Centrum voor Wiskunde en Informatica

REPORT RAPPORT

Relaxations of the Satisfiability Problem using Semidefinite Programming

Etienne de Klerk, Hans van Maaren, Joost P. Warners

Software Engineering (SEN)

SEN-R9903 January 31, 1999

Report SEN-R9903
ISSN 1386-369X

CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

Relaxations of the Satisfiability Problem using Semidefinite Programming

Etienne de Klerk[b] Hans van Maaren[b] Joost P. Warners[a,b]

[a] *CWI*
P.O. Box 94079, 1090 GB, Amsterdam, The Netherlands
e–mail: `Joost.Warners@cwi.nl`

[b] *Department of Technical Mathematics and Informatics, Faculty of Information Technology and Systems*
Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands
e–mail: `e.deklerk,h.vanmaaren,j.p.warners@twi.tudelft.nl`

ABSTRACT
We derive a semidefinite relaxation of the satisfiability (SAT) problem and discuss its strength. We give both the primal and dual formulation of the relaxation. The primal formulation is an eigenvalue optimization problem, while the dual formulation is a semidefinite feasibility problem. It is shown that using the relaxation, the notorious pigeon hole and mutilated chessboard problems are solved in polynomial time. As a byproduct we find a new 'sandwich' theorem that is similar to Lovász' famous ϑ-function. Furthermore, using the semidefinite relaxation 2SAT problems are solved. By adding an objective function to the dual formulation, a specific class of polynomially solvable 3SAT instances can be identified. We conclude with discussing how the relaxation can be used to solve more general SAT problems and some empirical observations.

1991 Mathematics Subject Classification: 03B05, 65K05, 68T15, 68T20, 90C27, 90C30
1991 Computing Reviews Classification System: F.2, G.2
Keywords and Phrases: Satisfiability, Propositional logic, Semidefinite programming, Automated reasoning, Polynomial algorithm
Note: The third author is supported by the Dutch Organization for Scientific Research (NWO) under grant SION 612-33-001.

1. Introduction

The satisfiability problem of propositional logic (SAT) is the original NP complete problem [5]. Many algorithms for SAT have been developed, both complete and incomplete; see for an overview [13]. Most of the incomplete algorithms are aimed at proving satisfiability; using some hillclimb strategy a satisfying solution is sought, and if one is found the algorithm terminates; if no solutions are found, no definite answers about the (un)satisfiability of the formula can be given. In this paper we are primarily concerned with developing an (incomplete) algorithm for detecting *un*satisfiability. It is based on elliptic approximations of propositional formulas. Elliptic approximations of propositional formulas were first introduced by van Maaren [20, 19]. Since then they have been used to recognize polynomially solvable instances of satisfiability [29] and to derive effective branching rules and relative clause–weightings [21, 30]. Here they are used to derive sufficient conditions for unsatisfiability of a formula. This condition can be expressed in terms of an *eigenvalue optimization* problem, which in turn can be cast as a *semidefinite program* (see e.g. [28, 8]). Semidefinite programming is a generalization of linear programming, and can be solved in polynomial time (to a given accuracy). Recently, much attention has been devoted to this field. Using semidefinite programming, efficient approximation algorithms to various hard combinatorial optimization problems have been developed, including the *maximum–satisfiability* problem [12, 11, 26, 16]. Using duality theory, we show that the dual of our formulation is a semidefinite feasibility problem, which is related to the formulation used by Karloff and Zwick for 3SAT formulas [16]. However, it is our aim to use it for proving unsatisfiability rather than finding approximate MAX–SAT solutions. We show that using the relaxation the notorious pigeon hole problems are solved in polynomial time in a truly automated way; i.e. without additional problem–specific tricks. As a byproduct, we obtain a new 'sandwich theorem' that is similar to Lovász' famous ϑ-function [18]. Furthermore, we show that it is complete for 2SAT formulas, and we indicate how it can be used to help solving 3SAT problems. In particular, a certain class of polynomially solvable 3SAT formulas can be recognized by adding an objective function to the dual formulation [23, 29].

This paper is organized as follows. In the next section we discuss the preliminaries and notation, and give a brief introduction to semidefinite programming. Subsequently, we derive a semidefinite relaxation of the satisfiability problem, derive its dual formulation and mention a number of properties. In Sections 4, 5 and 6 the strength of the relaxation is investigated by considering several subclasses of satisfiability problems, namely 2SAT problems, a class of covering problems (to which the pigeon hole and mutilated chess board problem belong) and 3SAT problems. We conclude with some empirical observations and suggestions for further research.

2. Preliminaries and notation

2.1 The SAT problem

We consider the satisfiability problem in conjunctive normal form (CNF). A propositional formula Φ in CNF is a conjunction of clauses, where each clause $\mathbf{C}_k$ is a disjunction of *literals*. Each literal is an *atomic proposition* (or *variable*) or its negation ($\neg$). Let m be the number of clauses and n the number of atomic propositions. A clausal propositional formula is denoted as $\Phi = \mathbf{C}_1 \wedge \mathbf{C}_2 \wedge \ldots \wedge \mathbf{C}_m$, where each clause $\mathbf{C}_k$ is of the form

$$\mathbf{C}_k = \bigvee_{i \in I_k} p_i \vee \bigvee_{j \in J_k} \neg p_j,$$

with $I_k,\ J_k \subseteq \{1, \ldots, n\}$ disjoint. The satisfiability problem of propositional logic is to determine whether or not an assignment of truth values to the variables exists such that each clause evaluates to true (i.e. one of its literals is true) and thus the formula is true.

Associating a $\{-1, 1\}$–variable x_i with each proposition letter p_i, a clause $\mathbf{C}_k$ can be written as a linear inequality in the following way.

$$C_k(x) = \sum_{i \in I_k} x_i - \sum_{j \in J_k} x_j \geq 2 - \ell(\mathbf{C}_k), \tag{2.1}$$

where $\ell(\mathbf{C}_k)$ denotes the length of clause k, i.e. $\ell(\mathbf{C}_k) = |I_k \cup J_k|$. Using matrix notation, the integer linear programming formulation of the satisfiability problem can be stated as

$$(\mathrm{IP}_{SAT}) \;\; \text{find } x \in \{-1, 1\}^n \text{ such that } Ax \geq b.$$

The matrix $A \in \mathbb{R}^{m \times n}$ is called the *clause–variable* matrix. We have that $a_k^T x = C_k(x)$, where a_k^T denotes the k^{th} row of A. Obviously, $a_{ki} = 1$ if $i \in I_k$, $a_{ki} = -1$ if $i \in J_k$, while $a_{ki} = 0$ for any $i \notin I_k \cup J_k$. Furthermore, $b_k = 2 - \ell(\mathbf{C}_k)$.

Note that in general it is not possible to solve integer linear programming problems in a direct fashion. The most common approach is to relax the integrality constraints to linear constraints (i.e. $x \in \{-1, 1\}$ is relaxed to $-1 \leq x \leq 1$), and subsequently solve the resulting *LP relaxation*. Unfortunately, the LP relaxation of (IP$_{SAT}$) is weak; it is easily checked that the trivial all–zero solution is always feasible when no unit clauses are present. By introducing some objective function the solution can be steered away from the trivial one, in the hope that an incumbent solution may be found by rounding the solution to the LP relaxation. A specific kind of roundable solutions is the so–called *linear autarky*; this is discussed in Section 4. For deciding *un*satisfiability however, the above LP relaxation is virtually useless (note that an alternative LP relaxation, namely the one based on polynomial representations of SAT problems, is capable of proving *un*satisfiability rather than satisfiability [31]).

