Ph.D. Program in

Automatic Control, Bioengineering and Operation Research

Cycle 29

Two Essays in Computational Optimization: Computing the Clar Number in Fullerene Graphs and Distributing the Errors in Iterative Interior Point Methods

---

Author:
Maryam SALAMI

Supervisors:
Dr. Giovanni Rinaldi
Dr. Claudio Gentile

June 15, 2018

Abstract

Fullerene are cage-like hollow carbon molecules graph of pseudospherical symmetry consisting of only pentagons and hexagons faces. It has been the object of interest for chemists and mathematicians due to its widespread application in various fields, namely including electronic and optic engineering, medical science and biotechnology. A Fullerene molecular, Γ_n of n atoms has a multiplicity of isomers which increases as $N_{iso} \sim \mathcal{O}(n^9)$. For instance, Γ_{180} has 79,538,751 isomers. The Fries and Clar numbers are stability predictors of a Fullerene molecule. These number can be computed by solving a (possibly NP-hard) combinatorial optimization problem. We propose several ILP formulation of such a problem each yielding a solution algorithm that provides the exact valued of the Fries and Clar numbers. We compare the performances of the algorithm derived from the proposed ILP formulations. One of this algorithm is used to find the Clar isomers, i.e., those for which the Clar number is maximum among all isomers having a given size. We repeated this computational experiment for all sizes up to 204 atoms. In the course of the study a total of 2 649 413 774 isomers were analyzed.

The second essay concerns developing an *iterative* primal dual infeasible path following (PDIPF) interior point (IP) algorithm for separable convex quadratic minimum cost flow network problem. In each iteration of PDIPF algorithm, the main computational effort is solving the underlying Newton search direction system. We concentrated on finding the solution of the corresponding linear system iteratively and inexactly. We assumed that all the involved inequalities can be solved inexactly and to this purpose, we focused on different approaches for distributing the error generated by iterative linear solvers such that the convergences of the PDIPF algorithm are guaranteed. As a result, we achieved theoretical bases that open the path to further interesting practical investigation.

Acknowladgement

A very special gratitude goes out to my advisors Dr. Giovanni Rinaldi and Dr. Claudio Gentile who gave me the great opportunity doing researches under their supervision and writing my PhD thesis with such an interesting and ambitious topics in the field of combinatorial optimization.

It is my honor working with Dr. Giovanni Rinaldi who has always gladly shared his insight in general as well as his expertise in combinatorial optimization in particular. I want to thank him for his enthusiasm about discussing mathematical issues. His ideas and advices has been always led me into the world of active research.

The mathematical optimization and operational research group of **IASI-CNR** provided me with a very welcome and supporting working environment. Over the years, it is pleasure to be surrounded by dedicated colleagues of IASI-CNR to whom I am grateful for their conveying complex concepts and for always being enthusiasm and supportive.

I am indebted the most to my friends, Dr. Paolo Ventura, Dr. Carmela Sinisgalli, Dr. Elahe Pourabass, Valentina Fustaino and Meysam Salami, who have proved in many occasions how precious they are to me.

Furthermore, I gratefully acknowledge financial support from the Mixed Integer Nonlinear Optimization **(MINO)**, European- founded **Marie Sklodowska-Curie**, Initial Training Network, $\mathbf{7^{th}}$ Framework project which gave me the possibility of learning, doing researcher, participation at various conferences and workshops, as well as my research stay in Dortmund University in Dortmund, Germany and industrial cooperation with **MAJOR** company in Luca, Italy.

Last but certainly not least, my heartfelt thanks go Prof. Gianluigi Conte who always lend a sympathetic ear to my concerns and put large problems into perspective.

CONTENTS

Introduction

This doctoral thesis offers a contribution to the study of two combinatorial optimization problems, namely, finding the Clar number in Fullerene graph and Distributing the errors in iterative interior point methods.

The problem of computing the Clar number for Fullerene graph consists in finding the maximum number of saturated hexagons that are pairwise independent over all possible perfect matching of Fullerene graph.

It has been studied in the literature so far that there is a significant correlation between the Clar number and the stability of Fullerene molecule. A Fullerene molecule of n atoms has a multiplicity of isomers which increases as $\sim \mathcal{O}(n^9)$. Not all isomers of a given size have the same Clar number. Thereafter, since a high Clar number denotes a higher molecule stability, it is of great interest to identify Clar isomers, for each Fullerene molecule, which are the isomers for which the Clar number has the maximum possible value among all isomers.

The task of finding the Clar isomers of Fullerene graphs with sizes in a given interval can be accomplished by solving quite a large number of optimization problems. For example, for Fullerene molecule with 200 atoms even a solution algorithm that would take a second of computing time per isomer, would take almost seven years on a single processor to examine all the isomers. Therefore, it is necessary to design efficient solution algorithms for improving the methods proposed so far in the literature which is the purpose of the present doctoral thesis. To do so, we are interested in computing Clar number as (possibly $NP-$hard) combinatorial optimization problem and taking a theoretical approaches that exploits the theory of Polyhedral Combinatorics as opposed to approximate solution. To this purpose, we propose several integer linear programming formulations and we try to strengthen the already proposed integer linear formulation in Ahmadi et.al (2015) by finding sets of valid inequalities that can be added to the formulation either "statically", if their number is small, or "dynamically" by a suitable separation algorithm.
Each of our established formulations results in a solution algorithm that provides an exact value of Clar number. These algorithms are applied under the framework of the well-knowing branch-and-cut method. Branch-and-Cut method is a generic solution technique whose performance for a given type of optimization problem is mainly determined by two key elements: a close but manageable approximation of the associated polyhedron and secondly, efficient methods to solve the corresponding separation problem which is to decide for an arbitrary point in the ambient space whether or not it lies inside the polyhedron just mentioned.

In the second essay, we investigate a nonlinear optimization problem on solving convex quadratic programming problem by iterative primal-dual infeasible path following interior point algorithm. This part of the thesis is dedicated to analyzing possible iterative approaches for solving inexactly the linear system of Newton directions one has to face in each outer iteration of primal dual interior point algorithm. Indeed, in our study, we consider a generic Newton linear system with the assumption that each of the involved equation can be solved

inexactly. Thereafter, the idea is to transform this generic system of equations to various subsystems through the Gaussian elimination procedure and to solve the obtained linear subsystems by exploiting appropriate inexact linear solvers. Our final contribution will be the proposal of two innovative methods for redistributing the corresponding error generated by inexact linear solver such that the convergence of iterative primal-dual infeasible path following algorithm is guaranteed.

In this study, we try to explain and discuss all the required theoretical bases of the proposed approaches. Though, due to the limitation of time, our research could not go deeper. In particular, the fully presents of the solution method and computational results including implementing the proposed algorithms on minimum quadratic cost flow problem instances will be the aim of the future research.

It is worth mentioning that, the motivation behind this study was to fulfill the purpose of Mixed Integer Nonlinear Optimization **(MINO)**, European-founded **Marie Sklodowska Curie**, Initial Training Network, $\mathbf{7^{th}}$ Framework project. The internship opportunity I had with **MINO** project was a great chance for learning and researching development.

Part I

Computing the Clar Number in Fullerene Graphs

Chapter 1

Preliminaries of Combinatorial Optimization

This chapter introduces the terminology and basic concepts used in this doctoral thesis. We start with a few notations. An introduction into some definitions in the field of graph theory is given in Section 1.1. Section 1.2 deals generaly with the concepts of polyhedral theory. Thereafter, the rough ideas of Integer Programming (IP) are discussed in Section 1.3. In order to solve IP problems we introduce the Branch & Cut algorithm in Section 1.3.2. Section 1.4 and 1.5 are describing briefly the matching and stable set polytopes. For an extended exposition of these arguments. see, e.g., Cornforti, Cornujols & Zambelli[3], Grotschel, Lovsz & Schrijver [4].

1.1 Graphs

In this section we deal with several definitions and notations in the field of graph theory which are based on [5, 6].

An **undirected graph** G is a pair (V, E) consisting of a nonempty, finite set V of **nodes** and a finite (possibly empty) set E of **edges** which are unordered pairs of distinct nodes. Unless otherwise stated, the graphs in this thesis are generally assumed to be undirected.An edge $e = \{u, v\}$ is usually written as uv, which is equal to vu. In our context the edge set E contains no multiple edges. The **order** of a graph G is the **cardinality** of its node set V , which is the number of elements of V . It is denoted by $|G|$.

A graph is called **finite** if both V and E are finite. In the scope of this thesis we are just dealing with finite graphs.

A node v is incident to an edge e if $e = uv$. The two nodes incident to an edge are its **endnodes**. Two nodes u, v of a graph G are **adjacent** or **neighbors** if uv is an edge of G. For a node set $W \subseteq V$, we define the set of neighbors $\Gamma(W)$ as the set of all nodes in V W which are adjacent to at least one node in W and call it the **neighborhood**. For a single node we write $\Gamma(v)$ instead of $\Gamma(v)$.

The **degree** of a node v, denoted by $\delta(v)$, in a graph G is the number of edges incident with v, or equivalently the number of adjacent nodes to v, which is the cardinality of $\Gamma(v)$. A node is called **isolated** if its degree is zero.

1.1.1 Cuts and Degree

Let $U, W \subseteq V$ be two node sets. We call the set of edges with precisely one end in U a cut of G and denote it by $\delta(U)$. In this context, the set U and its complement $\overline{U} := V\ U$ are referred to as the **shores** of the cut. We write $\delta(v)$ instead of $\delta(v)$ for a node $v \in V$ and call $\delta(v)$ the **star** of v. We will also use the abbreviation $(U : W) := \delta(U) \cap \delta(W)$ for the set of edges with one end in U and the other end in W . The degree $deg(v)$ of a node v is the number of edges incident with v, $|\delta(v)|$.

A path is a nonempty graph $P = (V, E)$ with the node set

$$V = \{v_i | i = 0, \dots k\} \tag{1.1}$$

of a pairwise distinct nodes and the edge set

$$E = \{v_i v_{i+1} | i = 0, \dots, k1\} \tag{1.2}$$

$E = \{v_i v_{i+1} | i = 0, \dots, k1\}$. The nodes v_0 and v_k are linked by P and are called its **ends**. The remaining nodes $v_1, \dots, v_{k1}$ are called the inner nodes of P. A path is often referred to by the sequence of its nodes,i. e., $P = v_0, v_1 \dots v_k$, and is called a path from v_0 to v_k or simply a (v_0, v_k)-path. The number of edges of a path is its **length**.

If $P = v_0, \dots, v_{k1}$ is a path of length at least 2 then the graph $C := (V, E \cup v_{k-1}v_0)$ is called a **cycle**. As with paths, a cycle is often referred to by the (cyclic) sequence of its nodes, e. g., the above cycle C could be written as $v_0 \dots v_{k1}v_0$. The length of a cycle is the number of its edges (or nodes) and a cycle of length k is also called a k-cycle. A **chord** of a cycle C is an edge that joins two nodes of C which are not adjacent in the cycle.

Two nodes u, v in a graph G are connected if they are linked by a path in G. The graph G itself is connected if this is true for any two of its nodes. The distance between two nodes u and v in a graph G is the length of a shortest (u, v)-path in G; if no such path exists, we set the distance to infinity.

1.1.2 Sub(Sup)graphs and contractibility

Let $G = (V, E)$ and $G' = (V', E')$ be two graphs. If $V' \subseteq V$ and $E' \subseteq E$ then G' is a subgraph of G (G a supergraph of G'), written as $G' \subseteq G$.

If $G' \subseteq G$ and G' contains all the edges $uv \in E$ with $u, v \in V'$ then G' is an **induced subgraph** of G. We say that $V sp\prime$ **induces** or **spans** G' in G and write $G' = G[V']$. Thus, for any node set $U \subseteq V$, the (node-)induced subgraph $G[U]$ is the graph on U whose edges are exactly the edges of G with both ends in U . Finally, $G' \subseteq G$ is a **spanning subgraph** of G if V' spans all of G, i. e., if $V' = V$.

A graph is said to be **complete** if it contains an edge to each pair of its nodes. A complete graph with n nodes is denoted by K_n . A **clique** is the node set of a complete subgraph. If it has n nodes, it is called $n-$clique, whereas a 3-clique is called **triangle**. We call a graph G bipartite if its node set V can be partitioned into two disjoint sets V_1 , V_2 with $V = V_1 \cup V_2$ such that neither two nodes of set V_1 nor two nodes of set V_2 are neighbors. The node sets V_1 , V_2 are called **bipartition** of V .

We call the operation of identifying a pair of adjacent nodes while preserving all other adjacencies between nodes an elementary contraction, or simply a **contraction**. We assume that multiple edges arising from a contraction are replaced by single edges. A graph G is called **contractible** to another graph G' if G' can be obtained from Gb by a sequence of contractions.

1.1.3 Complement, isomorphism and subtraction

The **complement** $\overline{G}$ of a graph G is the graph with the same node set as G and the **complement edge set** E, containing only the edges which are not in E. For a node set $W \subseteq V$, the graph $G - W$ denotes the graph obtained by removing all nodes of W and all edges adjacent with at least one node of W. In the special case when W contains only one node v, we simply write Gv.

1.1.4 Selected Classes of Graphs

A graph $G = (V, E)$ is **bipartite** if its node set V can be partitioned into two nonempty subsets V_1 and V_2 such that each edge has one end in V_1 and the other end in V_2 . G is **complete bipartite** if each node in V_1 is joined to each node in V_2 . We denote the complete bipartite graph with $|V_1| = m$ and $|V_2| = n$ by $K_{m,n}$.

A graph is **planar** if it can be drawn in a plane without any edges crossing, i. e., if it has genus 0. We call a nonplanar graph G almost planar if it contains a node v such that G becomes planar by removing v and all its incident edges.

A graph is k**-regular** if all its nodes have the same degree k. A 3-regular graph is called **cubic**.

1.2 Polyhedra

For pursing this thesis some basic knowledge of polyhedral theory is essential. Here we give an overview of the relevant concepts of affine geometry and polyhedral theory. These definition are made based on the introductory chapter of [7].

Let $, x_1, \ldots, x_n \in R^d$ and let $\mathbf{1}$ denote the appropriately sized vector of all ones. We call the linear combination

$$\sum_{i=1}^{n} \lambda_i x_i \tag{1.3}$$

an **affine combination** if the scalar product $\mathbf{1}^T\lambda = \sum_{i=1}^{n} \lambda_i$ equals 1. Furthermore 1.3 is called a **conic combination** if $\lambda \geq 0$ and a **convex combination** if it is both conic and affine. The vectors $x_1, \ldots, x_n$ are affinely independent if (1.3) with $\mathbf{1}^T\lambda = 1$ can only have the value 0 when $\lambda = 0$. In other words, none of the vectors is an affine combination of the remaining ones.

These combinations are called **proper** if neither $\lambda = 0$ nor $\lambda = e_j$ for all $j \in \{1, 2, \ldots, k\}$ with $\lambda = (\lambda_1, \ldots, \lambda_n)^T$.

For any set $I \subseteq R^d$ the **affine rank** of I, denoted by arank(I), is the cardinality of the largest affinely independent subset of I. For any subset $I \subseteq R^d$ the **dimension** of I, denoted by $dim(I)$, is the cardinality of a largest affinely independent subset of I minus one, $dim(I) = rank(I) - 1$. A set $I \subseteq R^d$ with $dim(I) = d$ is called **full-dimensional**.

The set of convex combinations of some points is called **convex hull** and is written as $conv(x_1, \ldots, x_n)$.

1.2.1 Hyperplane, Halfspace, Polyhedra and polytopes

For $a \in R^d$ and $\alpha \in R$, we denote the **hyperplane** defined by the equation $a^T x = \alpha$ by

$$H(a, \alpha) := \{x \in R^d | a^T x = \alpha\}. \tag{1.4}$$

The corresponding inequality $a^T x \leq \alpha$ defines a **closed halfspace**

$$K(a, \alpha) := \{x \in R^d | a^T x \leq \alpha\}. \tag{1.5}$$

which is bounded by the hyperplane $H(a, \alpha)$. In other words, the bounding hyperplane is the collection of all points for which the inequality is **tight**, i. e., for which the inequality is satisfied with equality. For a given inequality $a^T x \leq \alpha$, we call $a^T x$ the **left hand side** and α the **right hand side**. Where appropriate, we will use the abbreviation (a, α) instead of $a^T x \leq \alpha$.

The intersection of a finite number of closed halfspaces is called a **polyhedron**. Every polyhedron is closed and convex. The convex hull of a nonempty finite set is called a **polytope**.

A polytope is a bounded polyhedron. Consequently, there are two ways of describing a polytope: as the convex hull of a nonempty finite set, also called **ν-representation**, or as the bounded intersection of finitely many closed halfspaces, also called **H-representation**.

Let P be a polyhedron in R^d and let $K := K(a, \alpha)$ be a closed halfspace in R^d with the corresponding bounding hyperplane H. We say that K and its defining inequality $a^T x \leq \alpha$, respectively, are valid for P if $P \subseteq K$. If, in addition, $P \cap H \neq \emptyset$, we call K and H a **supporting halfspace** and a **supporting hyperplane** of P , respectively. If a supporting hyperplane H of P does not contain the entire polyhedron P , we call it a **proper supporting hyperplane**.

A **face** F of a polyhedron P is the intersection of P and the **bounding hyperplane** of a valid halfspace $K(a, \alpha)$ of P , i. e.

$$F = P \cap H(a, \alpha). \tag{1.6}$$

We also say that the inequality $a^T x \leq \alpha$ defines the face F.

If $H(a, \alpha)$ is a proper supporting hyperplane of P , we call F **proper**.

If v is a point in a polyhedron P such that $\{v\}$ is a face of P , then v is called a **vertex** of P . A **facet** is a nonempty face of P with dimension $dim(P) - 1$ and its inequality is called **facet-defining** inequality for P . Let x be a vector satisfying (β, b_0). In this case the **slack** of (β, b_0) is defined as the difference between b_0 and $\beta^T x$.

The dimension $dim(F)$ of a face F of P is the dimension of its affine hull $aff(F)$. The faces of dimension $0, 1, dim(P) - 1$ are called the vertices, edges and facets of P, respectively

1.2.2 Polynomial-time algorithm

An **algorithm** is defined as a finite set of basic arithmetical and logica operations that, given as input the encoded instance of the problem, produces as output the answer or the solution of such problem.

A **polynomial-time** algorithm, or simply **polynomial** algorithm, is an algorithm that terminates after anumber of elementary steps that is bounded by a polynomial in the size of the input.

A problem is called **polynomial-time** solvable, or **polynomial**, if it can be solved by a polynomial algorithm.

1.2.3 Complexity of Optimization Problem

A **decision** problem is a problem having only two possible answer, either "yes" or "no". For collections of certain decision problems, there are two **Complexity** classes, $\mathcal{P}$ and $\mathcal{NP}$. The class $\mathcal{P}$ is the collection of all polynomial-time solvable decision problems. If each positive / negative answer of a decision problem can be checked in polynomial time then the decision problem is in the class $\mathcal{NP}$.

A decision problem Π_1 is a **polynomially reducible** to Π_2 if there exists a polynomial algorithm which, given any encoded instance π_1 of Π_1, produce in a polynomial time an encoded instance π_2 of Π_2 such that the answer to π_1 is "yes" if and only if π_2 has positive answer.

A decision problem Π in $\mathcal{NP}$ is called $\mathcal{NP}$-complete if each problem in $\mathcal{NP}$ is polynomially reducible to it. In the other word, the $\mathcal{NP}$-complete problems are the most difficult problems in $\mathcal{NP}$. An optimization problem is defined to be $\mathcal{NP}$-hard if there exists an NP-complete decision problem that is polynomially reducible to it.This means that all the decision problems in $\mathcal{NP}$ have to be polynomially reducible to it.

1.3 Integer and Binary Programming

We define the **linear integer programming** problem, or simply **integer program** (IP) as follows:

$$\begin{aligned} \max \quad & c^T x \\ s.t. \quad & Ax \leq b \\ & x \in \mathbb{Z}^n \end{aligned}$$

where $\mathbb{Z}^n$ denotes the set of n-dimensional integral vectors. One may notice that the IP feasible set $S := \{x \in \mathbb{Z}^n | Ax \leq b\}$ is not a polyhedron. We define the **associate polyhedron** to (IP) as the convex hull of the feasible set S. Furthermore, **binary program** is obtained by restricting the variables to be either 0 or 1.

$$\begin{aligned} \max \quad & c^T x \\ s.t. \quad & Ax \leq b \\ & x \in \{0,1\}^n \end{aligned}$$

1.3.1 Combinatorial Optimization

Combinatorial optimization problems are concerned with the optimization of an objective function over collections of subsets of a finite set. Let E be a finite set of n elements and let $c \in \mathbb{R}$ be the vector of the weight assigned to the elements of the E.

For an arbitrary subset $F \subset E$ its **aggregate weight** is defined as $c(F) = \sum_{j \in F} c_j$. Suppose we are given a collection $\mathcal{F}$ of subsets $F \subset E$, the corresponding **combinatorial optimization** problem is defined as follows:

$$\max_{F \in \mathcal{F}} \; c(F) \tag{1.7}$$

Moreover, the associated polytope P_F of the combinatorial optimization problem is defined as the the convex hull of the incidence vectors χ^F of all $F \in \mathcal{F}$, i. e.,

$$P_{\mathcal{F}} = conv\{\chi^F | F \in \mathbb{F}\}. \tag{1.8}$$

Now the combinatorial optimization problem can be reformulated as

$$\begin{aligned} \max \quad & c^T x \\ s.t. \quad & x \in P_{\mathcal{F}} \\ & x \in \{0,1\}^n \end{aligned}$$

There is no any efficient algorithm for solving the above optimization problem whose feasible set is only described by a $\mathcal{V}$-representation.However, theoretically the combinatorial optimization problem can be transformed into a binary program because every polytope also has an H-representation, there exists a **linear description** which is a finite set of inequalities $Ax \leq b$ such that $P_{\mathcal{F}} = \{x \in R^n | Ax \leq b\}$. There are techniques such as the **Fourier-Motzkin** elimination, to transform the different representations of the polytope $P_{\mathcal{F}}$ into one another. However, these techniques are only practical for very small problem instances. In general, an explicit linear description of a combinatorial optimization problem is either not completely known or it compromise too many inequalities. Yet, for finding an optimum solution, a complete linear description is not necessarily needed and determining the local facet structure of the near the optimal vertex is sufficient. This is the idea behind the **branch & cut** method.

1.3.2 Branch & Cut method

As we already now there are polynomial time algorithms for linear programs.The reason why we can solve LPs in polynomial time even they have an infinite number of solutions, is the concept that we only have to consider the vertices of its polytope. If we want to adopt this idea to Integer Program(IP), we have to find the convex hull of all feasible solutions of the IP, which is $P_I := conv\{x \in \mathrm{Z}^n | Ax \leq b\}$. If we can describe this polytope with a polynomially sized list of inequalities, we can compute the optimum in polynomial time. Unfortunately, in general the number of facets of a polytope P_I increases exponentially. But in order to compute the optimal solution of IP, there is no need to know all facets. To compute only such necessary facets is one principle idea of the Branch & Cut algorithm. The use of that method was first published by Grotscel, Junger and Reinelt [8] in 1984. Its name was introduced by Padberg and Rinaldi [9]. In order to solve the IP system, we omit the condition of the integrality of variables and obtain a so-called **LP-relaxation**. The LP-relaxation solution may be fractional, but it provides an upper bound (UP) of IP. By adding special inequalities to the system $Ax \leq b$, the LP-relaxation polytope gets more and more close to P_I.Here "special" means that, if x^* is a vector satisfying $Ax^* \leq b$, we want a valid inequality (β, b_0), a so-called **cutting-plane**, separating x^* from P_I. The task to decide if such an inequality exists and to compute one if possible, is called the **separation problem**.There exist an important theorem in this context which declars that under some technical conditions, the optimization problem can be solved in polynomial time, iff the separation problem is polynomial [4]. In order to solve an IP, one could generate such cutting-planes until the optimal solution is found. In practice, however, it turned out that a combination of cutting-planes with the **Branch & Bound** technique is more successful. Its conjunction is called **Branch & Cut**. After the initialization and preprocessing, this algorithm uses a tree to maintain the subproblems generated during the process. It begins the **bounding process** with its root. After solving the LP-relaxation, it is checked whether the

solution is LP feasible. In the case of feasibility the **Local Upper Bound** (LUB) is updated. If the LUB is smaller than the **Lower Bound**(LB), which is at the beginning minus infinity, the subproblem is **fathomed**. Otherwise, the algorithm checks if the solution is integer feasible. Clearly, if it is an integer solution, the best solution for this *subproblem* has been computed and this node of the tree can be fathomed. If improvement of the LP solution after adding new constraints is too small (**tailing off**) or the separation routine could not compute a new constraints, the algorithm **branches** in new subproblems, e.g. by setting a variable to an integer value. If the selection of a subproblem of the tree is successful, which means that the tree is not empty and that the UB is greater than the LB, the algorithm reenters the bounding phase. If the selected subproblem does not bring new information, we fathom it and select a new one. This process terminates when the tree is empty.

While the LP-relaxation of the initial system provides a **Global Upper Bound** (GUB), each subproblem computes a local upper bound which is only valid for all its sons in the tree. In some cases the problems are too difficult to solve them exactly and therefore it is a good idea to stop the algorithm after a certain number of iterations. The advantage of a Branch & Cut algorithm is that this algorithm provides next to a solution also a quality of it. This quality is measured by the **gap**, which is calculated by

$$\frac{GUB - LB}{GUB} \tag{1.9}$$

It turns out that it is not helpful to add more and more inequalities to the LP-relaxation. The inequality system Ax b may grow too big, which slows down the computation time. Therefore after each separation phase, some inequalities are eliminated and stored in a pool. This pool can also be checked once in a while, if it contains a violated constraint which is added again. This process is called **pool separation**.

1.4 Perfect Matching Problem

The *perfect matching* problem in a graph $G = (V, E)$ is to determine whether there exists a set of edges meeting each node exactly once. More precisely, finding a set M of edges such that

$$|M \in \delta(u)| = 1 \quad \text{for all } u \in V \tag{1.10}$$

If $c \in \mathbb{Z}^E$ is a vector of edge weights, then the *the perfect matching problem* is to find a perfect matching of G of maximum total weight $\sum_{e \in M} c_e$. There is a proof of Padberg and Rao [10] that maximum weight perfect matching can be computed in polynomial time.

