A Note on the 2-Circulant Inequalities for the Max-Cut Problem

Konstantinos Kaparis* Adam N. Letchford†

To appear in *Operations Research Letters*

Abstract

The *max-cut* problem is a much-studied $\mathcal{NP}$-hard combinatorial optimisation problem. Poljak and Turzik found some facet-defining inequalities for this problem, which we call *2-circulant* inequalities. Two polynomial-time separation algorithms have been found for these inequalities, but one is very slow and the other is very complicated. We present a third algorithm, which is as fast as the faster of the existing two, but much simpler.

Keywords: max-cut problem, polyhedral combinatorics, branch-and-cut.

1 Introduction

Let G be an undirected graph, with vertex set V and edge set E. For any $S \subseteq V$, the set of edges having exactly one end-vertex in S is called an *edge-cutset* or simply *cut*. Given a weight $w_e \in \mathbb{R}$ for each edge $e \in E$, the *max-cut* problem calls for a cut in G of maximum total weight.

The max-cut problem is strongly $\mathcal{NP}$-hard [13]. It has many applications and has received much attention (see, e.g., [10, 18]). As present, the most successful exact algorithms for max-cut are based on either *linear programming* (LP) or *semidefinite programming* (SDP) relaxations (see, e.g., [4, 23, 25]). LP-based algorithms can solve sparse instances with up to around 500 nodes, and SDP-based ones can solve dense instances with up to around 150 nodes.

To construct strong LP relaxations of combinatorial optimisation problems, it helps to derive families of strong (preferably facet-defining) valid linear inequalities (see, e.g., [7]). Several such families have been derived

*Department of Business Administration, University of Macedonia, Greece. E-mail: k.kaparis@uom.edu.gr

†Department of Management Science, Lancaster University, United Kingdom. E-mail: A.N.Letchford@lancaster.ac.uk

for max-cut (see, e.g., [3, 10]). In this paper, we focus on some inequalities discovered by Poljak & Turzik [24], which we call *2-circulant* inequalities.

The *separation problem* for a given family of valid linear inequalities is this: given a fractional solution to an LP relaxation of an instance of the max-cut problem, either find an inequality in the family that is violated by the fractional solution, or prove that no such violated inequality exists (see, e.g., [7, 16]). In [24], Poljak and Turzik claimed (without proof) that the separation problem for the 2-circulant inequalities (or more precisely a generalisation of them) could be solved in polynomial time. Explicit algorithms for this purpose were however discovered only later on [20, 21].

Unfortunately, the separation algorithm in [20] is very slow, involving the solution of $O(n^2)$ very large LPs (where $n = |V|$). The algorithm in [21] is faster, with a running time of $O(n^5)$, but it is very hard to implement. The purpose of this paper is to present a third algorithm, which is much simpler than both of the other algorithms, yet still runs in $O(n^5)$ time.

The paper has a simple structure: the literature is reviewed in Section 2, the new algorithms are presented in Section 3, and concluding remarks are made in Section 5. Throughout the paper, we let K_n denote the complete undirected graph with vertex set $V_n = \{1, \ldots, n\}$ and edge set $E_n = \{e \subset V_n : |e| = 2\}$. Given a vector $x \in [0,1]^{E_n}$ and an edge $e = \{i, j\} \in E_n$, we sometimes write x_{ij} or $x(i, j)$ instead of x_e. We also assume that $n \geq 5$ throughout the paper, to avoid trivial cases.

2 Literature Review

Now we review the literature. Due to space limitations, we only review essential results, and refer the reader to [10, 18] for comprehensive surveys.

2.1 The cut polytope

By adding dummy edges of weight zero, if necessary, we can work with the complete graph K_n. The max-cut problem can then be formulated as the following 0-1 LP:

$$\begin{aligned} \max \quad & \textstyle\sum_{e \in E_n} w_e x_e & & \\ \text{s.t.} \quad & x_{ij} + x_{ik} + x_{jk} \leq 2 & 1 \leq i < j < k \leq n & \quad (1) \\ & x_{ij} - x_{ik} - x_{jk} \leq 0 & \{i, j\} \in E_n;\ k \in V_n \setminus \{i, j\} & \quad (2) \\ & x \in \{0,1\}^{E_n}. & & \end{aligned}$$

Here, x_e takes the value 1 if and only if edge e lies in the cut. The inequalities (1) and (2) are called *triangle* inequalities.