2.2 Semidefinite programming

Recently much attention has been devoted to the field of semidefinite programming. It was shown that efficient approximation algorithms for hard combinatorial optimization problems can be obtained using semidefinite relaxations [12, 1], while there are also applications in control theory [28]. Using interior point methods, semidefinite programs can be solved (to a given accuracy) in polynomial time. For the reader that is unfamiliar with semidefinite programming, we review some of the basic concepts.

The standard primal (P) and dual (D) semidefinite programming formulations can be denoted by (see e.g. [8])

$$
\text{(P)} \quad \begin{array}{ll} \inf & \mathbf{Tr}\, CX \\ \text{s.t.} & \mathbf{Tr}\, A_i X = b_i, \quad 1 \leq i \leq m \\ & X \succeq 0. \end{array}
\qquad\qquad
\text{(D)} \quad \begin{array}{ll} \sup & b^T y \\ \text{s.t} & \sum_{i=1}^{m} y_i A_i + S = C, \\ & S \succeq 0. \end{array}
$$

In the above programs, the A_i, C, X and S are symmetric real $(n \times n)$–matrices and b and y are m–vectors. The matrix X denotes the primal decision variables, while (S, y) are the dual decision variables; the constraint $X \succeq 0$ (resp. $S \succeq 0$) indicates that X (resp. S) must be positive semidefinite. Positive semidefiniteness of a matrix can be characterized in several ways (see e.g. [24]). A symmetric real matrix $A \in \mathbb{R}^{n \times n}$ is said to be positive semidefinite if (*i*) $x^T A x \geq 0$ for all $x \in \mathbb{R}^n$, (*ii*) all the eigenvalues of A are nonnegative, (*iii*) there exists a matrix R of full column rank such that $A = R^T R$. Furthermore, $\mathbf{Tr}$ denotes the *trace*–operator. The trace of a matrix A is equal to the sum of its diagonal elements. A useful easy–to–check property of the trace operator is $\mathbf{Tr}\, AB = \mathbf{Tr}\, BA$, where A and B are matrices of appropriate sizes. Also, the trace of a matrix is equal to the sum of its eigenvalues.

For the pair (P,D) weak duality holds, i.e. $b^T y \leq \mathbf{Tr}\, CX$ for any pair of feasible solutions. Perfect duality (i.e. $\inf \mathbf{Tr}\, CX = \sup b^T y$) holds if one of (P) and (D) is strictly feasible; (P) (resp. (D)) is strictly feasible if a strictly interior solution $X \succ 0$ (resp. $S \succ 0$) exists. Then infeasibility of one implies unboundedness of the other and vice versa. Strong duality holds when both (P) and (D) are strictly feasible; then the optimal solution (X, S) exists and is complementary. As stated before, the primal–dual pair (P,D) can be solved in polynomial time using an interior point algorithm.

3. A semidefinite relaxation of the SAT problem

3.1 Elliptic representations and approximations

In [19] elliptic approximations for satisfiability problems are introduced. Let us note that each individual clause has its own elliptic *representation*, given by

$$\mathcal{E}_k = \{x \in \mathbb{R}^n \mid (a_k^T x - 1)^2 \leq (\ell(\mathbf{C}_k) - 1)^2\}. \tag{3.1}$$

It is easy to check that any assignment $x \in \{-1, 1\}^n$ satisfying clause $\mathbf{C}_k$ lies in the interior or on the boundary of $\mathcal{E}_k$, since for such an assignment it holds that $-\ell(\mathbf{C}_k) + 2 \leq a_k^T x \leq \ell(\mathbf{C}_k)$. On the other hand if x does not satisfy clause $\mathbf{C}_k$ it lies outside of $\mathcal{E}_k$ (since then $a_k^T x = -\ell(\mathbf{C}_k)$). Note that in the case that $\ell(\mathbf{C}_k) \leq 2$ the inequality in (3.1) may be replaced by equality.

Thus, basically the satisfiability problem can be expressed as finding a $\{-1, 1\}$ vector x lying in the intersection of m ellipsoids, i.e.

$$x \in \bigcap_{k=1}^{m} \mathcal{E}_k \cap \{-1, 1\}^n.$$

However, it is hard to characterize the intersection of two or more ellipsoid explicitly, so we need to find another way to aggregate the information contained in the m separate ellipsoids. It appears plausible to take the sum over all these ellipsoids; this again yields an ellipsoid. Unfortunately, during summation the discriminative properties of the separate ellipsoids are partly lost, therefore we speak of an *approximation* of a propositional formula. Associating a nonnegative weight w_k with each individual clause, we obtain the following ellipsoid.

$$\mathcal{E}(w) = \{x \in \mathbb{R}^n \mid x^T A^T W A x - 2w^T A x \leq r^T w\}, \tag{3.2}$$

where $w \in \mathbb{R}^m$, $W = \operatorname{diag}(w)$ and $r_k = \ell(\mathbf{C}_k)(\ell(\mathbf{C}_k) - 2)$.

Note that $x \in \mathcal{E}(w)$, for all $w \geq 0$, is a necessary (but not sufficient) condition for any $x \in \{-1, 1\}^n$ to be a satisfying assignment; for 2SAT it is also a sufficient condition. Let us state this in a lemma.

Lemma 3.1 *Let Φ be a CNF formula. If $x \in \{-1, 1\}^n$ is a satisfying assignment of Φ, then $x \in \mathcal{E}(w)$ for any $w \geq 0$.*

3.2 A sufficient condition for unsatisfiability

Let us again consider (3.2). By Lemma 3.1, for any satisfying assignment $x \in \{-1, 1\}^n$ it must hold that

$$x^T A^T W A x - 2w^T A x - r^T w \leq 0, \tag{3.3}$$

for all $w \geq 0$. Thus, we have a necessary condition for satisfiability. Reversing this argument gives us a sufficient condition for unsatisfiability.

Lemma 3.2 *Let Φ be a CNF formula. If for some $w \geq 0$ it holds that $x \notin \mathcal{E}(w)$ for all $x \in \{-1,1\}^n$, then Φ is unsatisfiable.*

To illustrate the usefulness of this approach let us briefly consider an example.

Example We consider the well known *pigeon hole* formulas, which can be stated as follows:

Given $h+1$ pigeons and h holes, decide whether it is possible to put each pigeon in at least one hole, while no two pigeons may be put in the same hole.