Figure 1.1: The red edges define a perfect matching of the hexagons

With every perfect matching M, we can associate a characterize vector $\chi^M \in \mathbb{R}^E$ whose component has value 1 if $e \in M$ and 0 otherwise. Consider the polytope defined by the following inequalities:

$$\sum_{e \in (\delta(u))} x_e = 1 \quad \text{for all } u \in v \tag{1.11}$$

$$x \geq 0 \tag{1.12}$$

In this case, a vector $x \in \{0,1\}^E$, a characteristic vector of a perfect matching, is an extreme point of above polytope. If the graph is bipartite then a rich theory had already been developed which not only characterized those bipartite graphs which had perfect matchings [11], but showed that this problem could be formulated as a small linear program. However, the more general case of nonbipartite graphs, graphs that contain odd cardinality cycles, seemed different. A necessary condition was that the number of nodes had to be even, but that was far from sufficient.

Theorem 1. *(Hall's Bipartite matching Theorem). A bipartite graph $G = (V, E)$ has a perfect matching if and only if, for every $X \subseteq V$, the number of isolated nodes in $G - X$ is at most $|X|$.*

In 1947, Tutte [12] had generalized Hall's theorem to nonbipartite graphs. He proved that replacing "isolated nodes by "odd cardinality components yielded a characterization of which nonbipartite graphs have perfect matchings.

Theorem 2. *(Tutte 's matching Theorem). A (nonbipartite or bipartite) graph $G = (V, E)$ has a perfect matching if and only if, for every $X \subseteq V$, the number of odd cardinality components of $G - X$ is at most $|X|$.*

A well-known work of Edmonds [13] gives the minimal set of equations and inequalities which describes the perfect matching polytope as following :

$$x_e \geq 0 \quad \forall e \in E \tag{1.13}$$

$$x\left(\delta\left(v\right)\right) = 1 \quad \forall v \in V, \tag{1.14}$$

$$x\left(\delta\left(U\right)\right) \geq 1 \quad \forall U \subseteq V, \quad |U| \text{ is odd.} \tag{1.15}$$

where $\delta\left(U\right)$ denotes the set of edges that have exactly one endpoints in $U \subseteq V$. The last class of inequalities are called (*odd cut*) or (*blossom constraints*). The re-statement of these inequalities, derived by using the matching equality and considering the edges with both endpoints inside the odd set U, is as follows:

$$x(E(U)) = \sum_{e \in \gamma(U)} x_e \leq \frac{|U| - 1}{2} \tag{1.16}$$

where $\gamma(U)$ is the subset of edges with both endpoints inside the odd set $U \subseteq V$.

He also proved that the inequalities (1.13) - (1.15) describe a 0/1 polytope, means that the perfect matching problem can be solved by maximizing a linear function over the polytope defined by (1.13) - (1.15) . One may notice that the number of odd cut inequalities (1.15) or (1.16) grew exponentially with the size of G. Later on, in 1982, Padberg & Rao in [10] showed that the separation problem for the inequalities (1.15) can be solved in polynomial time. Let $x^* \in \mathbb{R}^E$ be the fractional solution satisfying system $(1.13), (1.14)$ that we want to separate from perfect matching polytope. Then, the separation problem is to find a subset $U \subseteq V$ of nodes such that

$$x\left(\delta\left(U\right)\right) < 1 \quad |U| \text{ is odd} \tag{1.17}$$

which is equivalent to solving a minimum cut problem on graph G whose weight of edges are the elements of x^*. To this purpose, there is no need to solve minimum $(s-t)$ cut problem for all the $\{s,t\}$ pairs of nodes of G, instead one may exploit the proposed algorithm by Gomory & Hu [14]. The algorithm proceed by construction of *Gomory-Hu tree*, obtained by solving a sequence of $|V|-1$ maximum $(s-t)$ flow problems on graphs derived from G by a sequence of contracting and uncontracting operations. Managing a data structure that efficiently performs all such operation is not trivial, therefore the implementation of Gomory-Hu algorithm results a not simple task to achieve.

The following lemma stated in [10] allowed Padberge & Rao to solve the minimum cut problem by just a little modification of the Gomory-Hu procedure.

Lemma 1. *Let $G = (V, E)$ and $c_e \geq 0$ for all $e \in E$ be given. Let $\delta(X)$ be a cut of G that is minimum with respect to the cost vector c. Then there exists an odd minimum cut $\delta(X_0)$ such that $X_0 \subseteq X$ or $X_0 \subseteq V \setminus X$ holds.*

1.5 Stable set problem

A stable set of a graph G is a set of nodes S with the property that the nodes of S are pairwise non adjacent. We abbreviate a stable set with S. Let G_c be a node-weighted graph. A stable set of G_c, which maximize $\sum_{v \in S} c(v)$, is called a **maximum set**. The size of a maximum-cardinality stable set is called the **stability set number** or **the stable set number** and indicated by $\alpha_{G(c)}$. In the case where all the elements of c is equal to 1, we neglect c and write more simply α_G. Let's define an incidence vector, χ^S, of a stable set S of graph G as a $|G|$ - dimensional vector with the following components

$$\chi_i{}^S := \begin{cases} 1, & \text{if} \quad v_i \in S \\ 0, & otherwise. \end{cases}$$

Definition 1. *The **stable set polytope** of a graph $G = (V, E)$ is the convex hull of the incidence vectors of all stable sets in G. It is denoted by*

$$P_{STAB} := conv\{\chi^S | S \subseteq V \quad \text{stable set}\} \tag{1.18}$$

The stable set polytope is a polytope because it is bounded by the n-dimensional unit cube. By choosing $S = V$, one may notice that unit vector is a stable set. Moreover, the zero vector is trivially also a stable set of the empty set and therefore, the stable set polytope is a full-dimensional polytope. This implies that all facets of P_{STAB} are inequalities and hence we do not have to consider equalities.

It has been proved that the maximum stable set corresponds to a vertex of P_{STAB}, let's denote this incidence vector χ^S by a binary variable x, therefore the stability number is $c^T x$. This inspire us for an IP formulation for obtaining the maximum stable set. To complete the IP formulation, we need inequalities which define P_{STAB} . Consider the following inequality system

$$x_i + x_j \leq 1 \quad \forall ij \in E \tag{1.19}$$

$$x_i \geq 0 \quad \forall i \in V, \tag{1.20}$$

The definition of an incidence vector of a stable set implies the so-called non-negativity inequalities (1.20). They are always facet-defining for P_{STAB} , since there are $n-1$ affinely independent solutions which have value zero in one entry. The inequalities (1.19) ensure that there cannot be a pair of adjacent nodes in one stable set which is a direct consequence of its definition. This type of constrains is called **edge inequality**. Hence, inequalities (1.20) and (1.19) are both valid for the stable set polytope. Inequalities (1.19) are not generally facet-defining. Inequalities (1.19) together with the non-negativity inequalities are referred to as **trivial inequalities**. Hence, maximum stable set problem can be formulated as a linear integer programming as follows:

$$\mathcal{IP}: \max \quad c^T x \tag{1.21}$$

$$s.t. \quad Ax \leq 1 \tag{1.22}$$

$$x \in \{0,1\}^{|V|} \tag{1.23}$$

where vertex x is the incidence vector of stable set and matrix A is a node-edge incident matrix for graph G.

It has been proved in [15] that determining a maximum stable set in an arbitrary graph is *NP-hard*. This is still true even when the weighting coefficient is equal to 1. Moreover, assuming that $P \neq NP$, there is no polynomial time algorithm for approximating the stability number within a factor of $|V|^{\epsilon}$ for fixed $\epsilon \geq 0$. Therefore, it is also *NP-* hard to find a solution of $\mathcal{IP}$. Nevertheless, the Branch and Cut algorithm is the approach that can be applied to solve the $\mathcal{IP}$ problem. To this purpose, we need to propose a linear programming relaxation and see how well it describes the stable set polytope. By looking carefully at the $\mathcal{IP}$, one may notice that the inequalities (1.23) are the reason why $\mathcal{IP}$ is challenging to solve. Hence, the easiest polynomial relaxation of $\mathcal{IP}$ to substitute the binary constraints (1.23) by (1.20).

$$\mathcal{LP}: \max \quad c^T x \tag{1.24}$$
$$s.t. \quad Ax \leq 1 \tag{1.25}$$
$$x \geq 0 \tag{1.26}$$

and its corresponding **stable set polytope relaxation** is described as follows:

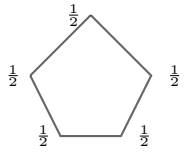
$$P_{RSTAB} := \{x \in \mathbb{R}^{|V|} | Ax \leq 1, x \geq 0\} \tag{1.27}$$

The following colloray by Balinski [16] indicates an important structure of the stable set polytope relaxation.

Collorary 1. *The vertices of P_{RSTAB} are $\left(0, \frac{1}{2}, 1\right)$-valued.*

Theorem 3. [4] *The trivial inequalities, the non-negative inequality* (1.20) *together with the edge inequality* (1.19) *are sufficient to describe P_{STAB} iff G is bipartite and has no isolated nodes.*

The minimal graph for which the trivial inequalities (1.20) and (1.19) are not defining the P_{STAB}, are the odd circuits. If G is an odd circuits then the point $\left(\frac{1}{2}, \frac{1}{2}, ..., \frac{1}{2}\right)^T \in \mathcal{R}^{|V|}$ satisfies all the inequalities (1.20) and (1.19) but it is not in P_{STAB}. This point is a vertex of P_{RSTAB} as well as the optimal values of the corresponding model in $\mathcal{LP}$-model.



Therefore each odd cycle C describes a new class of inequalities valid for P_{STAB}, so-called **odd cycle inequalities**:

$$\sum_{i \in V(C)} x_i \leq \frac{|V(C)| - 1}{2} \tag{1.28}$$

Let's define the **cycle-constraint** stayble set polytope as a polytope constructed by the inequalities (1.20) and (1.19) and (1.28)

$$P_{CSTAB} := \{x \in \mathbb{R}^{|V|} | x \quad satisfies\,(1.20), (1.19), (1.28)\} \tag{1.29}$$

t-perfect graphs, introduced by Chvátal in [17], are family of graphs for which the corresponding $P_{CSTAB}(G)$ has integral extreme points. Bipartite and almost bipartite are those among the t-perfect graphs. The problem of checking weather a graph is t-perfect or not belongs to *co-NP*. In 1986, Gerard and Schrijver [18] gave a polynomial time algorithm for separation problem of odd cycle inequalities; Given a fractional point x^*, to separate this point by

an odd cycle inequality one has to find a minimal weight odd cycle where the weight of each edge uv is given by $1 - x^*_u - x^*_v$. This problem is equivalent to find a shortest path in a bipartite graph $G^{'}$ obtained by making identical copies of vertices which are suitably connected. More precisely, for every arbitrary vertices u and v of G, the bipartite graph $G^{'}$ has vertices v, $v^{'}$, u, $u^{'}$ where v and u are in the one part and $^{'}v$ and $u^{'}$ are in the other one. The edges are defined as $vu^{'}$, $uv^{'}$. So, this graph has twice as many nodes and twice as many edges as the original graph, G. One can see easily that any path from v to $v^{'}$ in $G^{'}$ is corresponds to a union of cycles with odd number of total edges. Hence, one has to just look for the minimum odd cycle weather it has a length value exactly less than one (which amounts to running $|V|$ times a shortest path algorithm). In conclusion, we have the following theorem.

Theorem 4. [4] *Strong optimization problem for P_{CSTAB} can be solved in polynomial time.*

Collorary 2. *A maximum weight stable set problem for t-perfect graph can be solved in polynomial time.*

Among all the possible valid odd-cycle inequalities for P_{STAB}, we are mainly interested in those which are facet-defining because they are not dominated by any valid inequality of P_{STAB}. A cycle is called a **hole** if it is chordless. If an odd cycle induce an odd hole, then the corresponding odd-cycle inequality is called **odd-hole** inequality.

Collorary 3. [19] *Let G be an odd hole, then $\sum_{i \in V} x_i \leq \frac{|V|-1}{2}$ is facet-defining for $P_{STAB}(G)$.*

“ What is in this spaceship planet that needs doing that I know something about, that probably won't happen unless I take responsibility for it?”.

Buckminster Fuller

Chapter 2

Fullerene graph

2.1 Introduction

Buckminsterfullerene $\Gamma_{60} - I_h$ is a molecule made of 60 carbons that is convex and spherical with a highly symmetric icosahedral structure. This molecule was originally conjectured independently by Osawa in 1970 [20] and discovered in 1985 by Kroto et al. through laser evaporation of graphite [21, 22, 23, 24]. From the very beginning, the Fullerenes had attracted the attention of theoreticians because of their special structures and properties. The earliest theoretical work was performed in 1986 using semiempirical MNDO method [25]. Later on in 1998, a critical review was reported about the results obtained with different computational methods on the most classical Fullerenes, Γ_{60} and Γ_{70} [26].

Fullerenes are cage-like hollow carbon molecules of pseudospherical symmetry consisting of pentagons and hexagons. They are generally prepared by the vaporization of graphite in an electric arc in a low-pressure atmosphere.

It has been the object of interest for chemists and mathematicians due to its widespread application in various fields namely including electronic and optic engineering [27], medical science [28] and biotechnology [29, 30].

Fullerene derivatives is an attractive tool for biological applications. In [29], it has been explained how very appealing photo-, electro-chemical and physical properties of Fullerene family, and especially Γ_{60}, are exploited in many and different biological fields.

The interest in such a field is so intense that it has given rise to the journal *"Fullerene Science and Technology"* in 1993, currently known as "Fullerenes, Nanotubes and Carbon Nanostructures"[31], several books [32, 33, 34] and exponentially increasing number of papers [35].

2.1.1 Isomers of Fullerene

Fullerene graph, Γ_n made of n molecules, with all faces either pentagon or hexagon has the multiplicity of isomers. Soon after the discovery of Fullerene the simple Stone-Wales [36] transformation technique was proposed as a mechanism

of isomerization. Based on the distribution of the pentagonal faces in Fullerenes, Fullerene polyhedra can be arranged in multiple isomeric forms for all $n \geq 28$.

In [37] they consider a unified framework that allows construction and cataloging of the distinct topological possibilities for isomerization processes. There, all isomerization patches and isomerization pairs containing up to five pentagons and with an upper limit for the boundary length depending on the number of pentagons are listed and several infinite series of transformations are identified.

More recently, In 2012, the Buckygen program developed by Brinkmann et al. [38] constructs Fullerene isomers up to Γ_{400}. They also list special Fullerenes without ring spirals starting at a pentagon and Fullerenes without any ring spiral at all up to Γ_{400}.

These isomer comes in different many shapes, namely spherically or icosahedral, barrel, trigonal pyramidally (tetrahedral structure), trihedrally, nano-cone and cylindrically(nanotubes) shaped

The number of distinct isomers of a given dimension n, i.e., the number of pairwise non isomorphic Fullerene graphs of n nodes, grows with n quite rapidly.

The following table shows the number of isomers for each value of n in the range $[20, 204]$. Recall that no Fullerene graph exists for $n = 22$, for $n \leq 20$ and for n odd. All the values of columns "# of isomers" in Table 2.1.1, for $n \geq 60$, can be expressed by the function

$$f(n) = 1.7 \cdot 10^{-14} \cdot n^{9.59 \pm 0.03}.$$

This function gives a better idea about the growth rate of the number of isomers as n increases.

A Fullerene isomer belongs to the IPR (isolated pentagon rule) class if each pentagon joins only hexagons, and each hexagon alternatively join pentagons and hexagons. It has been known that the IPR property brings thermodynamic stability to a Fullerene cage [39]. Fullerenes with less than 60 carbon atoms cannot have an IPR structure, therefore $\Gamma_{60} - I_h$ is the smallest member in this class. Γ_{72} is the smallest Fullerene with an IPR cage besides the two classical Γ_{60} and Γ_{70}. It has been known that IPR Fullerenes yield the best stability and even focusing in this class, the number of candidate isomers for most thermodynamically stable isomers are huge.

2.1.2 Enumeration of Fullerene

Since the first Fullerene, the famous Γ_{60} was discovered, a perennial problem in the chemistry and physics of Fullerene is the question of how these organised cage structure emerges from chaotic carbon- vapour. In the transformation of Fullerene for construction of Fullerene road models, carbon atoms are added to or subtracted from Fullerene cages in multiples of two.

Experimental data and quantum mechanical calculation developed various mechanisms for carbon ingestion/ extrusion [40, 41] and isomerisation/annealing [36]. Other many attempts have been made to generate large lists of combinatorial Fullerenes [42, 43, 44]. The first tool was the spiral algorithm [42]. These

n	# isomers	n	# isomers	n	# isomers	n	# isomers
20	1	68	6 332	114	1 008 444	160	26 142 839
24	1	70	8 149	116	1 207 119	162	29 202 543
26	1	72	11 190	118	1 408 553	164	33 022 573
28	2	74	14 246	120	1 674 171	166	36 798 433
30	3	76	19 151	122	1 942 929	168	41 478 344
32	6	78	24 109	124	2 295 721	170	46 088 157
34	6	80	31 924	126	2 650 866	172	51 809 031
36	15	82	39 718	128	3 114 236	174	57 417 264
38	17	84	51 592	130	3 580 637	176	64 353 269
40	40	86	63 761	132	4 182 071	178	71 163 452
42	45	88	81 738	134	4 787 715	180	79 538 751
44	89	90	99 918	136	5 566 949	182	87 738 311
46	116	92	126 409	138	6 344 698	184	97 841 183
48	199	94	153 493	140	7 341 204	186	107 679 717
50	271	96	191 839	142	8 339 033	188	119 761 075
52	437	98	231 017	144	9 604 411	190	131 561 744
54	580	100	285 914	146	10 867 631	192	145 976 674
56	924	102	341 658	148	12 469 092	194	159 999 462
58	1 205	104	419 013	150	14 059 174	196	177 175 687
60	1 812	106	497 529	152	16 066 025	198	193 814 658
62	2 385	108	604 217	154	18 060 979	200	214 127 742
64	3 465	110	713 319	156	20 558 767	202	233 846 463
66	4 478	112	860 161	158	23 037 594	204	257 815 889

Table 2.1: Number of isomers as a function of n

proposed methods either could not guarantee that *every possible* structure is generated, so they are not complete or were not efficient enough to be able to generate lists of interesting size or both. A parallel line of research is the use of graph theoretical techniques to catalog the mathematically possible Fullerene structure. As an early mathematical approach the one in [45] towards enumerating three-regular spherical graphs, could not be used to obtain fast algorithms for this restricted class of structures. Therefore, base on an assumed set of rules for the construction and transformation of fulleren, in [34] and [46] two growth transformation technique were proposed. The one in [46] exploited a top-down approach, and it was considered as fast enough algorithm to generate, for example, all 1812 isomers of Γ_{60}in less than 20 seconds. Their suggested program is called *fullgen* which is the most successful practical method to date with proven completeness. It operates by stitching together 'patches' bounded by zigzag paths.

In [47] they present a carbon insertion/extrusion mechanisms for transforming one Fullerene to another by replacing a patch on the Fullerene surface.

They construct a systematic catalogue for the topologically distinct local insertion/extrusion transformation of Fullerene and they list all pairs of growing patches with the same boundary containing up to five pentagons.

This approach is extended in [48] by using a cut-and-paste approach for a family of structurally similar 'growth patches which are non-isomorphic patches with the same boundary but containing different numbers of vertices.

They explore the ways that Fullerenes can be formally generated from 'seed' polyhedra, using either a predefined set of graph transformations or a set that is restricted by a cost function intended to mimic the energetics of bond rearrangement and carbon insertion. There, it has been shown that a family of transformations based on the *Endo-Kroto* Γ_2 insertion mechanism [41] gives access to all isomers of all Fullerenes up to Γ_{200} from a Γ_{24} seed. This method fails in the general case.

This growth operation is called expansion. Fullerenes can be constructed from some simple set of starting isomers by successively applying expansion that replaces some fragment of Fullerene by a larger piece. It is shown in [49] that no finite set of expansion operation is sufficient, so instead in [50] they seek an easily-described infinite class of these operation that are probably complete.

The interested reader on the chemistry and physics of Fullerene can refer to the comprehensive reviewers and books on this subject highlighting many activities in years [51, 52, 53, 54, 55, 56, 57, 58].

The field of topological and graph theoretical descriptions of Fullerene has been received an intensive activity over the past 20 years [59, 34, 60, 61] to the extend that it has become a major sub-discipline within mathematical chemistry. One can mention here *An Atlas of Fullerenes* by Fowler and Manolopoulos [62] and *The Topology of Fullerenes* by Schwerdtfeger, Wirz and Avery [63] as good reviewers for topological and graph theoretical structures of Fullerenes.

2.2 Topological Properties of Fullerenes

Fullerene molecules can be represented as a 2-connected plane graph in which only carbon atoms are depicted as vertices while hydrogens are omitted. Edges are demonstrating chemical bonds between carbon atoms. The basic idea of considering a Fullerene graph is that many that physico-chemical properties of Fullerenes can be studied by using the information encoded in their corresponding graph [64]. Fullerene graph is both cubic, planar and three-connected whose faces are either pentagons or hexagons. Many properties about the Fullerene's topologies and indicators of their chemical behaviour can be derived directly from their graphs. For a three-connected planar graph, there is essentially a unique way to be embedded in the plane [65]. Due to this fact, the three-connected planar graph has a well-defined set of *faces*. Therefore a cubic three-connected Fullerene graph can be represented as $\Gamma = (\mathcal{V}, \mathcal{E}, \mathcal{F})$ where $\mathcal{V}$ defines a set of vertices, and $\mathcal{E}$ is a set of edges and $\mathcal{F}$ is a set of faces which is defined without ambiguity. Based on the *Euler's polyhedron formula* for the planar connected graph:

$$N - E + F = 2 \tag{2.1}$$

where $N = |\mathcal{V}|, E = |\mathcal{E}|$ and $F = |\mathcal{F}|$ being the number of vertices, edges and faces, respectively. According to the hand-shaking lemma in graph theory

$$\sum_{i=1}^{N} deg(v_i) = 2|E| \tag{2.2}$$

since fullernes are cubic so $deg(v_i) = 3$ for all vertices, therefore $N = \frac{2}{3}E$, and by replacing this in the equation (2.1) one may derive that

$$E = 3F - 6 \tag{2.3}$$

As we already mentioned the total number of faces is $F = F_5 + F_6$, where F_5 and F_6 are the number of pentagons and hexagons respectively. From (2.3) we derive

$$E = 3F_5 + 3F_6 - 6 \tag{2.4}$$

On the other hand, each pentagon(hexagon) has five(six) edges, which gives $E = \frac{5}{2}F_5 + 3F_6$. From this equation together with (2.4) one may obtain that $F_5 = 12$. This property is called *12 Pentagon Theorem* from which the number of hexagons are derived as $F_6 = (N - 20)/2$ with $N \geq 20$, and the general formula for Fullerenes can be describe as Γ_{20+2F_6}. In this formulation at least a Fullerene with every number of hexagons greater than two exists [66]. However $N = 22$ with just one hexagon and 12 pentagon is not a Fullerene though a cage-like fulleroid [67].

2.2.1 Dual of Fullerene

Generally talking, the dual of a planar graph might not be unique though in the case of three-connected graph such as Fullerene a dual is unique and the dual operation is well defined [65]. The dual graph of Fullerene is a triangulation graph with 12 vertices of degree 5 and the remaining of degree 6. In this thesis, Γ_n^* indicate the Fullerene dual of Γ_n. According to the fact that the dual is an involution operation, we have $\Gamma_n^* = \Gamma_n$, so one can think of the Fullerene dual as just another representation of the same graph. Figure (2.1) shows planer representation of Fullerene graph Γ_{60} and its corresponding Γ_{60}^*.

The Manolopoulos face spiral algorithm is one of the first methods for encoding Fullerene graphs which unwinds all the faces of Fullerene same as an orange peel [42]. More precisely, the algorithm starts with a sequence of three mutually adjacent faces and adds the new faces to the string such that the next face is adjacent to the previous one and the one that was added to the string earliest, and that has neighboring faces left which are not part of the spiral string (yet). As a result, the string length consists of 12 fives and $N/2 - 10$

Γ_{60} Γ^*_{60}

Figure 2.1: Planner embedding of Fullerene graph and dual: (a) $\Gamma_{60} - I_h$; (b) the dual of $\Gamma_{60} - I_h$ (pentakis-dodecahedron)

sixes, corresponding to the size of faces. There are $6N$ possible spiral starts resulting in up to $6N$ spirals per Fullerene graph. In most Fullerene, the number of spirals is less than $6N$ due to the fact that some of them cannot be completed since the two rules for selecting the next face to be added to the string can not be both fulfilled simultaneously. Since there is always 12 position in the spiral string for all Fullerene, it can be abbreviated as a list of the positions of the 12 pentagon, which is referred to as a Face Spiral Pentagon Indices ($FSPI$). The *canonical spiral* representation of Fullerene is the lexicographically smallest of all successful spirals. Although the initial conjecture was that all the Fullerene admits the face spiral, there is exceedingly small proportion of Fullerene which cannot be unwounded into spiral face.

One of the approaches for generating the Fullerene graph is by adding faces to an existing graph while considering the different position for this addition. [43] Brinkmann et al. [46] use patches,a subdivision of an existing Fullerene graph, rather than a single faces for generating Fullerene. Further development came in Brinkmann et al. [48] in which Γ_{20} is used as seed and all other Fullerenes up to Γ_{200} are generated from it. Their approach has a shortcoming that three graphs are not accessible by their proposed transformation and to overcome this approach, in Hasheminezhad et al. [50] a set of growth operation is defined in a way that starting from either Γ_{20} or Γ_{28} every Fullerene is generated in a systematic way. Furthermore, these set of growth operation is used in Brinkmann et al. [38] to develop a fast and complete algorithm that recursively generates all Fullerene isomerase up to a given maximal N. The method is called a patch replacement which is based on replacing a finite connected region inside a Fullerene with a larger patch with identical boundary. According to [68] without exploiting the patch replacement approach generating all Fullerene graph defeat to combinatorial explosion. The incredibly efficient generator based on the patch replacement is called *buckygen* and it has been used to generate databases up to Γ_{400}, this database are available at *House-of-Graphs* website [60].