Barahona and Mahjoub [3] call the convex hull of feasible solutions to the above 0-1 LP the *cut polytope*. We will denote it by CUT_n. They show that the triangle inequalities define facets of CUT_n, along with various other

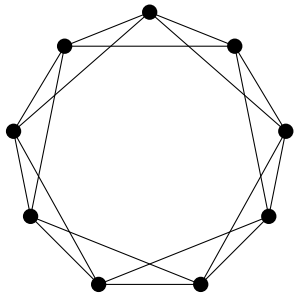

Figure 1: Representation of a 2-circulant inequality with $c = 9$.

families of inequalities, including so-called *odd clique* and *odd bicycle wheel* inequalities. Since then, a huge array of valid and facet-defining inequalities has been discovered (see Part V of [10]). We will be particularly interested in the following inequalities, due to Poljak & Turzik [24]:

Proposition 1 (Poljak & Turzik, 1992) *Let c be a positive integer with $5 \leq c \leq n$, and suppose that c is congruent to 1 modulo 4. Let $v_1, \ldots, v_c$ be distinct vertices in K_n. The inequality*

$$\sum_{i=1}^{c} \Big(x\big(v_i, v_{i+1}\big) + x\big(v_i, v_{i+2}\big) \Big) \;\leq\; 3(c-1)/2 \tag{3}$$

defines a facet of CUT_n, where indices are taken modulo c.

We will call the inequalities (3) *2-circulant* inequalities. Figure 1 gives a graphical representation of a 2-circulant inequality with $c = 9$. Small filled circles represent nodes, and edges represent variables with a coefficient of one on the left-hand side.

A point that will be important later on (see Subsection 2.3) is that 2-circulant inequalities are not valid for CUT_n when c is congruent to 3 mod 4. Indeed, to make them valid in that case, it is necessary to increase the right-hand side by one, to $(3c-1)/2$ (see Figure 2). Moreover, the resulting inequalities are not of interest, since they are easily shown to be implied by triangle inequalities.

We remark that the cut polytope is closely related to the so-called *Boolean quadric polytope*, which is a polytope associated with unconstrained quadratic programming in binary variables (see, e.g., [2, 8, 10, 22]). For brevity, we do not go into details, but simply point out that our new separation algorithms can be easily adapted to the Boolean quadric polytope.

2.2 Switching

An important operation, called *switching*, was also defined in [3]. It states that, if the inequality $\lambda^T x \leq \gamma$ is valid (or facet-defining) for CUT_n, then

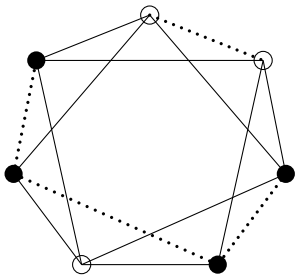

Figure 2: A 2-circulant with $c = 7$ contains a cut with $(3c-1)/2 = 10$ edges. Nodes in S represented by filled circles, others by hollow circles. Edges in $\delta(S)$ represented by solid lines, others by dotted lines.

the 'switched' inequality

$$\sum_{e \in E_n \setminus \delta(S)} \lambda_e x_e - \sum_{e \in \delta(S)} \lambda_e x_e \leq \gamma - \sum_{e \in \delta(S)} \lambda_e$$

is also valid (or facet-defining), for any $S \subset V_n$. As a simple example, taking a triangle inequality of the form (1) and switching on node k, we obtain a triangle inequality of the form (2). In a similar way, one can construct switched odd clique, odd bicycle wheel and 2-circulant inequalities.

2.3 Separation

Known results on exact separation algorithms include the following:

- Separation for the triangle inequalities (1), (2) can be solved in $O\big(n^3\big)$ time by mere enumeration.
- More generally, separation for (switched) odd clique inequalities on k nodes, for a fixed odd $k \geq 3$, can be solved by enumeration in $O\big(n^k\big)$ time.
- Separation for the odd bicycle wheel inequalities can be solved in $O\big(n^5\big)$ time [14]. The idea is to reduce the separation problem to $\binom{n}{2}$ minimum-weight odd cycle problems in K_{n-2}, and then use the known fact [3, 15] that the minimum-weight odd cycle problem can itself be reduced to a series of shortest (s,t)-path problems.
- The separation problem for a family of valid inequalities that arises from an SDP relaxation of max-cut can be solved (to arbitrary fixed precision) in polynomial time, by computing the minimum Eigenvalue of a certain matrix [19]. The inequalities in question are however never facet-defining.