For more details on this problem and its SAT encoding see Section 5. The set of clauses in such a formula can be divided into a set of *long* clauses and a set of *short* clauses. It can be shown that for the instance with h holes and $h+1$ pigeons, when all long clauses are given a weight of one, and all short clauses a weight of $h-1+\frac{2}{h+1}$, the minimal value of (3.3) over the sphere $x^Tx = n$ (this is a relaxation of the integrality constraints) is equal to $4(h-1+\frac{1}{h+1}) > 0$. These values can be explicitly computed by using the specific structure of the eigenspace of the matrix A^TWA associated with the pigeon hole formulas. Thus, even using a low dimensional weight vector, pigeon hole formulas can be shown to be contradictory.□

Now it must be our aim to try to formulate an optimization problem for finding a set of weights with the desired property in an automated way. To this end we rewrite (3.3). Introducing an additional Boolean variable $x_{n+1} \in \{-1,1\}$ we obtain the inequality

$$x^TA^TWAx - 2x_{n+1}w^TAx - r^Tw \leq 0. \tag{3.4}$$

Once again, if Φ is satisfiable then inequality (3.4) is satisfied by some $\{x_1,\dots,x_{n+1}\} \in \{-1,1\}^{n+1}$ for all $w \geq 0$.

We can rewrite condition (3.4) as $\tilde{x}^TQ(w)\tilde{x} \leq 0$, where $\tilde{x} := [x_1,\dots,x_n,x_{n+1}]$ and $Q(w)$ is the $(n+1)\times(n+1)$ matrix:

$$Q(w) := \begin{bmatrix} A^TWA - \frac{r^Tw}{n}I & -A^Tw \\ -w^TA & 0 \end{bmatrix}.$$

We can further add a so-called correcting vector $u \in \mathbb{R}^n$ to Q to obtain

$$\tilde{Q}(w,u) := \begin{bmatrix} A^TWA - \frac{r^Tw}{n}I - \operatorname{diag}(u) & -A^Tw \\ -w^TA & e^Tu \end{bmatrix}.$$

Note that $\tilde{x}^TQ(w)\tilde{x} = \tilde{x}^T\tilde{Q}(w,u)\tilde{x}$ because $\tilde{x} \in \{-1,1\}^{n+1}$.

Following Lemma 3.2 we are interested in minimizing $x^T\tilde{Q}(w,u)x$ over the $n+1$–dimensional $\{-1,1\}$–vectors. In particular, if the optimal value to this problem is positive for some $w \geq 0$ and u then Φ cannot be satisfiable; so we want to maximize it over u and w. Relaxing the integrality constraint to a single spherical constraint $x^Tx = n+1$, a lower bound of this problem is given by the minimal eigenvalue of $\tilde{Q}(w,u)$ multiplied by the squared norm of x.

So we are facing the problem of maximizing the minimal eigenvalue of $\tilde{Q}(w,u)$. This can be expressed as a semidefinite programming problem:

$$
\text{(P)} \quad \begin{array}{ll} \sup & (n+1)\lambda \\ \text{s.t.} & \tilde{Q}(w,u) \succeq \lambda I, \\ & w \geq 0. \end{array}
$$

We call the optimal value of this optimization problem the *gap* of formula Φ.

Definition 3.1 *The gap of a formula Φ is defined as the optimal value of the optimization problem (P).*

$$
gap(\Phi) := \sup_{w \geq 0, u} (n+1)\lambda_{\min}\left(\tilde{Q}(w,u)\right).
$$

Thus, by Lemma 3.2, we have the following corollary.

Corollary 3.1 *If a formula Φ has a positive gap, it is unsatisfiable.*

We will show that the converse is also true if Φ is a *2–SAT* formula. Furthermore, the formulas corresponding to a specific type of covering problems, which include the notorious *pigeon hole problems* and *mutilated chess boards*, have a positive gap.

3.3 The dual relaxation

The Lagrangian dual of the optimization problem (P) can be simplified to the following *semidefinite feasibility problem*:

$$
\text{(D)} \quad \begin{array}{ll} \text{find} & Y \in \mathbb{R}^{n \times n},\ y \in \mathbb{R}^n \\ \text{s.t.} & a_k^T Y a_k - 2a_k^T y \leq r_k, \quad 1 \leq k \leq m, \\ & \operatorname{diag}(Y) = e, \\ & Y \succeq yy^T. \end{array}
$$

To see that (D) is indeed the dual of (P), note that the constraint on the diagonal of Y is follows by dualizing the correcting vector, while the first set of constraints is obtained by rewriting the condition

$$
\mathbf{Tr} \begin{bmatrix} a_k a_k^T - \frac{r_k}{n} I & -a_k \\ -a_k^T & 0 \end{bmatrix} \begin{bmatrix} Y & y \\ y^T & 1 \end{bmatrix} \leq 0,
$$

and using that $\operatorname{diag}(Y) = e$. The semidefinite constraint follows using the *Schur complement* reformulation:

$$
\begin{bmatrix} Y & y \\ y^T & 1 \end{bmatrix} \succeq 0 \Leftrightarrow Y - yy^T \succeq 0.
$$

Note that (D) can be derived from the elliptic approximations of clauses in a more direct way; in fact it can be seen to be the 'standard' SDP relaxation of the conditions (3.1) as follows. Expanding (3.1), we obtain $a_k^T x x^T a_k - 2a_k^T x \leq r_k$. This equation can be linearized by replacing the variables x_i by vectors $v_i \in \mathbb{R}^{n+1}$ with the additional requirement that $\|v_i\| = 1$, adding a homogenizing vector v_0, and letting $Y = V^T V$ and $y = V^T v_0$ (here V is the matrix with the vectors v_i as columns). The constraint on the diagonal of Y follows immediately, and the semidefinite constraint follows using that $[V\ v_0]^T[V\ v_0] \succeq 0$ and the Schur complement.

Furthermore, note that (D) is closely related to MAX3SAT relaxation proposed by Karloff and Zwick [16]. It appears to be slightly weaker, since in their formulation the constraints are further disaggregated; see also Section 7. In addition, we are only interested in proving unsatisfiability, rather than in MAX–SAT solutions.

3.4 Some properties of the gap

Let us now consider (P) and (D) to derive some easy properties of the gap. If (D) is infeasible, then the associated formula Φ must be unsatisfiable. This observation also follows from the *perfect duality* relation which holds for (P) and (D): if (D) is infeasible then the primal (P) is unbounded ($\text{gap}(\Phi) = \infty > 0$) which implies that Φ is unsatisfiable. It may be noted that perfect duality holds for (P) and (D) because (P) is always strictly feasible. Let us state this in a lemma.

Lemma 3.3 *Given a formula Φ. It holds that gap (Φ) is either zero or infinity. If gap$(\Phi) = \infty$, then Φ is unsatisfiable.*

Furthermore, the gap is in a sense monotone.

Lemma 3.4 *Let Φ be a CNF–formula and let $\Psi \subseteq \Phi$. Then it holds that gap$(\Psi) \leq$ gap(Φ).*

Proof: Consider (P). Add to (P) the constraints that $w_k = 0$ for all clauses $\mathbf{C}_k$ which occur only in Φ. Solving this modified version of (P), gap(Ψ) is obtained. Obviously, it is a more restricted version of gap(Φ), so the lemma follows. □

This leads immediately to the following easy corollary.

Corollary 3.2 *The gap is monotone under unit resolution.*

Proof: Unit resolution can be regarded as the addition of unit clauses to a formula. By the previous lemma, the gap cannot decrease. □

In the next sections the gap for some specific SAT problems is considered.