A Fullerene graph can be transformed locally by replacing one Fullerene

patch [69] by either a different patch or the same patch in a different orientation [37, 47], or globally according to Goldberg-Coxeter transformation. The *boundary code* of patches is a sequence of free valencies of the vertices that lie on the boundary and two patches can be exchanged if they share the same boundary code. Isomerization operation is referred to replacing patches of the equal size while growing operation is referred to replaying patches by smaller or bigger one by insertion or deletion of vertices. Brinkmann et al. [37] generated a catalogue of isomerization with up to five pentagons. In this study, we are interested in the growing operation which is classified according to the number of the pentagons and the number of vertices which is added. The smallest example of growth operation which involves a patch with two pentagons and adds two vertices is called *Endo-Kroto transformation* [41]. In Brinkmann et al. an extensive list of growth pair can be found. The current fast Fullerene generator [38] is based on the three class of patch replacement defined by Hasheminezhad et al. [50].

Topological and Chemical Indicator Many useful properties of chemical system can be deduct by chemist just by considering its graph structure. For Fullerene, as an instance, the way on which the 12 pentagon are distributed on the surface indicates qualitatively the stability of the Fullerene or how it will be packed in the solid state. Moreover, by considering the symmetry of the underlying structure some spectroscopic properties of the Fullerene will be relived. *Topologicall indicator* is defined as a map τ_I from a Fullerene graph G into a finite series of number,

$$\tau_I : G \to \{x_1, \cdots x_n\} \tag{2.5}$$

where the number x_i is called *Topological Index* which might be integer, rational or real. More precisely, *Topological Index* connects graph theoretical properties of Fullerene directly to its physical properties. It is worth mentioning that there might not be a one-to-one corresponding between the topological indicators and the Fullerene graph and indeed most of the commonly used topological indicators are the same for many different isomers. Moreover, easily obtained *topological indices* is very useful since solving electronic structure problem for a large Fullerene is a daunting task. 12 face spiral pentagon indices is an example of *topological indices*.

Wiener in 1947 introduced the first topological indices with chemical concept called *Wiener Index* in which the paraffin boiling points is determined to exploit its graph structure [70] According to that the *Wiener Index* is defined as the sum of the entries in the *Topological distance matrix* which consists of the length of the shortest path between every pair of vertices in the chemical graph. The $WienerIndex$ gives an acceptable measurement of compactness acyclic alkanes and provides a reasonable correlation to boiling points [70]. For Fullerene, the lower the *Wiener Index* the higher is the compactness of the molecule. Since then, for the purpose of analysing the structural properties of the molecule,

many different topological indices have been introduced and studied yielding interesting mathematical properties. A number of very useful topological indices are arise from the distance matrix. To name a few, topological radius R and diameter D [71], Wiener Index, Hosoya Polynomial [72] and the Szeged Index [73] An interested reader is referred to Ref [74] for a list of *Topological Indices* and for more detailed discussion about *Topological Indices* is referred to [75] and [71].

2.3 Stability of Fullerene

Although there exists a number of heuristics that estimate the thermodynamic stability of a particular Fullerene, for example, by way of spectral analysis of its graph, it is not yet feasible to do so systematically for all isomers of large Fullerenes.

Kekulé structure and Perfect Matching *Kekulé (resonance) structure* is double bounds are drawn into the aromatic system [76]. Benzene has only two possible *Kekulé structure* while by increasing the size of Fullerene, one notice that Γ_{60} has as many as 12,500 *Kekulé structure* of which only 158 are non-isomorphic, [77], [78], [79]. A *Kekulé structure* is the same as *Perfect Matching* in the associated structural graph $G(\mathcal{V}, \mathcal{E}, \mathcal{F})$. In graph theory, a *Perfect Matching* is a subset of edges such that every vertex of the graph G is incident with exactly one edges from the subset. Over Fullerene, the edges of the perfect matching corresponds to the double bounds which are what referred to by the chemists as *Kekulé structure*. The existence of a perfect matching in Fullerene graph is the a corollary of a classical result of Petersen's Theorem [80] :

Theorem 5. *Every Connected cubic graph with no more than two cut-edges has a perfect matching.*

Before presenting a proof of this theorem a few remarks are in order. The original proof of the Peterson's Theorem was somewhat complicated and tedious, thereafter shorter proofs were given by Brahana(1917-18), by Frink(1925-26) and by Konig(1936), though with Tutte's Theorem at hand the shortest possible proof can be stated. Let $c_0(G)$ denote the number of odd components of a graph G.

Theorem 6. *(Tutte's Theorem). A graph G has a perfect matching if and only if $c_0(G - S) \leq |S|$, for all $S \subseteq V(G)$.*

The existence of the perfect matching can be characterized well exploiting the Tutte's Theorem and the important part of the theorem is the sufficiency which asserts that if G does not have a perfect matching, then there exist a subset of vertices S whose removal creates more than $|S|$ components with odd cardina The following is the proof of Peterson's Theorem based on the Tutte Theorem.

Proof. Let G be a connected cubic graph with no more than two cut-edges, it is trivial that $|V(G)|$ is even. Suppose indirectly that G has no perfect matching, according to Tutte's Theorem, G contains a cutset S with $c_0(G-S)$ odd components and $|S| \leq c_0(G-S)-2$. Since G is cubic each odd component is joined to S by some odd number of lines. If this odd number is one, the corresponding line must be cut-edge. By hypothesis there are at most two edge-cuts in G. Hence the odd components of $G-S$ send at least $3\left(c_0(G-S)-2\right)+2 = 3c_0(G-S)-4 \geq 3\left(|S|+2\right)-4 = 3|S|+2$ edges to S. On the other hand S sends at most $3|S|$ edges to the odd components and a contradiction results. □

The problem of counting the number of perfect matching in a given graph is a very difficult problem. Let's denote the set of all the *Kekulé structures* of the graph Γ by $\mathcal{K}(G)$ and $\phi(G) = |\mathcal{K}(G)|$, *Kekulé number*, as the cardinality of $\mathcal{K}(G)$. Not very sharp lower bound can be obtained just by taking into account the fact that Fullerene graph are 3-connected and using the method of *Cathedral Construction* ([12]).

Theorem 7. *Every Fullerene graph contains at least three different perfect matching.*

In ([81]) a much better lower bound is derived exploiting other structural properties of the Fullerene graph.

Theorem 8. *(Plesnik) [12] Let G be an r-regular with an even number of points which is $(r-1)-line-connected$. then if any $(r-1)$ lines are deleted from G, the resulting graph has a perfect matching.*

the proof details can be found in [12]

As a corollary of Plesnk's results, every edges of the Fullerene graph is contained in a perfect matching.

Definition 2. *Graph G is $1-$extendable if every edge of G appears in some perfect matching of G.*

Therefore Fullerene graphs are $1-$extendable. $1-$extendable graph deserve special attention because they permit *ear decomposition*. More details about *ear decomposition* can be found in [12].

The *Kekulé number* for several small Fullerene up to Γ_{84} is calculated and it has been shown that for Γ_{60} the least stable isomer has $K = 16,501$, while for its most stable isomer we have $K = 12,500$ [82].Moreover, it has been shown by Austin et al. that 20 isomers of the Γ_{60} have a higher *Kekulé number* than its most stable isomer [83]. As a conclusion, *Kekulé number* is not a good indicator for the stability of the graph. Cubic graphs has exponentially many perfect matching and hence exponentially many *Kekulé Structure* and the theoretical lower bounds is given in [83].

Chapter 3

Fries and Clar number of Fullerene graph

Let $\Gamma = (V, E, F)$ be a Fullerene graph and *Kekulé number*, $k = |\mathcal{K}|$ corresponds Kekulé structure $\mathcal{K} \subseteq E$ of Γ. Based on k, we have $|V| = 2k$, $|E| = 3k$ and $|F| = k + 2$.

For a given Kekulé structure $\mathcal{K} \subseteq E$ of Γ, each of its face might have 0,1,2 or 3 of its bounding edges in $\mathcal{K}$. Let's denote by $B_i(\mathcal{K})$ the set of faces that have exactly i of their bounding edges in $\mathcal{K}$. The *void faces* of $\mathcal{K}$ are those in B_0 and the faces in $B_3(\mathcal{K})$ are called the *full faces* or *benzenoid faces* of $\mathcal{K}$. The *Fries number* [84] or the *Kekulé parameter* of Γ, denoted by *Fris*(Γ), is defined to be the maximum of the number of benzenoid hexagonal faces overall Kekulé structures for Γ. The next lemma shows that *Fris*(Γ) $\leq \frac{2k}{3}$ and the *perfect* Kekulé structure is the one with $|B_3(\mathcal{K})| = \frac{2k}{3}$.

Lemma 2. *[85] Let $\mathcal{K} \subseteq E$ be a Kekulé structure for the Fullerene $\Gamma = (V, E, F)$ and, for $i = 0, 1, 2, 3$ let $B_i(\mathcal{K})$ denote the set of faces of Γ that have exactly i of their bounding edges in $\mathcal{K}$. Then:*

- $|B_3(\mathcal{K})| = \frac{2k}{3} - \frac{|B_1(\mathcal{K})|+2|B_2(\mathcal{K})|}{3}$;
- *Fris*(Γ) $\leq \frac{2k}{3}$

These banzenoid hexagons can have two isomers and Kekulé assumed that the three double bonds in these benzenoid hexagons are changing their place so fast that an isolation of these isomers would be impossible. According to Pauling assumption there could not exist two distinct isomers and the real state of benzene was the result of "resonance" between the two isomers.

Recent surveys on Fries number and face independent number of Fullerenes appear in Graver [85], Graver [85] and Vukicevic, et al. [86]

For a given perfect matching $M \subset E$ of Γ, a cycle C of Γ is M-alternating if the edges of C appear alternately in and off M. A set H of disjoint hexagons

of Γ is called a *resonant* set (or sextet pattern) if Γ has a perfect matching M such that all hexagons in H are M-alternating. A maximum sextet pattern is also called a *Clar formula*. The *Clar number* [87], $Clr(\Gamma)$, of Γ is the size of the largest resonance set of benzenoid faces overall Kekulé structures for Γ or equivalently, the largest independent set of benzenoid faces over all Kekulé structure for Γ.

The aromaticity denotes extra stability nature of a specific conjugated system and the resonance energy reflects the energy gain or loss due to the interaction between Kekulé structures. The resonance energy is calculated experimentally and represents the extra stability of the conjugated system. Distinct approaches have been developed to estimate the resonance energy [88] among them the probably most successful was the Clar's π-sextet theory stated in [87] In Clar's aromatic sextet theory, the aromatic π-sextext are defined as six π-electrons set in a single ring separated from adjacent rings by formal CC single bonds [89]. Moreover, Gutman in [90] stated: the Clar structure consisting of circles that satisfies the following three rules:

- Circles are never drawn in adjacent hexagons
- The remainder of the polyhex obtained by the deletion of the vertices of the hexagons that possess circles must be empty or have a Kekulé structure
- As many circles as possible are drawn subject to the constraints (a) and (b).

For building up his aromatic π-sexset theory, in [87] Clar provided many instances to support his observation that for isomeric benzenoid hydrocarbond when the number of the circles of Clar structures (Clar number) increases, the absorption bands shift to the shorter wavelength and the stability of the isomers also increases.

The Clar number of a benzenoid hydrocarbon was computed by Hansen and Zheng [91] by the integer linear programming.They conjectured that the linear programming relaxation of this model was sufficient for the general case. Later, this conjecture has been proved by Abeledo and Atkinson in [92].

The Clar and the Fries number for benzenoid hydrocarbons are good indicators for their stability. However for the Fullerene graph in Austin et al. [83] 20 distinct isomer of C_{60} was constructed whose Fries number suppress the on of icosahedral C_{60}. This leads to the fact that the maximality of Fries number may not indicate the higher stability of the molecule. On the other hand, in Zhang et al. [93] a significant correlation between Clar number and the stability of Fullerene have been studied.

One might naively assume that there is a Kekulé structure that simultaneously corresponds to the Fries and Clar numbers or in another word the Clar structure is the subset of the Fries structure. In [94] it has been discussed that this is generally not the case. Abeledo and Atkinson proved that *Clar number* of a 2-connected bipartite plane graph can be computed in polynomial time [92]. Note that a plane graph is bipartite if and only if all of its faces are even. On

the other hand, E.R Brczi-Kovcs and A.Bernth in [95] showed that determining the Clar number of a general 2-connected plane graph is NP-hard. Their aim was to determine the Clar number of fulllerene graph in polynomial time. They showed that determining the Clar number of a general 2-connected plane graph is NP-hard, if the number of odd faces is not bounded in the planar embedding. They present an algorithm that determines the Clar number of a 2-connected plane graph and has good running time unless that the odd faces are too far from each other in the planar representation. The problem weather computing a *Clar number* for Fullerene graph is NP-hard is however left open, since their NP-hardness reduction involves creating a lot of odd faces.

However, finding a Clar number for the planar graph is computationally NP-hard problem [96]. It is known that the Fries number of Fullerene is bounded above by $\frac{|V|}{3}$ and the Fries structures with $\frac{|V|}{3}$ numbers are referred to as *complete Fries structures* or *perfect Clar structures*. This happens only when the Clar and the Fries structures are the same. On the other hand, the corresponding bound for the Clar number is $\frac{|V|}{6} - 2$ and the Fullerene for which this bound is attained is characterized by Zhan and Ye in [97, 98] where they are referred to as *extremal Fullerene*.

Theorem 9. *[97] Let Γ_n be a Fullerene graph with n vertices. Then $Clr(\Gamma_n) \leq \lfloor \frac{n-12}{6} \rfloor$.*

They also showed that there are infinite many *extermal Fullerene*.

It might be worth mentioning that in contrast to various topological indices which are usually invariant under the automorphism operation, Fries number of Fullerenes depends strongly on the Fullerenes isomer, and different isomers of a certain Fullerene may have different Fries number [85].

The idea of expressing a mathematical model for the topological index goes back primarily to the work done by P. Hansen and M. Zheng [91]. They proposed an integer programming model for determining the Clar number of benzenoid hydrocarbon. Later on and more recently L. Pavlovic and T. Divnic introduced a quadratic programming approach for computing the Randic index [99].

In [100] we introduced a binary integer linear programming model for finding the Fries number of a Fullerene and here in this thesis, we improve solving approach for the proposed binary programming models such that their optimal solution specify the location of the double bounds in the associated Kekulé structure and its optimum value denote the Fries and Clar number for all Fullerenes. As a common technique, in this study, we applied the *branch-and-cut* procedure for reaching the optimal solution of the prescribed binary integer programming models.

3.1 Integer Linear Programming for Fries Number

Let $\Gamma = (V, E, F)$ be a Fullerene graph and $M \subset E$ be a perfect matching of Γ. Let $\mathcal{H}(\Gamma)$ the set of all *hexagons* of Γ, i.e., the set of all induced sub-graphs of Γ whose edge sets are 6-cycles of the graph. The set $\mathcal{H}(\Gamma)$ has $n/2 - 10$ elements, where n is the size of V.

In [100] the following integer programming formulation is used to compute Fries number:

ILP_{Fris} :

$$\max \quad \sum_{H \in \mathcal{H}(\Gamma)} y_H \tag{3.1a}$$

$$\text{s.t.} \quad x(\delta(v)) = 1 \qquad v \in V, \tag{3.1b}$$

$$x(E(H)) \geq 3\, y_H \quad H \in \mathcal{H}(\Gamma), \tag{3.1c}$$

$$0 \leq x_e \leq 1 \qquad e \in E, \tag{3.1d}$$

$$0 \leq y_H \leq 1 \qquad H \in \mathcal{H}(\Gamma), \tag{3.1e}$$

$$x \in \mathbb{Z}^E, \tag{3.1f}$$

$$y \in \mathbb{Z}^{\mathcal{H}(\Gamma)}. \tag{3.1g}$$

A variable x_e is associated with every edge e of Γ and $\delta(v)$ denotes the set of edges that have exactly one endpoints in $\{v\}$. Let's us call the polytope described by the above set of constrains as *Fries polytope*, $P_{Fries}(\Gamma)$.

Every vector $\bar{x}$ that satisfies the constraints (3.1b), (3.1d), and (3.1f) is the incidence vector of a perfect matching M of Γ, i.e., a vector of $\{0,1\}^E$ where, for every $e \in E$, the component $\bar{x}_e$ is equal to 1 if $e \in M$ and is equal to 0, otherwise. The reverse also holds. To count the number of saturated hexagons of a given perfect matching M the variables y_H come into play; one for each $H \in \mathcal{H}(\Gamma)$. If an hexagon H is saturated and $\bar{x}$ is the incident vector of M, then $\bar{x}(E(H)) = 3$; therefore both values 0 and 1 for variable y_H make constraint (3.1c) satisfied, and the direction of the objective function "pushes" its value to 1. If, to the contrary, H is not saturated, then $\bar{x}(E(H)) \leq 2$, and the integrality constraint (3.1g) along with (3.1c) force y_H to take the value zero. Let us refer to the set of inequalities (3.1c) as *Kekulé* inequalities.

In conclusion we can assert that the integer linear program (3.1a)-(3.1g) correctly computes the Fries number of the graph Γ.

Linear Programming Relaxation Formulation for Fries Number

Generally speaking, solving binary problems for large-scale instances are computationally complicated procedure. Therefore, as a first approach for solving (ILP_{Fris}), we start by considering its underlying linear relaxation problem. To this purpose, we solved the linear relaxation models for a variety instance of Fullerene graphs and our computational experiments clearly reveal that solving

the linear relaxation models do not always yield an *integer* optimum solutions; so, it can be stated generally that the underlying coefficient matrix is not always unimodular.

Quadratic Programming for Fries number

In this section, we show that a simple polynomial transformation allows us to reformulate the linear binary integer mathematical model (ILP_{Fris}), as a quadratic program. The idea behind it is that, the integrality constraints are enforced by the complementary constraints. To this purpose, let's start by replacing the integrality conditions $x \in \mathbb{Z}^E$ and $y \in \mathbb{Z}^{\mathcal{H}}$ with the following ones:

$$x_e\,(1 - x_e) = 0, \quad e \in E \tag{3.2}$$
$$y_H\,(1 - y_H) = 0, \quad H \in \mathcal{H}(\Gamma) \tag{3.3}$$

Let's call these set of equations, the complementary constraints. Substituting them in the programming problem (ILP_{Fris}) results in obtaining the following equivalent nonlinear version of the problem:

QP_{Fris} :

$$\max \quad \sum_{H \in \mathcal{H}(\Gamma)} y_H \tag{3.4a}$$
$$\text{s.t.} \quad x(\delta(v)) = 1 \qquad v \in V, \tag{3.4b}$$
$$x(E(H)) \geq 3\,y_H \quad H \in \mathcal{H}(\Gamma), \tag{3.4c}$$
$$x_e(1 - x_e) = 0 \qquad e \in E, \tag{3.4d}$$
$$y_H(1 - y_H) = 0 \quad H \in \mathcal{H}(\Gamma), \tag{3.4e}$$
$$0 \leq x_e \leq 1 \qquad e \in E, \tag{3.4f}$$
$$0 \leq y_H \leq 1 \qquad H \in \mathcal{H}(\Gamma), \tag{3.4g}$$

Furthermore, the complementary constraints $x_e\,(1 - x_e) = 0$ and $y_H\,(1 - y_H) = 0$ can be enforced by incorporating an exact penalty within the objective function, i.e., there exists a threshold value M^* such that whenever the scalar M exceeds M^* the solution of the following problem satisfies the complementary constraints and so on the integrality conditions of the involved variables are insured.

$$\max \quad \sum_{H \in \mathcal{H}(\Gamma)} y_H - M \sum_{H \in \mathcal{H}(\Gamma)} y_H(1 - y_H) - M \sum_{e \in E} x_e(1 - x_e) \tag{3.5a}$$
$$\text{s.t.} \quad x(\delta(v)) = 1 \qquad v \in V, \tag{3.5b}$$
$$x(E(H)) \geq 3\,y_H \quad H \in \mathcal{H}(\Gamma), \tag{3.5c}$$
$$0 \leq x_e \leq 1 \qquad e \in E, \tag{3.5d}$$
$$0 \leq y_H \leq 1 \qquad H \in \mathcal{H}(\Gamma), \tag{3.5e}$$

Over the past decades, Interior point methods (IPMs) have been proved practically efficient in solving linear and nonlinear optimization problems including quadratic programming problems [101]. IPMs have better complexity bounds than other available solving approaches for quadratic programming problem (the best iteration complexity bound found for accuracy $\epsilon > 0$ is $O(\sqrt{n}log\frac{1}{\epsilon})$ [101]. For more details of IPM appraoch the interested reader is referred to Byrd et al. [102], Byrd et al. [103], Waltz et al [104]. Experimental results of solving *concave* quadratic model for set of fullenere instances, under the frame work of interior point method, has been reported in **Table 1** in [100]. Note that different isomers of a certain Fullerene, due to their various structures, might have different Fries numbers. There, for each Fullerene the first isomer according to the is taken into account. Furthermore, as mentioned previously, there might be more than one Kekulé structure associated to Fries number of a specific Fullerenes isomer, known as alternative optimum solutions of the problem.

3.1.1 Integer Linear Programming for Clar number

Two hexagons of Γ are called *independent* if they do not have any nodes of edges in common. Observe that it would be sufficient to say that two hexagons are independent if they do not share a node because if two hexagons sharing a node v had no edges in common, then v would have four edges incident to it contradicting the assumption that Γ is a Fullerene graph, where every node has degree three. We call a *Clar system* associated with a perfect matching M of Γ a maximal set on saturated hexagons that are also pairwise independent. The cardinality of such a Clar system is denoted by $C_M(\Gamma)$. The *Clar number* $Clr(\Gamma)$ of Γ is the the maximum value $C_M(\Gamma)$ over the set $\mathcal{M}(\Gamma)$ of all perfect matchings of Γ, i.e.,

$$Clr(\Gamma) = \max\{C_M(\Gamma) \mid M \in \mathcal{M}(\Gamma)\}.$$

Developing on the model (3.1a)-(3.1g), a computation of the Clar number through integer linear programming techniques is proposed in [105], where the constraints

$$y_H + y_L \leq 1 \qquad \begin{array}{l} \textit{for all distinct} \quad H, L \in \mathcal{H}(\Gamma) \\ \text{with } V(H) \cap V(L) \neq \emptyset \end{array} \tag{3.6}$$

are simply added to the program (3.1a)-(3.1g). It is easy to see that in any solution to (3.1a)-(3.1g), (3.6) the y vector is the incidence vector of a subset of $\mathcal{H}(\Gamma)$ that qualifies as a *Clar system.*

Let's refer to the system described by the set of inequalities (3.1a)-(3.1g) together with (3.6) as ILP_{Clr} and the corresponding polytope as *Clar Polytope*, $P_{Clar}(\Gamma)$.

Not all isomers of a given size have the same Clar number. Therefore, since a high Clar number denotes a higher molecule stability, it is of great interest to identify, for each value of n within a reasonable range, which are the isomers for

which the Clar number has the maximum possible value $\bar{C}_n$ among all isomers of size n. We call these isomers the *Clar isomers* and we denote them by $\mathcal{I}_n$.

This task of finding the Clar isomers of Fullerene graphs with sizes in a given interval can be accomplished by solving quite a large number of optimization problems of the type described in the previous section. For example, for $n = 200$, even a solution algorithm that would take a second of computing time per isomer, which is a time in the range of those reported in [105], would take almost seven years on a single processor to examine all the isomers.

Therefore, it is necessary to design efficient solution algorithms improving on the methods proposed so far in the literature.

3.2 Strengthening the Clar number ILP formulation

3.2.1 Hex Inequality

Suppose an arbitrary hexagon $H \in \mathcal{H}(\Gamma)$ with its corresponding face variable y_H, let us indicate by $E(H) := \{1, 2, \cdots, 6\}$ the 6 edge cycle of H and by $\delta(V(H)) := \{a, b, c, d, e, f\}$ the set of 6 edges the have only one endpoint in H, as shown in Figure 3.1.

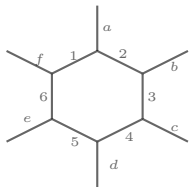


Figure 3.1: Hexagon H

Then, the *Kekulé* inequality for H can be written as:

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 - 3y_H \geq 0 \tag{3.7}$$

Consider the following valid solution for (3.7),

$$(x_1, x_2, x_3, x_4, x_5, x_6, y_H) = (1, 0, 1, 0, 0, 0) \tag{3.8}$$

The matching constraints (3.1b) imply that

$$(x_a, x_b, x_c, x_d, x_e, x_f) = (0, 0, 0, 1, 1, 0) \tag{3.9}$$

One may notice that this feasible solution is not among the interested ones since x_i are taking integral values and in the contrary, y_H has a fractional value. This fact motivates us to look for a set of tighter inequalities such that while *dominating* the inequalities (3.7), it also chops off such possible solutions as described previously from $P_{Fries}(\Gamma)$ and $P_{Clar}(\Gamma)$ polytopes. To this purpose, by matching constraint (3.1b) we have:

$$x_1 + x_2 = 1 - x_a, \tag{3.10}$$
$$x_3 + x_4 = 1 - x_c, \tag{3.11}$$
$$x_5 + x_6 = 1 - x_e. \tag{3.12}$$

Then, substituting set of equations (3.10) - (3.12) into the *Kekulé* inequality (3.7) results in:

$$1 - x_a + 1 - x_c + 1 - x_e \geq 3y_H \tag{3.13}$$

which can be split into the sum of the following three inequalities:

$$1 - x_a \geq y_H, \tag{3.14}$$
$$1 - x_c \geq y_H, \tag{3.15}$$
$$1 - x_e \geq y_H. \tag{3.16}$$

The set of inequalities (3.14) - (3.16) are valid for a given solution (3.8) and (3.9), nevertheless $1 - x_e \geq y_H$ is violated by (3.8) and (3.9) which means that (3.14) - (3.16) chop off such an undesirable feasible solutions from the Fries and Clar polytopes.

On the other hand, by considering the other three possible matching constraints:

$$x_1 + x_6 = 1 - x_f, \tag{3.17}$$
$$x_2 + x_3 = 1 - x_b, \tag{3.18}$$
$$x_4 + x_5 = 1 - x_d. \tag{3.19}$$

Following are the other three valid inequalities that (3.7) can be obtained by their positive combination:

$$1 - x_b \geq y_H, \tag{3.20}$$
$$1 - x_d \geq y_H, \tag{3.21}$$
$$1 - x_f \geq y_H. \tag{3.22}$$

Let's refer to inequalities of type (3.14) - (3.16) and (3.20) - (3.22) as *Hex* inequalities. It can be concluded that replacing (3.1c) by *Hex* inequalities in ILP formulations of Fries and Clar numbers result in more accurate description of $P_{Fries}(\Gamma)$ and $P_{Clar}(\Gamma)$, respectively.

