregen algoritmen competitief en vaak zelfs krachtiger zijn dan de huidige oplossingsstrategieën voor deze problemen.

In het proefschrift wordt bijzondere nadruk gelegd op de toepassing van onze methoden op problemen in combinatorische optimalisatie. Als terugkerende voorbeelden beschouwen we drie graafoptimaliseringsproblemen: het graafpartitie probleem (GPP), het kwadratisch cyclusdekkingsprobleem (QCCP) en het kwadratisch handelsreizigerprobleem (QTSP). De GPP en zijn varianten worden hoofdzakelijk behandeld in Hoofdstuk 4, waar we onze op SDP gebaseerde snijvlakmethode gebruiken om begrenzingen van grootschalige graafpartitie problemen te vinden. Dit zijn momenteel de sterkste GPP-begrenzingen in de literatuur. De QCCP en de QTSP zijn gerelateerde optimaliseringsproblemen die toepassingen hebben in onder meer bio-informatica, logistiek en energiedistributienetwerken. Ons onderzoek naar geheeltalligheid en snijvlakmethoden in SDP's initieert nieuwe en efficiënte benaderingen voor het oplossen van beide problemen. Naast deze methoden verkennen we ook alternatieve oplossingsstrategieën. In Hoofdstuk 2 bestuderen we het linearisatieprobleem van de QCCP, waarbij de vraag wordt gesteld of een instantie van de QCCP kan worden opgelost als een equivalent lineair cyclusdekkingsprobleem. We bepalen diverse voorwaarden waaronder de kostmatrix van de QCCP lineairiseerbaar is en gebruiken deze voorwaarden om nieuwe begrenzingen te vinden die zowel sterk als efficiënt berekenbaar zijn. Daarnaast introduceren we verschillende nieuwe op LP en SDP gebaseerde bovengrenstechnieken voor de QCCP, zoals gerandomiseerde afrondingsmethoden en een reinforcement learning methode.

In het laatste hoofdstuk van het proefschrift beschouwen we een ander combinatorisch optimaliseringsprobleem. Hoewel de voorgestelde oplossingsstrategie niet direct gerelateerd is met de andere onderdelen in het proefschrift, houdt het probleem zelf verband met een variant van de QTSP, namelijk het gegeneraliseerd handelsreizigersprobleem (GTSP). Een interessante toepassing van dit probleem is te vinden in de studie naar kwantumcircuits, waarbij het doel is om een toewijzing van qubits over een gegeven architectuur te vinden zodanig dat de totale overheadkosten voor het compileren van het circuit geminimaliseerd worden. Door de symmetrieën in het onderliggende probleem te bestuderen, kan het aantal variabelen en randvoorwaarden in het model aanzienlijk worden beperkt. Voor bepaalde speciale kwantumarchitecturen stelt deze reductie ons in staat om optimale qubit toewijzingen voor extreem grote kwantumcircuits te verkrijgen.
[1] E. Adams, M.F. Anjos, F. Rendl, and A. Wiegele. A hierarchy of subgraph projectionbased semidefinite relaxations for some NP-hard graph optimization problems. INFOR: Inf. Syst. Oper. Res., 53(1):40-48, 2015.
[2] W. P. Adams, R. J. Forrester, and F. W. Glover. Comparisons and enhancement strategies for linearizing mixed 0-1 quadratic programs. Discrete Optim., 1(2):99-120, 2004.
[3] W. P. Adams and H. D. Sherali. A tight linearization and an algorithm for zero-one quadratic programming problems. Manage Sci., 32(10):1274-1290, 1986.
[4] W. P. Adams and H. D. Sherali. Linearization strategies for a class of zero-one mixed integer programming problems. Oper. Res., 38(2):217-226, 1990.
[5] W.P. Adams and L. Waddell. Linear programming insights into solvable cases of the quadratic assignment problem. Discrete Optim., 14:46-60, 2014.
[6] A. Aggarwal, D. Coppersmith, S. Khanna, R. Motwani, and B. Schieber. The angularmetric traveling salesman problem. SIAM J. Comput., 29:697-711, 1999.
[7] O. Aichholzer, A. Fischer, F. Fischer, J.F. Meier, U. Pferschy, A. Pilz, and R. Staněk. Minimization and maximization versions of the quadratic travelling salesman problem. Optim., 66(4):521-546, 2017.
[8] A. Ajagekar and F. You. Quantum computing for energy systems optimization: Challenges and opportunities. Energy, 179:76-89, 2019.
[9] M.G. Alfailakawi, I. Ahmad, and S. Hamdan. Harmony-search algorithm for 2D nearest neigbor quantum circuits realization. Expert Syst. Appl., 61:16-27, 2016.
[10] N. Alhagi. Synthesis of reversible functions using various gate libraries and design specifications. Technical report, Portland State University, 2000.
[11] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. SIAM J. Optim., 5:13-51, 1995.
[12] E. Amaldi, G. Galbiati, and F. Maffioli. On minimum reload cost paths, tours and flows. Networks, 57:254-260, 2011.
[13] M.S. Ang, J. Ma, N. Liu, K. Huang, and Y. Wang. Fast projection onto the capped simplex with applications to sparse regression in bioinformatics. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, Advances in Neural Information Processing Systems, volume 34, pages 9990-9999. Curran Associates, Inc., 2021.
[14] M. F. Anjos, B. Ghaddar, L. Hupp, F. Liers, and A. Wiegele. Solving $k$-way graph partitioning problems to optimality: The impact of semidefinite relaxations and the bundle method. In M. Jünger and G. Reinelt, editors, Facets of Combinatorial Optimization: Festschrift for Martin Grötschel, pages 355-386. Springer Berlin Heidelberg, Berlin and Heidelberg, Germany, 2013.
[15] M.F. Anjos and J.B. Lasserre. Handbook on semidefinite, conic and polynomial optimization, volume 166. Springer Science \& Business Media, 2011.
[16] M.F. Anjos and H. Wolkowicz. Strengthened semidefinite relaxations via a second lifting for the max-cut problem. Discrete Appl. Math., 119(1-2):79-106, 2002.
[17] K. Anstreicher and H. Wolkowicz. On Lagrangian relaxation of quadratic matrix constraints. SIAM J. Matrix Anal. Appl., 22(1):41-55, 2000.
[18] MOSEK ApS. MOSEK Optimization Suite 10.0.40, 2022.
[19] E. M. Arkin, M. A. Bender, E. D. Demaine, S. P. Fekete, J. S. B. Mitchell, and S. Sethia. Optimal covering tours with turn costs. SIAM J. Comput., 35(3):531-566, 2005.
[20] M. Armbruster. Branch-and-cut for a semidefinite relaxation of large-scale minimum bisection problems. PhD thesis, Technische Universität Chemnitz, Chemnitz, Germany, 2007.
[21] M. Armbruster, M. Fügenschuh, C. Helmberg, and A. Martin. LP and SDP branch-and-cut algorithms for the minimum graph bisection problem: A computational comparison. Math. Program. Comput., 4:275-306, 2012.
[22] A. Assad and W. Xu. The quadratic minimum spanning tree problem. Nav. Res. Logist. (NRL), 39(3):399-417, 1992.
[23] A. Atamtürk and V. Narayanan. Cuts for conic mixed-integer programming. In M. Fischetti and D.P. Williamson, editors, Integer Programming and Combinatorial Optimization, pages 16-29, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.
[24] A. Atamtürk and V. Narayanan. Conic mixed-integer rounding cuts. Math. Program., 122(1):1-20, 2010.
[25] A. Atamtürk, G. L. Nemhauser, and M. W. P. Savelsbergh. A combined Lagrangian, linear programming and implication heuristic for large-scale set partitioning problems. J. Heuristics, 1:247-259, 1995.
[26] V. Balakrishnan and L. Vandenberghe. Semidefinite programming duality and linear time-invariant systems. IEEE Trans. Autom. Control, 48(1):30-41, 2003.
[27] A. Balog and I. Bárány. On the convex hull of the integer points in a disc. In Proceedings of the Seventh Annual Symposium on Computational Geometry, pages 162-165, 1991.
[28] H.J. Bandelt, M. Oosten, J.H.G.C. Rutten, and F.C.R. Spieksma. Lifting theorems and facet characterization for a class of clique partitioning inequalities. Oper. Res. Lett., 24:235-243, 1999.
[29] J. Bang-Jensen, S. Bessy, B. Jackson, and M. Kriesell. Antistrong digraphs. J. Comb. Theory Ser. B., 122:68-90, 2016.
[30] J. Bang-Jensen and G. Gutin. Digraphs: Theory, Algorithms and Applications. Springer, 2009.
[31] A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVincenzo, P.W. Margolus, P.W. Shor, T. Sleator, J. Smolin, and H. Weinfurter. Elementary gates for quantum computation. Phys. Rev., 52:3457-3467, 1995.
[32] G.P. Barker and D. Carlson. Cones of diagonally dominant matrices. Pac. J. Math., 57:15-32, 1975.
[33] T.D. Barrett, W.R. Clements, J.N. Foerster, and A.I. Lvovsky. Exploratory combinatorial optimization with reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 3243-3250, 2020.
[34] H. H. Bauschke and J. M. Borwein. Dykstra's alternating projection algorithm for two sets. J. Approx. Theory, 79:418-443, 1994.
[35] H. H. Bauschke and V. R. Koch. Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces. Contemp. Math., 636:140, 2015.
[36] J.E. Beasley. Heuristic algorithms for the unconstrained binary quadratic programming problem. Technical report, Technical Report, The Management School, Imperial College, London, England, 1998.
[37] A. Beck. Quadratic matrix programming. SIAM J. Optim., 17:1224-1238, 2007.
[38] E.T. Bell. Exponential numbers. Amer. Math. Monthly, 41:411-419, 1934.
[39] P. Belotti, J.C. Góez, I. Pólik, T.K. Ralphs, and T. Terlaky. A complete characterization of disjunctive conic cuts for mixed integer second order cone optimization. Discrete Optim., 24:3-31, 2017.
[40] A. Ben-Tal and A. Nemirovski. Lectures on modern convex optimization: analysis, algorithms, and engineering applications. SIAM, 2001.