4. The gap for 2SAT problems

In this section we prove that $\text{gap}(\Phi) = \infty$ if Φ is an unsatisfiable 2SAT formula. It is well known that 2SAT problems are in fact solvable in linear time [2]. Furthermore, Goemans and Williamson [12] and Feige and Goemans [11] gave approximation algorithms with performance guarantee for the MAX2SAT problem, based on semidefinite programming. So it is not our aim to improve on any of the above algorithms; we merely want to show some properties of the gap approach. To this end we first characterize infeasibility of 2SAT formulas.

Definition 4.1 (Autarky) *[22, 17] Let Φ be any CNF formula. A vector $z \neq 0$ is called a linear autarky of Φ if $Az \geq 0$.*

Lemma 4.1 *An autarky z indicates a satisfiable subformula Ψ of Φ. The formula $\Phi \backslash \Psi$ is satisfiability equivalent to Φ.*

Proof: For each linear inequality one of two possibilities is applicable:

1. All $z_i = 0$, $i \in I_k \cup J_k$ (so $a_k^T z = 0$).
2. At least one $z_i \neq 0$, $i \in I_k \cup J_k$ (so $a_k^T z \geq 0$).

In the second case, $a_k^T \text{sgn}(z) \geq 2 - |I_k \cup J_k| = b_k$, implying that sgn($z$) restricted to its nonzero entries is a satisfying assignment for the subformula Ψ that is obtained by taking all clauses of Φ associated with linear inequalities with property 2. The formula $\Phi \backslash \Psi$ is satisfiability–equivalent to the original formula (i.e. it is satisfiable if and only if Φ is satisfiable). □

We also make use of the notion of minimal unsatisfiability.

Definition 4.2 (Minimal unsatisfiability) *An unsatisfiable CNF formula Φ is called minimally unsatisfiable if a satisfiable formula is obtained by omitting any given clause from Φ.*

By Lemma 3.4 we can restrict ourselves to the case that Φ is a minimally unsatisfiable formula.

Lemma 4.2 *If a formula Φ is minimally unsatisfiable, then $Az \not\geq 0$ for all $0 \neq z \in \mathbb{R}^n$.*

Proof: If the condition in this lemma does not hold, the formula has an autarky, contradicting the fact that it is minimally unsatisfiable. □

Corollary 4.1 *If Φ is minimally unsatisfiable, then A is of full rank.*

Now we can prove the key lemma of this section.

Lemma 4.3 *Let Φ be a minimally unsatisfiable 2SAT formula. Then gap(Φ) $= \infty$.*

Proof: From the semidefinite constraint we conclude that $a^T Y a \geq (a^T y)^2$ for any vector a. This implies that

$$(a_k^T y)^2 - 2a_k^T y \leq a_k^T Y a_k - 2a_k^T y \leq r_k \equiv 0$$

since $r_k = 0$ for 2SAT (for all $k = 1, \ldots, m$), from which it follows that $0 \leq a_k^T y \leq 2$. The minimal unsatisfiability of Φ implies that $y = 0$. So, $a_k^T Y a_k = 0$ for all $k = 1, \ldots, m$. Since $Y \succeq 0$ the a_k's must lie in the nullspace of Y[1], and since A is of full rank (rank(A) $= n$) this implies $Y = 0$, contradicting the condition diag $(Y) = e$ of (D). We conclude that (D) is infeasible, implying that gap(Φ) $= \infty$. □

The main theorem of this section follows easily.

[1]Let $Y = LL^T$, then $0 = a_k^T Y a_k = \|L^T a_k\|^2$, or $L^T a_k = 0$ which implies $Y a_k = 0$.

Theorem 4.1 *Let Φ be any 2SAT formula. It holds*

$$gap(\Phi) = \begin{cases} \infty & \text{if } \Phi \text{ unsatisfiable;} \\ 0 & \text{if } \Phi \text{ satisfiable.} \end{cases}$$

Proof: Let Φ be a unsatisfiable 2SAT formula and let Ψ be a minimally satisfiable subformula of Φ. By Lemma 4.3, $\text{gap}(\Psi) = \infty$, Lemma 3.4 implies that $\text{gap}(\Phi) = \infty$. Conversely, assume now that Φ is a satisfiable 2SAT formula. Using any satisfying assignment x, a feasible solution $Y = xx^T$, $y = x$ to (D) can be constructed, implying that $\text{gap}(\Phi) = 0$. □

If $\text{gap}(\Phi) = 0$ we can use the dual solution (Y, y) to construct a satisfying assignment x. First we set the entries of x corresponding to nonzero entries of y to $x = \text{sgn}(y)$ from the proof of Lemma 4.3 we know that y is an autarky. For all clauses that are not yet satisfied, a satisfying assignment can be constructed by considering Y^*, which denotes the matrix Y restricted to the rows and columns corresponding to the zero entries of y. Note that $a_k^T y = 0$ implies that $a_k^T Y a_k = 0$, so $Y_{ij} = -a_{ki}a_{kj}$ where $I_k \cup J_k = \{i, j\}$. So we can assume that each of the rows and columns of Y^* contains an off-diagonal element that is equal to ± 1. Fixing all ± 1 elements, such a matrix Y^* can be completed to a rank one $\{-1, 1\}$ matrix that is feasible in (D). From this matrix a $\{-1, 1\}$ solution x can easily be deduced. This construction is equivalent to the one given in [11].

5. The gap for a class of covering problems

In this section we consider SAT encodings of a particular class of covering problems, and show that these can be shown to be contradictory by our SDP approach.

5.1 A specific class of covering problems

Let V be a set of n propositional variables. Let $\mathcal{S} = \{S_1, \ldots, S_N\}$ and $\mathcal{T} = \{T_1, \ldots, T_M\}$ be sets of subsets of V. Both $\mathcal{S}$ and $\mathcal{T}$ form a partition of V. Furthermore, let us assume that $M < N$. Consider the following CNF formula Φ_{CP}.

$$\bigvee_{i \in S_k} p_i, \qquad 1 \le k \le N \tag{5.1}$$

$$\neg p_i \vee \neg p_j, \quad i, j \in T_k, i \neq j, \ 1 \le k \le M. \tag{5.2}$$

Let us now verify that this problem is unsatisfiable (when $M < N$), using *cutting planes* (this construction is well known, see e.g. [6]). To this end we introduce some more notation. By e_S (e_{ij}) we denote the vector with ones in the positions $i \in S$ (i and j) and zeros elsewhere. The integer linear feasibility formulation can be stated as

$$(\text{IP}_{CP}) \quad \begin{array}{ll} \text{find} & x \in \{-1, 1\}^n \\ \text{s.t.} & e_{S_k}^T x \ge 2 - |S_k|, \quad 1 \le k \le N \\ & e_{ij}^T x \le 0, \qquad\qquad i, j \in T_k, i \neq j \ 1 \le k \le M. \end{array}$$

Obviously, it is easy to find a solution of the linear relaxation of this problem (i.e. when the integrality constraints are relaxed to $-e \le x \le e$), by setting all variables to 0. However, using cutting planes it can be shown that for $\{-1,1\}$–variables, the set of inequalities for a set T_k implies that

$$e_{T_k}^T x \leq 2 - |T_k|, \; 1 \leq k \leq M. \tag{5.3}$$

A cutting plane is obtained by taking nonnegative combinations of the constraints, and subsequently adjusting the right hand side of the resulting equality, such that it is as sharp as possible. To determine this right hand side, it is used that we are dealing with binary variables [4].