- One can separate in polynomial time over families of valid inequalities that include not only all odd bicycle wheel and 2-circulant inequalities, but also all of their switchings [20, 21]. The algorithm in [20] is based on lift-and-project (see [1]), and involves the solution of $\binom{n}{2}$ LPs, each with $O(n^3)$ variables and $O(n^2)$ constraints. The algorithm in [21] is based on repeated applications of the covariance map, along with arguments based on "$\{0, \frac{1}{2}\}$-cuts" (see [5]). It runs in $O(n^5)$ time, but is very hard both to understand and to implement.

Poljak & Turzik [24] noted (Remark 5.4, page 391) that 2-circulant inequalities remain valid even when the nodes $v_1, \ldots, v_c$ are not all distinct, provided that $v_i \neq v_{i+1}$ for $i = 1, \ldots, c$. They also stated that the separation problem for the resulting generalised 2-circulant inequalities could be transformed to a minimum-weight odd cycle problem on a suitable graph, but they did not give a proof. (Actually, we believe that such a transformation is impossible, due to the fact that 2-circulant inequalities are valid not when c is odd, but only when c is congruent to 1 mod 4.)

There are also several separation *heuristics* available for CUT_n and related polyhedra; see, e.g., [2, 4, 6, 9, 12, 17]. For the sake of brevity, we do not go into details.

3 The New Separation Algorithms

In this section, we describe our new separation algorithms. Subsection 3.1 deals with the 2-circulant inequalities themselves, whereas Subsection 3.2 deals with switched 2-circulant inequalities. The inputs to both algorithms are an integer $n \geq 5$ and a fractional point $x^* \in [0,1]^{E_n}$ that lies outside CUT_n.

3.1 Separation for 2-circulant inequalities

Our separation algorithm is based on the following lemmas and definition.

Lemma 1 *Given any ordered triple (i, j, k) of nodes in V, the "weakened triangle" inequality*

$$x_{ij} + x_{jk} + 2x_{ik} \leq 3 \tag{4}$$

is valid for CUT_n.

Proof. It is the sum of the triangle inequality (1) and the upper bound $x_{ik} \leq 1$. □

Definition 1 *Given any ordered triple (i, j, k) of nodes in V, let $\Delta(i, j, k)$ denote the slack of the weakened triangle inequality (4). That is:*

$$\Delta(i, j, k) = 3 - x_{ij} - x_{jk} - 2x_{ik}.$$

Lemma 2 *The 2-circulant inequalities (3) can be written as*

$$\sum_{i=1}^{c} \Delta\big(v_i, v_{i+1}, v_{i+2}\big) \geq 3,$$

where indices are again taken mod c.

Proof. Multiplying the inequality (3) by minus two and adding $3c$ to both sides yields

$$3c - \sum_{i=1}^{c} \Big(2x\big(v_i, v_{i+1}\big) + 2x\big(v_i, v_{i+2}\big)\Big) \geq 3.$$

The result then follows easily. □

Lemma 2 immediately suggests using Algorithm 1 to solve the separation problem for 2-circulant inequalities.

Algorithm 1: Exact separation of 2-circulant inequalities.

Input: An integer $n \geq 5$ and a point $x^* \in [0,1]^{E_n}$ that satisfies all of the weakened triangle inequalities (4).

Output: A collection of valid inequalities violated by x^*, or a proof that no violated 2-circulant inequalities exist.

```
Construct a digraph G+ = (V+, A+), initially empty;
for each ordered pair (i, j) of nodes in V do
    insert a node into V+ and label it "(i, j)";
end
for each ordered triple (i, j, k) of nodes in V do
    let Δ = 3 − x*_ij − x*_jk − 2x*_ik;
    insert an arc from node (i, j) to node (j, k);
    give the arc a weight of Δ;
end
for each node (i, j) ∈ V+ do
    Find the minimum-weight dicycle in G+ that passes through the
      node (i, j) and whose cardinality is congruent to 1 mod 4;
    if the weight of the dicycle is less than 3 then
        Output the corresponding violated inequality;
    end
end
```

Note that Algorithm 1 solves the separation problem for a *generalisation* of the 2-circulant inequalities, rather than the 2-circulant inequalities themselves, since we cannot guarantee that every dicycle in G^+ found during the course of the algorithm corresponds to a sequence of distinct nodes $v_1, \ldots, v_c$ in G.