[41] S.J. Benson, Y. Ye, and X. Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. SIAM J. Optim., 10(2):443-461, 2000.
[42] A. Berman and C. Xu. $\{0,1\}$ completely positive matrices. Linear Algebra Appl., 399:35-51, 2005.
[43] A. Bhardwaj, P. Rostalski, and R. Sanyal. Deciding polyhedrality of spectrahedra. SIAM J. Optim., 25:1873-1884, 2015.
[44] A. Bhattacharjee, C. Bandyopadhyay, R. Wille, R. Drechsler, and H. Rahaman. A novel approach for nearest neighbor realization of 2D quantum circuits. In 2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), pages 305-310, Hong Kong, 2018. IEEE.
[45] A. Bhattacharjee, C. Bandyopadhyay, R. Wille, R. Drechsler, and H. Rahaman. Improved look-ahead approaches for nearest neighbor synthesis of 1D quantum circuits. In 2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID), pages 203-208, Delhi, NCR, India, 2019. IEEE.
[46] D. Bhattacharjee and A. Chattopadhyay. Depth-optimal quantum circuit placement for arbitrary topologies. arXiv:1703.08540, 2017.
[47] C.E. Bichot and P. Siarry. Graph partitioning. John Wiley \& Sons, 2013.
[48] G. Birkhoff. Three observations on linear algebra. Univ. Nac. Tacumán. Revista A., 5:147-151, 1946.
[49] R. Bödi, K. Herr, and M. Joswig. Algorithms for highly symmetric linear and integer programs. Math. Program., 137(1-2):65-90, 2013.
[50] I.M. Bomze and M. Gable. Optimization under uncertainty and risk: Quadratic and copositive approaches. Eur. J. Oper. Res., 310:449-476, 2023.
[51] P. Bonami, M. Kilinç, and J. Linderoth. Mixed Integer Nonlinear Programs, volume 154 of The IMA Volumes in Mathematics and its Applications, chapter Algorithms and Software for Convex Mixed Integer Nonlinear Programs, pages 1-39. Springer, New York, NY, 2012.
[52] J. M. Borwein and H. Wolkowicz. Facial reduction for a cone-convex programming problem. J. Aust. Math. Soc., 30(3):369-380, 1980.
[53] S.P. Boyd, Neal P., E.K. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn., 3:1-122, 2011.
[54] S.P. Boyd and L. Vandenberghe. Convex optimization. Cambridge university press, 2004.
[55] J. P. Boyle and R. L. Dykstra. A method for finding projections onto the intersection of convex sets in Hilbert spaces. In R. Dykstra, T. Robertson, and F. T. Wright, editors, Advances in Order Restricted Statistical Inference, Lecture Notes in Statistics, volume 37. Springer, New York, 1985.
[56] G. Braun and S. Pokutta. A short proof for the polyhedrality of the Chvátal-Gomory closure of a compact convex set. Oper. Res. Lett., 42(5):307-310, 2014.
[57] L Brunetta, M. Conforti, and G. Rinaldi. A branch-and-cut algorithm for the equicut problem. Math. Program., 78:243-263, 1997.
[58] T. Bulhões, A. Pessoa, F. Protti, and E. Uchoa. On the complete set packing and set partitioning polytopes: Properties and rank 1 facets. Oper. Res. Lett., 46:389-392, 2018.
[59] A. Buluç, H. Meyerhenke, I. Safro, P. Sanders, and C. Schulz. Recent advances in graph partitioning. In L. Kliemann and P. Sanders, editors, Algorithm Engineering: Selected Results and Surveys, volume 9220. Springer International Publishing, Cham, Switzerland, 2016.
[60] S. Burer. On the copositive representation of binary and continuous nonconvex quadratic programs. Math. Program., 120:479-495, 2009.
[61] S. Burer and D. Vandenbussche. Solving lift-and-project relaxations of binary integer programs. SIAM J. Optim., 16(3):726-750, 2006.
[62] R. Burkard, M. Dell'Amico, and S. Martello. Assignment Problems. SIAM, Philadelphia, USA, 2009.
[63] Y. Büyükçolak, D. Gözüpek, and S. Özkan. On minimum reload cost paths, tours and flows. Networks, 74(3):274-286, 2019.
[64] P.M. Camerini, L. Fratta, and F. Maffioli. On improving relaxation methods by modified gradient techniques. Nondiffer. Optim., pages 26-34, 1975.
[65] M.K. de Carli Silva and L. Tunçel. A notion on total dual integrality for convex, semidefinite, and extended formulations. SIAM J. Discrete Math., 34(1):470-496, 2018.
[66] P. Carraresi and F. Malucelli. A new lower bound for the quadratic assignment problem. Oper. Res., 40:22-27, 1992.
[67] J. S. Caughman and J. J. P. Veerman. Kernels of directed graph Laplacians. Electron. J. Comb., 13:R39, 2006.
[68] M.T. Çezik and G. Iyengar. Cuts for mixed 0-1 conic programming. Math. Program., 104:179-202, 2005.
[69] A. Cegielski. Iterative Methods for Fixed Point Problems in Hilbert Spaces, volume 2057 of Lecture Notes in Mathematics. Springer, Berlin, Heidelberg, 2012.
[70] E. Cela, V.G. Deineko, and G.J. Woeginger. Linearizable special cases of the QAP. J. Comb. Optim., 31:1269-1279, 2016.
[71] A. Cerveira, A. Agra, F. Bastos, and J. Gromicho. A new branch and bound method for a discrete truss topology design problem. Comput. Optim. Appl., 54(1):163-187, 2013.
[72] V.G. Chagas. Exact algorithms for the quadratic bin packing problem. PhD thesis, Universidade Estadual de Campinas, Campinas, Brazil, 2021.
[73] X. Cheng, Z. Guan, and W. Ding. Mapping from multiple-control Toffoli circuits to linear nearest neighbor quantum circuits. Quantum Inf. Process., (17):169, 2018.
[74] S. Chiba and T. Yamashita. On directed 2-factors in digraphs and 2-factors containing perfect matchings in bipartite graphs. SIAM J. Discrete Math., 32(1):394-409, 2018.
[75] B.S. Choi and R. Van Meter. An $\theta(\sqrt{n})$-depth quantum adder on a 2D NTC quantum computer architecture. J. Emerg. Technol. Coput. Syst., 8:1-22, 2012.
[76] S. Chopra and M.R. Rao. The partition problem. Math. Program., 59:87-115, 1993.
[77] V. Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. Discrete Math., 4:305-337, 1973.
[78] C. Coey, M. Lubin, and J.P. Vielma. Outer approximation with conic certificates for mixed-integer convex problems. Math. Program. Comput., 12(2):249-293, 2020.
[79] O. Collins, S. Dolinar, R. McEliece, and F. Pollara. A VLSI decomposition of the De Bruijn graph. J. ACM, 39:931-948, 1992.
[80] F. Comellas, Dalfó C., and M. A. Fiol. Multidimensional Manhattan street networks. SIAM J. Discrete Math., 22(4):1428-1447, 2008.
[81] M. Conforti, G. Cornuéjols, and G. Zambelli. Integer Programming. Graduate Texts in Mathematics. Springer International Publishing, 2014.
[82] G. Cornuéjols, D. Lee, and Y. Li. On the rational polytopes with Chvátal rank 1. Math. Program., 179(1-2):21-46, 2020.
[83] A. Cowtan, S. Dilkes, R. Duncan, A. Krajenbrink, W. Simmons, and S. Sivarajah. On the qubit routing problem. arXiv:1902.08091, 2019.
[84] A. Ćustić and A. P. Punnen. A characterization of linearizable instances of the quadratic minimum spanning tree problem. J. Comb. Optim., 35(2):436-453, 2018.
[85] A. Ćustić, V. Sokol, A.P. Punnen, and B. Bhattacharya. The bilinear assignment problem: Complexity and polynomially solvable special cases. Math. Program., 166:185205, 2017.
[86] D. Cvetković, M. Čangalović, and V. Kovačević-Vujčić. Semidefinite programming methods for the symmetric traveling salesman problem. In G. Cornuéjols, R.E. Burkard, and G.J. Woeginger, editors, Integer programming and Combinatorial Optimization (IPCO 1999), volume 1610 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 1999.
[87] J. Czyzyk, M.P. Mesnier, and J.J. Moré. The NEOS server. IEEE Comput. Sci. Eng., 5(3):68-75, 1998.
[88] D. Dadush, S.S. Dey, and J.P. Vielma. The Chvátal-Gomory closure of a strictly convex body. Math. Oper. Res., 36(2):227-239, 2011.
[89] D. Dadush, S.S. Dey, and J.P. Vielma. On the Chvátal-Gomory closure of a compact convex set. Math. Program., 145:327-348, 2014.
[90] E. R. van Dam and R. Sotirov. Semidefinite programming and eigenvalue bounds for the graph partition problem. Math. Program., 151:379-404, 2015.
[91] G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. Oper. Res., 2:393-410, 1954.
[92] G.B. Dantzig. The Simplex Method. RAND Corporation, Santa Monica, CA, 1956.
[93] S. Dash, O. Günlük, and D. Lee. Generalized Chvátal-Gomory closures for integer programs with bounds on variables. Math. Program., 190:393-425, 2021.
[94] D. Davendra. Traveling salesman problem: Theory and applications. BoD-Books on Demand, 2010.
[95] C. Delorme and S. Poljak. Laplacian eigenvalues and the maximum cut problem. Math. Program., 62:557-574, 1993.
[96] F. Deutsch and H. Hundal. The rate of convergence of Dykstra's cyclic projections algorithm: The polyhedral case. Numer. Funct. Anal. Optim., 15(5-6):537-565, 1994.
[97] S.S. Dey and D.A. Morán R. Some properties of convex hulls of integer points contained in general convex sets. Math. Program., 141(1-2):507-526, 2013.
[98] S.S. Dey and J.P. Vielma. The Chvátal-Gomory closure of an ellipsoid is a polyhedron. In F. Eisenbrand and F.B. Shepherd, editors, 14 th International Conference on Integer Programing and Combinatorial Optimization (IPCO2010), volume 6080 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2010.