ILP^{H}_{Clr} :

$$\max \quad \sum_{H\in\mathcal{H}(\Gamma)} y_H = (1^{\mathcal{H}(\Gamma)})^T\, y \tag{3.23a}$$

$$\text{s.t.} \quad x(\delta(v)) = 1 \qquad v \in V, \tag{3.23b}$$

$$x_e + y_H \le 1 \qquad H \in \mathcal{H}(\Gamma),\ e \in \delta(V(H)) \tag{3.23c}$$

$$y_H + y_L \le 1 \qquad \textit{for all distinct } H, L \in \mathcal{H}(\Gamma) \textit{ with, } \mathrm{V}\,(H) \cap V(L) \neq \emptyset \tag{3.23d}$$

$$x_e \ge 0 \qquad e \in E, \tag{3.23e}$$

$$x_e \le 1 \qquad e \in E \setminus \bigcup_{H\in\mathcal{H}(\Gamma)} \delta(V(H)), \tag{3.23f}$$

$$x \in \mathbb{Z}^E, \tag{3.23g}$$

$$y \in \mathbb{Z}^{\mathcal{H}(\Gamma)}. \tag{3.23h}$$

where for $U \subseteq V$, by $\delta(U) \subseteq E$ we denote the set of edges that have one endpoint in U and the other in $V \setminus U$. Thus $\delta(V(H))$ is the set of 6 edges the have only one endpoint in H. Observe that, because of the constraints (3.23c), the upper bound on the variables y is redundant while for the variable x the upper bound has to be stated explicitly only for those components that appear in none of the constraints (3.23c).

3.2.2 The Blossom Inequality

A well-known work of Edmonds [13] gives the minimal set of (in)equalities which describes the *perfect matching* polytope as following:

$$x_e \ge 0 \quad \textit{for all } e \in E, \tag{3.24}$$

$$x\left(\delta\left(v\right)\right) = 1 \quad \textit{for all } v \in V, \tag{3.25}$$

$$x\left(\delta\left(U\right)\right) \ge 1 \quad \textit{for all } U \subseteq V, \quad |U| \textit{ is odd.} \tag{3.26}$$

where $\delta\left(U\right)$ denotes the set of edges that have exactly one endpoints in $U \subseteq V$. The last set of t inequalities (3.26) are (*blossom inequalities*) and it is proved to be one of the effective cutting plane for both Fries and Clar polytopes. One may observe obviously that:

$$x\left(\delta\left(U\right)\right) = \sum_{v\in U} x_{(\delta(U))} - 2\sum_{e\in\gamma(U)} x_e \tag{3.27}$$

where $\gamma(U)$ is the subset of edges with both endpoints inside the odd set $U \subseteq V$. Matching constraint implies that

$$\sum_{v\in U} x_{(\delta(U))} \le 1 \tag{3.28}$$

Therefor, an equivalent re-statment of *blossom* inequalities are derived easily as:

$$\sum_{e\in\gamma(U)} x_e \leq \frac{|U|-1}{2} \quad \textit{for all } U \subseteq V, \quad |U| \textit{ is odd} \tag{3.29}$$

Here we are going to consider the necessary and sufficient conditions under which *blossom* inequalities are facet inducing cutting planes for general matching polytope.

Theorem 10. *Let $G = (V, E)$ be the undirected graph and $U \subseteq V$ be any odd subset of nodes of G, $|U|$ is odd. If $G[U]$ which is the subgraph of G induced by U, is not connected, then the corresponding blossom is not facet inducing inequality.*

Proof. Since $G[U]$ is not connected, there exist $\overline{U} \subseteq S$ with an odd cardinality such that $\overline{U} \cup \big(U \setminus \overline{U}\big) = U$ and $U \setminus \overline{U}$ is an even node set. The *blossom* inequality for the odd subset $\overline{U}$ says that:

$$\sum_{e\in\gamma(\overline{U})} x_e \leq \frac{|\overline{U}|-1}{2} \tag{3.30}$$

and the following valid inequality for the even subset $U \setminus \overline{U}$:

$$\sum_{e\in\gamma(U\setminus\overline{U})} x_e \leq \frac{|U \setminus \overline{U}|}{2} \tag{3.31}$$

Summing up the two inequalities (3.30) and (3.31) results a corresponding blossom inequality for subset U. This indicates the fact that it can not be a facet inducing one. □

3.2.3 The Odd Hole Inequality

A *hole* of a graph G is an induced subgraph of G whose edges form a cycle that does not have chords. Let $\mathcal{C}_G$ be the family of all holes of G with an odd number of edges. Then the family of the *odd hole inequalities* is the following linear system:

$$\sum_{v\in V(C)} x_v \leq \frac{|V(C)|-1}{2}, \quad C \in \mathcal{C}_G. \tag{3.32}$$

Let the stable set polytope $P^{stab}(\Gamma)$ of Fullerene graph Γ to be defined as the convex hull of all characteristic vectors of the stable sets of Γ.

Optimizing a linear function over $P^{stab}(\Gamma)$ is an NP-hard problem, [15]. Therefore, a complete external description of $P^{stab}(\Gamma)$ may not be obtainable generally speaking. However several other families of valid inequalities are known for this polytope such as odd hole inequalities. Since the number of odd cycle grow up exponentially in the dual graph of Fullerene, they are exploited as cutting planes.

3.2.4 The Clique Inequality

Based on the stable set definition, *Clique* inequality are valid for the stable set polytope:

$$\sum_{v_i \in K} v_i \leq 1 \quad \textit{for all clique } K \subseteq V \tag{3.33}$$

Considering Clar polytope P_{Clar}, since the Fullerene dual is planar there is no clique of size more than three in the Fullerene dula. This suggest that the maximum cliques are triangles. Fig. (3.2) provides a subraph H_3 of three adjacent hexagons namely h_1, h_2 and h_3 of an arbitrary Fullerene graph Γ_n.

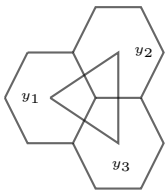


Figure 3.2: Triangle Inequality

their associated variables y_1, y_2 and y_3 makes a triangle in underlying the Fullerene dual Γ_n^*. This suggest the following *clique* inequality for Clar polytope P_{Clar}:

$$y_1 + y_2 + y_3 \leq 1 \quad \textit{for all triangles in } \Gamma_n^* \tag{3.34}$$

Let's refer to this set of inequality as *Triangle* ones.

Theorem 11. *Let G be a graph with node set (X, Y) and $K \subseteq Y$. Inequalities (3.33) is valid for $P_{Clar}(X)$.*

Proof. Let K be a clique in $G = (V, E)$, $K \subseteq Y$. Since there is an edge $y_i y_j \in H$ for all the nodes $y_i, y_j \in K$, a stable set of G can only contain one node of K which implies that inequality $\sum_{h_i \in K} y_i \leq 1$ holds for all $(x, y) \in P_{Clar}(X)$.

□

Sine there is a y variable associated to each hexagonal face, by just looking at the dual graph of the Fullerene, which is planar and cubic, therefore, there is no subgraph of K_4 in the dual graph of Fullerene. So one might conclude that the maximal clique set of y variables are the triangle.

Another proof approach will be by looking at the primal graph G, suppose the three hexagons adjacent pairwise as shown in (3.3), for the fourth hexagon to be adjacent to the current three ones, it will require a common edge with each of them. The adjacency with two of them can be built easily through the third common edge is only possible by making a tube which is contrary to the structure of Fullerene graphs.

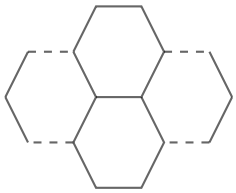


Figure 3.3: Maximum Clique in Fullerene

The problem of finding three set of pairwise adjacent hexagons in Fullerene graph Γ_n or equivalently a 3-clique (triangle) in dual Fullerene Γ_n^* is a polynomial time as it is a problem of finding a maximum clique in the dual Fullerene.

3.2.5 The Earing Inequality

Suppose (P, H) be a pair of adjacent *through one edge* pentagon and hexagon, $|P \cap H| = 1$ then the inequality

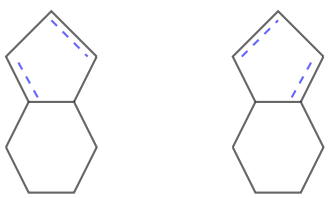
$$x_{(P/P\cap H)} + y_H \leq 2 \tag{3.35}$$

is called *Earring Inequalities*.

Theorem 12. *The earring inequality is valid for the Fries polytope.*

Proof Consider 3.4. For any arbitrary matching, red lines indicate the matched edges in hexagons and dashed blue lines are matched ones in pentagons.for any arbitrary matching, this figure consider all possible cases. More precisely, in the case when $y_H = 0$ at most two of the edges of the pentagons can be matched in which neither edges of the adjacent hexagon is in the matching, $y_H^1 = 0$ and $y_H^2 = 0$.

In the case where $y_H = 1$, there are exactly two ways of matching edges in the corresponding hexagon indicated in figures $y_H^1 = 1, \cdots, y_H^4 = 1$ in which as a consequence, the number of the matched edges in the adjacent pentagon can not exceed one.



$y_H^1 = 0$ $y_H^2 = 0$

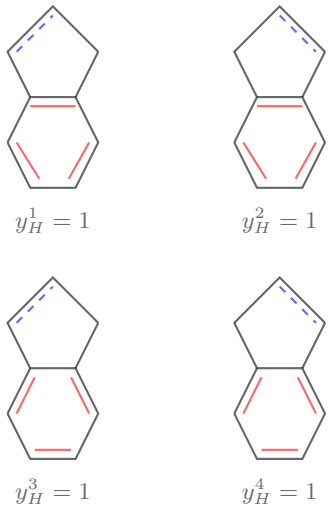


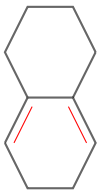
Figure 3.4: Earring Inequality

Since every Fullerene graph has exactly twelve number of pentagon, it is easily to find this pair of faces. Moreover due to the fact that there will not be two many number of these set of inequalities they are exploited to strengthen the linear relaxation of the Fries Polytope.

3.2.6 The Twin Inequality

For each hexagon H, its associated *edge cut* set, $\delta(H)$, is defines as

$$\delta(H) = \{e := uv \subseteq E : |\{u, v\} \cap V(H)| = 1\} \tag{3.36}$$



Suppose $\left(H, H^{'}\right)$ be a pair of adjacent through *exactly by one edge* hexagons such that $|H \cap H^{'}| = 1$, then the inequality

$$x_e - x_{e^{'}} + y_H + y_{H^{'}} \leq 1 \quad \left\{e, e^{'}\right\} \in \delta(H) \cap H^{'} \tag{3.37}$$

is referred to as twin inequalities. It should be noticed that for each pair of hexagons there are two set of *Twin Inequality*.

Theorem 13. *The Twin Inequality is valid for Clar Polytope.*

Proof The proof is based on considering the variant values of $y_H, y_{H'} \in \{0,1\}$ which are restricted to the cases where y_H $y_{H'}$ can not take value equal to one simultaneously according to the stable set constraint.

- $y_H = y_{H'} = 0$; trivial.
- $y_H = 0, y_{H'} = 1$; there are two possible matchings for covering the vertices in hexagon H when $y_{H'} = 1$ and in both cases it is easy to see that $x_e - x_{e'} = 0$ (they both either covered or not covered by the corresponding matching), which verifies the validity of the twin inequalities.
- $y_H = 1, y_{H'} \in \{0,1\}$, in this case one have $x_e = x_{e'} = 0$, according to *Hex Inequality* which reflects the fact that when $y_H = 1$ neither of the edges in the edge cut set of H is covered by any feasible perfect matching.

In the following subsections some inequalities that are dominated by the matching equality and *Hex inequalities* are stated.

3.2.7 The 3-path Inequality

3-path inequality for the Clar polytope gives the relation between y variables of two adjoint hexagons and 3 of the edges inside and in the boundary of one of the hexagons, $H^{'}$. Based on the labeling indicated in Figure (3.38) this inequality is as follows:

$$y_H + y_{H'} + x_a + x_b + x_c \leq 2 \tag{3.38}$$

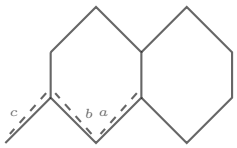


Figure 3.5: The 3-path inequality

Theorem 14. *3-path inequality is a valid inequality for* $P_{Clar}(\Gamma) \cap \mathbb{Z}^{|E|+|V|}$.

Proof. Let $(x, y) \in P_{Clar}(\Gamma) \cap \mathbb{Z}^{|E|+|V|}$ be a arbitrary feasible integer point in P_{Clar}. To prove the validity of (3.38), it is sufficient to show that for this arbitrary integer point in the polytope, the left hand side value of (3.38) can not be three. So let us consider the maximum feasible values can be allowed to assign to the variables in left hand side of the equation. Based one the stable set constraint $y_H + y_{H'} \leq 1$, so at most of the variables y_H and $y_{H'}$ could take value one, suppose $y_{H'} = 1$. Then, the *Hex inequality* $y_{H'} + x_a \leq 0$ force that $x_a = 0$. Moreover, by matching constraint one have $x_a + x_b \leq 1$, since x_a is forced to be zero, let x_b takes its maximum value which is one but this leads x_c to take only value zero based on the matching inequality. Hence, at most one might have $y_{H'} = x_b = 1$ and as a result $y_H = x_a = x_c = 0$. □

Obviously,the number of adjacent hexagons can not exceed the number of edges in Fullerene dual, $\frac{3|V|}{2}$. One may notice that for each pair of adjacent hexagon there will be four set of 3-path inequalities and we are going to exploit them as cutting planes in branch and cut procedure.

3.2.8 The Paralell-edge Inequality

Considering each hexagon H and its two paralell edges a and b, the paralell edge inequality is described as

$$-x_a - x_b + y_H \leq 0 \tag{3.39}$$

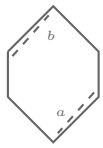


Figure 3.6: Parallel-edge inequality

Theorem 15. *Parallel- edge inequality is valid for* $P_{Clar}(\Gamma) \cap \mathbb{Z}^{|E|+|V|}$.

Proof. Suppose $(x, y) \in P_{Clar}(\Gamma) \cap \mathbb{Z}^{|E|+|V|}$. One needs to take the similar approach as in the proof of previous theorem. To this, when $y_H = 1$, then exactly one of the variables x_a or x_b are definitely one.This leads to the zero value at the left hand side and in the case when $y_H = 0$, $x_a + x_b \geq 0$ is definitely a valid constraint for P_{Clar}. □

There are at most $3\frac{3n}{2}$ inequalities of this type and as 3-path ones they are used as cutting planes.

3.3 A stable set formulation for the Clar number

Every feasible solution $(\bar{x}, \bar{y})$ of Problem (3.23a)-(3.23g) defines a perfect matching $M = \{e \in E \mid \bar{x}_e = 1\}$ of Γ whose edges are partitioned into two sets: the set of edges M' that belong to one of the saturated hexagons, i.e., $M' = \{e \in M \mid e \in E(H)$ and $\bar{y}_H = 1\}$ and $M'' = M \setminus M'$, its complement with respect to M. Consequently, since every node of V is the endpoint of some edge in M, such a matching induces a partition of V into 6-node sets ($V(H)$ for some $H \in \mathcal{H}(\Gamma)$ with $\bar{y}_H = 1$) and 2-node sets (the endpoint of e for $e \in M''$), which are all pairwise disjoint.

Therefore, we can reformulate Problem (3.23a)-(3.23g) as the following *set*

partioning problem, based on the nodes sets of these two categories.

$$\max \quad (1^{\mathcal{H}(\Gamma)})^T y \tag{3.40a}$$

$$\text{s.t.} \quad A^\Gamma x + W^\Gamma y = 1^V \tag{3.40b}$$

$$x \geq 0^E \tag{3.40c}$$

$$y \geq 0^{\mathcal{H}(\Gamma)} \tag{3.40d}$$

$$x \in \mathbb{Z}^E \tag{3.40e}$$

$$y \in \mathbb{Z}^{\mathcal{H}(\Gamma)}. \tag{3.40f}$$

Here for a finite set U, by 0^U (or 1^U) we denote a vector of $\mathbb{R}^U$ with all its components equal to 0 (or equal to 1). The $|V| \times |E|$ matrix A^Γ is the node-edge incidence matrix of the graph Γ, i.e., a matrix where the column corresponding to edge $e = (i, j)$ is the incidence vector of the set $\{i, j\}$ as a subset of V. The matrix W^Γ, that we call the *hexagon matrix*, is the $|V| \times |\mathcal{H}(\Gamma)|$ matrix where the column corresponding to hexagon H is the incidence vector of $V(H)$ as a subset of V.

The constraints (3.40b)-(3.40f) guarantee that the solution is a 0–1 vector; indeed, since A^Γ and W^Γ have components in 0–1, to satisfy (3.40b) no variable can exceed value 1. Constraint (3.40b) imposes that a 0–1 vector is feasible if and only if the sets defined by the columns of A^Γ and W^Γ, corresponding to the solution components that have value 1, is a partition of V. The rows of the two matrices are never empty and a feasible soution to the system always exists for the values of n we consider, since Γ has at least one perfect matching.

The formulation (3.40a)–(3.40f) was already used in [91] by Hansen and Zheng to formulate the Clar number problem for benzenoid hydrocarbons (mole cules made of carbon and hydrogen atoms). The graphs associated with these molecules are bipartite (all cycles have length 6) thus the authors conjectured that the linear programming relaxation might always yield integral optimal solutions. The conjecture was proved to be true in [92] by Abeledo and Atkinson that showed that the linear programming relaxation (3.40a)–(3.40d) is totally unimodular, thus an optimal slution of the corresponding linear program is always integral. This implies that the problem of computing the Clar number of a benzenoid hydrocarbon is solvable in polynomial time.

In the case of Fullerene graphs odd cycles are also present and the result of Abeledo and Atkinson does not hold. The question whether the Clar problem for Fullerene graphs is solvable in polynomial time or not has not been settled yet; see, e.g., [95].

Since, as observed, the polytope defined by linear programming relaxation (3.40a)–(3.40d) has fractional vertices, in order to strengthen the formulation it is necessary to find valid inequalities that can be added to the formulation either "statically", if their number is small, or "dynamically" by a suitable separation algorithm. This can be done by introducing the *intersection graph of a matrix*.

Let us denote by $B = [A^\Gamma \mid W^\Gamma]$ the coefficient matrix of constraint (3.40b). Then the *intersection graph* $G_B = (V_B, E_B)$ of matrix B is defined as follows:

every element of the node set V_B is associated with a column of B. For two distinct nodes i and j of V_B the edge (i, j) exists if and only if $B_i^T B_j \neq 0$, where B_i and B_j are the columns of B associated with i and j, respectively. In other words, the subsets of V defined by the two columns are disjoint.

Since the columns selected by a feasible solution of (3.40a)–(3.40f) are such that their sum is the vector of all ones $1^{(V)}$, then for any pairs of them there is no edge in E_B connecting the corresponding nodes in V_B. Consequently, these nodes are a *stable set* of the graph G_B. Let us call $STAB(G)$ the convex hull of the incidence vectors of all the stable sets of a graph $G = (N, J)$. Being these vectors finitely many, $STAB(G) \subset \mathbb{R}^N$ is a polytope and is called the *stable set polytope*.

Thus, Problem (3.40a)–(3.40f) can be reformulated as follows:

$$\begin{array}{ll} \max & (1^{\mathcal{H}(\Gamma)})^T y \\ \text{s.t.} & A^\Gamma x + W^\Gamma y = 1^V \\ & (x, y) \in STAB(G_B), \quad \text{where } B = [A^\Gamma \mid W^\Gamma]. \end{array} \tag{3.41}$$

A well known linear programming relaxation of the stable set problem is based on two families of inequalities that we are going to briefly describe.

A *clique* of a graph G is a set K of pairwise adjacent nodes. A clique is called a *maximal clique* if it is not properly contained in another clique. Since the intersection of a stable set and a clique of G has at most one element, the following inequality, called the *clique inequality*, is valid for $STAB(G)$:

$$\sum_{v \in K} z_v \leq 1, \qquad \text{where } K \text{ is a clique of } G, \tag{3.42}$$

where $z \in \mathbb{R}^N_+$ is a non-negative vector indexed by the nodes of G. If the clique K is maximal, then the inequality (3.42) cannot be dominated by any other inequality and actually it defines a facet of $STAB(G)$. Let us call $\mathcal{K}_G$ the family of all maximal cliques of G. Then

$$\sum_{v \in K} z_v \leq 1, \quad K \in \mathcal{K}_G \tag{3.43}$$

is a system of inequality whose constraint matrix is called the *clique matrix* of $STAB(G)$. The inequalities defined by the clique matrix along with the non-negativity constraints provide a relaxation of $STAB(G)$ that often approximates $STAB(G)$ pretty well. For a well know class of graphs, the *perfect graphs*, the relaxation coincides with the polytope since all its extremal feasible points are integral. For general graphs, when combined with the integrality restrictions, it yields an integer linear programming formulation of the stable set problem.

Unfortunately, for general graphs $\mathcal{K}_G$ has exponentially many elements. Therefore the clique matrix cannot be entirely given explicitly in a formulation that has to be used in actual computations. Therefore, maximizing (or minimizing) a linear function over the set defined by (3.43) and $z \geq 0$ must be

done by cutting plane techniques based on a repeated solution of a separation problem for the system (3.43). Such a separation problem amounts to finding a clique of maximum weight on a suitable weighted graph, which is an NP-hard problem for general graphs.

The situation is much more favorable when the stable set problem is defined on the intersection graph of a Fullerene graph, as stated in the following.

Proposition 1. *For a Fullerene graph $\Gamma = (V, E)$ with adjacency matrix A^Γ and hexagon matrix W^Γ, the family of maximal cliques of its intersection graph G_B, where $B = [A^\Gamma \mid W^\Gamma]$, has $|V|$ elements whose incidence vectors are given by the rows of matrix B.*

Proof. For a node v of Γ, the corresponding row of A^Γ has a 1 in the column indexed by an edge incident with v. The corresponding row of W^Γ has a 1 in the column indexed by an hexagon with v in its node set. The node sets of all these columns have a common intersection and thus their corresponding nodes in G_B define a maximal clique K_v.

If the claim is false, then there must exist a node w of G_B that is not in K_v but is adjacent to two nodes u and v in K_v. Moreover, the node sets corresponding to u, v, and w need not have a common intersection t, because otherwise they would be contained in the clique K_t defined by row t of B. It is easy to see that in a Fullerene graph three pairwise intersecting node sets that are either the endpoints of an edge or the nodes of an hexagon have always a common intersection. □

Because of Proposition 1, no clique inequalities need to be added to the formulation (3.40a)-(3.40f).

The second family of additional constraints are the *odd hole inequalities* as described in section 3.2.3.

The maximum cardinality of a stable set S in a hole $C \in \mathcal{C}$ is obtained, for example, by numbering the nodes of C, as they appear along the cycle, with consecutive integers starting from 1 and by assigning the nodes with even label to the stable set. Thus the inequalities of (3.32) are valid also when $|V(C)|$ is even. In this case, though, the inequality is dominated by the one obtained by summing up the clique inequalities defined by each of the cliques made by the edges incident to each of the even labeled nodes. If $|V(C)|$ is odd, but C has a chord e, the the edges of C and the chord make two cycles C' and C'', one of even size and the other of odd size. The inequality obtained by summing up the inequalities (3.32) defined by C' and C'' dominates the inequality (3.32) defined by C. Since we are interested in families of inequalities that do not contain member trivially dominated by positive combinations of other known inequalities, the members of $\mathcal{C}_G$ are requested to have odd sizes and to be chordless.

The inequalities (3.32) are exponentially many, in general. However a polynomial time separation algorithm exists for them (see [4] (8.3.6) and (9.1.11)). Consequently, we can optimize a linear function in polynomial time over the

polytope described by the equations (3.40b), the non-negativity constraints and the whole family of odd hole inequalities.

For completeness we recall here the polynomial time algorithm that solves the separation problem of the odd hole inequalities. Suppose that we are given a point $\bar{z} \in \mathbb{R}^V$ with $0^V \leq z \leq 1^V$. We assign a weight $\bar{s}_e = 1 - \bar{z}_u - \bar{z}_v$ to every edge $e \in E$ of G. The inequalities (3.32) are satisfied by $\bar{z}$ if and only if the following holds:

$$\sum_{e \in E(C)} \bar{s}_e \geq 1, \quad C \in \mathcal{C}_G. \tag{3.44}$$

It is therefore sufficient to find the shortest (according to the edge length $\bar{s}$) odd hole in G and check whether or not its length is less than 1. To find the shortest odd hole we do the following. For every node v of G we create two copies: one at the "top" level v' and another at the "bottom" level v''. For every edge $e(u, v)$ of G we create two edges (u', v'') and (u'', v'), both with weight $\bar{s}_e$. In every simple path from v' to v'' nodes alternate from "top" to "bottom" and eventually terminates at the "bottom" level; therefore, every simple path from v' to v'' has odd length. Every such a path induces a cycle of odd length in G. For every $v \in V$ we compute the shortest v'–v'' path. The shortest among these $|V|$ shortest paths defines the desired shortest odd hole in G. Since a shortest path in a graph with non-negative edge lengths can be found in polynomial time, the whole procedure has polynomial time complexity.

3.4 Empirical evaluation of the algorithms

Based on the formulations described in the previous sections, we have designed several solution algorithms to solve the Clar problem to optimality. The pattern of each such algorithms is the same and can be outlined as follows.

First the optimal value over the linear programming relaxation is computed by providing the linear programming (LP) solver with a static linear system. This is the case of the formulations whose LP relaxation contains a small (polynomial in n) number of constraints. The formulations (3.1b)–(3.1g) plus (3.6) and (3.23a)–(3.23h) are of this type.

If the LP relaxation contains a number of constraints that is exponential in n, then its optimal solution is obtained dynamically by an iterative cutting-plane algorithm. This is the case, for example, of the LP relaxation (3.40a)–(3.40f), where the odd hole inequalities are used to strengthen the LP formulation.