For completeness, it is shown how to derive (5.3). Taking a subset $U \subseteq T_k$, $|U| = 3$, and summing the (three) inequalities associated with this set, the inequality $2e_U^T x \leq 0$ is obtained. Since for any $\{-1,1\}$ vector x it holds that $e_U^T x$ is *odd*, the right hand side may be rounded down to the largest odd integer smaller than zero, thus we find that $e_U^T x \leq -1$. More generally, suppose we are given a set $U \subset T_k$ and an inequality $e_U^T x \leq 2 - |U|$. In addition let $j \in T_k \backslash U$, and denote $\overline{U} = U \cup \{j\}$. Summing for all $i \in U$ the inequalities $e_{ij}^T x \leq 0$ (with weight 1) and the initial inequality with weight $|U| - 1$, we obtain that $|U| e_{\overline{U}}^T x \leq (|U| - 1)(2 - |U|)$. Dividing both sides by $|U|$, and rounding the right hand side down to the nearest integer with same parity as $|\overline{U}|$ (thus the right hand side becomes $1 - |U|$, which is valid, since $((|U| - 1)/|U|)(2 - |U|) < 3 - |U|$) we obtain

$$e_{\overline{U}}^T x \leq 1 - |U| = 2 - |\overline{U}|.$$

We conclude that (5.3) is indeed implied by the inequalities in (IP$_{CP}$).

Summing all the inequalities (5.3), and using that $\mathcal{T}$ partitions V, we find that $-n \leq e^T x \leq 2M - n$. Similarly, taking the sum over the first set of inequalities in (IP$_{CP}$), we have $2N - n \leq e^T x \leq n$. Combining, we get

$$2N \leq e^T x + n \leq 2M, \tag{5.4}$$

from which it follows that $N \leq M$, implying the infeasibility of (IP$_{CP}$) and thus the unsatisfiability of Φ_{CP} when $M < N$. We conclude that using this cutting plane technique, formulas of the form Φ_{CP} are solved in polynomial time.

Surprisingly, other techniques often require large running times to solve formulas of this type. Indeed, Haken [14] proved that *resolution* requires a running time that is exponential in the size of the formula to solve the *pigeon hole formulas*, which fit the format Φ_{CP} (as is shown below). Note that to find the cutting plane proof in an efficient way, additional problem–specific information is required. It is our aim to show in this section that using our semidefinite programming approach formulas of the format Φ_{CP} are proved contradictory in polynomial time, while no additional problem–specific information whatsoever is required. Before proving this, let us first give two examples of problems that fit in this format.

Pigeon hole formulas Let us state again the pigeon hole principle:

Given $h+1$ pigeons and h holes, decide whether it is possible to put each pigeon in at least one hole, while no two pigeons may be put in the same hole.

We now argue that the standard encoding of the pigeon hole problem fits the format Φ_{CP}.

For each pigeon–hole pair a proposition is introduced; thus, we obtain a total of $h(h+1)$ variables in the set V. The long (positive) clauses (5.1) now express that each pigeon is put in at least one hole; thus there are exactly $N = h+1$ of such clauses that all have length h. The short (negative) clauses (5.2) model that no two pigeons may be put in the same hole simultaneously; for each hole there is a set of short clauses, giving rise to $M = h$ separate sets T_k, each of size $h+1$. It is easy to see that each proposition occurs both exactly once in the sets S_k and exactly once in the sets T_k, thus both $\mathcal{S}$ and $\mathcal{T}$ are a partition of V.

Mutilated chess boards The problem of the mutilated chess board can be expressed as follows:

Given a chess board of size $2s \times 2s$ *squares. Two of its diagonally opposite vertices are removed. Can the resulting 'mutilated' chess board be covered by rectangular dominoes of size* 2×1 *(i.e. a single domino covers exactly two adjacent squares), such that each square is covered exactly once?*

The standard satisfiability coding for this problem is obtained by introducing a proposition for each pair of adjacent squares; thus we need $4(2s^2 - s - 1)$ variables. For each square there is a positive clause (of length 2, 3 or 4) expressing that it must be covered at least once, and a set of negative (2–)clauses expressing that it may be covered at most once. It is strongly conjectured that resolution requires exponential running time on this problem as well (despite the absence of 'long' clauses) [27]. Taking a subset of this set of clauses, a formula of the form Φ_{CP} is obtained. For all the *black* squares we keep the positive clauses, while for all the *white* squares we only use the negative clauses. The first set corresponds to (5.1) and the second set to (5.2). See also the graph in Figure 1; the nodes correspond to individual variables, the drawn edges indicate the *positive* clauses (all cliques with drawn edges constitute one positive clause) and the dotted edges the *negative* clauses (each pair of nodes connected by a dotted edge corresponds to a negative clause). Again, it is easy to see that each variable occurs in exactly one of the $N = 2s^2$ positive clauses, and in exactly one of the $M = 2s^2 - 2$ sets of negative clauses.

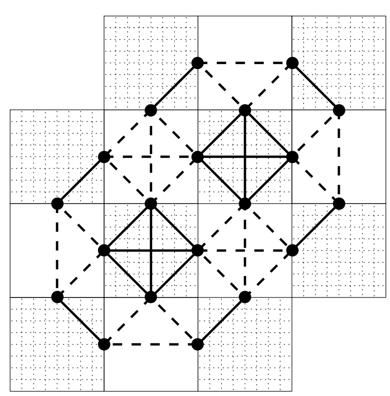

Figure 1: The mutilated chess board for $s = 2$.

Let us now consider the semidefinite relaxation of Φ_{CP}. The SDP relaxation can be denoted as (see also (D))

$$
(\mathrm{SD}_{CP}) \qquad \begin{array}{lll}
\text{find} & Y \in \mathbb{R}^{n\times n}, y \in \mathbb{R}^n & \\
\text{s.t.} & e_{S_k}^T Y e_{S_k} - 2e_{S_k}^T y \le |S_k|(|S_k|-2), & 1 \le k \le N \\
& e_{ij}^T Y e_{ij} + 2e_{ij}^T y \le 0, & i,j, i \ne j \in T_k,\ 1 \le k \le M \\
& \mathrm{diag}(Y) = e, & \\
& Y \succeq yy^T. &
\end{array}
$$

We prove the following Theorem.