[99] J. Ding and S. Yamashita. Exact synthesis of nearest neighbor compliant quantum circuits in 2-D architecture and its application to large-scale circuits. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 39(5):1045-1058, 2019.
[100] Y. Ding, D. Ge, and H. Wolkowicz. On equivalence of semidefinite relaxations for quadratic matrix programming. Math. Oper. Res., 36(1):88-104, 2011.
[101] Y. Ding and H. Wolkowicz. A low-dimensional semidefinite relaxation for the quadratic assignment problem. Math. Oper. Res., 34(4):1008-1022, 2009.
[102] D.P. DiVincenzo and IBM. The physical implementation of quantum computation. Fortschr. der Phys., (48):771-783, 2000.
[103] J.D. Dixon and A. Majeed. Coset representatives for permutation groups. Port. Math., 45(1):61-68, 1988.
[104] W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. IBM J. Res. Dev., 17:420-425, 1973.
[105] J. van Doornmalen and C. Hojny. A unified framework for symmetry handling. 2022.
[106] J. Douglas and H. H. Rachford. On the numerical solution of heat conduction problems in two and three space variables. Trans. Am. Math. Soc., 82(2):421-439, 1956.
[107] S. Drewes and S. Ulbrich. Mixed integer second order cone programming. Verlag Dr. Hut Germany, 2009.
[108] D. Drusvyatskiy and H. Wolkowicz. The many faces of degeneracy in conic optimization. Found. Trends Optim., 3(2):77-170, 2017.
[109] B.P.M. Duarte. Exact optimal designs of experiments for factorial models via mixedinteger semidefinite programming. Math., 11(4):854, 2023.
[110] I. Dukanovic and F. Rendl. Semidefinite programming relaxations for graph coloring and maximal clique problems. Math. Program., 109:345-365, 2007.
[111] A.L. Dulmage and N.S. Mendelsohn. Matrices associated with the Hitchcock problem. J. $A C M, 9: 409-418,1962$.
[112] J. Dunkel and A.S. Schulz. The Gomory-Chvátal closure of a nonrational polytope is a rational polytope. Math. Oper. Res., 38(1):63-91, 2012.
[113] I. Dunning, J. Huchette, and M. Lubin. JuMP: A modeling language for mathematical optimization. SIAM Rev., pages 295-320, 2017.
[114] C. Dürr, M. Heiligman, P. Hoyer, and M. Mhalla. Quantum query complexity of some graph problems. SIAM J. Comput., 35:1310-1328, 2006.
[115] R. L. Dykstra. An algorithm for restricted least squares regression. J. Am. Stat. Assoc., 78(384):837-842, 1983.
[116] J. Eckstein. Splitting methods for monotone operators with applications to parallel optimization. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1989.
[117] J. Eckstein. Deriving solution value bounds from the ADMM. Optim. Lett., 14:12891303, 2020.
[118] J. Eckstein and D.P. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. Math. Program., 55:293318, 1992.
[119] J. Eckstein and M. Fukushima. Some reformulations and applications of the alternating direction method of multipliers. Large Scale Optim. State Art, pages 115-134, 1994.
[120] J. Edmonds. Maximum matching and a polyhedron with 0, 1-vertices. J. Res. Natl. Bur. Stand. B, 69(125-130):55-56, 1965.
[121] J. Edmonds and F.R. Giles. A min-max relation for submodular functions on graphs. Ann. Discrete Math., 1:185-204, 1977.
[122] A. Eisenblätter. Frequency Assignment in GSM Networks: Models, Heuristics, and Lower Bounds. PhD thesis, Technische Universität Berlin, Berlin, Germany, 2001.
[123] K. Ellrott, C. Yang, F.M. Sladek, and T. Jiang. Identifying transcription factor binding sites through Markov chain optimization. Bioinform., 18:100-109, 2002.
[124] A. Engau, M.F. Anjos, and A. Vannelli. On handling cutting planes in interior-point methods for solving semi-definite relaxations of binary quadratic optimization problems. Optim. Methods Softw., 27(3):539-559, 2012.
[125] P. Erdős and A. Rényi. On random graphs. Publicationes Mathematicae, 6(2):290-197, 1959.
[126] A. Farghadan and N. Mohammadzadeh. Mapping quantum circuits on 3D nearestneighbor architectures. Quantum Sci. Technol., 4:035001, 2019.
[127] S.P. Fekete and D. Krupke. Covering tours and cycle covers with turn costs: Hardness and approximation. In Algorithms and Complexity: 11th International Conference, CIAC 2019, Rome, Italy, May 27-29, 2019, Proceedings 11, pages 224-236. Springer, 2019.
[128] S.P. Fekete and D. Krupke. Practical methods for computing large covering tours and cycle covers with turn cost. In S. Kobourov and H. Meyerhenke, editors, Meeting on Algorithm Engineering and Experiments (ALENEX2019), 2019.
[129] R. Feldmann, B. Monien, P. Mysliwietz, and S. Tschöke. A better upper bound on the bisection width of De Bruijn networks. In Rüdiger Reischuk and Michel Morvan, editors, STACS 97, pages 511-522, Berlin and Heidelberg, Germany, 1997. Springer Berlin Heidelberg.
[130] Y.Q. Feng. Automorphism groups of Cayley graphs on symmetric groups with generating transposition sets. J. Comb. Theory Ser. B, 96:67-72, 2006.
[131] C. E. Ferreira, A. Martin, C.C. de Souza, R. Weismantel, and L. A. Wolsey. The node capacitated graph partitioning problem: A computational study. Math. Program., 81:229-256, 1998.
[132] M. Fiedler. Algebraic connectivity of graphs. Czechoslovak Math. J., 23(2):298-305, 1973.
[133] A. Fischer. A polyhedral study of quadratic traveling salesman problems. PhD thesis, Chemnitz University of Technology, Chemnitz, Germany, 2013.
[134] A. Fischer. An analysis of the asymmetric quadratic traveling salesman polytope. SIAM J. Discrete Math., 28(1):240-276, 2014.
[135] A. Fischer, F. Fischer, G. Jäger, J. Keilwagen, P. Molitor, and I. Grosse. Exact algorithms and heuristics for the quadratic traveling salesman problem with an application in bioinformatics. Discrete Appl. Math., 166:87-114, 2014.
[136] A. Fischer, F. Fischer, G. Jäger, J. Keilwagen, P. Molitor, and I. Grosse. Computational recognition of RNA splice sites by exact algorithms for the quadratic traveling salesman problem. Computation, 3(2):285-298, 2015.
[137] A. Fischer and C. Helmberg. The symmetric quadratic traveling salesman problem. Math. Program., 142(1-2):205-254, 2013.
[138] F. Fischer, G. Jäger, A. Lau, and P. Molitor. Complexity and algorithms for the traveling salesman problem and the assignment problem of second order. Lecture Notes in Comput. Sci., 5165:211-224, 2009.
[139] M. Fischetti and A. Lodi. Optimizing over the first Chvátal closure. Math. Program., 110:3-20, 2007.
[140] M.L. Fisher. The Lagrangian relaxation method for solving integer programming problems. Manage Sci., 27(1):1-18, 1981.
[141] M. Fortin and R. Glowinski. Augmented Lagrangian methods: Applications to the numerical solution of boundary-value problems. Elsevier, 2000.
[142] A.G. Fowler, S.J. Devitt, and L.C.L. Hollenberg. Implementation of Shor's algorithm on a linear nearest neighbour qubit array. Quantum Inf. Comput., 4:237-251, 2004.
[143] E. Fredkin and T. Toffoli. Conservative logic. Int. J. Theor. Physics, 21:219-253, 1982.
[144] M.L. Fredman and R.E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. J. ACM, 34(3):596-615, jul 1987.
[145] E.J. Friedman. Fundamental domains for integer programs with symmetries. In Combinatorial Optimization and Applications: First International Conference, $C O C O A$ 2007, Xi'an, China, August 14-16, 2007. Proceedings 1, pages 146-153. Springer, 2007.
[146] E. Gaar and F. Rendl. A bundle approach for SDPs with exact subgraph constraints. In Integer Programming and Combinatorial Optimization: 20th International Conference, IPCO 2019, Ann Arbor, MI, USA, May 22-24, 2019, Proceedings 20, pages 205-218. Springer, 2019.
[147] E. Gaar and F. Rendl. A computational study of exact subgraph based SDP bounds for max-cut, stable set and coloring. Math. Program., 183:283-308, 2020.
[148] E. Gaar, M. Siebenhofer, and A. Wiegele. An SDP-based approach for computing the stability number of a graph. Math. Meth. Oper. Res., 95(1):141-161, 2022.
[149] D. Gabay. Applications of the method of multipliers to variational inequalities, pages 299-331. North-Holland, Amsterdam, 1989.
[150] N. Gaffke and R. Mathar. A cyclic projection algorithm via duality. Metrika, 36:29-54, 1989.
[151] G. Galbiati. The complexity of a minimum reload cost diameter problem. Discrete Appl. Math., 156:3494-3497, 2008.
[152] G. Galbiati, S. Gualandi, and F. Maffioli. On minimum reload cost cycle cover. Discrete Appl. Math., 164:112-120, 2014.
[153] P. Galinier and J. K. Hao. Hybrid evolutionary algorithms for graph coloring. J. Comb. Optim., 3:379-397, 1999.
[154] P. Galinier and A. Hertz. A survey of local search methods for graph coloring. Comput. Oper. Res., 33:2547-2562, 2006.
[155] T. Gally and M.E. Pfetsch. Computing restricted isometry constants via mixed-integer semidefinite programming. Optimization Online: http://www.optimization-online. org/DB_FILE/2016/04/5395.pdf, 2016.
[156] T. Gally, M.E. Pfetsch, and S. Ulbrich. A framework for solving mixed-integer semidefinite programs. Optim. Methods Softw., 33(3):594-632, 2018.
[157] T.M. Gally. Computational Mixed-Integer Semidefinite Programming. PhD thesis, Technische Universität Darmstadt, Darmstadt, Germany, 2019.
[158] L. M. Gambardella and M. Dorigo. Ant-Q: A reinforcement learning approach to the traveling salesman problem. In Proc. 12th Internat. Conf. Mach. Learn. (ML 1995), pages 252-260, 1995.