In the cutting plane phase of the algorithm that involves an exponentially sized family of inequalities $\mathcal{L}$, we solve the first linear program (derived from a static polynomially sized relaxation) and we check whether the optimal solution satisfies all the inequalities in $\mathcal{L}$. If this is the case, we have solved the relaxation that includes (implicitly) all the inequalities in $\mathcal{L}$. Otherwise, we find one or more inequalities in $\mathcal{L}$ that are violated by the current solution, we add them to the linear program, and we iterate

Once the computation of the optimal solution of the relaxation is completed, two scenarios are possible. In the first one, such an optimal solution is integral

and we are done: the corresponding optimal value is the Clar number of the instance. In the second one, the solution has at least one fractional component. Then we submit the formulation to a mixed linear programming (MIP) solver. Solvers of this kind are able to solve linear programs with the additional requirement that a subset of the variables can only have integral values.

Recent versions of commercial MIP solvers are able to strengthen the LP formulation of the mixed integer linear program that are supposed to solve. The inequalities that they generate are produced with different techniques that typically are not based on the combinatorial structure of the problem (like, for example, the one we described for the generation of the odd hole inequalities). If the best relaxation that they are able to produce is not integral, then they proceed with a *branch and bound* phase, i.e., with a pseudo-enumeration algorithm based on decomposing the original problem into two subproblems. Each of them is characterized by having one of the solution variables, that has a fractional value at the optimal solution of the relaxation, fixed to 0 and to 1, respectively (here we assume that we are in the special case where all variables are bounded to have values between 0 and 1).

This recursive process can be described by a binary tree where every node represents the original linear programming relaxation with the addition of a set of variable fixings. The proliferation of the tree is interrupted when a node corresponds to an infeasible linear program, or when the LP solution is integral, or when the optimal LP value at the node (*upper bound*) falls below the current best integral solution found during the execution of the whole process (*lower bound*).

Actually, state-of-the-art MIP solvers implement the technique known as *branch and cut*, whose main difference with respect to branch and bound is that at every node of the tree the LP relaxation is further strengthened with additional inequalities. They can either be *global*, i.e., valid for the original problem and thus useful to improve the relaxation of every other node, or *local*, i.e., valid only for the subproblems corresponding to node and to all its descendants.

It is important to stress the fact that it is of paramount importance to keep the size of the enumeration tree small. If the size of the tree grows exponentially fast, then so does the computation time. Therefore, it will become practically infeasible to solve instances even of moderate size if the tree is not kept small. The principal devise used to limit the tree proliferation is the "pruning" of a node for which the upper bound falls below the lower bound. Therefore, it is important to have relaxations that yield upper bounds that are as low as possible. Since the better the relaxation the lower is the value of the bound, most of the efforts of the present paper are focused on identifying strong relaxations that, at the same time, are not too time consuming to solve for an LP solver.

For our computational experiments we used IBM CPLEX Optimizer Version 12.71 both as an LP and an MIP solver, with its default settings, with the exception of the preprocessing that we turned off because no advantages from this feature, which is most beneficial in general mixed integer programming, are to be expected in our case.

All the computations were carried out with an Intel i7 4960X with 6 cores,

clock at 3.6Ghz, and 64Gb of core memory, equipped with the operating system Linux, Version 4.4.0. The solution algorithm was implemented in C and compiled with the gcc compiler with optimization switch "`-O2`".

3.4.1 Comparing the three models

Our first experiment was to compare the performances of the three algorithms based on the linear programming relaxation (3.1b)–(3.1g) plus (3.6), that will be referred to as Algorithm $\mathcal{A}1$; on the relaxation (3.23b)–(3.23h) (Algorithm $\mathcal{A}2$), and on the relaxation (3.40b)–(3.40f) (Algorithm $\mathcal{A}3$). No additional inequalities were used in this experiment to strengthen the formulation and the CPLEX cut generation feature was disabled.

The tree algorithms were tested on Fullerene graphs with 120, 122, and 124 nodes. For each size we examined the first 2 000 isomers generated by the code `fullgen`. As it will be more evident in the next section, the hardness of the Clar problem varies quite substantially among instances of similar sizes depending on whether the reminder of the division $n/6$ is 0, 2, or 4. For this reason we selected sizes for which the reminders have all these three values. The results are shown in Table 3.1.

	$n = 120$		$n = 122$		$n = 124$	
Algorithm	BB	Time	BB	Time	BB	Time
$\mathcal{A}1$	8 817	1 846	9 639	2 025	10 682	2 308
$\mathcal{A}2$	7 796	2 610	8 508	2 874	9 555	3 282
$\mathcal{A}3$	50	47	40	45	20	48

Table 3.1: Comparison of the three basic algorithms

The columns of the table labeled "BB" report the average number of nodes of the branch and bound tree, while those labeled "Time" refer to the average computation time in milliseconds. Both averages are computed over the set of 2 000 isomers considered in the computational study.

The clear winner among the three algorithms is $\mathcal{A}3$, which is based on the stable set formulation described in Section 3.3. It is from two to three orders of magnitude faster that the other two "matching-based" algorithms both in terms of generated branch and bound nodes and in terms of computation time. About these two algorithms, $\mathcal{A}2$ seems to be superior to $\mathcal{A}1$ in terms of generated nodes. This was to be expected since the corresponding formulation is stronger than the other. However, it requires a longer computing time per node. Possibly because the corresponding relaxation is about six times larger than the other. However, it is conceivable that for larger values of n the difference in the number of generated nodes compensates the higher demand of computing resources at every node and that thus also the total CPU time for $\mathcal{A}2$ becomes shorter compared to the one of the other algorithm.

We repeated the same computational experiment by strengthening the three formulations. In particular, we enabled the automatic cut generation in CPLEX. Therefore, in this new setting the algorithm was of a branch and cut type, because the solver could add inequalities at every node of the enumeration tree. In addition, we statically added all the 3-clique inequalities to the formulation of Algorithm $\mathcal{A}2$ and all the 5-hole inequalities (defined by every 5-cycle in the Fullerene graph that intersects 5 hexagons) to the formulation of $\mathcal{A}3$.

Due to the different types of inequalities it is possible that the strengthening automatically operated by CPLEX changes the performance ratios of the three algorithms. The results reported in Table 3.2 might help settling this issue. In the table the columns labeled "BC" give the average number of the branch and cut nodes generated by the MIP solver. To the names of the algorithms we appended the character "C" to recall the fact that "CPLEX inequalities" were generated.

	$n = 120$		$n = 122$		$n = 124$	
Algorithm	BC	Time	BC	Time	BC	Time
$\mathcal{A}1C$	4 251	1 152	2 173	1 272	2 855	1 387
$\mathcal{A}2C$	1 268	788	1 023	882	1 056	957
$\mathcal{A}3C$	1.95	149	0.43	169	0.43	166

Table 3.2: Comparison of the strengthened version of three algorithms

Compared to the first experiment, in this one we observe a substantial reduction in the number of nodes in the enumeration tree, in particular for Algorithm $\mathcal{A}3C$ that is able to solve many instances without any recourse to enumeration (the average number of nodes is less than 0.5). The reduction is remarkable also for the computing time, but only for the algorithms $\mathcal{A}1C$ and $\mathcal{A}2C$. As noted for the first set of experiments, it is conceivable that for larger values of n also Algorithm $\mathcal{A}3C$ becomes faster than $\mathcal{A}3$.

3.5 Hunting for Clar isomers

The time needed to compute the Clar number, i.e., to solve Problem (3.23a)-(3.23g) to optimality, varies from one isomer to another quite substantially. In Table 3.5 we show the results of the Clar number computation for all the isomers of size 76, using Algorithm $\mathcal{A}3C$. The column "Mean CPU ratio" is the ratio of the average computing time for a single isomer, evaluated over the set of all isomers sharing the same Clar number, and the same average for the group of Clar isomers.

It is clear from the table that most of the computation effort is essentially useless, since it is not actually necessary to compute the Clar number exactly for any instance belonging to the first four groups. It is only necessary to have a mathematical proof that an isomer is not a Clar isomer. Once this proof is

Clar	# Isomers	Mean CPU ratio	Total CPU share
6	37	7.72	0.23%
7	2 121	9.52	16.04%
8	12 673	7.28	73.31%
9	3 898	3.26	10.09%
10	422	1	0.34%

Table 3.3: Clar computation breakdown for $n = 76$

provided, the computation of the Clar number can be abandoned since it cannot contribute to the construction of the class of the Clar isomers.

Fortunately, this mathematical proof is readily available: if the upper bound on the value of the Clar number falls below the value of the largest Clar number found among the isomers that have been analyzed, we have a proof that the instance cannot be a Clar isomer. The upper bound is the optimal value of the relaxation at hand if the algorithm is in the first phase. Otherwise, if we are in the enumeration phase, it is the maximum of the optimal values of the relaxations associated with all the leaves of the current enumeration tree.

All isomers of a given size are generated by the computer code `fullgen` (see [46]) with the command "`fullgen <n> code 6`", where "`<n>`" is the isomer size. The code generates all the isomers sequentially. We give each of them an integer identifier that corresponds to the position of the isomer in the sequence. The output of our algorithm contains the list of the integer identifiers of all the Clar isomers of a given size. For each such isomer we also record the set of hexagons that make the optimal solution.

The algorithm that constructs the set of Clar isomers of a given size is based on $\mathcal{A}3C$ and works as follows. The isomers are submitted to the algorithm in increasing order of identifier. Suppose that we are processing isomer j, that the maximum Clar number of the isomers from 1 to $j-1$ is $\bar{C}$ and that we have collected the set of all isomers from 1 to $j-1$ whose Clar number equals $\bar{C}$. We add the following new constraint to formulation:

$$1000 \cdot (1^{\mathcal{H}(\Gamma)})^T y \geq 1000 \cdot \bar{C} - 1. \tag{3.45}$$

This constraints has the effect of making infeasible the problem if the Clar number of the instance is less that $\bar{C}$. Finally we execute $\mathcal{A}1C$. If the problem is infeasible, then we proceed to the next isomer. If it is feasible and the optimal value equals $\bar{C}$, then we add the isomer to the collection. If the optimal value is bigger than $\bar{C}$, then we reset $\bar{C}$ to the new value and we replace the current collection with the one made by the current isomer.

The advantage of this approach is that the time taken by $\mathcal{A}1C$ to prove that the problem is infeasible because of (3.45) is typically much shorter that the time needed to find the optimal solution in the formulation without (3.45).

The computational results are reported in Table 3.5. The table is split into three groups of columns. Each group corresponds to one of the possible values

of the reminder of the division $n/6$. As a matter of fact, the cardinality of the Clar isomers $|\mathcal{I}_n|$ and the average CPU time grow quite regularly with n within each of such groups. To the contrary, these two values, as functions of n, are not monotone at all.

The columns headed with "CPU", report the average computing time for a single isomer evaluated over the whole set of isomers of size n, whose number is given in Table 2.1.1. The CPU time spent for an isomer amounts to the time necessary either to find its Clar number of to prove that its Clar number is strictly less than the largest one found among the preceding isomers in the sequence.

All the computations were carried out with an Intel i7 4960X with 6 cores, clock at 3.6Ghz, and 64Gb of core memory, equipped with a Linux 4.4.0 operating system. The solution algorithm was implemented in C and compiled with the gcc compiler with optimization switch "`-O2`". The MIP solver employed by the algorithm was the IBM CPLEX Optimizer Version 12.71.

According to the values of CPU reported in Table 3.5 and to the number of isomers given in Table 2.1.1, the total computation time amounts to 334 days. Indeed the actual computation time, that includes setup and other auxiliary operation, was 395 days. Since we could run the experiments in parallel using the 6 cores available in the processor, the total wall clock time was about 9.4 weeks.

Let us now analyze the results of Table 3.5 in more details.

The least regular behavior, as $|\mathcal{I}_n|$ and CPU as a function of n are concerned, appears when n is a multiple of 6. When n falls in the range $[24, 204]$, the value $|\mathcal{I}_n|$ falls in the following interval defined by two cubic functions (see Figure 3.7):

$$\left[0.02 \cdot \left(\frac{n-24}{6}\right)^3,\ 40 + 0.03 \cdot \left(\frac{n-20}{6}\right)^3\right].$$

The value CPU does not seem to depend on n, for $n \in [24, 204]$, and has an average of 3.5 milliseconds.

If $n \equiv 2 \mod 6$, then $|\mathcal{I}_n|$ is well approximated by the function

$$0.5 \cdot \left(\frac{n-18}{6}\right)^{3.76}$$

(see Figure 3.8), while the value of CPU is approximated by the linear function

$$1.75 + 0.8 \cdot \left(\frac{n-18}{6}\right).$$

Finally, when $n \equiv 4 \mod 6$, the number of Clar isomers $|\mathcal{I}_n|$ is approximated by the function

$$0.65 \cdot \left(\frac{n-18}{6}\right)^3$$

(see Figure 3.9), while the value of CPU is substantially independent of n, for $n \in [24, 204]$, with an average of 5.23 milliseconds.

$n \equiv 0 \mod 6$			$n \equiv 2 \mod 6$			$n \equiv 4 \mod 6$		
n	$\lvert\mathcal{I}_n\rvert$	CPU	n	$\lvert\mathcal{I}_n\rvert$	CPU	n	$\lvert\mathcal{I}_n\rvert$	CPU
24	1	4.00	26	1	0.00	28	2	6.00
30	3	8.00	32	6	2.67	34	6	6.00
36	9	6.89	38	15	5.11	40	18	5.03
42	16	4.26	44	72	5.07	46	27	4.13
48	16	3.62	50	159	8.62	52	67	5.28
54	39	4.02	56	439	8.94	58	100	4.91
60	18	2.78	62	718	10.42	64	195	4.82
66	11	6.13	68	1,408	10.52	70	267	4.56
72	22	2.86	74	2,107	10.29	76	422	5.11
78	19	3.29	80	3,530	10.48	82	558	5.05
84	47	2.64	86	4,795	10.67	88	812	4.94
90	50	2.24	92	7,216	10.76	94	1,065	5.01
96	72	2.07	98	9,340	11.10	100	1,442	4.89
102	66	2.64	104	13,187	11.50	106	1,703	4.93
108	101	2.13	110	16,142	11.96	112	2,233	4.91
114	80	2.82	116	21,394	12.62	118	2,647	4.93
120	150	2.35	122	25,687	13.26	124	3,301	4.99
126	144	2.45	128	32,833	14.39	130	3,826	5.01
132	186	2.68	134	38,015	14.85	136	4,655	5.06
138	208	2.65	140	47,293	15.78	142	5,418	5.07
144	273	2.77	146	54,290	16.72	148	6,407	5.13
150	188	2.90	152	65,668	17.81	154	7,120	5.19
156	360	3.00	158	73,728	19.26	160	8,385	5.93
162	378	3.41	164	88,081	20.70	166	9,498	5.34
168	388	3.57	170	97,705	22.02	172	10,913	6.28
174	431	3.66	176	114,864	23.28	178	11,962	5.48
180	559	3.47	182	125,981	24.89	184	13,719	5.56
186	407	3.58	188	146,187	26.43	190	15,114	5.65
192	732	3.74	194	159,356	27.88	196	17,107	5.75
198	685	3.80	200	183,402	28.15	202	18,722	5.87
204	662	4.07						

Table 3.4: Clar isomers and average computing time

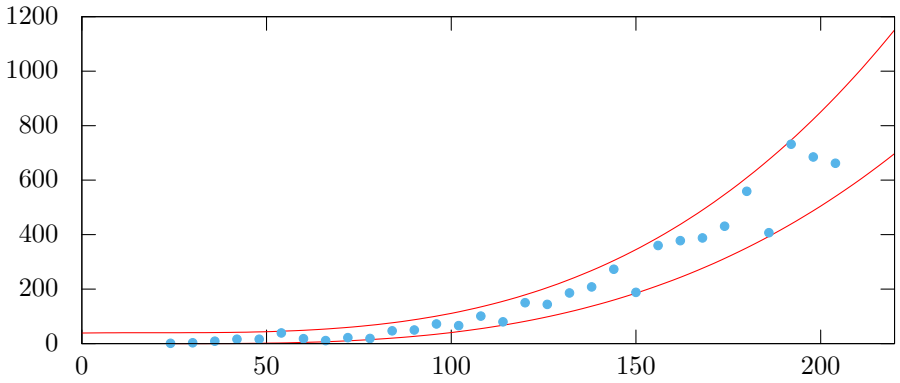


Figure 3.7: $|\mathcal{I}_n|$ for $n \equiv 0 \mod 6$

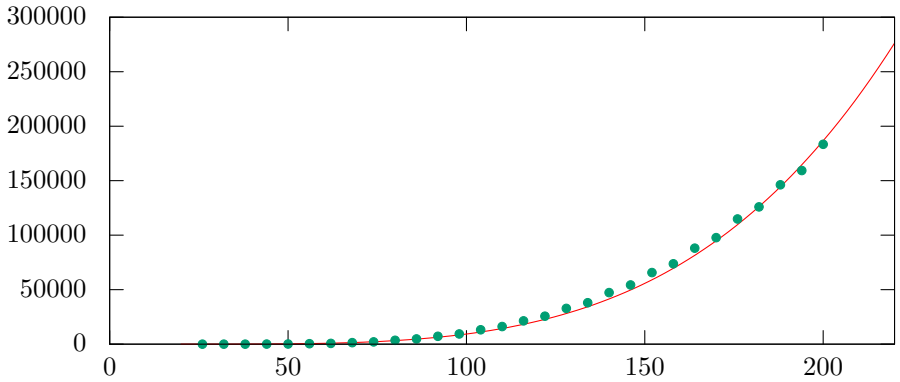


Figure 3.8: $|\mathcal{I}_n|$ for $n \equiv 2 \mod 6$

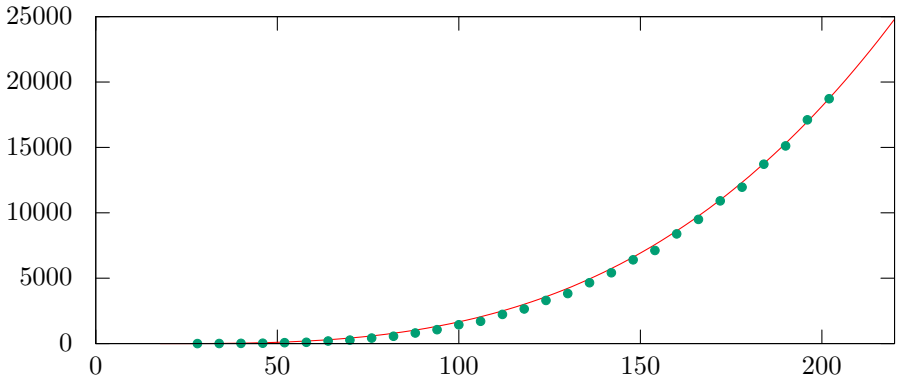


Figure 3.9: $|\mathcal{I}_n|$ for $n \equiv 4 \mod 6$

Rather then reporting the values of the $\bar{C}_n$ in a table, we summarize them in the following

Proposition 2. *For $n \leq 204$ the maximum Clar number $\bar{C}_n$ of a Fullerene graph over the set of all isomers of n nodes is*

$$\bar{C}_n = \begin{cases} 3\left(\frac{n-2}{6}\right) - 2\left\lfloor \frac{n-2}{6} \right\rfloor - 3 & \text{if } n \geq 20 \text{ and } n \notin \{22, 30\}, \\ 2 & \text{if } n = 30. \end{cases} \tag{3.46}$$

In [97] Zhang and Ye prove the following

Theorem 16 (Theorem 1.1 in [97])**.**

$$\bar{C}_n \leq \left\lfloor \frac{n-12}{6} \right\rfloor. \tag{3.47}$$

They call *extremal Fullerene* those for which the Clar number is the one given by Theorem 16 and in [97] they show examples of extremal Fullerene with 60 and 70 nodes. In [98] the same authors characterize the extremal Fullerenes and apply this characterization to the case $n = 60$, for which they find and show all the 18 isomers that are extremal. All these results are in agreement with those of Table 3.5.

In [106] Gao, Li, and Zhang prove the following

Theorem 17 (Theorem 1.2 in [106])**.**

$$\bar{C}_n \leq \begin{cases} \left\lfloor \frac{n}{6} \right\rfloor - 3 & \text{if } n \equiv 2 \mod 6, \\ \left\lfloor \frac{n}{6} \right\rfloor - 2 & \text{otherwise.} \end{cases} \tag{3.48}$$

They also call *extremal* a Fullerene for which the Clar number is the one given by (3.48). Quite remarkably, they provide a complete characterization of extremal Fullerene.

With this characterization the problem of constructing an extremal Fullerene graph of n nodes is reduced to the one of constructing a suitable plane graph of roughly $n/3$ nodes. However, in [106], it is not said if at least one such plane graph exists for all n; only a few examples are shown with $n = 70$ and $n = 80$. From Proposition 1 we know that this is indeed the case for all n except $n = 30$, but only if we are in the range $[20, 204]$ (the case $n = 22$ is excluded, of course, since no Fullerene graph exists with this size). We believe that Proposition 1 can be extended to all sizes $n > 204$, possibly using the techniques and the results of [106]. Therefore, we conclude with the following

Conjecture 1. *The maximum Clar number $\bar{C}_n$ of a Fullerene graph over the set of all isomers of n nodes is*

$$\bar{C}_n = \begin{cases} 3\left(\frac{n-2}{6}\right) - 2\left\lfloor \frac{n-2}{6} \right\rfloor - 3 & \text{if } n \geq 20 \text{ and } n \notin \{22, 30\}, \\ 2 & \text{if } n = 30. \end{cases} \tag{3.49}$$

Part II

Distributing the Errors in Iterative Interior Point Methods

Chapter 4

Preliminaries of Interior Point Methods

The first polynomial algorithm for Linear Programming (LP) was developed by Khachiyan [107] called ellipsoid algorithm. According to this algorithm a series of ellipsoids are constructed whose centers form a sequence of points which converge to an optimal solution of an LP problem. The ellipsoids are constructed in a way that progressing towards optimality from one iteration to another is guaranteed. Updating the ellipsoids requires a high pre-iteration algebra operations therefore it has never become a competitive alternative to the simplex method [108].

Later, interior point method (IPM) was proposed by Karmarkar [109]. The idea beyond the Karmarkar's algorithm was that instead of inscribing a ball into the ill-conditioned corner of the feasible polytope, it employs projective geometry to transform this corner into a regular well-conditioned simplex polytope, inscribes a ball into it, and exploits the fact that optimization on a ball is a trivial operation.

Moreover, using a potential function in this method insures the reduction of a distance to optimality at each iteration. This algorithm is computationally attractive because the optimality is reached after a relatively small number of iterations [101] even though a single iteration of Karmarkar's method is expensive due to requiring to apply a projection operation.

Karmarkar's projective method and the projected Newton barrier method were proved to be equivalent in Gill et al. [110]. This fact made an increasing influence in the role of *barrier functions* in the theory of interior point method.

Interior point methods for linear and (convex) quadratic programming have low-degree polynomial worst-case complexity and it has the ability to derive the optimal solutions in an almost constant number of iterations which depends very little on the problem dimension. These impressive features makes the interior point methods applicable for very large scale optimization.

Interior point methods are competitive when dealing with small problems

of dimensions below one million constraints and variables and are beyond competition when applied to large problems of dimensions going into millions of constraints and variables [101].

Primal-dual techniques are usually faster and more reliable than pure primal or pure dual approaches [111, 112]. *Infeasible - primal- dual* interior point algorithm have impressive and attractive features for which it is widely accepted that it is the most efficient interior point method. These features follows from the fact that the logarithmic barrier method is applied to the primal and the dual problems at the same time. This method was discussed in Megiddo [113] and then later, its theoretical background and the first complexity results are given in Kojima et al. [114]

Many scientists interpreted the interior point methods as algorithms which follows a central path on their way to an optimal solution [115]. In the late 80s, the research on the implementation of the interior point methods was impressively progressed and the theoretical crucial role of the logarithmic barrier functions was studied in [116] and later on, by the early 90s, sufficient evidence was given in [117, 118] to prove that the IPMs are efficient enough for solving very large scale linear programming problems.

Mixed Integer Programming solvers contain implementations of both simplex and interior point methods and have been led to impressive developments since the presence of interior point methods, over the last 25 years [119, 120, 121, 122]. Both methods are widely used and continue to compete with each other. It is worth mentioning that problems of large scale size generally seems to be solved preferably by interior point methods, though it is not always possible to predict the winner on a particular class of problems. For example, one of important factors that clarifies the efficiency of a given algorithm is the sparsity structure of the problem which determines the cost of linear algebra operations. The simplex method easily takes advantage of any hyper-sparsity in a problem [123] but its sequential nature makes it difficult to parallelise [124].

The key reason of IPMs success is solving *linear* optimization problem using logarithmic barrier function as a *nonlinear* programming technique. Soon after, a similar logarithmic barrier function methodology has been extended to solve quadratic [125] and nonlinear optimization problems [126]. The reason why the logarithmic function is a well type of barrier function was explained in Nesterov and Nemirovskii [127]. Indeed, Forsgren et al. [128] stated nicely that an especially appealing aspect of the interior-point revolution is its spirit of unification, which has brought together areas of optimization that for many years were treated as firmly disjoint.

The interested reader in nonlinear and semidefinite property are referred to surveys of Forsgren et al. [128], Vandenberghe and Boyd [129], respectively.

In this thesis the theory and implementation of interior point methods for solving convex quadratic programming problem will be considered.

4.1 Duality Theorem

Considering the primal convex quadratic programming problem as follows:

$$P_G : \min_x \quad c^T x + \frac{1}{2} x^T Q x \tag{4.1}$$

$$s.t \quad Ax \geq b \tag{4.2}$$

where $A \in \mathbb{R}^{m \times n}, Q \in \mathbb{R}^{n \times n}, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and the primal variable is $x \in \mathbb{R}^n$. Since we are interested in the convex quadratic problem, it requires that matrix $Q \in \mathbb{R}^{n \times n}$ is positive semidefinite which means that there is a factorization (not necessary Cholesky one) such that $Q = VE^2V^T$ is explicitly given, where $V \in \mathbb{R}^{n \times l}$ and $E \in Diag(\mathbb{R}^l{}_{++})$ is a positive diagonal matrix. Let's denote by $\nu_{P_G}(x)$ as a value of the objective function of the problem P_G at x.