Theorem 5.1 *The semidefinite relaxation (SD_{CP}) of Φ_{CP} is infeasible (if $M < N$). Equivalently, gap(Φ_{CP}) $= \infty$.*

Proof: Note that from the semidefinite constraint it follows that $a^T Y a \ge (a^T y)^2$ for any n–vector a. Thus it follows that

$$(e_{S_k}^T y)^2 - 2e_{S_k}^T y \le e_{S_k}^T Y e_{S_k} - 2e_{S_k}^T y \le |S_k|(|S_k|-2),$$

implying that

$$2 - |S_k| \le e_{S_k}^T y \le |S_k|,\ 1 \le k \le N. \tag{5.5}$$

Now we consider the inequalities corresponding to the sets T_k. Taking the sum over all the inequalities corresponding to set T_k, k fixed, we find that

$$e_{T_k}^T Y e_{T_k} + (|T_k|-2)e_{T_k}^T \mathrm{diag}(Y) + 2(|T_k|-1)e_{T_k}^T y \le 0.$$

To verify this, note that each diagonal element Y_{ii}, $i \in T_k$, occurs in exactly $|T_k|-1$ inequalities; similarly, each linear term y_i, $i \in T_k$, occurs in exactly $|T_k|-1$ inequalities as well. Simplifying this expression using that $\mathrm{diag}(Y) = e$, we obtain

$$e_{T_k}^T Y e_{T_k} + 2(|T_k|-1)e_{T_k}^T y \le -|T_k|(|T_k|-2).$$

Using the semidefinite constraint again, we conclude that

$$(e_{T_k}^T y)^2 + 2(|T_k|-1)e_{T_k}^T y \le e_{T_k}^T Y e_{T_k} + 2(|T_k|-1)e_{T_k}^T y \le -|T_k|(|T_k|-2),$$

implying that

$$-|T_k| \le e_{T_k}^T y \le 2 - |T_k|,\ 1 \le k \le M.$$

Summing these inequalities we find that $-n \le e^T y \le 2M - n$ while from (5.5) we have that $2N - n \le e^T y \le n$, implying that $2N \le e^T y + n \le 2M$ (note that this is equivalent to what we obtained using cutting planes, see (5.4)). Thus we conclude that (SD$_{CP}$) is infeasible when $N > M$. $\square$

Thus we have the following corollary.

Corollary 5.1 *Pigeon hole formulas and the mutilated chess board problem have infinite gap, and thus are proven to be contradictory in polynomial time.*

It may be noted that the proof of Theorem 5.1 and the cutting plane refutation of Φ_{CP} are essentially very similar. Indeed, the cutting planes (5.3) are automatically implied in (SD$_{CP}$).

5.2 Application to graph coloring

A famous result by Lovász is his 'sandwich' theorem [18], which states that for an undirected graph $G = (V, E)$ in polynomial time (using semidefinite programming), a number $\vartheta(G)$ can be computed which is bounded from above by the graph's coloring number $\gamma(G)$ (i.e. the minimal number of colors required to color the vertices of the graph such that no two adjacent vertices have the same color), and from below by its clique number $\omega(G)$ (i.e. the maximal complete subgraph of G). Applying our result from the previous section to the graph coloring problem (GCP) we obtain a similar result.

Suppose we are given a graph $G = (V, E)$ and a set of colors C. We introduce a proposition for each vertex–color combination. Then the GCP can be modelled as a formula Φ_{GCP} containing a set of $|V|$ long clauses, expressing that each vertex should be colored by at least one color, and a set of short clauses expressing that no two vertices may get the same color. We can construct the semidefinite relaxation of Φ_{GCP} in the usual way; we refer to it as (SD_{GCP}). Now let C^* be the smallest set of colors for which (SD_{GCP}) is feasible. Such a set must exist, since for $|C^*| \geq |V|$ the GCP and thus its relaxation (SD_{GCP}) are trivially feasible. We have the following Theorem.

Theorem 5.2 *It holds that $\omega(G) \leq |C^*| \leq \gamma(G)$.*

Proof: First note for any set of color C with $|C| < \omega(G)$, Φ_{GCP} has a subformula of the form Φ_{CP}, hence by Lemma 3.4 it has gap infinity. This subformula corresponds to the set of clauses corresponding to a clique of size $|C| + 1$ or larger. Now consider set C^*. It holds that $|C^*| \geq \omega(G)$, since otherwise (SD_{GCP}) would be infeasible. Also, since removing one color from C^* implies infeasibility of (SD_{GCP}) (by assumption), it holds that $|C^*| \leq \gamma(G)$. This proves the theorem. □

So by applying a binary search on the size of C, a number similar to Lovász ϑ–number can be computed. Obviously, $\vartheta(G)$ can be computed more efficiently. The dimension of its associated semidefinite programming problem is $|V|$, where as the dimension of the gap relaxation is $|V||C|$. A relaxation of dimension n can be solved in $O(\sqrt{n})$ iterations, where each iteration requires $O(n^3)$ operations, and its actual practical performance also depends on the density of the graph under consideration. For graphs with relatively small coloring number, say $\gamma(G) = O(\sqrt{n})$, the difference in computational efficiency might be acceptable. Based on some experiments on random graphs, we can say that $|C^*|$ and $\vartheta(G)$ give quite different bounds on $\gamma(G)$.

6. The gap for 3SAT problems

6.1 A modified formulation

So far we have seen that several classes of formulas can be solved by the gap approach. Let us now turn our attention to the most general class of CNF formulas, the 3SAT problems. First we have a negative result for pure 3SAT problems (the class of CNF formulas in which all clauses have length 3).

Lemma 6.1 *Suppose Φ is a pure 3SAT problem. It holds that gap $(\Phi) = 0$.*

Proof: It is easy to verify that the solution $Y = I$, $y = 0$ is a feasible assignment of (D) since $a_k^T a_k = 3 = r_k$ for all clauses. By duality, gap$(\Phi) = 0$. $\square$

So it is not possible to prove an unsatisfiable 3SAT formula to be contradictory straightaway. Of course the gap can be computed in nodes of a branching tree; during the branching process 2–clauses are created, thus making it possible for the SDP approach to be successful in specific cases. Note that (also) in this respect, the SDP approach is stronger than the LP approach (as mentioned in Section 2.1). However, computing the gap is computationally rather expensive, and especially in a DPLL–like branching algorithm (including unit resolution, although this can be simulated by the semidefinite relaxation as well) [7] the overhead will be substantial with the current state–of–the–art implementations. Several possibilities in this respect are discussed in the next section.

For now, let us slightly reformulate our gap relaxation to be able to say a little more about 3SAT formulas. Consider again the elliptic representation $\mathcal{E}_k$ associated with a clause k (3.1). By introducing a *parity*–variable, the inequality is turned into an equality constraint.

$$\mathcal{E}_k^p = \{x \in \mathbb{R}^n, 0 \leq s_k \leq 1 \mid (a_k^T x - 1)^2 + 4s_k = 4\},$$

where for all feasible $\{-1, 1\}$ assignments to x it holds that $s_k \in \{0, 1\}$. The ellipsoid $\mathcal{E}_k^p$ has the semidefinite relaxation

$$a_k^T Y a_k - 2a_k^T y + 4s_k = 3. \tag{6.1}$$

Obviously, the trivial solution (Lemma 6.1), is still feasible when simply setting all s_k to 0. However, if we are now going to maximize the sum of the s_k's, an other solution than the trivial solution may be obtained. Thus we define the *parity* gap. Consider the semidefinite optimization problem (D').

$$(\mathrm{D'}) \quad \begin{array}{rll} \max & \displaystyle\sum_{k=1}^{m} s_k & \\ \text{s.t.} & a_k^T Y a_k - 2a_k^T y + 4s_k = 3, & 1 \leq k \leq m, \\ & \operatorname{diag}(Y) = e, & \\ & Y \succeq yy^T, & \\ & 0 \leq s_k \leq 1, & 1 \leq k \leq m. \end{array}$$

Definition 6.1 *The optimal value of optimization problem (D') is called the parity gap of a formula Φ.*

We then have the following lemma.