One component of Algorithm 1 that needs explaining is the minimum-weight dicycle computation in step 10. This can be accomplished via a minor modification of the minimum-weight odd cycle algorithm given in [3, 15]. We create another graph H, which has four copies of each node in V^+. For each arc $\big((i,j),(j,k)\big)$ in A^+, we insert four arcs into H, with the same weight, going:

- from the first copy of node (i,j) to the second copy of (j,k);
- from the second copy of node (i,j) to the third copy of (j,k);
- from the third copy of node (i,j) to the fourth copy of (j,k);
- from the fourth copy of node (i,j) to the first copy of (j,k).

A minimum-weight dicycle in G^+ passing through node (i,j) and having cardinality congruent to 1 mod 4 can now be found by computing a shortest path in H from the first copy of (i,j) to the second copy.

Now, provided that our fractional point x^* satisfies all of the weakened triangle inequalities, then all arc-weights in A^+ will be non-negative. This leads to our main theorem:

Theorem 1 *If the fractional point $x^* \in [0,1]^{E_n}$ satisfies all of the weakened triangle inequalities (4), then the separation problem for a generalisation of the 2-circulant inequalities can be solved in $O\big(n^5\big)$ time.*

Proof. The digraph G^+ has $O\big(n^2\big)$ nodes and $O\big(n^3\big)$ arcs, and therefore so does H. To solve the separation problem, we must solve $O\big(n^2\big)$ shortest (s,t)-path problems in H. The implementation of Dijkstra's algorithm described in [11] takes $O(p + q \log q)$ time to compute a shortest path in a digraph with p arcs and q nodes. Thus, each shortest-path computation in H can be performed in $O\big(n^3\big)$ time. □

We remark that, although a running time of $O\big(n^5\big)$ time is rather high, our separation algorithm can generate several violated inequalities in a single call. (We conjecture that it can generate $\Theta\big(n^2\big)$ of them.) Moreover, the practical performance of the algorithm can be improved a little by exploiting the structure of x^*:

Proposition 2 *Suppose that x^* satisfies all of the triangle inequalities (1), (2), and let f denote the number of variables that are fractional at x^*. Then Algorithm 1 can be implemented to run in $O\big(nf^2\big)$ time.*

Proof. Theorem 4 of [20] states that 2-circulant inequalities are implied by the triangle inequalities together with the disjunction

$$\Big(x\big(v_1,v_2\big)=0\Big) \vee \Big(x\big(v_1,v_2\big)=1\Big).$$

By symmetry, this implies that, if x^* satisfies all triangle inequalities, then a 2-circulant inequality cannot be violated unless $x^*\big(v_i, v_{i+1}\big)$ is fractional for $i = 1, \ldots, c$. This implies in turn that we only need to include a node (i, j) in V^+ if x^*_{ij} is fractional. This reduces the number of nodes in V^+ to $2f$, and reduces the number of arcs in A^+ to $O(nf)$. Each shortest-path computation then takes only $O(nf)$ time, and the number of shortest-path calls is only $O(f)$. □

One can speed up the practical performance of the separation algorithm further, by aborting any given Dijkstra call as soon as a node in H receives a permanent distance label of 3 or more. We omit details, for brevity.

3.2 Separation for switched 2-circulant inequalities

Now we show how to extend the separation algorithm to cover *switched* 2-circulant inequalities, while preserving the same (asymptotic) running time.

First, we enumerate all possible switchings of the weakened triangle inequality (4). For example, if we switch nodes i and k, we obtain

$$-x_{ij} \; - \; x_{jk} \; + \; 2x_{ik} \leq 1. \tag{5}$$

Then, we extend the notation $\Delta(i, j, k)$, by putting a small bar above any nodes that have been switched. For example, $\Delta(\bar{i}, j, \bar{k})$ will denote the slack of the switched inequality (5), i.e., the quantity $1 + x_{ij} \; + \; x_{jk} \; - \; 2x_{ik}$.