[159] G. Gamrath, D. Anderson, K. Bestuzheva, W. Chen, L. Eifler, M. Gasse, P. Gemander, A. Gleixner, L. Gottwald, K. Halbig, G. Hendel, C. Hojny, T. Koch, P. Le Bodic, S.J. Maher, F. Matter, M. Miltenberger, E. Mühmer, B. Müller, M.E. Pfetsch, F. Schlösser, F. Serrano, Y. Shinano, C. Tawfik, S. Vigerske, F. Wegscheider, D. Weninger, and J. Witzig. The SCIP Optimization Suite 7.0. Technical report, Optimization Online, March 2020.
[160] I. Gamvros, L. Gouveia, and S. Raghavan. Reload cost trees and network design. Networks, 59:365-379, 2012.
[161] A. Ganesan. Automorphism groups of Cayley graphs generated by connected transposition sets. Discrete Math., 313:2482-2485, 2013.
[162] A. Ganesan. Automorphism group of the complete transposition graph. J. Algebr. Comb., 42:793-801, 2015.
[163] A. Ganesan. On the automorphism group of Cayley graphs generated by transpositions. Australas. J. Comb., 64(3):432-436, 2016.
[164] A. Ganesan. Cayley graphs and symmetric interconnection networks. arXiv:1703.08109, 2017.
[165] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. Theor. Comput. Sci., 1:237-267, 1976.
[166] K. Gatermann and P.A. Parrilo. Symmetry groups, semidefinite programs, and sums of squares. J. Pure Appl. Algebra, 192(1-3):95-128, 2004.
[167] D.R. Gaur, R. Krishnamurti, and R. Kohli. The capacitated max $k$-cut problem. Math. Program., 115:65-72, 2008.
[168] B. Gavish and S.C. Graves. The travelling salesman problem and related problems. 1978. Operations Research Center Working Paper:OR 078-78.
[169] A. M. Geoffrion. Lagrangean relaxation for integer programming, pages 82-114. Springer Berlin Heidelberg, Berlin, Heidelberg, 1974.
[170] W. Gil-González, A. Molina-Cabrera, O.D. Montoya, and L.F. Grisales-Noreña. An MI-SDP model for optimal location and sizing of distributed generators in dc grids that guarantees the global optimum. Appl. Sci., 10(21):7681, 2020.
[171] F.R. Giles and W.R. Pulleyblank. Total dual integrality and integer polyhedra. Linear Algebra Appl., 25:191-196, 1979.
[172] P.C. Gilmore. Optimal and suboptimal algorithms for the quadratic assignment problem. J. Soc. Ind. Appl. Math., 10(2):305-313, 1962.
[173] P. Giselsson and S. Boyd. Linear convergence and metric selection for DouglasRachford splitting and ADMM. IEEE Trans. Autom. Control, 62(2):532-544, 2016.
[174] F. Glover. Improved linear integer programming formulations of nonlinear integer problems. Manage Sci., 22(4):455-460, 1975.
[175] C. Godsil and G. Royle. Algebraic graph theory, volume 207 of Graduate Texts in Mathematics. Springer, New York, 2001.
[176] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. J. ACM, 42(6):1115-1145, 1995.
[177] D. Goldfarb, S. Ma, and K. Scheinberg. Fast alternating linearization methods for minimizing the sum of two convex functions. Math. Program., 141:349-382, 2013.
[178] G.H. Golub and C.F. Van Loan. Matrix computations. JHU press, 2013.
[179] R.E. Gomory. Outline of an algorithm for integer solutions to linear programs. Bull. Am. Math. Soc., 64(5):275-278, 1958.
[180] L. Gourvès, A. Lyra, C. Martinhon, and J. Monnot. The minimum reload s-t path, trail and walk problems. Discrete Appl. Math., 158(13):1404-1417, 2010.
[181] R. Grone and R. Merris. Ordering trees by algebraic connectivity. Graphs Combin., 6:229-237, 1990.
[182] M. Grötschel and O. Holland. A cutting plane algorithm for minimum perfect 2matchings. Comput., 39:327-344, 1987.
[183] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. Combinatorica, 1:169-197, 1981.
[184] M. Grötschel, L. Lovász, and A. Schrijver. Geometric algorithms and combinatorial optimization, volume 2. Springer, Berlin, 1988.
[185] M. Grötschel and Y. Wakabayashi. A cutting plane algorithm for a clustering problem. Math. Program., 45:59-96, 1989.
[186] M. Grötschel and Y. Wakabayashi. Facets of the clique partitioning polytope. Math. Program., 47:367-387, 1990.
[187] L.K. Grover. A fast quantum mechanical algorithm for database search. Theory Comput., pages 212-219, 1996.
[188] L.K. Grover. Quantum mechanics helps in searching for a needle in a haystack. Phys. Rev. Lett., 79(2):325-328, 1997.
[189] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2021. https://www. gurobi.com.
[190] B. Guta. Subgradient optimization methods in integer programming with an application to a radiation therapy problem. PhD thesis, Technische Universität Kaiserslautern, Kaiserslautern, Germany, 2003.
[191] S.W. Hadley, F. Rendl, and H. Wolkowicz. A new lower bound via projection for the quadratic assignment problem. Math. Oper. Res., 17(3):727-739, 1992.
[192] H. Häffner, C.F. Roos, and R. Blatt. Quantum computing with trapped ions. Phys. Rep., 469(4):155-203, 2008.
[193] W.W. Hager, D. T. Phan, and H. Zhang. An exact algorithm for graph partitioning. Math. Program., 137:531-556, 2013.
[194] S. P. Han. A successive projection method. Math. Program., 40:1-40, 1988.
[195] B. He, H. Liu, Z. Wang, and X. Yuan. A strictly contractive Peaceman-Rachford splitting method for convex programming. SIAM J. Optim., 24(3):1011-1040, 2014.
[196] B. He, F. Ma, and X. Yuan. Convergence study of the symmetric version of ADMM with larger step sizes. SIAM J. Imaging Sci., 9(3):1467-1501, 2016.
[197] M. Held and R.M. Karp. The traveling-salesman problem and minimum spanning trees. Oper. Res., 18(6):1138-1162, 1970.
[198] M. Held and R.M. Karp. The traveling-salesman problem and minimum spanning trees: Part II. Math. Program., 1(1):6-25, 1971.
[199] M. Held, P. Wolfe, and H. P. Crowder. Validation of subgradient optimization. Math. Program., 6(1):62-88, 1974.
[200] C. Helmberg. Fixing variables in semidefinite relaxations. SIAM J. Matrix Anal. Appl., 21(2):952-969, 2000.
[201] C. Helmberg. Semidefinite programming for combinatorial optimization. habilitation, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Germany, 2000.
[202] C. Helmberg. A cutting plane algorithm for large scale semidefinite relaxations. In M. Grötschel, editor, The Sharpest Cut: The Impact of Manfred Padberg and His Work, pages 233-256. SIAM, Philadelphia, PA, 2004.
[203] A. Hertz and D. de Werra. Using tabu search techniques for graph coloring. Comput., 39:345-351, 1987.
[204] M.C. Heydemann. Cayley graphs and interconnection networks, pages 167-224. Springer Netherlands, Dordrecht, 1997.
[205] N. J. Higham. Computing a nearest symmetric positive semidefinite matrix. Linear Algebra Appl., 103:103-118, 1988.
[206] A. Hiley and B.A. Julstrom. The quadratic multiple knapsack problem and three heuristic approaches to it. In P. Heggernes, editor, Genetic and Evolutionary Computation Conference (GECCO), pages 547-552, 2006.
[207] Y. Hirata, M. Nakanishi, S. Yamashita, and Y. Nakashima. An efficient conversion of quantum circuits to a linear nearest neighbour architecture. Quantum Inf. Comput., 11:142-166, 2011.
[208] D.S. Hochbaum. Monotonizing linear programs with up to two nonzeroes per column. Oper. Res. Lett., 32:49-58, 2004.
[209] A.J. Hoffman and J.B. Kruskal. Integral boundary points of convex polyhedra. Annals of Mathematical Studies, 38, 1956.
[210] AJ Hoffman and HW Kuhn. Systems of distinct representatives and linear programming. Amer. Math. Month., 63(7):455-460, 1956.
[211] C. Hojny and M.E. Pfetsch. Handling symmetries in mixed-integer semidefinite programs. Optimization Online: https://optimization-online.org/?p=21236, 2022.
[212] M. Hong and Z.-Q. Luo. On the linear convergence of the alternating direction method of multipliers. Math. Program., 162(1-2):165-199, 2017.
[213] R.A. Horn and C.R. Johnson. Matrix analysis. Cambridge university press, 2012.
[214] R. van Houte, J. Mulderij, T. Attema, I. Chiscop, and F. Phillipson. Mathematical formulation of quantum circuit design problems in networks of quantum computers. Quantum Inf. Process, 18, 2020.
[215] H. Hu and R. Sotirov. Special cases of the quadratic shortest path problem. J. Comb. Optim., 35(3):754-777, 2018.
[216] H. Hu and R. Sotirov. On solving the quadratic shortest path problem. INFORMS J. Comput., 32(2):219-233, 2020.
[217] H. Hu and R. Sotirov. The linearization problem of a binary quadratic problem and its applications. Ann. Oper. Res., 307:229-249, 2021.
[218] H. Hu, R. Sotirov, and H. Wolkowicz. Facial reduction for symmetry reduced semidefinite and doubly nonnegative programs. Math. Program., pages 1-55, 2022.
[219] W.N. Hung, X. Song, G. Yang, J. Yang, and M. Perkowski. Optimal synthesis of multiple output boolean functions using a set of quantum gates by symbolic reachability analysis. IEEE Trans. Comput.-Aid. Des., 25:1652-1663, 2006.
[220] IBM. IBM ILOG CPLEX 22.1 User's manual, 2022. IBM ILOG CPLEX Division, Incline Village, NV.
[221] The MathWorks Inc. MATLAB version: 9.14.0 (R2023a). Natick, Massachusetts, USA, 2022.
[222] T. Itoko, R. Raymond, T. Imamichi, and A. Matsuo. Optimization of quantum circuit mapping using gate transformation and commutation. Integr., 70:43-50, 2020.