Lemma 6.2 *If a formula Φ has parity gap zero it is polynomially solvable.*

Proof: If the parity gap is zero, $s_k = 0$ for all $1 \leq k \leq m$. Suppose that a solution exists for which at least one clause is satisfied in exactly two literals. Using this solution, a solution of (D') can be constructed, with strictly positive objective value. If (D') has objective value zero, no such solution exists, so any satisfying assignment satisfies each clause in exactly one or three literals; so the formula is equivalent to an XOR–SAT formula. Checking whether such a solution exists can be done in polynomial time [23, 29]. □

A specific class of formulas that appears to be a likely candidate to have parity gap zero, is the class of *doubly balanced formulas* [19, 29]. By definition, for doubly balanced formulas it holds that A^TA is a diagonal matrix and $A^Te \equiv 0$. Eliminating the s_k variables from (D') it can be rewritten to

$$\text{(D")} \quad \begin{array}{lll} \min & \mathbf{Tr} \begin{bmatrix} A^TA & -A^Te \\ -e^TA & 0 \end{bmatrix} \begin{bmatrix} Y & y \\ y^T & 1 \end{bmatrix} & \\ \text{s.t.} & -1 \leq a_k^TYa_k - 2a_k^Ty \leq 3, & 1 \leq k \leq m, \\ & \operatorname{diag}(Y) = e, & \\ & Y \succeq yy^T. & \end{array}$$

It holds that $\text{opt(D')} = \frac{3}{4}m - \frac{1}{4}\text{opt(D")}$. Note that the objective function of (D") is essentially the same objective function as the one used by Goemans and Williamson in their MAX2SAT algorithm, only we are presently considering 3SAT formulas. Using formulation (D") it is straightforward to prove the next lemma.

Lemma 6.3 *A doubly balanced formula has parity gap zero.*

Proof: By the definition of doubly balancedness and the constraint on the diagonal of Y the objective function of (D") reduces to the constant $3m$, implying that the parity gap is equal to zero. □

We also have an autarky result.

Lemma 6.4 *If all $s_k > \frac{3}{4}$, $x = sgn(y)$ is a satisfiable assignment of Φ.*

Proof: Note that

$$(a_k^Ty)^2 - 2a_k^Ty + (4s_k - 3) \leq 0.$$

This implies that

$$1 - 2\sqrt{1 - s_k} \leq a_k^Ty \leq 1 + 2\sqrt{1 - s_k}.$$

If $s_k > \frac{3}{4}$, it holds that $a_k^Ty > 0$. If all s_k have this property, then y is a linear autarky. □

Note that y might be a linear autarky while not all s_k are larger than $\frac{3}{4}$. In this respect, a slightly stronger formulation is obtained by using *a single* slack variable t for all clauses, rather than a separate slack variable s_k for each of the clauses. The objective then becomes to maximize t; all s_k's must be replaced by t, and equality must be replaced by inequality. A drawback of this approach is that an optimum of zero only implies that a polynomially solvable *sub*formula is present. On the other hand, solving this subformula first, may speed up solving the full formula [31].

While it appears that the semidefinite relaxation as developed in this paper is not quite strong enough to completely solve 3SAT problems by itself, it can be used in various other ways. Apart from including it in a complete algorithm (see next section), both its primal and dual solution can be used for heuristic purposes. This is briefly discussed in Section 7.

6.2 Incorporating the gap approach in a complete algorithm

As mentioned in the previous section, the gap can be computed in each node of a branching tree to detect unsatisfiability early, so as to reduce the size of the search tree. Even though at present it appears that this might be computationally too expensive, there are several ways to reduce computational cost. We mention two possibilities.

1. Instead of using a primal–dual algorithm, it is possible to use a dual scaling algorithm to exploit sparsity of $\tilde{Q}$ to the full [3]. Computational experience with MAX2SAT problems indicates that for small sized problems this approach is competitive with other dedicated algorithms for the MAX2SAT problem [9].

2. For larger problems, spectral bundle methods can be used to solve the eigenvalue optimization problem (P) [15]. These have been shown to be able to handle problems with thousands of variables. Such methods solve only (P), so that the dual information is lost.

7. Remarks and further research

7.1 Some empirical observations

Both the primal and dual solutions obtained by solving a semidefinite relaxations can be used for heuristic purposes. The dual solution (Y, y) might be used to try to obtain good approximate MAX–SAT solutions. To illustrate the quality of the solutions thus obtainable, see Figure 2. We restricted ourselves to a set of random pure 3SAT formulas with 100 variables and a varying number of clauses, and used the dual formulation with a single slack variable. The SDPs were solved using the public–domain solver SeDuMi [25]. Approximate solutions are constructed by (i) taking the dual solution and rounding y (the drawn line), (ii) by applying a Goemans–Williamson–like randomized rounding procedure (the dotted line). For comparison we also included the solutions obtained by simply drawing random solutions; note that this in fact corresponds to the Karloff–Zwick algorithm for approximating (pure) MAX3SAT solutions. To obtain a good *ad hoc* lower bound on the optimal solution we applied a greedy weighted local search procedure. Considering the results, it is clear that the local search procedure gives the best results, but interestingly enough the drawn line is consistently above the dotted line, implying that the deterministic one–step procedure of rounding y gives better solutions than the randomized rounding procedure, which in turn is better than just drawing random solutions.

Let us finish this section by mentioning that using the primal optimal solution w, it is possible to identify hard subformulas. To this end, the sum of the weights is bounded (this corresponds to using a single slack variable in the dual formulation). Removing the clauses with ‘small’ weights, a subformula is obtained that appears to be the ‘core’ of the original formula. This yields a technique for finding approximately minimal unsatisfiable subformulas.

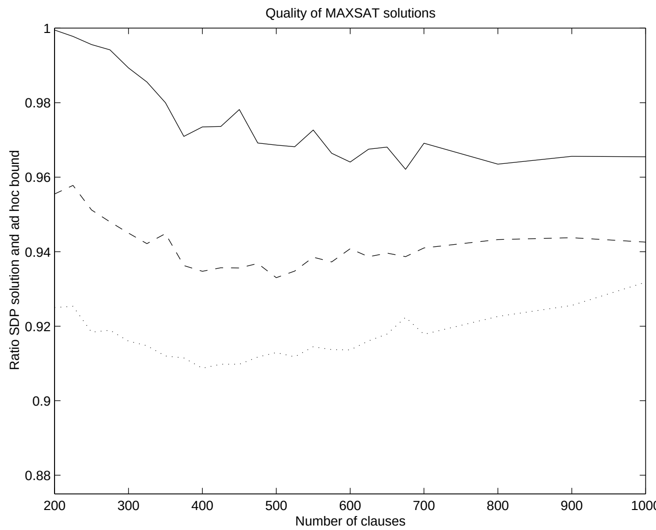

Figure 2: Comparison of MAXSAT solutions.

7.2 More general clause models

So far we have considered a fixed quadratic clause–model based on an elliptic approximation. Obviously there exist many other quadratic clause–models. Basically, all quadratic models $q(x)$ with the following property are valid.