Let Γ_G be a Lagrangian problem associated with the primal problem P_G, i.e.:

$$\Gamma_G : \min_x \quad c^T x + \frac{1}{2} x^T Q x + y^T (b - Ax) \tag{4.3}$$

$$s.t \quad y \geq 0. \tag{4.4}$$

Let's define the *Lagrange dual function* (or just a *lagrange function*) $\nu_{\Gamma_G} : \mathbb{R}^m \to \mathbb{R}$ as the minimum value of Lagrangian problem over x; for $y \in \mathbb{R}^m$,

$$\nu_{\Gamma_G}(y) := y^T b + \min_x \{ \quad (c^T - y^T A)x + \frac{1}{2} x^T Q x \} \tag{4.5}$$

one may notice that since Q is positive semidefinite, for each given y, the term $(c^T - y^T A)x + \frac{1}{2} x^T Q x$ is convex and differentiable in x and therefore the minimum, if exists, is given by the Fermat rule, which in our case becomes :

$$Qx + (c^T - y^T A) \quad = 0 \qquad \Longrightarrow \tag{4.6}$$

$$x^T Q x + cx - y^T A x = 0 \qquad \Longleftrightarrow \tag{4.7}$$

$$y^T A x \qquad = \left(x^T Q x + cx \right) \tag{4.8}$$

By substituting the equation (4.8) into the lagrange dual function (4.5), we have

$$c^T x - y^T A x + \frac{1}{2} x^T Q x \quad = c^T x - \left(x^T Q x + c^T x \right) + \frac{1}{2} x^T Q x \tag{4.9}$$

$$= -\frac{1}{2} x^T Q x \tag{4.10}$$

Now, by taking into account this substitution into the dual function we may write the *Dual problem* as:

$$D_G : \max_{x,y} \quad b^T y - \frac{1}{2} x^T Q x \tag{4.11}$$

$$s.t \quad A^T y - Qx = c, \tag{4.12}$$

$$y \geq 0. \tag{4.13}$$

Notice that the dual problem also turns out to be a linearly constrained convex quadratic program.

We continue our discussion by stating two fundamental properties used for deriving the necessary and sufficient condition for optimality of convex quadratic programing problems; namely the **Weak** and **Strong** duality properties.

Theorem 18. (***Weak Duality Property***) *Let (P_G) be a feasible quadratic program with positive semidefinite symmetric matrix Q in the objective and (D_G) be its corresponding dual problem. If x and $(\widehat{x}, y))$ are feasible solutions to this pair of problems, respectively, then the primal objective at x is greater than or equal to the dual objective at $(\widehat{x}, y)$, i.e, $\nu_{P_G}(x) \geq \nu_{D_G}(\widehat{x}, y)$.*

Proof. Let x and $(\widehat{x}, y)$ be feasible solutions to the problems (P_G) and (D_G), respectively. Then,

$$c^T x + \frac{1}{2} x^T Q x \overset{eq(4.12)}{=} (A^T y - Q\widehat{x})^T x + \frac{1}{2} x^T Q x \tag{4.14}$$

$$= y^T A x - x^T Q \widehat{x} + \frac{1}{2} x^T Q x \tag{4.15}$$

$$\overset{eq(4.2)}{\geq} y^T b - x^T Q \widehat{x} + \frac{1}{2} x^T Q x \tag{4.16}$$

Since Q is positive semidefinite, $(x - \widehat{x})^T Q (x - \widehat{x}) \geq 0$ this implies that $\frac{1}{2} x^T Q x - x^T Q \widehat{x} \geq -\frac{1}{2} \widehat{x}^T Q \widehat{x}$, applying it to the equation (4.16) results in

$$c^T x + \frac{1}{2} x^T Q x \geq y^T b - \frac{1}{2} \widehat{x}^T Q \widehat{x} \tag{4.17}$$

$$\nu_{P_G}(x) \geq \nu_{D_G}(\widehat{x}, y) \tag{4.18}$$

□

Here after, we assume that both pair of problems P_G and D_G are nonempty and for Theorem (18) the primal and dual objectives value are both finite.

Let's refer to the **nonnegative** gap between the primal and dual objectives,

$$\triangle := \nu_{P_G}(x) - \nu_{D_G}(\widehat{x}, y), \tag{4.19}$$

as the *duality gap* associated with the primal and dual feasible pair x and $(\widehat{x}, y)$, respectively.

If the duality gap at the primal dual feasible pair (x), $(\widehat{x}, y)$ is zero, i.e., $\nu_{P_G}(x) = \nu_{D_G}(\widehat{x}, y)$, then we say that the *strong duality* property holds.

Now based on the weak duality property, it can be derived easily that if the primal dual feasible pair x, $(\widehat{x}, y)$ satisfy the strong duality property, then x is primal optimal and $(\widehat{x}, y)$ is dual optimal.

In the following parts, we will go through the proof that strong duality property holds for our underlying problems (P_G) and (D_G).

We continue our discussion by stating the preliminary required to the proof of strong duality property for pair of problems P_G and D_G.

Lemma 3. *(Farkas Lemma): [130]*
Considering the system of inequalities,

$$Ax \geq 0, \quad c^T x < 0 \tag{4.20}$$

and the system of equalities and inequalities,

$$A^T y = c, \quad y \geq 0, \tag{4.21}$$

Where $A \in \mathbb{R}^{m \times n}$ and $c \in \mathbb{R}^n$, then, exactly one of these two systems has solution.

Proposition 3. *Let x^* be the primal optimum for the problem P_G, then there exists y^* such that (x^*, y^*) is feasible for D_G and, moreover, this pair of solutions satisfy the* ***Complementry Slackness Condition*** *(C.S.C):*

$$y^{*T} (Ax^* - b) = 0. \tag{4.22}$$

Proof. We are going to consider two cases, first, if $Qx^* + c = 0$, then since $y \geq 0$, let's define $y^* = 0$. So $A^T y^* - Qx^* = c$, i.e., $(x^*, 0)$ is a feasible solution for D_G. Moreover, $y^{*T}(Ax^* - b) = 0$. Otherwise, if $Qx^* + c \neq 0$, then obviously x^* is on the boundary of primal feasible set, i.e.,

$$I(x^*) = \{i| \quad A_i x^* = b_i\} \neq \emptyset. \tag{4.23}$$

Now based on the linear programming version of the Farkas Lemma 3 , exactly one of the two following systems has a solution.

$$\begin{array}{lll} (I): & d^T (Qx^* + c) < 0 & \qquad (II): \ {y_I}^T A_I = Qx^* + c \\ & A_I d \geq 0 & \qquad\qquad y_I \geq 0 \end{array}$$

Where A_I is rows of matrix A indexed in set I. Due to the fact that x^* is an optimal solution for P_G, we claim that the system (I) can not be solvable. To proof this, it is only sufficient to show that d is an improvement direction for P_G only if $A_I d \geq 0$, $d^T(Q^* + c) < 0$. Indeed, first notice that since $A_I d \geq 0$, it can be seen easily that $(x^* + \epsilon d)$ is a feasible solution for P_G, for a small enough value of ϵ. Now by comparing the value of the primal objective function at both x^* and $(x^* + \epsilon d)$, one can observe that for $\epsilon \geq$ sufficiently small:

$$c^T(x^* + \epsilon d) + \frac{1}{2}((x^* + \epsilon d)^T Q(x^* + \epsilon d)) \tag{4.24}$$

$$= c^T x^* + \frac{1}{2} x^{*T} Q x^* + \epsilon d^T (Qx^* + c) + \frac{1}{2}\epsilon^2 d^T Q d \tag{4.25}$$

$$< c^T x^* + \frac{1}{2} x^{*T} Q x^* \tag{4.26}$$

Hence, as a result, by Farkas lemma, system (II) has a solution, i.e.

$$\exists \ y_I \geq 0 : \ \ {y_I}^T A_I = Qx^* + c \tag{4.27}$$

By defining $y^* = \begin{bmatrix} y_I \\ 0 \end{bmatrix}$, it can be easily seen that (x^*, y^*) is feasible for D_G and moreover ${y^*}^T (Ax^* - b) = 0$.

□

Theorem 19. *Let x^* be the optimum solution for P_G, then there exists y^* such that (x^*, y^*) is optimum for D_G and for this pair of optimum solutions the strong duality property holds.*

Proof. Let y^* be chosen according to the proof scheme of Proposition 3, in the case if $Qx^* + c = 0$, then $y^* = 0$ and by considering the primal and dual objective values at this pair, one may obtain.

$$\nu_{P_G}(x^*) := c^T x^* + \frac{1}{2} x^{*T} Q x^* \tag{4.28}$$

$$= (-Qx^*)^T x^* + \frac{1}{2} x^{*T} Q x^* \tag{4.29}$$

$$= -\frac{1}{2} x^{*T} Q x^* \tag{4.30}$$

$$=: \nu_{D_G}(x^*, y^*) \tag{4.31}$$

Otherwise if $Qx^* + c \neq 0$, then $y^* = \begin{bmatrix} y_I \\ 0 \end{bmatrix}$ where y_I is obtained from the equation (4.27). In this case let's start by looking at the dual objective value:

$$\nu_{D_G}(x^*, y^*) := b_I^T y_I^* - \frac{1}{2} x^{*T} Q x^* \overset{eq(4.22)}{=} (A_I x^*)^T y_I^* - \frac{1}{2} x^{*T} Q x^* \tag{4.32}$$

$$= x^{*T}((y_I{}^*)^T A_I) - \frac{1}{2} x^{*T} Q x^* \tag{4.33}$$

$$\overset{eq(4.27)}{=} x^{*T}(Qx^* + c) - \frac{1}{2} x^{*T} Q x^* \tag{4.34}$$

$$= c^T x^* + \frac{1}{2} x^{*T} Q x^* \tag{4.35}$$

$$=: \nu_{P_G}(x^*) \tag{4.36}$$

□

So the primal and dual objective functions has the same values, this concludes that strong duality for the pair of solutions x^* and (x^*, y^*) holds. Moreover, based on the weak duality theorem, the feasible (x^*, y^*) solution for D_G can not be any thing but an optimal solution.

In the following section, the strong duality property will be exploited to drive the necessary and sufficient optimality condition for a quadratic convex programing problem in which the primal feasibility set is bounded.

4.2 The KKT condition for Convex Quadratic Programing

As a contribution of solving convex quadratic programing problem, we are going to experiment our results on minimum cost flow problems with quadratic costs. Due to this fact, for the rest of this thesis, we are interested in defining the following pair of problems as primal P and dual D problems, respectively.

$$
\begin{aligned}
P : \min_{x,t} \quad & c^T x + \frac{1}{2} x^T Q x && && (4.37)\\
s.t \quad & Ax && = b, && (4.38)\\
& -x - t && = -u, && (4.39)\\
& x \geq 0, && && (4.40)\\
& t \geq 0. && && (4.41)
\end{aligned}
$$

$$
\begin{aligned}
D : \max_{\widehat{x},y,w,s_1,s_2} \quad & b^T y - \frac{1}{2} \widehat{x}^T Q \widehat{x} - u^T w && && (4.42)\\
s.t \quad & A^T y - w - Q\widehat{x} + s_1 && = c, && (4.43)\\
& -w + s_2 && = 0, && (4.44)\\
& s_1 \geq 0,\ s_2 \geq 0, && && (4.45)\\
& \widehat{x}, w, y \quad free. && && (4.46)
\end{aligned}
$$

where $t \in \mathbb{R}^n$, $w \in \mathbb{R}^n$ and $s_1, s_2 \in \mathbb{R}^n_+$. It can be easily seen that these pair of problems can be obtained from the more general forms P_G and D_G, hence the strong duality property also holds for them, i.e. the duality gap at the optimality is equal to zero. More precisely by computing the duality gap, Δ, between the values of the primal objective of P at primal feasible solution (x, t) and the dual objective of D at the dual feasible solution $(\widehat{x}, y, w, s)$, one may have:

$$\Delta = c^T x + \frac{1}{2} x^T Q x - \left[b^T y - \frac{1}{2} \widehat{x}^T Q \widehat{x} - u^T w \right] \tag{4.47}$$

$$\overset{eq(4.43)}{=} \left[A^T y - w - Q\widehat{x} + s_1 \right] x - b^T y + u^T w + \frac{1}{2} x^T Q x + \frac{1}{2} \widehat{x}^T Q \widehat{x} \tag{4.48}$$

$$= y^T (Ax - b) - w (x - u) + s_1 x + \frac{1}{2} (x - \widehat{x})^T Q (x - \widehat{x}) \tag{4.49}$$

$$\tag{4.50}$$

Keeping in mind that the statement $y^T (Ax - b) = 0$ is satisfied by free because of the feasibility of (x, t), we get

$$\Delta \overset{eq(4.39,4.44)}{=} s_2 t + s_1 x + \frac{1}{2} (x - \widehat{x})^T Q (x - \widehat{x}) . \tag{4.51}$$

By taking into account the resulting expression for Δ, one may notice that all the terms there, are non-negative due to the primal and dual feasibility of (x, t) and $(\widehat{x}, y, w, s)$ and also the semidefinite of matrix Q. This means that $\Delta = 0$ if and only if each non-negative term in the expression for Δ is equal to zero. Moreover, according to the factorization of $Q := VE^2V^T$ with the assumption that E^2 is a positive definite matrix ,the term $(x - \widehat{x})^T Q (x - \widehat{x})$ being equal to zero becomes equivalent to $V^T x = V^T \widehat{x}$ as seen below:

$$(x - \widehat{x})^T Q (x - \widehat{x}) = 0 \tag{4.52}$$

$$(x - \widehat{x})^T VE^2V^T (x - \widehat{x}) = 0 \tag{4.53}$$

$$(x - \widehat{x})^T V = 0 \tag{4.54}$$

$$\tag{4.55}$$

To summarize, for optimization problem P with differentiable objective and constraint function for which the strong duality obtains, any pair of primal and dual optimal points must satisfy the following system of equations which are called *Karush-Kuhn-Tucker* (KKT) condition.

$$Ax = b, \tag{4.56}$$

$$-x - t = -u, \tag{4.57}$$

$$A^T y - w - Q\widehat{x} + s_1 = c, \tag{4.58}$$

$$-w + s_2 = 0, \tag{4.59}$$

$$XS_1 e = 0, \tag{4.60}$$

$$TS_2 e = 0, \tag{4.61}$$

$$V^T x - V^T \widehat{x} = 0. \tag{4.62}$$

Where $X = diag(x)$, $T = diag(t)$, $S_1 = diag(s_1)$ and $S_2 = diag(s_2)$. It should be noted that usually in the literatures related to this topic, a less general

form of KKT system are used, in particular where the constraint $V^T x = V^T \widehat{x}$ is replaced with

$$x = \widehat{x} \tag{4.63}$$

If one is always going to enforce the constraint $x = \widehat{x}$ then there is no real reason rather than clarity of exposition to explicitly distinguish between x and $\widehat{x}$.

For the sake of simplicity in the notations of the vectors and parameters which are involved in $(4.37) - (4.41)$ and $(4.42) - (4.46)$, through the rest of our discussion, let's assume the following re-expression of them:

$$x^{'} = \begin{bmatrix} x \\ t \end{bmatrix}, \widehat{x}^{'} = \begin{bmatrix} \widehat{x} \\ 0 \end{bmatrix}, \quad y^{'} = \begin{bmatrix} y \\ w \end{bmatrix}, \quad s^{'} = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}, \quad b^{'} = \begin{bmatrix} b \\ u \end{bmatrix} \tag{4.64}$$

$$Q^{'} = \begin{bmatrix} Q & 0 \\ 0 & 0 \end{bmatrix} \quad , A^{'} = \begin{bmatrix} A & 0 \\ -I & -I \end{bmatrix}, V^{'} = \begin{bmatrix} V & 0 \\ 0 & 0 \end{bmatrix}. \tag{4.65}$$

Based on the above re-expression of the variables and parameters, the corresponding general KKT linear system describing the necessary and sufficient optimality condition for the convex quadratic programming problem is as follows:

$$A^{'} x^{'} = b^{'}, \tag{4.66}$$

$$A^{'T} y^{'} - Q^{'} \widehat{x}^{'} + s^{'} = c^{'}, \tag{4.67}$$

$$V^{'} x^{'} - V^{'} \widehat{x}^{'} = 0, \tag{4.68}$$

$$X^{'} S^{'} e = 0, \tag{4.69}$$

$$x^{'}, s^{'} \geq 0. \tag{4.70}$$

where the equations (4.66), (4.67) and (4.69) will be referred to as the primal feasibility, the dual feasibility and the complementary slackness condition, respectively. In the remaining part of our discussion, for the sake of simplicity, we will drop the prim symbol of all the involved variables. Moreover, let us re-state the above KKT system by introducing an additional variable $z := -E^2 V^T \widehat{x} \in \mathbb{R}^l$. According to this definition, the substituting of term $-Q^{'} \widehat{x}^{'}$ by Vz in (4.67) and restating the equation (4.68) result in the following two update conditions in the KKT system:

$$A^T y + s + Vz = c, \tag{4.71}$$

$$EV^T x + E^{-1} z = 0 \tag{4.72}$$

Now, let's define a linear function $\varphi : \mathbb{R}^{m+2n+l} \to \mathbb{R}^{m+n+l}$ as:

$$\varphi := \begin{bmatrix} Ax - b \\ (A^T y + s + Vz) - c \\ EV^T x + E^{-1} z \end{bmatrix} \tag{4.73}$$

4.3 Primal-Dual Path Following IP Method for CQP

IPMs for solving the linear system (4.66)-(4.70) are based on introducing a barrier parameter μ to perturb the complementary slackness feasibility condition (4.69):

$$XSe = \mu e, \tag{4.74}$$

where the logarithmic barrier parameter μ reduce gradually. This barrier parameter μ forces keeping the solution x in the interior of the positive orthant that is preventing any of the components x_j from approaching their boundary value of zero. Thereafter, the algorithm proceeds by gradually reducing the logarithmic barrier parameter which means minimizing the original objective and approaching the optimal solution in which some of the component x_j might take zero value. A system of equations (4.66), (4.71), (4.72) and (4.74) is referred to as *slackened KKT* system. Let's introduce $\phi : \mathbb{R}^{m+2n+l} \to \mathbb{R}^{m+2n+l}$ as a function corresponds to the slackened KKT system.

$$\phi := \begin{bmatrix} \varphi \\ \mu e - XSe \end{bmatrix} \tag{4.75}$$

Therefore, the system of equation to be considered in IPMs for CQP can described as:

$$\phi(x, y, s, z) = 0 \tag{4.76}$$

which for any $\mu > 0$, has a unique solution $(x(\mu), y(\mu), s(\mu), z(\mu))$ referred to as μ-centre solution. A family of these μ-centre solution provide a (continuos) primal- dual *central path* i.e. $\{x(\mu), y(\mu), s(\mu), z(\mu) : \mu \geq 0\}$. The clever choice of a suitable sequence of μ is what the IPMs focuses on and one may notice that the practical efficiency, the implementation and the theoretical worst-case complexity of the interior point methods depends strongly on the way of handling the perturbed complementary slackness condition. At the early steps of the optimization process assigning the large values to μ enforce the *centrality* concept that prevents the components of X and S from approaching the zero boundary. While progressing the reduction of μ makes a path from centrality to optimality which may be at the boundary of the feasible region. The convergence to the optimality in the interior point methods are guaranteed by gradually reducing the barrier parameter μ. Usually, IPMs terminate when an ϵ-accurate solution ($\mu \leq \epsilon$) is found.

The best known IPM algorithms find the ϵ-accurate optimal solutions to the problem with n variables in $\mathrm{O}(\sqrt{n}log(\frac{1}{\epsilon}))$ iterations or $\mathrm{O}(nlog(\frac{1}{\epsilon}))$ iteration, depending on how aggressive steps to optimality are allowed. Computational experience provides evidence that the algorithm which uses a more aggressive strategy (the so-called long-step method) solves linear and quadratic

programming problems in a number of iterations which may be expressed as $O(log(n)log(\frac{1}{\epsilon}))$ [131].

It is worth mentioning that, those IPMs which are using a self-concordant barrier function such as the logarithmic barrier has the best complexity term $\sqrt{n}$, referring to the general theory in Ch. 4 of [132].

The interesting issue is that practically IPMs performs even much better than their theoretical analysis and in many of the applications they converge to the optimality in a constant number of iterations which is independent of the dimension size of the problem [133].

IPMs pursue by computing *Newton* direction and making *one* step in this direction before reducing the barrier parameter μ.

Let's w denote the quadruple (x, y, s, z) and for a given w, let's define (r_p, r_d, r_{cs}, r_v) as follows:

$$\mu := \mu(w) = x^T s/n, \tag{4.77}$$

$$r_p := r_p(w) = Ax - b, \tag{4.78}$$

$$r_d := r_d(w) = A^T y + s + Vz - c, \tag{4.79}$$

$$r_{cs} := r_{cs}(w) = XSe - \sigma\mu e \tag{4.80}$$

$$r_v := r_v(w) = EV^T x + E^{-1} z, \tag{4.81}$$

$$r := r(w) = (r_p(w), r_d(w), r_{cs}(w), r_v(w)) \tag{4.82}$$

In the *kth* iteration of the *exact* IPMs, for given μ_k, the Newton direction $(\triangle x, \triangle y, \triangle s, \triangle z)$ is derived via the solution of the following system of equations:

$$\phi^{'}(x_k, y_k, s_k, z_k) \begin{bmatrix} \triangle x \\ \triangle y \\ \triangle s \\ \triangle z \end{bmatrix} = - \begin{bmatrix} r_p \\ r_d \\ r_{cs} \\ r_v \end{bmatrix}, \tag{4.83}$$

where $\phi^{'}$ is a Jacobian matrix of ϕ.

4.3.1 Primal-Dual Feasible Path Following IP

For the *feasible* interior point algorithm, it will be assumed that all the primal-dual iterates belongs to set $\mathcal{F}^0 = \{(x, y, s, z) \,|Ax = b, A^T y + s + Vz = c, (x, s) > 0\}$ i.e. $(r_p, r_d) = 0$ and the convergence is guaranteed by swift control of the complementarity product $x^T s$ and maintaining the second order term $\triangle x^T \triangle s$ small enough. This is achieved by being enough close to the central path, namely by keeping the the error in the complementarity slackness equation $XSe = \mu e$ small enough. In this context, the Newton step is taken in a way that the next point stays in specific neighborhood. The neighborhood conception should be defined to guarantee the (x, s) being strictly positive. In [112] the following neighborhood is defined to which the points at each iteration is belonging :

$$N_2(\theta) = \{(x, y, s, z) \in \mathcal{F}^0 | \|XSe - \mu e\| \leq \theta\mu\} \tag{4.84}$$

where $\theta \in (0,1)$ is an arbitrary parameter and the barrier term is equal to the average complementarity product $\mu = x^T s/n$. Based on this definition, the neighborhood of the central path becomes smaller as long as the central path approaches an optimal solution. This neighborhood definition is not preventing the $x_i s_i$ from being too small and actually small fraction point in $\mathcal{F}^0$ is contained in the N_2 no matter how small θ is selected.

Various definitions of the neighborhood to stay close to the central path inspired alternative IPMs as *path-following* methods. Indeed, all these algorithms follow the central path on their way to optimality [115].

IPMs operating in the $N_2(\theta)$ neighborhood is known in the literature as the *short-step* path-following algorithm. In feasible primal dual path following algorithm, all iterates belong to $N_2(\theta)$ of the central path, i.e. all iterates are strictly primal-dual feasible which implies that the right hand side vector in the linear system (4.83) has the form $(0, 0, 0, \mu e - XSe)$. In [101] it has been proved that the short-step path-following algorithm applied to CQP converges to an ϵ-accurate solution in $\mathcal{O}(\sqrt{n}ln(1/\epsilon))$ iterations.

4.3.2 Primal-Dual Infeasible Path Following IP

In practice there is no need to force the the feasibility of all iterates and to impose the IPM algorithm to stay in the primal-dual feasible set $\mathcal{F}^0$. For the *primal dual infeasible path following* (PDIPF) algorithm each of the right hand side term of Newton direction system (4.83) can be nonzero, i.e. $(r_p, r_d, r_{cs}, r_v) \neq 0$. In this algorithm by making a step of size α then the primal and dual infeasibility will be reduced $(1-\alpha)$ times. Indeed, since all the equations in $\varphi(x, y, s, z)$ are linear, it is expected that the residuals are reduced progressively. In particular, if PDIPF algorithm reaches a feasible step, then all the infeasibility are vanished for remaining iterations. In practice, as noted in [133], usually the primal dual feasibility is reached before the optimality criteria is achieved. One may notice that in PDIPF algorithms, it is required to choose the step size α significantly smaller than 1 that results in a damped Newton direction $[[112], Ch.6]$.

In literature [125], [134] and [135] a system in which $r_v = 0$ is considered. Based on this assumption and by applying systematic Gaussian elimination procedure to this system, one may drive a *reduced Newton* system in which $\triangle s$ is obtained as a term of $\triangle x$ from complementary slackness equation and replacing it in other equations. By proceeding the Gaussian elimination, a *normal* equation system with off-diagonal fill in matrix is obtained.

The most significant development between 1994 and 1998 was that $LOQO$ software package, described in [125], could solve linear and quadratic programming problems by implementing an infeasible primal dual IPM.

4.4 Iterative Inexact PDIPF IP

In each iteration of IPMs a linear system (4.83) has to be solved which is the main computation effort of IP iteration. Therefore, in recent decades exten-

sive researches have been devoted to develop techniques for solving this linear system that becomes extremely ill-conditioned as the IPM approaches its solution. This fact leads to numerical instability and motivates the researchers for finding alternative techniques based on a family of search directions which approximate the solution of the system (4.83). Considering these approaches, it has been claimed in [136] that the iterative approaches without *preconditioning* might fail to obtain the solution. One by the term preconditioning means transforming a linear system into another system with more favorable properties for iterative methods. Therefore, introducing a good preconditioner is significant ingredient for solving a linear system iteratively that causes impressive number of researches focusing on this issue [137, 138, 139].

In iterative PDIPF methods Newton direction is computed approximately rather than accurately. Therefore, instead of a pure Newton iteration (4.83), it consider the following system to be solved at inner kth iteration:

$$\phi^{'}(w^k) \triangle w_k = -\phi(w_k) + \epsilon_k, \tag{4.85}$$

where $\epsilon := (\epsilon_p, \epsilon_d, \epsilon_{cs}, \epsilon_v)$ is a vector of describing the inexactness of equations. In [140], they assumed that $\|\epsilon_k\|/\|\phi(w_k)\| \leq \eta_k$ and they showed that when the forcing sequence η_k is uniformly less than one then the inexact Newton method is locally convergent and the order of the convergence is derived in terms of the rate of the convergence of the relative residuals, i.e, $\|\epsilon_k\|/\|\phi(w_k)\|$. In [101], the inexact *feasible* IPM is applied for CQP where it has been assumed that $\epsilon_p = \epsilon_d$ and also the worst case algorithm complexity is discussed there.