[223] A. N. Iusem and A. R. De Pierro. On the convergence of Han's method for convex programming with quadratic objective. Math. Program., 52:265-284, 1991.
[224] G. Jäger and P. Molitor. Algorithms and experimental study for the traveling salesman problem of second order. Lect. Notes Comput. Sci., 5165:211-224, 2008.
[225] C. Jansson, D. Chaykin, and C. Keil. Rigorous error bounds for the optimal value in semidefinite programming. SIAM J. Numer. Anal., 46(1):180-200, 2007.
[226] M.R. Jerrum. The complexity of finding minimum-length generator sequences. Theor. Comp. Sci., 36:265-289, 1985.
[227] V. Jeyakumar. A note on strong duality in convex semidefinite optimization: Necessary and sufficient conditions. Optim. Lett., 2:15-25, 2008.
[228] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation; part I, graph partitioning. Oper. Res., 37:865-892, 1989.
[229] S. N. Kabadi and A. P. Punnen. An $\mathcal{O}\left(n^{4}\right)$ algorithm for the QAP linearization problem. Math. Oper. Res., 36:754-761, 2011.
[230] V. Kaibel and M. Pfetsch. Packing and partitioning orbitopes. Math. Program., 114(1):1-36, 2008.
[231] S. E. Karisch and F. Rendl. Semidefinite programming and graph equipartition. In P. M. Pardalos and H. Wolkowicz, editors, Topics in Semidefinite and Interior-Point Methods, pages 77-95. American Mathematical Soc., 1998.
[232] S. E. Karisch, F. Rendl, and J. Clausen. Solving graph bisection problems with semidefinite programming. INFORMS J. Comput., 12:177-191, 2000.
[233] E. de Klerk, D. Pasechnik, R. Sotirov, and C. Dobre. On semidefinite programming relaxations of maximum $k$-section. Math. Program., 136:253-278, 2012.
[234] K. Kobayashi and Y. Takano. A branch-and-cut algorithm for solving mixed-integer semidefinite optimization problems. Comput. Optim. Appl., 75:493-513, 2020.
[235] A. Kole, K. Datta, and I. Sengupta. A heurstic for linear nearest neighbor realization of quantum circuits by SWAP gate insertion using $n$-gate lookahead. IEEE J. Emerg. Sel. Top. Circuits Syst., 6:62-72, 2016.
[236] A. Kole, K. Datta, and I. Sengupta. A new heuristic for $n$-dimensional nearest neighbor realization of a quantum circuit. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 37(1):182-192, 2017.
[237] A. Kole, K. Datta, I. Sengupta, and R. Wille. Towards a cost metric for nearest neighbor constraints in reversible circuits. In Reversible Computation ( $R C$ ): 7th International Conference, pages 273-278, Grenoble, France, 2015.
[238] T. C. Koopmans and M. J. Beckmann. Assignment problems and the location of economic activities. Econometrica, 25:53-76, 1957.
[239] M. Kočvara. Truss topology design with integer variables made easy. Optimization Online: https://optimization-online.org/2010/05/2614/, 2010.
[240] N. Krislock, J. Malick, and F. Roupin. A semidefinite branch-and-bound method for solving binary quadratic problems. ACM Trans. Math. Softw., 43(4), 2017.
[241] O. Kuryatnikova, R. Sotirov, and J.C. Vera. The maximum $k$-colorable subgraph problem and related problems. INFORMS J. Comput., 34(1):656-669, 2021.
[242] S. Lakshmivarahan, J.S. Jho, and S.K. Dhal. Symmetry in interconnection networks based on Cayley graphs of permutation groups: A survey. Parallel Comput., (19):361407, 1993.
[243] T. Larsson, M. Patriksson, and A.B. Strömberg. Conditional subgradient optimization: Theory and applications. Eur. J. Oper. Res., 88:382-403, 1996.
[244] M. Laurent and S. Poljak. The metric polytope. In Integer programming and combinatorial optimization: Proceedings of a conference held at Carnegie Mellon University, May 25-27, 1992, pages 274-286. Carnegie Mellon University, Graduate School of Industrial Administration, 1992.
[245] M. Laurent, S. Poljak, and F. Rendl. Connections between semidefinite relaxations of the max-cut and stable set problems. Math. Program., 77:225-246, 1997.
[246] E. L. Lawler. The quadratic assignment problem. Manage Sci., 9(4):586-599, 1963.
[247] J. Lee and F. Margot. On a binary-encoded ILP coloring formulation. INFORMS J. Comput., 19(3):406-415, 2007.
[248] S. Lendl, Ćustić, and A. P. Punnen. Combinatorial optimization problems with interaction costs: Complexity and solvable cases. Discrete Optim., 33:101-117, 2019.
[249] A.N. Letchford and M.M. Sørensen. Binary positive semidefinite matrices and associated integer polytopes. Math. Program. Series A, 131:253-271, 2012.
[250] L. Létocart, A. Nagih, and N. Touati-Moungla. Dantzig-Wolfe and Lagrangian decompositions in integer linear programming. Int. J. of Math. Oper. Res., 4(3):247-262, 2012.
[251] G. Li, Y. Ding, and Y. Xie. Tackling the qubit mapping problem for NISQ-era quantum devices. In Proceedings of the 24th International Conference on Architectural Support for Programming Languages and Operating Systems, pages 1001-1014, 2019.
[252] X. Li, T. K. Pong, H. Sun, and H. Wolkowicz. A strictly contractive PeacemanRachford splitting method for the doubly nonnegative relaxation of the minimum cut problem. Comput. Optim. Appl., 78:853-891, 2021.
[253] L. Liberti. Symmetry in mathematical programming. In Mixed Integer Nonlinear Programming, pages 263-283. Springer, 2012.
[254] F. Lieder, F.B.A. Rad, and F. Jarre. Unifying semidefinite and set-copositive relaxations of binary problems and randomization techniques. Comput. Optim. Appl., 61:669-688, 2015.
[255] P.-L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. SIAM J. Num. Anal., 16(6):964-979, 1979.
[256] A. Lisser and F. Rendl. Graph partitioning using linear and semidefinite programming. Math. Program., 95:91-101, 2003.
[257] J. Löfberg. YALMIP : A toolbox for modeling and optimization in MATLAB. In Proceedings of the CACSD Conference, Taipei, Taiwan, 2004.
[258] D. A. Lorenz and Q. Tran-Dinh. Non-stationary Douglas-Rachford and alternating direction method of multipliers: Adaptive stepsizes and convergence. Comput. Optim. Appl., 74:67-92, 2019.
[259] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. SIAM J. Optim., 1:166-190, 1991.
[260] N. H. A. Mai, J. B. Lasserre, V. Magron, and J. Wang. Exploiting constant trace property in large-scale polynomial optimization. ACM Trans. Math. Soft., 48(4):1-39, 2022.
[261] N. H. A. Mai, V. Magron, and J. B. Lasserre. A hierarchy of spectral relaxations for polynomial optimization. Math. Program. Comput., pages 1-51, 2023.
[262] J.F. Marcotorchino. Aggregation of Similarities in Automatic Classification (in French). PhD thesis, Université Paris VI, Paris, France, 1981.
[263] F. Margot. Pruning by isomorphism in branch-and-cut. Math. Program., 94:71-90, 2002.
[264] F. Margot. Exploiting orbits in symmetric ILP. Math. Program., 98:3-21, 2003.
[265] F. Margot. Small covering designs by branch-and-cut. Math. Program., 94:207-220, 2003.
[266] F. Margot. Symmetry in integer linear programming, pages 647-686. Springer, 2009.
[267] S. Mars. Mixed-integer semidefinite programming with an application to truss topology design. PhD thesis, FAU Erlangen-Nürnberg, Erlangen, Germany, 2013.
[268] R.E. Marsten. The use of the box step method in discrete optimization. Technical report, National Bureau of Economic Research, 1975.
[269] A. Matsuo and S. Yamashita. Changing the gate order for optimal LNN conversion. In A. De Vos and R. Wille, editors, Reversible Computation, pages 89-101, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
[270] F. Matter and M.E. Pfetsch. Presolving for mixed-integer semidefinite optimization. INFORMS J. Optim., 5(2):131-154, 2023.
[271] F. de Meijer. Bounds on the minimum reload cost cycle cover problem. Master's thesis, Tilburg University, 2018.
[272] F. de Meijer, D. Gijswijt, and R. Sotirov. Exploiting symmetries in optimal quantum circuit design. Working paper, 2023.
[273] F. de Meijer, M. Siebenhofer, R. Sotirov, and A. Wiegele. Integer semidefinite programming for the quadratic minimum spanning tree problem. Working paper, 2023.
[274] F. de Meijer and R. Sotirov. The quadratic cycle cover problem: Special cases and efficient bounds. J. Comb. Optim., 39:1096-1128, 2020.
[275] F. de Meijer and R. Sotirov. SDP-based bounds for the quadratic cycle cover problem via cutting-plane augmented Lagrangian methods and reinforcement learning. INFORMS J. Comput., 33:1262-1276, 2021.
[276] F. de Meijer and R. Sotirov. The Chvátal-Gomory procedure for integer SDPs with applications in combinatorial optimization. arXiv:2201.10224v2, 2023.
[277] F. de Meijer and R. Sotirov. On integrality in semidefinite programming for discrete optimization. arXiv:2306.09865, 2023.
[278] F. de Meijer, R. Sotirov, A. Wiegele, and S. Zhao. Partitioning through projections: Strong SDP bounds for large graph partition problems. Comput. Oper. Res., 151:106088, 2023.
[279] P. Meurdesoif. Strengthening the Lovász bound for graph coloring. Math. Program., 102:577-588, 2005.
[280] R.R. Meyer. On the existence of optimal solutions to integer and mixed integer programming problems. Math. Program., 7:223-235, 1974.
[281] C.E. Miller, A.W. Tucker, and R.A. Zemlin. Integer programming formulation of traveling salesman problems. J. ACM, 7(4):326-329, 1960.
[282] A. Montanaro. Quantum algorithms: An overview. NPJ Quantum Inf., 2(1):1-8, 2016.