Property 7.1 *x is a satisfying assignment* $\Rightarrow q(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij}x_ix_j + \sum_{k=1}^{n} q_kx_k + q_0 \leq 0.$

As an example, let us consider the clause $\mathbf{C} = p_1 \vee p_2 \vee p_3$. Substituting the satisfying assignments of $\mathbf{C}$ in $q(x)$, a set of 7 inequalities is obtained. These inequalities, which are linear in the coefficients q_{ij} and q_k, define a cone. The extreme rays of this cone give rise to the following generic set of valid quadratic cuts (S).

$$\begin{array}{rcl}
x_1x_2 + x_1x_3 - x_2 - x_3 & \leq & 0 \\
x_1x_2 + x_2x_3 - x_1 - x_3 & \leq & 0 \\
x_1x_3 + x_2x_3 - x_1 - x_2 & \leq & 0 \\
-x_1x_2 - x_1x_3 - x_2x_3 - 1 & \leq & 0 \\
-x_1x_2 + x_1 + x_2 - 1 & \leq & 0 \\
-x_1x_3 + x_1 + x_3 - 1 & \leq & 0 \\
-x_2x_3 + x_2 + x_3 - 1 & \leq & 0
\end{array}$$

An equivalent set is given by Karloff and Zwick [16]. So, each valid quadratic model of $\mathbf{C}$ is a linear combination of this set of inequalities. In particular, the elliptic approximation is obtained by taking the sum of the first three inequalities (see (3.1) and use that $x^2 = 1$ for $x \in \{-1, 1\}$). Karloff and Zwick show that using the first three inequalities a 7/8 approximation algorithm for MAX3SAT problems can be obtained. Thus, our relaxation is an aggregated version of the Karloff–Zwick relaxation.

Note that the SDP relaxations can be further strengthened by adding valid inequalities. An example of such cuts are the *triangle inequalities* [11]; note that many of these are in fact implied by (S) (namely those concerning variables that occur jointly in some clause). Another possibility is using (see also (3.1))

$$(a_k^T x - 1)(a_l^T x - 1) \leq (\ell(\mathbf{C}_k) - 1)(\ell(\mathbf{C}_l) - 1) \tag{7.1}$$

for any pair of clauses $\mathbf{C}_k$, $\mathbf{C}_l$ which share one or more variables. The cuts (7.1) are capable of cutting off the trivial feasible solution. This is an example of a valid quadratic cut for *pairs* of clauses. Similar to (S), all valid quadratic cuts may be derived for pairs of clauses.

References

1. F. Alizadeh. *Combinatorial optimization with interior point methods and semi–definite matrices.* PhD thesis, University of Minnesota, Minneapolis, USA, 1991.
2. B. Aspvall, M.F. Plass, and R.E. Tarjan. A linear–time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979.
3. S.J. Benson, Y. Ye, and X. Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. Technical report, Computational Optimization Lab, Department of Management Science, University of Iowa, Iowa City, USA, 1997.
4. V. Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete Mathematics*, 4:305–337, 1973.
5. S.A. Cook. The complexity of theorem proving procedures. In *Proceedings of the 3rd annual ACM symposium on the Theory of Computing*, pages 151–158, 1971.
6. W. Cook, C.R. Coullard, and G. Turan. On the complexity of cutting plane proofs. *Discrete Applied Mathematics*, 18:25–38, 1987.
7. M. Davis, M. Logemann, and D. Loveland. A machine program for theorem proving. *Communications of the ACM*, 5:394–397, 1962.
8. E. de Klerk. *Interior point methods for semidefinite programming.* PhD thesis, Delft University of Technology, Delft, The Netherlands, 1997.
9. E. de Klerk and J.P. Warners. Semidefinite programming techniques for MAX–2–SAT and MAX–3–SAT: Computational perspectives. Technical Report 98–34, Department of Technical Mathematics and Informatics, Faculty of Information Technology and Systems, Delft University of Technology, Delft, The Netherlands, 1998.
10. D. Du, J. Gu, and P.M. Pardalos, editors. *Satisfiability problem: Theory and applications*, volume 35 of *DIMACS series in Discrete Mathematics and Computer Science.* American Mathematical Society, 1997.
11. U. Feige and M. Goemans. Approximating the value of two prover proof systems with applications to MAX 2SAT and MAX DICUT. In *Proc. Third Israel Symposium on Theory of Computing and Systems*, pages 182–189, 1995.
12. M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.
13. J. Gu, P.W. Purdom, J. Franco, and B.W. Wah. Algorithms for the satisfiability (SAT) problem: a survey. In Du et al. [10], pages 9–151.

14. A. Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.

15. C. Helmberg and F. Rendl. A spectral bundle method for SDP. Technical Report ZIB Preprint SC–97–37, Konrad–Zuse–Zentrum, Berlin, 1997.

16. H. Karloff and U. Zwick. A 7/8–approximation algorithm for MAX3SAT? In *Proceedings of the 38th Symposium on the Foundations of Computer Science*, pages 406–415, 1997.

17. O. Kullmann. Investigations on autark assignments. Technical report, Johann Wolfgang Goethe–Universität, Fachbereich Mathematik, 60054 Frankfurt, Germany, 1998. Submitted.

18. L. Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information Theory*, 25:1–7, 1979.

19. H. Van Maaren. Elliptic approximations of propositional formulae. Technical Report 96–65, Faculty of Technical Mathematics and Informatics, Delft University of Technology, Delft, The Netherlands, 1996. To appear in *Discrete Applied Mathematics*.

20. H. Van Maaren. On the use of second order derivatives for the satisfiability problem. In Du et al. [10], pages 677–687.

21. H. Van Maaren and J.P. Warners. Solving satisfiability problems using elliptic approximations - a note on volumes and weights. Technical Report 98-32, Department of Technical Mathematics and Informatics, Faculty of Information Technology and Systems, Delft University of Technology, Delft, The Netherlands, 1998.

22. B. Monien and E. Speckenmeyer. Solving satisfiability in less than 2^n steps. *Discrete Applied Mathematics*, 10:287–295, 1985.

23. T.J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the Tenth Symposium on the Theory of Computing*, pages 216–226, 1978.

24. G. Strang. *Linear algebra and its applications.* Harcourt Brace Jovanovich, third edition, 1988.

25. J.F. Sturm. Using SeDuMi 1. 02, a MATLAB toolbox for optimization over symmetric cones. Technical report, Communications Research Laboratory, McMaster University, Hamilton, Canada, 1998.

26. L. Trevisan, G. Sorkin, M. Sudan, and D. Williamson. Gadgets, approximation and linear programming. In *Proceedings of the 37th Symposium on the Foundations of Computer Science*, pages 617–626, 1996.

27. A. Urquhart. Open problem posed at SAT'98, May 10–14, Paderborn, Germany. 1998.

28. L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM review*, 38:49–95, 1996.

29. J.P. Warners and H. Van Maaren. Recognition of tractable satisfiability problems through balanced polynomial representations. Accepted for publication in *Discrete Applied Mathematics*.

30. J.P. Warners and H. Van Maaren. Solving satisfiability problems using elliptic approximations - Effective branching rules. Technical Report 98–18, Department of Technical Mathematics and Informatics, Faculty of Information Technology and Systems, Delft

University of Technology, Delft, The Netherlands, 1998.

31. J.P. Warners and H. Van Maaren. A two phase algorithm for solving a class of hard satisfiability problems. Technical Report SEN-R9802, Centre for Mathematics and Computer Science (CWI), Amsterdam, The Netherlands, 1998. Accepted for publication in *Operations Research Letters*.