Now, for each ordered pair (i, j) of nodes in V, there are four possibilities, according to which of the two nodes (if any) is to be switched. To handle this, we insert four nodes into V^+ for each such pair, labelled (i, j), $(\bar{i}, j)$, $(i, \bar{j})$ and $(\bar{i}, \bar{j})$. Then, when constructing A^+, in addition to inserting an arc from node (i, j) to node (j, k) with weight equal to $\Delta^*(i, j, k)$, we insert seven other arcs, as follows:

- from node (i, j) to node $(j, \bar{k})$, with weight $\Delta^*(i, j, \bar{k})$;
- from node $(i, \bar{j})$ to node $(\bar{j}, k)$, with weight $\Delta^*(i, \bar{j}, k)$;
- from node $(i, \bar{j})$ to node $(\bar{j}, \bar{k})$, with weight $\Delta^*(i, \bar{j}, \bar{k})$;
- from node $(\bar{i}, j)$ to node (j, k), with weight $\Delta^*(\bar{i}, j, k)$;
- from node $(\bar{i}, j)$ to node $(j, \bar{k})$, with weight $\Delta^*(\bar{i}, j, \bar{k})$;
- from node $(\bar{i}, \bar{j})$ to node $(\bar{j}, k)$, with weight $\Delta^*(\bar{i}, \bar{j}, k)$;
- from node $(\bar{i}, \bar{j})$ to node $(\bar{j}, \bar{k})$, with weight $\Delta^*(\bar{i}, \bar{j}, \bar{k})$.

It is then an easy (but tedious) exercise to show that x^* violates a switched 2-circulant inequality if and only if there is a dicycle in G^+ with weight less than 3 and cardinality congruent to 1 mod 4.

			%Gap Closed	
$\|V\|$	#OPT	%Gap	Circs	Sw. Circs
35	0	15.70	73.89	73.89
45	0	26.79	54.11	54.11
55	0	29.89	43.60	43.60

Table 1: Computational results for MC_A

4 Computational Results

To explore the potential of 2-circulant inequalities, we have implemented a cutting-plane algorithm and run it on a set of 60 randomly generated max-cut instances. Our test set is composed of two sets of fully dense instances having 30 members each. The two sets, called "`MC_A`" and "`MC_B`", differ only with respect to their weight coefficients. For the instances in `MC_A`, each edge weight is a random integer uniformly selected from $\{1, \ldots, 10\}$. For the instances in `MC_B`, each edge weight is set to either $+1$ or -1 with equal probability. Each set contains ten instances for each value of n in $\{35, 45, 55\}$.

For each instance, we computed three upper bounds: the one obtained when using only the triangle inequalities (1) - (2), the one obtained when 2-circulant inequalities were added, and the one obtained when switched 2-circulant inequalities were included as well. We also computed a lower bound, by running branch-and-bound with a time limit of 1200 seconds.

The code was written in `C` and calls on the callable library of `CPLEX` (v. 12.71) both to solve the LP relaxations and to perform branch-and-bound. The experiments were run on an Intel i3 processor at 3.60GHz, under Ubuntu 16.04, with 8GB of RAM.

Table 1 and table 2 summarize the computational results obtained for `MC_A` and `MC_B`, respectively. Each of the three rows in each table corresponds to a batch of 10 instances. The first column gives the number of nodes. The second column reports the number of instances solved to proven optimality by the branch-and-bound algorithm within the time limit. The column headed "%Gap" gives the average gap between the initial upper bound (triangle inequalities only) and the lower bound, expressed as a percentage of the lower bound. The columns headed "%Gap Closed" report the average percentage of the gap closed by the 2-circulant inequalities and switched 2-circulant inequalities.

For the instances with positive edge weights, the 2-circulant inequalities are quite useful, but the switched inequalities add no value. For the instances with positive and negative weights, however, the switched inequalities are extremely useful, closing nearly all of the gap.

			%Gap Closed	
$\|V\|$	#OPT	%Gap	Circs	Sw. Circs
35	10	33.90	41.27	100
45	10	55.74	28.38	99.48
55	5	80.57	22.90	93.68

Table 2: Computational results for MC_B

5 Concluding Remarks

We have presented a (relatively) simple exact separation algorithm for (a generalisation of) the 2-circulant inequalities of Poljak & Turzik [24], which runs in $O(nf^2)$ time, where n is the number of nodes and f is the number of fractional variables. We have also shown how to extend that algorithm to separate over a broader family of inequalities, that can be derived by switching.

We are currently working on faster separation heuristics, and of extensions to the case of max-cut on general (rather than complete) graphs. Ultimately, our goal is to create an open-source C library of exact and heuristic separation routines for the max-cut problem and related problems.