[283] D.R. Morrison, S.H. Jacobson, J.J. Sauppe, and E.C. Sewell. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. Discrete Optim., 19:79-102, 2016.
[284] A. P. S. MOSEK. The MOSEK optimization toolbox for MATLAB manual. Version 8.0., 2018.
[285] J. Mulderij. Nearest neighbor compliance in quantum circuit design. Master's thesis, Delft University of Technology, 2019.
[286] J. Mulderij, K.I. Aardal, I. Chiscop, and F. Phillipson. A polynomial size model with implicit SWAP gate counting for exact qubit reordering. arXiv:2009.08748, 2020.
[287] G. Narasimhan. The maximum $k$-colorable subgraph problem. PhD thesis, University of Wisconsin-Madison, Madison, WI, USA.
[288] V. Narayanan. Facial structure and representation of integer hulls of convex sets. In Integer Programming and Combinatorial Optimization: 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18-20, 2013. Proceedings 16, pages 302-313. Springer, 2013.
[289] Y. Nesterov. Nonsmooth convex optimization, pages 139-240. Springer, 2018.
[290] Y. Nesterov and A. Nemirovskii. Interior-point polynomial algorithms in convex programming. SIAM, 1994.
[291] M.A. Nielsen and I. Chuang. Quantum computation and quantum information. Cambridge University Press, New York, USA, 2010.
[292] D. E. Oliveira, H. Wolkowicz, and Y. Xu. ADMM for the SDP relaxation of the QAP. Math. Program. Comput., 10:631-658, 2018.
[293] M. Oosten, J.H.G.C. Rutten, and F.C.R. Spieksma. The clique partitioning problem: Facets and patching facets. Networks, 38:209-226, 2001.
[294] R. Orús, S. Mugel, and E. Lizaso. Quantum computing for finance: Overview and prospects. Rev. Phys., 4:100028, 2019.
[295] J. Ostrowski, J. Linderoth, F. Rossi, and S. Smriglio. Orbital branching. Math. Program., 126:147-178, 2011.
[296] C. Outeiral, M. Strahm, J. Shi, G. M. Morris, S. C. Benjamin, and C. M. Deane. The prospects of quantum computing in computational molecular biology. WIREs Comput. Mol. Sci., 11:e1481, 2021.
[297] M. Padberg. The Boolean quadric polytope: Some characteristics, facets and relatives. Math. Program., 45:139-172, 1989.
[298] G. Pataki. On the closedness of the linear image of a closed convex cone. Math. Oper. Res., 32(2):395-412, 2007.
[299] M. Pedram and A. Shafaei. Layout optimization for quantum circuits with linear nearest neighbor architectures. IEEE Circuits Syst. Mag., 16:62-74, 2016.
[300] J. Peña, J.C. Vera, and L.F. Zuluaga. Linear convergence of the Douglas-Rachford algorithm via a generic error bound condition. arXiv:2111.06071, 2021.
[301] A. Peres. Reversible logic and quantum computers. Phys. Rev. A, Gen. Phys., 32:32663276, 1985.
[302] A. Philipp, S. Ulbrich, Y. Cheng, and M. Pesavento. Multiuser downlink beamforming with interference cancellation using a SDP-based branch-and-bound algorithm. In Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing, pages 7724-7728. IEEE, 2014.
[303] A. Del Pia and D. Zhou. An SDP relaxation for the sparse integer least square problem. arXiv:2203.02607, 2023.
[304] G. Pierra. Decomposition through formalization in a product space. Math. Program., 28:96-115, 1984.
[305] M. Pilanci, M.J. Wainwright, and L. El Ghaoui. Sparse learning via Boolean relaxations. Math. Prog., 151:63-87, 2015.
[306] S. Poljak and F. Rendl. Nonpolyhedral relaxations of graph-bisection problems. SIAM J. Optim., 5(3):467-487, 1995.
[307] B.T. Polyak. Minimization of unsmooth functionals. USSR Comput. Math. Math. Phys., 9:14-29, 1969.
[308] J. Povh, F. Rendl, and A. Wiegele. A boundary point method to solve semidefinite programs. Comput., 78(3):277-286, 2006.
[309] A. P. Punnen and S. N. Kabadi. A linear time algorithm for the Koopmans-Beckmann QAP linearization and related problems. Discrete Optim., 10(3):200-209, 2013.
[310] A. P. Punnen, P. Pandey, and M. Friesen. Representations of quadratic combinatorial optimization problems: A case study using quadratic set covering and quadratic knapsack problems. Comput. Oper. Res., 112, 2019.
[311] A.P. Punnen, M. Walter, and B.D. Woods. A characterization of linearizable instances of the quadratic traveling salesman problem. arXiv:1708.07217, 2018.
[312] Qiskit contributors. Qiskit: An open-source framework for quantum computing, 2023.
[313] P. Raghavan and C. D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. Combinatorica, 7(4):365-374, 1987.
[314] M. Ramana and A.J. Goldman. Some geometric results in semidefinite programming. J. Glob. Optim., 7:33-50, 1995.
[315] M.V. Ramana. Polyhedra, spectrahedra, and semidefinite programming. In Topics in semidefinite and interior-point methods, Fields Institute Communications, volume 18, pages 27-38, Toronto, ON, 1997.
[316] B. Recht, M. Fazel, and P.A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. SIAM Rev., 52(3):471-501, 2010.
[317] G. Reinelt. TSPLIB - A traveling salesman problem library. INFORMS J. Comput., 3:376-384, 1991.
[318] F. Rendl, G. Rinaldi, and A. Wiegele. Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. Math. Program., 121(2):307-335, 2010.
[319] F. Rendl and R. Sotirov. The min-cut and vertex separator problem. Comput. Optim. Appl., 69:159-187, 2018.
[320] F. Rendl, R. Sotirov, and C. Truden. Lower bounds for the bandwidth problem. Comput. Oper. Res., 135:105422, 2021.
[321] F. Rendl and H. Wolkowicz. A projection technique for partitioning the nodes of a graph. Ann. Oper. Res., 58(3):155-179, 1995.
[322] G. Rinaldi. Rudy: A generator for random graphs, 1998.
[323] N. Rontsis, P. Goulart, and Y. Nakatsukasa. Efficient semidefinite programming with approximate ADMM. J. Optim. Theory Appl., 192:292-320, 2022.
[324] B. Rostami, A. Chassein, M. Hopf, D. Frey, C. Buchheim, F. Malucelli, and M. Goerigk. The quadratic shortest path problem: Complexity, approximability and solution methods. Eur. J. Oper. Res., 268(2):473-485, 2018.
[325] B. Rostami and F. Malucelli. Lower bounds for the quadratic minimum spanning tree problem based on reduced cost computation. Comput. Oper. Res., 64:178-188, 2015.
[326] B. Rostami, F. Malucelli, P. Belotti, and S. Gualandi. Lower bounding procedure for the asymmetric quadratic traveling salesman problem. Eur. J. Oper. Res., 253(3):584592, 2016.
[327] S. Sahni and T. Gonzalez. P-complete approximation problems. J. ACM, 23:555-565, 1976.
[328] A. Schrijver. A comparison of the Delsarte and Lovász bounds. IEEE Trans. Inform. Theory, 25:425-429, 1979.
[329] A. Schrijver. On cutting planes. Ann. Discrete Math., 9:291-296, 1980.
[330] A. Schrijver. Theory of linear and integer programming. John Wiley \& Sons, 1998.
[331] A. Schrijver. Combinatorial optimization: polyhedra and efficiency, volume 24. Springer, 2003.
[332] A. Shafaei, M. Saeedi, and M. Pedram. Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures. In 2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC), pages 1-6, 2013.
[333] A. Shafaei, M. Saeedi, and M. Pedram. Qubit placement to minimize communication overhead in 2D quantum architectures. In 2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC), pages 495-500, 2014.
[334] H.D. Sherali and J.C. Smith. Improving discrete model representations via symmetry considerations. Manage. Sci., 47(10):1396-1407, 2001.
[335] H.D. Sherali and O. Ulular. A primal-dual conjugate subgradient algorithm for specially structured linear and convex programming problems. Appl. Math. Optim., 20:193-221, 1989.
[336] N.Z. Shor. Quadratic optimization problems. Soviet J. Comput. Syst. Sci., 25:1-11, 1987.
[337] P.W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings 35th Annual Symposium on Foundations of Computer Science, pages 124-134. IEEE Computer Society Press, 1994.
[338] ACM SIGKDD and Netflix. Proceedings of KDD Cup and Workshop. Available at http://www.cs.uic.edu/~liub/KDD-cup-2007/proceedings.html, 2007.
[339] L. Sinjorgo and R. Sotirov. On solving the MAX-SAT using sum of squares. arXiv:2302.06931, 2023.
[340] M. Sion. On general minimax theorems. Pac. J. Math., 8(1):171-176, 1958.
[341] M.Y. Siraichi, V.F. dos Santos, S. Collange, and F.M.Q. Pereira. Qubit allocation. In Proceedings of the 2018 International Symposium on Code Generation and Optimization (CGO 2018), pages 113-125. ACM, 2018.
[342] R. Sotirov. SDP relaxations for some combinatorial optimization problems. In M.F. Anjos and J.B. Lasserre, editors, Handbook on Semidefinite, Conic and Polynomial Optimization, pages 795-819. Springer US, Boston, MA, 2012.
[343] R. Sotirov. An efficient semidefinite programming relaxation for the graph partition problem. INFORMS J. Comput., 26:16-30, 2014.
[344] R. Sotirov. Graph bisection revisited. Ann. Oper. Res., 265:143-154, 2018.
[345] V.J.R. de Sousa, M.F. Anjos, and S. Le Digabel. Computational study of valid inequalities for the maximum $k$-cut problem. Ann. Oper. Res., 265:5-27, 2018.
[346] C.C. de Souza. The graph equipartition problem: Optimal solutions, extensions and applications. PhD thesis, Université Catholique de Louvain, Louvain-la-Neuve, Belgium, 1993.
[347] R. Staněk, P. Greistorfer, K. Ladner, and U. Pferschy. Geometric and LP-based heuristics for angular travelling salesman problems in the plane. Comp. Oper. Res., 108:97111, 2019.