An infeasible inexact potential reduction method for CQP is described in [141] focuses on generic KKT conditions in which $x = \widehat{x}$. In this approach the modified system of equations (4.85) is solved in a way that only complementary slackness equation is solved inexactly, i.e., $\epsilon_p = \epsilon_d = \epsilon_v = 0$. They also discussed the convergence attitude of this method. Later on, We are going to review two inexact PDIPF algorithm proposed in [1] and [2] for CQP and general constraint system of equations, respectively.

4.4.1 Inexact PDIPF IP algorithm [1]

In inexact PDIPF algorithm [1], it is assumed that in system of equation (4.85), $\epsilon_p = \epsilon_d = 0$ the inexact search direction satisfy (4.85) and ϵ_{cs} and ϵ_v are relatively small based on the following definition.

Definition 3. *Given a point $w \in \mathbb{R}^{5n}_{++} \times \mathbb{R}^m$ and positive scalars $\tau_{\epsilon_{cs}}$ and τ_{ϵ_v}, an inexact direction $\triangle w$ is referred to as a $(\tau_{\epsilon_{cs}}, \tau_{\epsilon_v})$-search direction if it satisfies* (4.85) *for some ϵ_{cs} and ϵ_v satisfying $\|\epsilon_{cs}\|_\infty \leq \tau_{\epsilon_{cs}}\mu$ and $\|\epsilon_v\| \leq \tau_{\epsilon_v}\sqrt{\mu}$.*

In inexact PDIFP algorithms, one extra condition should be guaranteed which is reducing r_p, r_d and r_v at each iteration. This can be controlled by defining a generalized neighborhood. For a given parameters $\eta \geq 0$, $\gamma \in [0, 1]$ and $\theta > 0$, define the following set:

$$N_{w^0}(\eta,\gamma,\theta) = \{w \in \Re^{5n}{}_{++} \times \Re^m : Xs \geq (1-\gamma)\,\mu e \quad , (r_p, r_d) = \eta({r_p}^0, {r_d}^0), \\ \|r_v - \eta r_v^0\| \leq \theta\sqrt{\mu}, \quad \eta \leq \mu/\mu_0\}$$

then, the generalized central path neighborhood was given by

$$N_{w^0}(\gamma,\theta) = \cup_{\eta[0,1]} N_{w^0}(\eta,\gamma,\theta) \tag{4.86}$$

Based on the above definition of the generalized neighbourhood, following is the inexact Primal Dual Path Following algorithm as stated in [1]:

1. **Start:** Let $\epsilon \geq 0$ and $0 < \underline{\sigma} \leq \overline{\sigma} < 4/5$ be given. Let $\gamma \in (0,1)\,, \theta > 0$ and $w^0 \in N_{w^0}(\gamma,\theta)$. Set $k = 0$.
2. **While** $\mu_k := \mu(w^k) > \epsilon$ **do**
   (a) Let $w := w^k$ and $\mu := \mu_k$; choose $\sigma := \sigma_k \in [\underline{\sigma}\,, \overline{\sigma}]$.
   (b) Set

$$\tau_{\epsilon_{cs}} = \sigma\gamma/4 \tag{4.87}$$

$$\tau_{\epsilon_v} = \left(\sqrt{1 + (1 - 0.5\gamma)\,\sigma} - 1\right)\theta \tag{4.88}$$

   (c) Set $r_p = b - Ax$, $r_d = A^T y + s + Vz - c$, $r_v := EV^T x + E^{-1}z$, and $\eta = \| r_p \| / \| {r_p}^0 \|$.
   (d) compute a $(\tau_{\epsilon_{cs}}, \tau_{\epsilon_v})$ - search direction $\triangle w$.
   (e) compute $\tilde{\alpha} = argmax\left\{\alpha \in [0\ 1] : w + \alpha \triangle w \in N_{w^0}(\gamma,\theta), \forall \alpha^{'} \in [0,\alpha]\right\}$
   (d) compute $\overline{\alpha} = argmin\left\{(x + \alpha \triangle x)^T (s + \alpha \triangle s) : \alpha \in [0,\tilde{\alpha}]\right\}$
   (e) Let $w^{k+1} = w + \overline{\alpha} \triangle w$ and set $k \longleftarrow k + 1$.

- **End**(while)

The following results gives the bound on the number of iterations needed by the inexact PDIPF algorithm to obtain ϵ-solution to the KKT system (4.66) - (4.68) .

Theorem 20. *Assume that the constants $\gamma, \underline{\sigma}, \overline{\sigma}$ and θ are such that :*

$$\max\left\{\gamma^{-1}, (1-\gamma)^{-1}, \underline{\sigma}^{-1}, (1 - 5/4\overline{\sigma})^{-1}\right\} = \mathcal{O}(1)\,, \theta = \mathcal{O}\left(\sqrt{n}\right) \tag{4.89}$$

and that the initial point $w^0 \in \Re^{2n}_{++} \times \Re^{m+l}$ satisfies $\left(x^0, s^0\right) \geq (x^*, s^*)$ for some feasible vector w^*. Then, the inexact PDIF algorithm finds an iterate w^k satisfies $\mu_k \leq \epsilon\mu_0$ and $\| \left({r_p}^k, {r_d}^k\right) \| \leq \epsilon \| {r_p}^0, {r_d}^0 \|$ and $\| r_V^k \| \leq \epsilon \| {r_v}^0 \| + \epsilon^{1/2}\theta {\mu_0}^{1/2}$ within $\mathcal{O}\left(n^2 log(1/\epsilon)\right)$ iteration.

Proof. [1] □

4.4.2 Inexact IP algorithm [2]

In [2], a global inexact interior point method for a general nonlinear system is proposed. Here, as follows, we modify the iterative step described there, according to our linear system of equation (4.75).

Step 0. Given data: $w_k = (x_k, y_k, s_k, z)$, $\overline{\tau}_1 = \min(X_0 S_0 e)/\left[x_0^T s_0/n\right]$, $\overline{\tau}_2 = x_0^T s_0/\|F(w_0)\|$, $\gamma_{k-1} \in \left[\frac{1}{2}, 1\right)$, and $\eta_{max}, \beta, \theta \in (0,1)$.

Step 1. Choose σ_k and $\widehat{\eta}_k$ such that $(\sigma_k + \widehat{\eta}_k) \in (0, \eta_{max})$, $\gamma_k \in \left[\frac{1}{2}, \sigma_{k-1}\right]$ and put $\mu_k = \sigma_k (x_k s_k)/n$ and $\tilde{\eta}_k = \sigma_k + \widehat{\eta}_k$.

Step 2. Solve the linear system

$$\phi^{'}(x_k, y_k, s_k, z_k) \begin{bmatrix} \triangle x \\ \triangle y \\ \triangle s \\ \triangle z \end{bmatrix} = -\phi(x_k, y_k, s_k, z_k) + \epsilon_k \tag{4.90}$$

where $\|\epsilon_k\| \leq \widehat{\eta}_k (x_k s_k)/n$.

Step 3. Choose $\overline{\alpha}_1^k$ such that

$$min\left(X_k\left(\overline{\alpha}_1^k\right) S_k\left(\overline{\alpha}_1^k\right)\widehat{e}\right) \geq \overline{\tau}_1 \gamma_k \left(x_k\left(\overline{\alpha}_1^k\right)^T s_k\left(\overline{\alpha}_1^k\right)\right)/n. \tag{4.91}$$

Step 4. Choose $\overline{\alpha}_2^k$ such that

$$x_k\left(\overline{\alpha}_2^k\right)^T s_k\left(\overline{\alpha}_2^k\right) \geq \overline{\tau}_1 \gamma_k \|F(w\left(\overline{\alpha}_2^k\right)\|. \tag{4.92}$$

Step 5. Let $\tilde{\alpha}_k = min\left(\overline{\alpha}_1^k, \overline{\alpha}_2^k\right)$.

Step 6. Let $p_k = \tilde{\alpha}_k \triangle w$, $\eta_k = 1 - \tilde{\alpha}_k (1 - \tilde{\eta}_k)$

Step 7. While

$\|H(w_k + p_k)\| > (1 - \beta(1 - \eta_k))\|H(w_k)\|$,
set $p_k = \theta p_k$ and $\eta_k = 1 - \theta(1 - \eta_k)$.

Step 8. Set $w_k = w_k + p_k$. Return to Step 0.

Convergence Analysis of Inexact IP [2] Here, we are going to discuss the convergence properties of the Inexact IP [2] algorithm exploited for solving (4.76). We assume the following definition of the neighborhood for the given $\epsilon \geq 0$ as stated in [2]:

$$\mathcal{N}(\varepsilon) = \{w = (x, y, s, z) | \varepsilon \leq \|\phi(w)\| \leq \|\phi(w_0)\|, \\ min(XSe) \geq (\overline{\tau}_1/2) s^T x/n, s^T x \geq (\overline{\tau}_2/2)\|\varphi(w)\|\}$$

Under the following assumptions, we will prove that $\|\phi(w_k)\| \longrightarrow 0$ and any limit point of $\{w_k\}$ is a solution of problem (4.76) provided that σ_k and $\widehat{\eta}_k$ are chosen to satisfy

$$\sigma_k > max\left[(\sqrt{n}+\overline{\tau}_1\gamma_k)/(\sqrt{(}n)(1-\overline{\tau}_1\gamma_k), (\overline{\tau}_2\gamma_k+\sqrt{n})/n\right]\widehat{\eta}_k \tag{4.93}$$

(A1) The function ϕ is continuously differentiable in $\mathcal{N}(0)$.

(A2) The iteration sequence $\{w_k\}$ is bounded.

(A3) The Jacobian matrix $\phi^{'}(w)$ is nonsingular in $\mathcal{N}(\epsilon)$, $\epsilon > 0$

(A4) $\varphi^{'}$ is Lipschitz continuous in $\mathcal{N}(0)$ with constant L.

It can be checked easily that assumptions $(A1), (A2)$ and $(A4)$ hold because of, respectively, the linearity of the function ϕ , the boundedness of the iterations sequence w_k and the $\varphi^{'}$ being constant. On the other hand, the assumption $(A3)$ might not be hold always because it can be checked easily that the non singularity of $\phi^{'}$ depends directly on the non singularity of the full rankness of matrix A. To this purpose, we first redeclare here the fundamental theorem 3.1 stated in [2].

Theorem 21. *If w^* is a limit point of $\{w_k\}$ such that $\phi^{'}(w^*)$ is non singular, then the sequence $\{w_k\}$ generated by Inexact IP algorithm converge to w^*.*

Proof. [2], p:114. □

Proposition 4. *The jacobian matrix $\phi^{'} \in \mathbb{R}^{m+2n+l} \times \mathbb{R}^{m+2n+l}$ is nonsingular if and only if the matrix A is full rank.*

Proof.

$$\phi' := \begin{bmatrix} A & & & & \\ & A^T & I & V \\ S & & X & \\ EV^T & & & E^{-1} \end{bmatrix} \tag{4.94}$$

To proof this we just need to apply a sequance of Gaussien elimination operators on the jacobbian matrix $\phi^{'}$. To this puropose, let's start by multiplying the term $-VE$ to the last row of $\phi^{'}$ and adding it to its second row. This modifyies the jacobian matrix as follows:

$$\begin{bmatrix} A & & & \\ -Q & A^T & I & \\ S & & X & \\ EV^T & & & E^{-1} \end{bmatrix} \tag{4.95}$$

Thereafter, as the second step, let us multiply the term $-X^{-1}$ to the third row of (4.95) and adding it to its second row. This obtains

$$\begin{bmatrix} A & & & \\ -(Q+X^{-1}S) & A^T & & \\ S & & X & \\ EV^T & & & E^{-1} \end{bmatrix} \tag{4.96}$$

One may notice that matrix (4.96) is nonsingular if and only if the block $\begin{bmatrix} -(Q+X^{-1}S) & A^T \\ A & \end{bmatrix}$ is nonsingular. According to Schur complement property (4.4.2) is nonsingular if and only if $A(Q+X^{-1}S)^{-1}A^T$ is nonsingular. Since Q and $X^{-1}S$ are positive semidefinte and positive definte matrices, respectively, so $(Q+X^{-1}S) \succ 0$. Hence $A(Q+X^{-1}S)^{-1}A^T$ is nonsingular if and only if A is a full rank matrix. Therefore, one may conclude that the jacobian matrix $\phi^{'}$ is nonsingular if and only if the matrix A is full rank. □

Chapter 5

Convex Quadratic Programming, Iterative Path-Following IP approaches

This chapter is dedicated to discuss and analyze approaches for solving convex quadratic programming (CQP) problems by iterative primal dual infeasible path following (PDIPF) methods. Our main purpose will be considering possible methods for solving inexactly the following linear system of Newton directions raised in each iteration of iterative PDIPF.

$$A \triangle x = -r_p + \epsilon_p, \tag{5.1}$$
$$A^T \triangle y - Q \triangle x + \triangle s = -r_d + \epsilon_d, \tag{5.2}$$
$$X \triangle s + S \triangle x = -r_{cs} + \epsilon_{cs}, \tag{5.3}$$

where

$$\mu := x^T s/n, \tag{5.4}$$
$$r_p := Ax - b, \tag{5.5}$$
$$r_d := A^T y + s - Qx - c, \tag{5.6}$$
$$r_{cs} := XSe - \sigma\mu e \tag{5.7}$$
$$\tag{5.8}$$

obtained simplified by (4.85) with the assumption that $x = \widehat{x}$.

The final contribution of this chapter will be the proposal of two methods for redistributing the error generated by inexact linear solvers such that the iterative PDIPF IP algorithm convergence is guaranteed.

5.1 Agumented System

Inexact Augmented system can be obtained from generic system of Newton direction in various form depends on assuming whether the complementary slackness equation (5.3) is solved exactly or not, i.e, whether ϵ_{cs} has value zero or not. Let us first consider the case where $\epsilon_{cs} = 0$, by following the Gaussian elimination procedure as described in Proposition (4), our interested Newton directions will be modified as the following inexact augmented system:

$$\begin{bmatrix} D^{-1} & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \triangle x \\ \triangle y \end{bmatrix} = \begin{bmatrix} -r_d + X^{-1} r_{cs} \\ -r_p \end{bmatrix} + \begin{bmatrix} h_y \\ h_x \end{bmatrix} \tag{5.9}$$

where $D^{-1} := -(X^{-1}S + Q)$ and h_x, h_y are errors generated by solving the augmented system inexactly. It can be checked easily that whatever h_x and h_y will be, they transform exactly to the tolerance of the generic Newton direction system, more precisely:

$$\begin{bmatrix} \epsilon_p \\ \epsilon_d \end{bmatrix} = \begin{bmatrix} h_x \\ h_y \end{bmatrix} \tag{5.10}$$

In addition to this, as a second case, one might consider computing $\triangle s$ from perturbed complementary slackness condition (5.3) with ϵ_{cs} as a perturbation parameter:

$$\begin{aligned} \triangle s\,(\epsilon_{cs}) &= X^{-1}(-r_{cs} - S \triangle x + \epsilon_{cs}) && (5.11) \\ &= X^{-1}(-r_{cs} - S \triangle x) + X^{-1}\epsilon_{cs} && (5.12) \end{aligned}$$

Then substituting (5.12) in (5.2) implies that:

$$\begin{aligned} A^T \triangle y + \triangle s - Q \triangle x &= -r_d + \epsilon_d && (5.13) \\ A^T \triangle y + \left[-X^{-1} r_{cs} - X^{-1} S \triangle x + X^{-1}\epsilon_{cs}\right] - Q \triangle x &= -r_d + \epsilon_d && (5.14) \end{aligned}$$

$$A^T \triangle y - \left[X^{-1}S + Q\right] \triangle x = -r_d + X^{-1} r_{cs} + \epsilon_d - X^{-1}\epsilon_{cs} \tag{5.15}$$

Hence, it shows that by solving the augmented system inexactly with generated error h_x and h_y, the error distribute to the perturbation errors ϵ_p and ϵ_d of equations (5.1) and (5.2) in generic Newton direction system as follows:

$$\begin{bmatrix} \epsilon_p \\ \epsilon_d \end{bmatrix} = \begin{bmatrix} h_x \\ h_y + X^{-1}\epsilon_{cs} \end{bmatrix} \tag{5.16}$$

By considering (5.16), one may notice that ϵ_{cs} can be chosen freely to balance the error of equation (5.2) and complementary slackness constraint (5.3).

The best option for choosing ϵ_{cs} will be

By choosing norm two, i.e. $\|.\|_2$, the ϵ_{cs} that minimizes the total error is given as follows:

$$(\epsilon_{cs})_i = \frac{h_{y_i} X_{ii}^{-1}}{1 + {X_{ii}}^{-2}} \quad \forall i = 1, \cdots, n. \tag{5.17}$$

5.2 Normal Equation System

In this section, we are going to consider *Normal equation* system obtained from applying a Gaussian procedure to the Augmented system. There, we also compare the advantage of solving each of these system above the other.

Inexact normal equation system is obtained by computing the $\triangle x$ from (5.2) and (5.3) where $\epsilon_{cs} = 0$, in terms of $-X^{-1} \triangle y$:

$$\triangle x = D\left(-r_d + X^{-1} r_{cs}\right) - DA^T \triangle y + D\epsilon_d, \tag{5.18}$$

And replacing it in the second set of equation of (5.9),

$$ADA^T \triangle y = AD\left(-r_d + X^{-1} r_{cs}\right) + r_p + AD\epsilon_d - \epsilon_p \tag{5.19}$$

Assuming that $\epsilon_d = 0$, the error in (5.19) becomes equal to ϵ_p. This means that the termination of solving inexactly *normal equation* system depends only on ϵ_p.

As some advantages of working with augmented system one may state the following features. First, the augmented system is sparse compared with normal equation system, more precisely if the constraint matrix A has a dense column this results straightforward in the dense normal equation system. The second reason is about the ill-conditioning which is due to matrix D. This happens because some of its elements moves toward zero and the others move toward infinity. Taking into account the position of D in the augmented system makes it easier to control the ill-conditioning of the augmented system when designing a preconditioner.

On the other hand, there are some reasons that motivate us solving normal equation system at each iteration of IPDIPF method. The first is that, normal equation system is symmetric and positive semi-definite, secondly its dimension is smaller compared to the augmented system [136, 125]. The third reason comes from the fact that in normal equation system, the Cholesky decomposition for positive definite matrix D costs $\mathcal{O}(\frac{n^3}{6})$ which is faster than the possibly selected QR and LU factorization with the complexity cost $\mathcal{O}(\frac{2}{3}n^3)$ and $\mathcal{O}(\frac{n^3}{3})$, respectively.

As a more general case, one may assume that the complementary slackness equation (5.3) is also allowed to be solved inexactly. For this, let us obtain $\triangle x$ from the equation (5.15) in the the term of $\triangle y$ as

$$\left[X^{-1}S + Q\right] \triangle x = A^T \triangle y - \left[-r_d + X^{-1} r_{cs} + \epsilon_d - X^{-1}\epsilon_{cs}\right] \tag{5.20}$$

$$\triangle x = \left[X^{-1}S + Q\right]^{-1} A^T \triangle y - \left[X^{-1}S + Q\right]^{-1}\left(r_d - X^{-1} r_{cs}\right) - \left[X^{-1}S + Q\right]^{-1}\left(\epsilon_d - X^{-1}\epsilon_{cs}\right) \tag{5.21}$$

Thereafter, multiplying (5.21) by matrix A, results in the following re-written form of the augmented system:

$$A\big[X^{-1}S+Q\big]^{-1}A^T \triangle y = A\big[X^{-1}S+Q\big]^{-1}\left(r_d - X^{-1}r_{cs}\right) + \tag{5.22}$$

$$A\big[X^{-1}S+Q\big]^{-1}\left(\epsilon_d - X^{-1}\epsilon_{cs}\right) + \epsilon_p - r_p \tag{5.23}$$

The equation (5.23) can be expressed as:

$$A\big[X^{-1}S+Q\big]^{-1}A^T \triangle y = A\big[X^{-1}S+Q\big]^{-1}\left(r_d - X^{-1}r_{cs}\right) - r_p + \tilde{h}_y, \tag{5.24}$$

where $\tilde{h}_y$ depend on ϵ_p, ϵ_d and ϵ_{sx} and is defined as :

$$\tilde{h}_y := A\big[X^{-1}S+Q\big]^{-1}\left(\epsilon_d - X^{-1}\epsilon_{cs}\right) + \epsilon_p \tag{5.25}$$

and so on, ϵ_p can be described in terms of ϵ_d and ϵ_{cs} from (5.25) as below,

$$\epsilon_p := \tilde{h}_y - A\big[X^{-1}S+Q\big]^{-1}\epsilon_d + A\big[X^{-1}S+Q\big]^{-1}X^{-1}\epsilon_{cs} \tag{5.26}$$

For the sake of simplicity in the formulation, let's introduce matrices B and C as:

$$B := A\big[X^{-1}S+Q\big]^{-1} \tag{5.27}$$

$$C := BX^{-1} \tag{5.28}$$

Hence, one have $\epsilon_p := \tilde{h}_y - B\epsilon_d + C\epsilon_{cs}$.

At this point, if one desire to minimize the squared norm of error vector ϵ, then

$$\min \| [\epsilon_p, \epsilon_d, \epsilon_{cs}] \|^2 : = \min\left(\|\tilde{h_y} - B\epsilon_d + C\epsilon_{cs}\| + \epsilon_d^T\epsilon_d + \epsilon_{cs}^T\epsilon_{cs}\right) \tag{5.29}$$

$$:= E(\epsilon_d, \epsilon_{sx}) \tag{5.30}$$

Now by considering the derivation of function $E(\epsilon_d, \epsilon_{cs})$ with respect to ϵ_d and ϵ_{cs} one have:

$$\frac{1}{2}\frac{\partial E}{\partial \epsilon_d} = -B^T\left(\tilde{h_y} - B\epsilon_d + C\epsilon_{cs}\right) + \epsilon_d \tag{5.31}$$

$$= -B^T\tilde{h_y} + \left(B^TB + I\right)\epsilon_d - B^TC\epsilon_{cs}$$

$$= 0$$

$$\frac{1}{2}\frac{\partial E}{\partial \epsilon_{cs}} = C^T\left(\tilde{h_y} - B\epsilon_d + C\epsilon_{cs}\right) + \epsilon_{cs} \tag{5.32}$$

$$= C^T\tilde{h_y} - C^TB\epsilon_d + \left(C^TC + I\right)\epsilon_{cs}$$

$$= 0 \tag{5.33}$$

Now by obtaining ϵ_{cs} from (5.33) as below

$$-C^T\tilde{h_y} + C^TB\epsilon_d = (C^TC + I)\epsilon_{cs} \rightarrow \epsilon_{cs} = \tilde{C}(C^TB\epsilon_d - C^T\tilde{h_y}), \tag{5.34}$$

where $\tilde{C} := \left(I + C^TC\right)^{-1}$ and substituting (5.34) in (5.32), one may have

$$-B^T\tilde{h_y} + \left(B^TB + I\right)\epsilon_d - B^TC\tilde{C}(C^TB\epsilon_d - C^T\tilde{h_y}) = 0 \tag{5.35}$$

$$-B^T\tilde{h_y} + \left(B^TB + I\right)\epsilon_d - B^TC\tilde{C}C^TB\epsilon_d + B^TC\tilde{C}C^T\tilde{h_y} = 0 \tag{5.36}$$

Here, a simple factorization results in

$$\left(B^TB + I - B^TC\tilde{C}C^TB\right)\epsilon_d = \left(B^T - B^TC\tilde{C}C^T\right)\tilde{h_y} \tag{5.37}$$

From which ϵ_d can be described as

$$\epsilon_d = \tilde{\tilde{B}}B^T(I - C\tilde{C}C^T)\tilde{h_y}, \tag{5.38}$$

where $\tilde{\tilde{B}} := \left[B^TB + I - B^TC\tilde{C}C^TB\right]^{-1}$. Hence, we can rewrite ϵ_{cs} in terms of $\tilde{C}$ and $\tilde{\tilde{B}}$ as follows:

$$\epsilon_{cs} = \tilde{C}(C^TB\tilde{\tilde{B}}B^T(I - C\tilde{C}C^T)\tilde{h_y} - C^T\tilde{h_y}) \tag{5.39}$$

As a weak feature of this approach one may notice that the computations of the inverse matrices $\tilde{C}$ and $\tilde{\tilde{B}}$, required in deriving ϵ_d and ϵ_{cs} are as complex as solving Newton search direction system exactly.

To develop this algorithm, as an alternative, suppose that $\epsilon_{cs} = 0$ and let us have a more precise look at $\tilde{h}_y$,

$$\tilde{h}_y := A\left[X^{-1}S + Q\right]^{-1}\epsilon_d + \epsilon_p \tag{5.40}$$

From the above equation ϵ_p can be obtained in terms of ϵ_d and $\tilde{h}_y$ as below:

$$\epsilon_p = \tilde{h}_y - A\left[X^{-1}S + Q\right]^{-1}\epsilon_d \tag{5.41}$$

Therefore,

$$\begin{aligned}
\| (\epsilon_p, \epsilon_d) \|^2 &:= \left(\tilde{h}_y - A\left[X^{-1}S + Q\right]^{-1}\epsilon_d\right)^T \left(\tilde{h}_y - A\left[X^{-1}S + Q\right]^{-1}\epsilon_d\right) + \epsilon_d^T\epsilon_d \\
&= \tilde{h}_y^T\tilde{h}_y + \tilde{h}_y^T\left(-A\left[X^{-1}S + Q\right]^{-1}\epsilon_d\right) - \left(A\left[X^{-1}S + Q\right]^{-1}\epsilon_d\right)^T\tilde{h}_y \\
&+ \left(A\left[X^{-1}S + Q\right]^{-1}\epsilon_d\right)^T\left(A\left[X^{-1}S + Q\right]^{-1}\epsilon_d\right) + \epsilon_d^T\epsilon_d
\end{aligned}$$

Minimizing this norm also have high computational complexity due to the appearance of the term $\left(A\left[X^{-1}S + Q\right]^{-1}\epsilon_d\right)^T\left(A\left[X^{-1}S + Q\right]^{-1}\epsilon_d\right)$.