References

[1] E. Balas, S. Ceria & G. Cornuéjols (1993) A lift-and-project cutting plane algorithm for mixed 01 programs. *Math. Program.*, 58, 295–324.

[2] F. Barahona, M. Jünger & G. Reinelt (1989) Experiments in quadratic 0-1 programming. *Math. Program.*, 44, 127–137.

[3] F. Barahona & A.R. Mahjoub (1986) On the cut polytope. *Math. Program.*, 36, 157–173.

[4] T. Bonato, M. Jünger, G. Reinelt & G. Rinaldi (2014) Lifting and separation procedures for the cut polytope. *Math. Program.*, 146, 351–378.

[5] A. Caprara & M. Fischetti (1996) $\{0, \frac{1}{2}\}$-Chvátal-Gomory cuts. *Math. Program.*, 74, 221–235.

[6] E. Cheng (1998) Separating subdivision of bicycle wheel inequalities over cut polytopes. *Oper. Res. Lett.*, 23, 13–19.

[7] M. Conforti, G. Cornuéjols & G. Zambelli (2014) *Integer Programming.* Cham, Switzerland: Springer.

[8] C. De Simone (1989) The cut polytope and the Boolean quadric polytope. *Discr. Math.*, 79, 71–75.

[9] C. De Simone & G. Rinaldi (1994) A cutting plane algorithm for the max-cut problem. *Opt. Methods and Software*, 3, 195–214.

[10] M.M. Deza & M. Laurent (1997) *Geometry of Cuts and Metrics*. Berlin: Springer-Verlag.

[11] M.L. Fredman & R.E. Tarjan (1987) Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM*, 34, 596–615.

[12] L. Galli, K. Kaparis & A.N. Letchford (2012) Gap inequalities for the max-cut problem: a cutting-plane algorithm. In A.R. Mahjoub, V. Markakis, I. Milis & V.T. Paschos (eds.) *Combinatorial Optimization: Proceedings of ISCO 2012*, pp. 178–188. Berlin: Springer.

[13] M.R. Garey, D.S. Johnson & L.J. Stockmeyer (1976) Some simplified $\mathcal{NP}$-complete graph problems. *Theoret. Comput. Sci.*, 1, 237–267.

[14] A.M.H. Gerards (1985) Testing the odd bicycle wheel inequalities for the bipartite subgraph polytope. *Math. Oper. Res.*, 10, 359–360.

[15] A.H.M. Gerards & A.J. Schrijver (1986) Matrices with the Edmonds–Johnson property. *Combinatorica*, 6, 365–379.

[16] M. Grötschel, L. Lovász & A.J. Schrijver (1988) *Geometric Algorithms in Combinatorial Optimization*. New York: Wiley.

[17] C. Helmberg & F. Rendl (1998) Solving quadratic (0,1)-programs by semidefinite programs and cutting planes. *Math. Program.*, 82, 291–315.

[18] M. Laurent (1997) Max-cut problem. In M. Dell'Amico, F. Maffoli & S. Martello (eds.) *Annotated Bibliographies in Combinatorial Optimization*, pp. 241–259. Chichester: Wiley.

[19] M. Laurent & S. Poljak (1995) On a positive semidefinite relaxation of the cut polytope. *Lin. Alg. Appl.*, 223/224, 439–461.

[20] A.N. Letchford (2001) On disjunctive cuts for combinatorial optimization. *J. Combin. Optim.*, 5, 299–315.

[21] A.N. Letchford & M.M. Sørensen (2014) A new separation algorithm for the Boolean quadric and cut polytopes. *Discr. Optim.*, 14, 61–71.

[22] M.W. Padberg (1989) The Boolean quadric polytope: some characteristics, facets and relatives. *Math. Program.*, 45, 139–172.

[23] L. Palagi, V. Piccialli, F. Rendl, G. Rinaldi & A. Wiegele (2012) Computational approaches to max-cut. In M.F. Anjos & J.B. Lasserre (eds.) *Handbook on Semidefinite, Conic and Polynomial Optimization*, pp. 821–847. Boston, MA: Springer US.

[24] S. Poljak & D. Turzik (1992) Max-cut in circulant graphs. *Discr. Math.*, 108, 379–392.

[25] F. Rendl, G. Rinaldi & A. Wiegele (2010) Solving max-cut to optimality by intersecting semidefinite and polyhedral relaxations. *Math. Program.*, 121, 307–335.