[348] C. Stein and D.P. Wagner. Approximation algorithms for the minimum bends traveling salesman problem. In K. Aardal and B. Gerards, editors, Integer Programming and Combinatorial Optimization (IPCO 2001), volume 2081 of Lecture Notes in Computer Science, pages 406-425. Springer, Berlin, Heidelberg, 2001.
[349] R.A. Stubbs and S. Mehrotra. A branch-and-cut method for 0-1 mixed convex programming. Math. Program., 86:515-532, 1999.
[350] D. Sun, K. C. Toh, Y. Yuan, and X. Y. Zhao. SDPNAL +: A Matlab software for semidefinite programming with bound constraints (version 1.0). Optim. Methods Softw., 35(1):87-115, 2020.
[351] R.J. Tibshirani. Dykstra's algorithm, ADMM, and coordinate descent: Connections, insights, and extensions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems (NIPS 2017)), volume 30, Long Beach, CA, 2017. Curran Associates, Inc.
[352] T. Toffoli. Reversible computing. Technical report, MIT Lab for Computer Science, 1980. Technical memo MIT/LCS/TM-151.
[353] A.W. Tucker. Solving a matrix game by linear programming. IBM J. Res. Dev., 4(5):507-517, 1960.
[354] A.W. Tucker. Combinatorial theory underlying linear programs. Recent Adv. Math. Program., pages 1-16, 1963.
[355] L. Tunçel. On the Slater condition for the SDP relaxations of nonconvex sets. Oper. Res. Lett., 29:181-186, 2001.
[356] R. Van Houte, J. Mulderij, T. Attema, I. Chiscop, and F. Phillipson. Mathematical formulation of quantum circuit design problems in networks of quantum computers. Quantum Inf. Process., 19:1-22, 2020.
[357] J. J. P. Veerman and R. Lyons. A primer on Laplacian dynamics in directed graphs. Nonlinear Phenom. Complex Syst., 23(2):196-206, 2020.
[358] D. Venturelli, M. Do, E. Rieffel, and J. Frank. Temporal planning for compilation of quantum approximate optimization circuits. In Scheduling and Planning Applications Workshop (SPARK), page 58, 2017.
[359] S. de Vries and B. Perscheid. Tight compact extended relaxations for nonconvex quadratic programming problems with box constraints. J. Glob. Optim., 84(3):591606, 2022.
[360] A.L. Wang and F. Kilinç-Karzan. On the tightness of SDP relaxations of QCQPs. Math. Program., 193:33-73, 2022.
[361] C. J. C. H. Watkins. Learning with delayed rewards. PhD thesis, University of Cambridge, Cambridge, United Kingdom, 1989.
[362] K.D. Wayne. A polynomial combinatorial algorithm for generalized minimum cost flow. Math. Oper. Res., 27:445-459, 2002.
[363] Z. Wen, D. Goldfarb, and W. Yin. Alternating direction augmented Lagrangian methods for semidefinite programming. Math. Program. Comput., 2:203-230, 2010.
[364] A. Wiegele. Biq Mac Library - A collection of max-cut and quadratic 0-1 programming instances of medium size. Preprint, 2007.
[365] A. Wiegele and S. Zhao. SDP-based bounds for graph partition via extended ADMM. Comput. Optim. Appl., 82:251-291, 2022.
[366] R. Wille, L. Burgholzer, and A. Zulehner. Mapping quantum circuits to IBM QX architectures using the minimal number of SWAP and H operations. In Proceedings of the 56th Annual Design Automation Conference, pages 1-6, Las Vegas, NV, USA, 2019.
[367] R. Wille, D. Große, L. Teuber, G. W. Dueck, and R. Drechsler. RevLib: An online resource for reversible functions and reversible circuits. In Int'l Symp. on Multi-Valued Logic, pages 220-225, 2008. RevLib is available at http://www.revlib.org.
[368] R. Wille, O. Keszocze, M. Walter, P. Rohrs, A. Chattopadhyay, and R. Drechsler. Look-ahead schemes for nearest neighbor optimization of 1D and 2D quantum circuits. In 2016 21st Asia and South Pacific design automation conference (ASP-DAC), pages 292-297, Macao, 2016. IEEE.
[369] R. Wille, A. Lye, and R. Drechsler. Exact reordering of circuit lines for nearest neighbor quantum architectures. IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 33(12):1818-1831, 2014.
[370] H. Wirth and J. Steffan. Reload cost problems: Minimum diameter spanning tree. Discrete Appl. Math., 113:73-85, 2001.
[371] H. Wolkowicz and Q. Zhao. Semidefinite programming relaxations for the graph partitioning problem. Discrete Appl. Math., 96/97:461-479, 1999.
[372] L.A. Wolsey. Integer programming. John Wiley \& Sons, 2020.
[373] L.A. Wolsey and G.L. Nemhauser. Integer and combinatorial optimization, volume 55. John Wiley \& Sons, 1999.
[374] B.D. Woods and A.P. Punnen. A class of exponential neighbourhoods for the quadratic travelling salesman problem. J. Comb. Optim., 40(2):303-332, 2020.
[375] C.W. Wu. Algebraic connectivity of directed graphs. Linear Multilinear Algebra, 53(3):203-223, 2005.
[376] K. Yonekura and Y. Kanno. Global optimization of robust truss topology via mixed integer semidefinite programming. Optim. Eng., 11(3):355-379, 2010.
[377] A. Yurtsever, J.A. Tropp, O. Fercoq, M. Udell, and V. Cevher. Scalable semidefinite programming. SIAM J. Math. Data Sci., 3(1):171-200, 2021.
[378] M. Zamani, H. Abbaszadehpeivasti, and E. de Klerk. The exact worst-case convergence rate of the alternating direction method of multipliers. arXiv:2206.09865v4, 2023.
[379] H. Zhang, Z. Zhang, J. Chen, and J. Wang. Solving quadratic traveling salesman problem with deep reinforcement learning. In 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pages 2508-2513, 2022.
[380] S. Zhao. Splitting into Pieces: Alternating Direction Methods of Multipliers and Graph Partitioning. PhD thesis, Alpen-Adria-Universität Klagenfurt, Klagenfurt, Austria, 2022.
[381] X. Zhao, H. Huang, and T.P. Speed. Finding short DNA motifs using permuted Markov models. J. Comput. Biol., 12:894-906, 2005.
[382] X. Zhao, D. Sun, and K. Toh. A Newton-CG augmented Lagrangian method for semidefinite programming. SIAM J. Optim., 20(4):1737-1765, 2010.
[383] X. Zheng, H. Chen, Y. Xu, Z. Li, Z. Lin, and Z. Liang. A mixed-integer SDP solution to distributionally robust unit commitment with second order moment constraints. CSEE J. Power Energy Syst., 6(2):374-383, 2020.

## CENTER DISSERTATION SERIES

CentER for Economic Research, Tilburg University, the Netherlands

| No. | Author | Title | ISBN | Published |
| :---: | :---: | :---: | :---: | :---: |
| 672 | Joobin Ordoobody | The Interplay of Structural and Individual Characteristics | $\begin{aligned} & 97890 \\ & 56686741 \end{aligned}$ | $\begin{aligned} & \text { February } \\ & 2022 \end{aligned}$ |
| 673 | Lucas Avezum | Essays on Bank Regulation and Supervision | $\begin{aligned} & 97890 \\ & 56686758 \end{aligned}$ | March 2022 |
| 674 | Oliver Wichert | Unit-Root Tests in High-Dimensional Panels | $\begin{aligned} & 97890 \\ & 56686765 \end{aligned}$ | April 2022 |
| 675 | Martijn de Vries | Theoretical Asset Pricing under Behavioral Decision Making | $\begin{aligned} & 97890 \\ & 56686772 \end{aligned}$ | June 2022 |
| 676 | Hanan Ahmed | Extreme Value Statistics using Related Variables | $\begin{aligned} & 97890 \\ & 56686789 \end{aligned}$ | June 2022 |
| 677 | Jan Paulick | Financial Market Information Infrastructures: Essays on Liquidity, Participant Behavior, and Information Extraction | $\begin{aligned} & 97890 \\ & 56686796 \end{aligned}$ | June 2022 |
| 678 | Freek van Gils | Essays on Social Media and Democracy | $\begin{aligned} & 97890 \\ & 56686802 \end{aligned}$ | June 2022 |
| 679 | Suzanne Bies | Examining the Effectiveness of Activation Techniques on Consumer Behavior in Temporary Loyalty Programs | $\begin{aligned} & 97890 \\ & 56686819 \end{aligned}$ | July 2022 |
| 680 | Qinnan Ruan | Management Control Systems and Ethical Decision Making | $\begin{aligned} & 97890 \\ & 56686826 \end{aligned}$ | June 2022 |
| 681 | Lingbo Shen | Essays on Behavioral Finance and Corporate Finance | $\begin{aligned} & 97890 \\ & 56686833 \end{aligned}$ | August 2022 |
| 682 | Joshua Eckblad | Mind the Gales: An Attention-Based View of Startup Investment Arms | $\begin{aligned} & 97890 \\ & 56686840 \end{aligned}$ | August 2022 |
| 683 | Rafael Greminger | Essays on Consumer Search | $\begin{aligned} & 97890 \\ & 56686857 \end{aligned}$ | August 2022 |
| 684 | Suraj Upadhyay | Essay on policies to curb rising healthcare expenditures | $\begin{aligned} & 97890 \\ & 56686864 \end{aligned}$ | $\begin{aligned} & \text { September } \\ & 2022 \end{aligned}$ |


| No. | Author | Title | ISBN | Published |
| :---: | :---: | :---: | :---: | :---: |
| 685 | Bert-Jan Butijn | From Legal Contracts to Smart Contracts and Back Again: An Automated Approach | $\begin{aligned} & 97890 \\ & 56686871 \end{aligned}$ | $\begin{aligned} & \text { September } \\ & 2022 \end{aligned}$ |
| 686 | Sytse Duiverman | Four essays on the quality of auditing: Causes and consequences | $\begin{aligned} & 97890 \\ & 56686888 \end{aligned}$ | $\begin{aligned} & \text { October } \\ & 2022 \end{aligned}$ |
| 687 | Lucas Slot | Asymptotic Analysis of Semidefinite Bounds for Polynomial Optimization and Independent Sets in Geometric Hypergraphs | $\begin{aligned} & 97890 \\ & 56686895 \end{aligned}$ | September $2022$ |