This fact inspired us to consider minimizing the infinity norm of error vector $(\epsilon_p, \epsilon_d, \epsilon_{sx})$ instead of norm two:

$$\min \|(\epsilon_p, \epsilon_d, \epsilon_{cs})\|_\infty \tag{5.42}$$

This minimization problem can be expanded as follows:

$$\min z \tag{5.43}$$

$$z \geq \left(\tilde{h}_y - A\big[X^{-1}S + Q\big]^{-1}\epsilon_d + A\big[X^{-1}S + Q\big]^{-1}X^{-1}\epsilon_{cs}\right)_i \tag{5.44}$$

$$z \geq (\epsilon_d)_j \tag{5.45}$$

$$z \geq (\epsilon_{cs})_k \tag{5.46}$$

For $\forall i = 1, 2, \cdots, m,\ \forall j, k = 1, 2, \cdots, n$. Supposing $\epsilon_d = \epsilon_{cs} := ze$ and substituting them into the (5.43), one may have:

$$\min z \tag{5.47}$$

$$e.z \geq \tilde{h}_y - A\big[X^{-1}S + Q\big]^{-1}e.z + A\big[X^{-1}S + Q\big]^{-1}X^{-1}e.z \tag{5.48}$$

$$\tag{5.49}$$

By a simple factorization, we derive our interested version which will be considered:

$\min z$

$$z \geq \frac{(\tilde{h}_y)_i}{1 + \left(A[X^{-1}S + Q]^{-1}e\right)_i - \left(A[X^{-1}S + Q]^{-1}X^{-1}e\right)_i} \quad \forall i = 1, 2, \cdots, m$$

Let z^* be the optimum solution of the above problem

$$z^* = \max_i \frac{(\tilde{h}_y)_i}{1 + \left(A[X^{-1}S + Q]^{-1}e\right)_i - \left(A[X^{-1}S + Q]^{-1}X^{-1}e\right)_i} \quad \forall i = 1, 2, \cdots, m \tag{5.50}$$

According to Step 2 of the inexact interior point algorithm described in [2], for the convergence guarantee the inexact linear solver continues iterating until the following condition hold:

$$\|\epsilon_k\|_\infty = \|\epsilon_p, \epsilon_d, \epsilon_{sx}\|_\infty \leq \widehat{\eta}_k \mu_k / \sigma_k. \tag{5.51}$$

Or, more precisely:

$$\max_i |(\epsilon_p)_i| \leq \widehat{\eta}_k \frac{\mu_k}{\sigma_k} \quad \forall i = 1, 2, \cdots, m \tag{5.52}$$

$$\max_j |(\epsilon_d)_j| \leq \widehat{\eta}_k \frac{\mu_k}{\sigma_k} \quad \forall j = 1, 2, \cdots, n \tag{5.53}$$

$$\max_k |(\epsilon_{cs})_k| \leq \widehat{\eta}_k \frac{\mu_k}{\sigma_k} \quad \forall k = 1, 2, \cdots, n \tag{5.54}$$

Let's define $\beta_i := A\big[X^{-1}S + Q\big]^{-1}e$ and $\Gamma_i := A\big[X^{-1}S + Q\big]^{-1}X^{-1}e$. Consider substituting the expressions of ϵ_p by (5.41) , ϵ_d and ϵ_{cs} by (5.50), respectively in (5.52), (5.53) and (5.54), one may conclude that for the the inexact linear solver iteration continues until the following conditions hold.

$$\max_i \frac{(\tilde{h}_y)_i}{1 + \beta_i - \Gamma_i} \leq \widehat{\eta}_k \frac{\mu_k}{\sigma_k} \quad \forall i = 1, 2, \cdots, m, \tag{5.55}$$

$$(\tilde{h}_y)_i - (\beta_i - \Gamma_i) \max_i \frac{(\tilde{h}_y)_i}{1 + \beta_i - \Gamma_i} \leq \widehat{\eta}_k \frac{\mu_k}{\sigma_k} \quad \forall i = 1, 2, \cdots, m. \tag{5.56}$$

The effectiveness of implying this approach considering the infinity norm, on the convergence of PDIPF IP algorithm requires more investigation.

Bibliography

[1] Z. Lu, R.D.S. Monteiro, and J.W. O'Neal. An iterative solver-based infeasible primal–dual path-following algorithm for convex quadratic programming, . *SIAM Journal on Optimization*, (17):287–310, 2006.

[2] S. Bellavia. Inexact Interior-Point Method. *Journal of optimization and applications*, 96(17):109–121, 1998.

[3] M. Conforti, G. Cornuéjols, and G. Zambelli. Extended Formulations in Combinatorial Optimization . *4OR*, 89(1):1–48, 2010.

[4] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Algorithms and Combinatorics 2. Springer, 1988.

[5] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 2000.

[6] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24. Alexander Schrijver. Combinatorial OptimizatAlgorithms and Combinator, 2003.

[7] A. Brøndsted. An Introduction to Convex Polytopes. *Graduate Texts in Mathematics*, 90, 1983.

[8] M. Grötschel, M. Jünger, and G. Reinelt. A Cutting Plane Algorithm for the Linear Ordering Problem. *Operations Research*, 329:1195–1220, 1984.

[9] G. Rinaldi. and M. W. Padberg. Optimization of a 532 City Symmetric Traveling Salesman Problem by Branch and Cut. *Operations Research Letters*, 69:1–7, 1987.

[10] M.W. Padberg and M. R. Rao. Odd Minimum Cut-Sets and b-Matchings. *Mathematics of Operations Research*, 7(1):67–80, feb 1982.

[11] P. Hall. On representatives of subsets. *J. London Math. So.*, 105:26–30, 1935.

[12] L. Lovasz and M.D. Plummer. *Matching Theory*. ANNALS OF DISCRETE MATHEMATICS, 86.

[13] J. Edmonds. Maximum Matching and a Polyhedron With O,1-Vertices . *JOURNAL OF RESEARCH of the National Bureau of Standards-B. Mathematics and Mathematical Physics*, 69B(1-2), 1965.

[14] R. E. Gomory and T.C. Hu. Multi-Terminal Network Flows . *Journal of the Society for Industrial and Applied Mathematics*, 9(4-6), 551-570.

[15] M.R. Garey and D.S. Johnson. *Computers and Intractability, A guide to the Theory of NP-Completeness*. A series of books in the mathematical sciences,Vicot Klee, Editor. W. H. Freeman and Company, 1979.

[16] M. L. Balinski. On Maximum Matching, Minimum Covering and their Connections. pages 303–312. Princeton University Press, 1970.

[17] V. Chvátal. On certain polytopes associated with graphs. *Journal of Combinatorial Theory, Series B*, 18(2):138 – 154, 1975.

[18] A. M. H. Gerards and A. Schrijver. Matrices with the edmonds-Johnson property. *Combinatorica*, 6(4):365–379, Dec 1986.

[19] G.L. Nemhauser and L.E. Trotter. Properties of vertex packing and independence system polyhedra. *Mathematical Programming*, 69:48–61, 1974.

[20] E. Osawa. Superaromaticity. *Kagaku (Chem)*, 25:854–863, 1970.

[21] H. W. Kroto, J. R. Heath, S. C. O'Brien, R. F. Curl, and R. E. Smalley. C60: Buckminsterfullerene. *Nature*, 318:162–163, November 1985.

[22] H.W. Kroto. C60 : Buckminsterfullerene, The Celestial Sphere that Fell to Earth . *Angew Chem Int*, 31(2):111–246, February 92.

[23] J.P. Hare and H.W. Kroto. A Postbuckminsterfullerene View of Carbon in the Galaxy . *Acc Chem Res*, 25:106–112, 1992.

[24] H. W. Kroto. Symmetry, space, stars and C60. *Angew Chem Int Ed*, 36:1578–1593, 1997.

[25] M.D. Newton and R.E. Stanton. Stability of buckminsterfullerene and related carbon clusters. *Journal of the American Chemical Society*, 108(9):2469–2470, 1986. PMID: 22175615.

[26] W. Andreoni. Computational approach to the physical chemistry of fullerenes and their derivatives. *Annual Review of Physical Chemistry*, 49(1):405–439, 1998. PMID: 15012433.

[27] J. F. Eckert, J. F. Nicoud, J. F. Nierengarten, S. G. Liu, L. Echegoyen, F. Barigelletti, N. Armaroli, L. Ouali, V. Krasnikov, and G. Hadziioannou. Fullerene-oligophenylenevinylene hybrids: synthesis, electronic properties, and incorporation inphotovoltaic devices. *Journal of American Chemistry society*, 122:7467–7479, 2000.

[28] A.H. Faraji and P. Wipf. Nanoparticles in cellular drug delivery. *Bioorg Med Chem*, 17:2950–2962, 2009.

[29] S. Bosi, T. DaRos, G.Spalluto, and M.Prato. Fullerene derivatives: an attractive tool for biological applications. *Eur J. Med. Chem.*, 38:913–923, 2003.

[30] A. W. Jensen, S. R. Wilson, and D. I. Schuster. Biological Applications of Fullerenes . *Bioorganic & Medicinal Chemistry*, 4(6):767–779, 1996.

[31] T. Braun. Fullerene Science and Technolopv. 1994.

[32] W. E Billups and M.A. Ciufolini. *Buckminsterfullerenes*. New York, vch edition, 1993.

[33] G.S. Hammond and V.J. Kerch. *Fullerenes. Synthesis, Properties, and Chemistry of Large Carbon Clusters*. American Chemical Society, acs symposium series 481 edition, 1992.

[34] P. W. Fowler and D. E. Manolopoulos. *An Atlas of Fullerene*. Clarenden Press, 1995.

[35] T. Braun. The epidemic spread of fullerene research. *Angew. Chem., Int. Ed. Engl.*, 31:588–589, 1992.

[36] A.J. Stone and D.J. Wales. Theoretical studies of icosahedral C60 and some related species. *Chemical Physics Letters*, 128(5):501 – 503, 1986.

[37] G.F. Brinkmann, P.W. Fowler, and C. Justus. A Catalogue of Isomerization Transformations of Fullerene Polyhedra. *J Chem Inf Comput Sci*, 43:917–927, 2003.

[38] G. Brinkmann, J. Goedgebeur, and B.D. McKay. The Generation of Fullerenes. *J Chem Inf Model*, 52:2910–2918, 2012.

[39] H. W. Kroto. The stability of the fullerenes Cn. *Nature*, 329:529–531, October 1987.

[40] K. Balasubramanian. Enumeration of isomers of polysubstituted C60 and application to NMR. *Chemical Physics Letters*, 182(3):257 – 262, 1991.

[41] M. Endo and H.W. Kroto. Formation of carbon nanofibers. *J Phys Chem*, 96:6941–6944, 1992.

[42] D.E. Manolopoulos, J.C. May, and S.E. Down. Theoretical studies of the fullerenes: C34 to C70 . *Chemical Physics Lettters*, 181, June 1991.

[43] X. Liu, D.J. Klein, T.G. Schmalz, and W.A. Seitz. Generation of Carbon-Cage Polyhedra . *J Comput Chem*, 12:1252–1259, 1991.

[44] Chih-Han Sah. Combinatorial construction of fullerene structures. *Croatica Chemica Acta*, 66:1–12, 1993.

[45] D. Barnette. On generating planar graphs. *Discrete Mathematics*, 7(3):199 – 208, 1974.

[46] G. Brinkmann and A. Dress. A Constructive Enumeration of Fullerenes . *Journal of Algorithms*, 23:345–358, 1997.

[47] G. Brinkmann and P. W. Fowler. A Catalogue of Growth Transformations of Fullerene Polyhedra . *J Chem Inf Comput Sci*, 43:1837–1843, 2003.

[48] G. Brinkmann, D. Franceus, P. W. Fowler, and J.E. Graver. Growing fullerenes from seed: Growth transformations of fullerene polyhedra . *Chemical Physics Letters*, 428:386–393, 2006.

[49] G. Brinkmann, J.E. Graver, and C. Justus. Numbers of faces in disordered patches . *J Math Chem*, 45:263–278, 2009.

[50] M. Hasheminezhad, H. Fleischner, and D. McKay. A universal set of growth operations for fullerenes . *Chemical Physics Letters*, 464:118–121, 2008.

[51] A.A. Popov, S. Yang, and L. Dunsch. Endohedral fullerenes. *Chem Rev*, 113:5989–6113, 2013.

[52] M. Yamada, T. Akasaka, and S. Nagase. Carbene additions to fullerenes. *Chem Rev*, 113:7209–7264, 2013.

[53] MS. Dresselhaus, G. Dresselhaus, and PC. Eklund. *Science of Fullerenes and Carbon Nanotubes*. NewYork: Academic Press.

[54] KM. Kadish and RS. Ruoff. New York: Wiley- Interscience, 2000.

[55] A. Hirsch, M. Brettreich, and F. Wudl. Weinheim: Wiley-VCH, 2005.

[56] A. Rodríguez-Fortea, S. Irle, and JM. Poblet. Fullerenes: formation, stability, and reactivity. *WIREs Comput Mol Sci*, 105:350–367, 2011,.

[57] EF. Sheka. *Fullerenes: Nanochemistry, Nanomag- netism, Nanomedicine, Nanophotonics*. Taylor & Francis, 2011.

[58] AD. Darwish. Fullerenes. *Annu Rep Prog Chem, Sect A Inorg Chem*, 108::464–477, 2012.

[59] A. T. Balaban, X. Liu, D. J. Klein, D. Babic, T. G. Schmalz, W. A. Seitz, and M. Randit. Graph Invariants for Fullerenes . *Journal of Chemical Information and Modeling*, 35:396–404, 1995.

[60] G. Brinkmann, J. Goedgebeur, H H. Mélot, and K. Coolsaet. House of graphs: a database of interesting graphs. *Discrete Appl Math*, 161:311–314, 2013.

[61] F. Cataldo, A. Graovac, and O. Ori. *The mathematics and topology of fullerenes*. Springe, 2011.

[62] PW. Fowler and DE. Manolopoulos. Dover Publications Inc, 2 edition, 006.

[63] P. Schwerdtfeger, L. N. Wirz, and J. Avery. The topology of fullerenes. 5:96–145.

[64] N. Trinajstić. *Chemical graph theory*. CRC Press, 1992.

[65] H. Whitney. Non-separable and planar graphs. *Trans Am Math Soc*, 34:339–362, 1932.

[66] G. Grunbaum and TS. Motzkin. The number of hexagons and the simplicity of geodesics on certain polyhe- dra. *Canadian journal of Mathematics*, 15:744–751, 1963.

[67] C. Killblane, Yi. Gao, N. Shao, and X. Cheng Zeng. Search for Lowest-Energy Nonclassical Fullerenes III: C22. *J. Phys. Chem*, 113:8839–8844, 2009.

[68] G. Brinkmann, J. Goedgebeur, and N. Van Cleemput. The history of the generation of cubic graphs. *J Chem Inf Model*, 5:67–89, 2013.

[69] J.E. Graver and CM. Graves. Fullerene patches I. *Ars Math Contemp*, 3:109–120., 2010.

[70] H. Wiener. Structural determination of paraffin boil- ing points. *J Am Chem Soc*, 69:17–20, 1947.

[71] P.W. Fowler, G. Caporossi, and P. Hansen. Distance matrices, Wiener indices, and related invariants of fullerenes. *J Phys Chem A*, 105:6232–6242, 2001.

[72] H. Hosoya. On some counting polynomials in chemistry. *Discrete Appl Math*, 19:239–257, 1988.

[73] I. Gutman. A formula for the Wiener number of trees and its extension to graphs containing cycles. *Graph Theory Notes NY*, 27:9–15, 1994.

[74] A.R. Katritzky, M. Karelson, and R. Petrukhin. The CODESSA PRO project. 2005.

[75] MV. Diudea, I. Gutman, and L. Jantschi. *Molecular Topolgy*. Nova Science, 2001.

[76] D.J. Klein and A.T. Balaban. Clarology for Conjugated Carbon Nano-Structures: Molecules, Polymers, Graphene, Defected Graphene, Fractal Benzenoids, Fullerenes, NanoTubes, Nano-Cones, Nano-Tori, etc. *The Open Organic Chemistry Journal,*, 5:27–61, 2011.

[77] D.J. Klein, T.G. Schmalz, G.E. Hite, and Seitz W.A. Resonance in C60 buckminsterfullerene. *J Am Chem Soc*, 108:1301–1302, 1986.

[78] D. Vukicevic, H.W. Kroto, and M. Randic. Atlas of kekule valence structures of buckminsterfulleren. *CROATICA CHEMICA ACTA*, 78(2):223–234, 2005.

[79] M. Randic, H.M. Kroto, and D. Vukicevc. Numerical Kekulé structures of fullerenes and partitioning of -electrons to pentagonal and hexagonal rings. *J Chem Inf Model*, 47:897–904, 2007.

[80] J. Petersen. Die Theorie der Regul aren Graphen. *Acta Mathematica*, 15(1):193–220, 1891.

[81] T. Došlić. On lower bounds of number of perfect matchings in fullerene graphs . *Journal of Mathematical Chemistry*, 24:359–364, 1998.

[82] T.G. Schmalz, W.A. Seitz, D.J Klein, and G.E. Hite. Elemental carbon cages. *J Am Chem Soc*, 110:1113–1127, 1988.

[83] S.J. Austin, P.W. Fowler, P. Hansen, D.E. Manolopoulos, and M. Zheng. Fullerene isomers of C60. Kekule counts versus stability . *Chemical Physics Letters*, 228:478–484, 1994.

[84] K. Fries. Über byclische verbindungen und ihren vergleich mit dem naphtalin. *Ann. Chem*, 454:121–324, 1927.

[85] J.E. Graver. Kekulé structures and the face independence number of a fullerene. *European Journal of Combinatorics*, 28(4-6):1115–1130, 2007.

[86] D. Vukicevic and M. Randic. On kekulè structure of buckminsterfullerene,. *Chemical Physics Letters*, 401(4-6):446–450, January 2005.

[87] E. Clar. Aromatic sextet. 1972.

[88] G.W. Wheland. *Resonance in organic chemistry*, pages 279–290. Wiley, 1995.

[89] F. Zhang, X. Guoa, and H. Zhangb. Advances of Clar's Aromatic Sextet Theory and Randic´'s Conjugated Circuit Model. *The Open Organic Chemistry Journal,*, 5:87–111, 2011.

[90] I. Gutman. Topological Properties of Benzenoid Systems, An Identity tbr the Sextet Polynomial . *Theoret. Chim. Acta*, 45:309–315, 1977.

[91] P. Hansen and M. Zheng. The Clar number of a benzenoid hydrocarbon and linear programming . *Journal of Mathematical Chemistry*, 15:93–107, 1994.

[92] H. Abeledo and G. Atkinson. Unimodularity of the Clar number problem. *Linear Algebra and its Applications*, 420:441–448, 2007.

[93] H. Zhang, D. Ye, and Y. Lui. A combination of Clar number and Kekulé count as an indicator of relative stability of fullerene isomers of C60. *J Math Chem*, 48:733–740, 2010.

[94] J.E. Graver, E.J. Hartung, and A.Y. Souid. Clar and Fries numbers for benzenoids. *J Math Chem*, 51:1981–1989, 2013.

[95] E.R. Bérczi-Kovács and A. Bernáth. The complexity of the Clar number problem and an FPT algorithm . 2015.

[96] A. Bernáth and E.R. Kovács. NP-hardness of the Clar number in general plane graphs. *Available at http://www. cseltehu/egres/qp/egresqp-11-07*, 2011.

[97] H. Zhang and Y. , Dong. An upper bound for the Clar number of fullerene graphs . *Journal of Mathematical Chemistry*, 41(2):123–132, February 2007.

[98] D. Ye and H. Zhang. Extremal fullerene graphs with the maximum Clar number. *Discrete Applied Mathematics*, 157:3152–3173, 2009.

[99] L. Pavlovic and T. Divnic. A Quadratic Programming Approach to the Randic index. *European Journal of Operational Research*, 176:435–444, 2007.

[100] M.A. Ahmadi and M. Salami. A mathematical programming model for computing the Fries number of a fullerene. *Applied Mathematical Modelling*, 39:5473–5479, 2015.

[101] J. Gondzio. Interior point methods 25 years later. *European Journal of Operational Research*, 218:587–601, 2012.

[102] R.H. Byrd, M.E. Hribar, and J. Nocedal. An Interior Point Algorithm for Large Scale Nonlinear Programming. *SIAM J. Optim*, 9:877–900, 1999.

[103] R.H. Byrd, J. Charles Gilbert, and J. Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Math. Program*, 89:149–185, (2000).

[104] R.A. Waltz, J. Nocedal, and D. Orban. An Interior Algorithm for Nonlinear Optimization That Combines Line Search and Trust Region Steps . *J. Math. Program*, 107:391–408, 2006.

[105] M.A. Ahmadi, E. Farhadi, and V. Amiri Khorasani. On Computing the Clar Number of a Fullerene Using Optimization Techniques. *MATCH*, 75:695–701, 2016.

[106] Y. Gao, Q. Li, and H. Zhang. Fullerenes with the maximum Clar number. *Discrete Applied Mathematics*, 202:58–69, 2015.

[107] L.G. Khaciyan. Polynomial time algorithms for linear programmings. *Computational Mathematics and Mathematical Physics*, 20:53–72, 1980.

[108] D. Goldfarb and M.J. Todd. Linear programming. *Optimization*, pages 73–170, 1989.

[109] N. Karmarkar. a new polynomial-time algorithm for linear programming. *Cominatoric*, 4(4):373–395, 1984.

[110] P.E. Gill, W. Murray, M.A. Saunders, J.A. Tomlin, and M.H. Wright. On the projected Newton barrier methods for linear programming and an equivalence to Karmarkar's projective method. *Mathematical Programming*, 36:183–209, 1986.

[111] E.D. Andersen, J. Gonzio, and C. Mészàros. *Implementatio of interior point methods for large scale linear programming*, pages 189–252. Kluwer Academic Publishers, 1996.

[112] S.J. Wright. *Primal-Dual Interior Point Methods*. SIAM, 1997.

[113] N. Megiddo. *Pathways to the optimal set in linear programming*. Springer-Verlag, 98.

[114] M. Kojima, S. Mizuno, and A. Yoshise. *A primal–dual interior point algorithm for linear proramming*, pages 29–47. Springer- Verlag, 1989.

[115] C.C. Gonzaga. Path–following methods for linear programming,. *SIAM Review*, 34:167–224, 1992.

[116] R.E. Marsten, R. Subramanian, I.J. Lustig, and D.F. Shanno. Interior point methods for linear programming: Just call Newton, Lagrange, and Fiacco and McCormick. *Interfaces*, 20:105–116, 1990.

[117] S. Mehrotra. On the implementation of a primal–dual interior point method. *SIAM Journal on Optimization 2*, pages 575–601, 1992.

[118] I.J. Lustig, R.E. Marsten, and D.F. Shanno. Interior point methods for linear programming: Computational state of the art. *ORSA Journal on Computing 6*, pages 1–14, 1994.

[119] R.E. Bixby. Progress in linear programming. *ORSA Journal on Computing 6*, pages 15–22, 1994.

[120] J.J.H. Forrest and D. Goldfarb. Steepest-edge simplex algorithms for linear programming, Mathematical Programming. *Mathematical Programming 57*.

[121] J.A.J. Hall and K.I.M. McKinnon. Hyper-sparsity in the revised simplex method and how to exploit it. *Computational Optimization and Applications*, 32:259–283, 2005.

[122] I. Maros. *Computational Techniques of the Simplex Method*. Kluwer Academic Publishers, first edition, 2003.

[123] J.A.J. Hall and K.I.M. Mckinnon. Hyper-Sparsity in the Revised Simplex Method and How to Exploit it. *Computational Optimization and Applications*, 32:259–283, 2005.

[124] J.A.J. Hall. Towards a practical parallelisation of the simplex method. *Computational Management Science 7*, 7:139–170, 2010.

[125] R. Vanderbei. LOQO: An interior point code for quadratic programming. *Optimization Methods and Software*, pages 451–484, 1999.

[126] R.J Vanderbei and D.F. Shanno. An interior-point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications 1*, 13:231–252, 1999.

[127] Y. Nesterov and A. Nemirovskii. *Interior Point Polynomial Algorithms in Convex Programming* . SIAM, 1994.

[128] A. Forsgren, P.E. Gill, and M.H. Wright. Interior Methods for Nonlinear Optimization. *Society for Industrial and Applied Mathematic*, 44 (:525–597, 2002.

[129] L. Vandenberghe and S. Boyd. Semidefinite programmin. *Semidefinite programming*, pages 49–95, 1996.

[130] S. Boyd and L. Vandenberghe. *Convex Optimization*. CAMBRIDGE UNIVERSITY PRESS, 2006.

[131] J. Renegar. A polynomial-time algorithm, based on Newton's method, for linear programming. *J. Renegar, A polynomial-time algorithm, based Mathematical Programming*, 40:59–93, 1988.

[132] Y. Nesterov. *Introductory Lectures on Convex Optimization*, chapter 4. Kluwer, 1998.

[133] M. Colombo and J. Gondzio. Further development of multiple centrality correctors for interior point methods. *Further development of multiple centrality correctors for interior point methods, Computational Optimization and Applications*, 41:277–305, 2008.

[134] Renato D. C. Monteiro and Jerome W. ONeal. Convergence analysis of a long-step primal-dual infeasible interior-point lp algorithm based on iterative linear solvers, 2003.

[135] A. Altman and J. Gondzio. Regularized symmetric inde nite systems in interior point methods for linear and quadratic optimization. 1998.

[136] G. Al-Jeiroudi. *On Inexact Newton Directions in Interior Point Methods for Linear Optimization*. PhD thesis, University of Edinburgh Graduate School of Mathematics, 2008.

[137] L. Bergamaschi, J. Gondzio, and G. Zilli. Preconditioning indefinite systems in interior point methods for optimization. *Computational Optimization and Applications,*, (28):149–171, 2004.

[138] J. J. Júdice, J. Patricio, L. F. Portugal, M. G. C. Re- sende, and G. Veiga. A study of preconditioners for network inte- rior point methods. *Computational Optimization and Applications*, (24):5–35, 2003.

[139] N. Karmarkar and K. Ramakrishnan. Computational results of an interior point algorithm for large scale linear programming. *Mathematical Programming 52*, (52):555–58, 1991.

[140] R. S. Dembot, S.C. Eisenstat, and T. Steihaug. Inexact Newton Methods. *SIAM Journal on Numerical Analysis*, 19:400–408, 1982.

[141] S. Cafieri, M. D'Apuzzo, V. De Simone, D. di Serafino, and G. Toraldo. Convergence Analysis of an Inexact Potential Reduction Method for Convex Quadratic Programming . *Journal of Optimization Theory and Applications*, 135:355–366, 2007.