| 688 | Daniel Brosch | Symmetry reduction in convex optimization with applications in combinatorics | $\begin{aligned} & 97890 \\ & 56686901 \end{aligned}$ | $\begin{aligned} & \text { October } \\ & 2022 \end{aligned}$ |
| 689 | Emil Uduwalage | Essays on Corporate Governance in Sri Lanka | $\begin{aligned} & 97890 \\ & 56686918 \end{aligned}$ | $\begin{aligned} & \text { October } \\ & 2022 \end{aligned}$ |
| 690 | Mingjia Xie | Essays on Education and Health Economics | $\begin{aligned} & 97890 \\ & 56686925 \end{aligned}$ | $\begin{aligned} & \text { October } \\ & 2022 \end{aligned}$ |
| 691 | Peerawat Samranchit | Competition in Digital Markets | $\begin{aligned} & 97890 \\ & 56686932 \end{aligned}$ | $\begin{aligned} & \text { October } \\ & 2022 \end{aligned}$ |
| 692 | Jop Schouten | Cooperation, allocation and strategy in interactive decision-making | $\begin{aligned} & 97890 \\ & 56686949 \end{aligned}$ | $\begin{aligned} & \text { December } \\ & 2022 \end{aligned}$ |
| 693 | Pepijn Wissing | Spectral Characterizations of Complex Unit Gain Graphs | $\begin{aligned} & 97890 \\ & 56686956 \end{aligned}$ | November $2022$ |
| 694 | Joris Berns | CEO attention, emotion, and communication in corporate financial distress | $\begin{aligned} & 97890 \\ & 56686963 \end{aligned}$ | November $2022$ |
| 695 | Tom Aben | The (long) road towards smart management and maintenance: Organising the digital transformation of critical infrastructures | $\begin{aligned} & 97890 \\ & 56686970 \end{aligned}$ | $\begin{aligned} & \text { December } \\ & 2022 \end{aligned}$ |
| 696 | Gülbike Mirzaoğlu | Essays in Economics of Crime Prevention and Behavior Under Uncertainty | $\begin{aligned} & 97890 \\ & 56686987 \end{aligned}$ | $\begin{aligned} & \text { February } \\ & 2023 \end{aligned}$ |
| 697 | Suwei An | Essays on incentive contracts, M\&As, and firm risk | $\begin{aligned} & 97890 \\ & 56686994 \end{aligned}$ | $\begin{aligned} & \text { February } \\ & 2023 \end{aligned}$ |
| 698 | Jorgo Goossens | Non-standard Preferences in Asset Pricing and Household Finance | $\begin{aligned} & 97890 \\ & 56687007 \end{aligned}$ | $\begin{aligned} & \text { February } \\ & 2023 \end{aligned}$ |


| No. | Author | Title | ISBN | Published |
| :---: | :---: | :---: | :---: | :---: |
| 699 | Santiago Bohorquez Correa | Risk and rewards of residential energy efficiency | $\begin{aligned} & 97890 \\ & 56687014 \end{aligned}$ | April 2023 |
| 700 | Gleb Gertsman | Behavioral Preferences and Beliefs in Asset Pricing | $\begin{aligned} & 97890 \\ & 56687021 \end{aligned}$ | May 2023 |
| 701 | Gabriella Massenz | On the Behavioral Effects of Tax Policy | $\begin{aligned} & 97890 \\ & 56687038 \end{aligned}$ | May 2023 |
| 702 | Yeqiu Zheng | The Effect of Language and Temporal Focus on Cognition, Economic Behaviour, and Well-Being | $\begin{aligned} & 97890 \\ & 56687045 \end{aligned}$ | May 2023 |
| 703 | Michela Bonani | Essays on Innovation, Cooperation, and Competition Under Standardization | $\begin{aligned} & 97890 \\ & 56687052 \end{aligned}$ | June 2023 |
| 704 | Fabien Ize | The Role of Transparency in Fairness and Reciprocity Issues in Manager-Employee Relationships | $\begin{aligned} & 97890 \\ & 56687069 \end{aligned}$ | June 2023 |
| 705 | Kristel de Nobrega | Cyber Defensive Capacity and Capability: A Perspective from the Financial Sector of a Small State | $\begin{aligned} & 97890 \\ & 56687076 \end{aligned}$ | July 2023 |
| 706 | Christian Peters | The Microfoundations of Audit Quality | $\begin{aligned} & 97890 \\ & 56687083 \end{aligned}$ | June 2023 |
| 707 | Felix Kirschner | Conic Optimization with Applications in Finance and Approximation Theory | $\begin{aligned} & 97890 \\ & 56687090 \end{aligned}$ | July 2023 |
| 708 | Zili Su | Essays on Equity Incentive and Share Pledging in China | $\begin{aligned} & 97890 \\ & 56687106 \end{aligned}$ | $\begin{aligned} & \text { September } \\ & 2023 \end{aligned}$ |
| 709 | Rafael Escamilla | Managing the Nanostore Supply Chain: Base-of-the-Pyramid Retail in Emerging Markets | $\begin{aligned} & 97890 \\ & 56687113 \end{aligned}$ | $\begin{aligned} & \text { September } \\ & 2023 \end{aligned}$ |
| 710 | Tomas Jankauskas | Essays in Empirical Finance | $\begin{aligned} & 97890 \\ & 56687120 \end{aligned}$ | August 2023 |
| 711 | Tung Nguyen Huy | Fostering Sustainable Land Management in Sub-Saharan Africa: Evidence from Ghana and Burkina Faso | $\begin{aligned} & 97890 \\ & 56687137 \end{aligned}$ | $\begin{aligned} & \text { September } \\ & 2023 \end{aligned}$ |
| 712 | Daniel Karpati | Essays in Finance \& Health | $\begin{aligned} & 97890 \\ & 56687144 \end{aligned}$ | $\begin{aligned} & \text { September } \\ & 2023 \end{aligned}$ |
| 713 | Mylène Struijk | IT Governance in the Digital Era: Insights from Meta-Organizations | $\begin{aligned} & 97890 \\ & 56687151 \end{aligned}$ | $\begin{aligned} & \text { September } \\ & 2023 \end{aligned}$ |


| No. | Author | Title | ISBN | Published |
| :---: | :---: | :---: | :---: | :---: |
| 714 | Albert Rutten | Essays on Work and Retirement | $\begin{aligned} & 97890 \\ & 56687168 \end{aligned}$ | November $2023$ |
| 715 | Yan Liu | Essays on Credit Rating Agencies in China | $\begin{aligned} & 97890 \\ & 56687175 \end{aligned}$ | $\begin{aligned} & \text { October } \\ & 2023 \end{aligned}$ |
| 716 | Xiaoyue Zhang | Distortions and industrial upgrading in China | $\begin{aligned} & 97890 \\ & 56687182 \end{aligned}$ | $\begin{aligned} & \text { September } \\ & 2023 \end{aligned}$ |
| 717 | Andries van Beek | Solutions in multi-actor projects with collaboration and strategic incentives | $\begin{aligned} & 97890 \\ & 56687199 \end{aligned}$ | $\begin{aligned} & \text { October } \\ & 2023 \end{aligned}$ |
| 718 | Andries Steenkamp | Polynomial Optimization: Matrix Factorization Ranks, Portfolio Selection, and Queueing Theory | $\begin{aligned} & 97890 \\ & 56687205 \end{aligned}$ | $\begin{aligned} & \text { October } \\ & 2023 \end{aligned}$ |
| 719 | Luis Vargas | Sum-of-Squares Representations for Copositive Matrices and Independent Sets in Graphs | $\begin{aligned} & 97890 \\ & 56687212 \end{aligned}$ | November $2023$ |
| 720 | Zhenshu Wu | Essays in Corporate Finance and ESG | $\begin{aligned} & 97890 \\ & 56687229 \end{aligned}$ | November $2023$ |
| 721 | Frank de Meijer | Integrality and Cutting Planes in Semidefinite Programming Approaches for Combinatorial Optimization | $\begin{aligned} & 97890 \\ & 56687236 \end{aligned}$ | November $2023$ |

Frank de Meijer (Zundert, The Netherlands, 1996) received his Bachelor's degree in Econometrics and Operations Research from Tilburg University in 2017, followed by a Master's degree (2018) and a Research Master's degree (2019), both in Operations Research, from the same university. In September 2019, he became a PhD candidate in Operations Research at Tilburg University. During the PhD period, Frank made academic visits to the Alpen-Adria-Universität Klagenfurt (Austria) and the University of Waterloo (Canada).

Many real-life decision problems are discrete in nature. To solve such problems as mathematical optimization problems, integrality constraints are commonly incorporated in the model to reflect the choice of finitely many alternatives. At the same time, it is known that semidefinite programming is very suitable for obtaining strong relaxations of combinatorial optimization problems. In this dissertation, we study the interplay between semidefinite programming and integrality, where a special focus is put on the use of cutting-plane methods. Although the notions of integrality and cutting planes are well-studied in linear programming, integer semidefinite programs (ISDPs) are considered only recently. We show that many combinatorial optimization problems can be modeled as ISDPs. Several theoretical concepts, such as the Chvátal-Gomory closure, total dual integrality and integer Lagrangian duality, are studied for the case of integer semidefinite programming. On the practical side, we introduce an improved branch-and-cut approach for ISDPs and a cutting-plane augmented Lagrangian method for solving semidefinite programs with a large number of cutting planes. Throughout the thesis, we apply our results to a wide range of combinatorial optimization problems, among which the quadratic cycle cover problem, the quadratic traveling salesman problem and the graph partition problem. Our approaches lead to novel, strong and efficient solution strategies for these problems, with the potential to be extended to other problem classes.

ISBN: 9789056687236
DOI: 10.26116/dpxe-ha35


[^0]:    ${ }^{1}$ Strictly speaking, a SWAP gate does only interchange the state of the involved qubits, while the actual hardware entities remain unchanged in the architecture.

