Approximating the Value of Two Prover Proof Systems, With Applications to MAX 2SAT and MAX DICUT

Uriel Feige*
Dept. of Appl. Math. and Comp. Sci.
The Weizmann Institute
Rehovot 76100, Israel

Michel Goemans†
Dept. of Math.
M.I.T.
Cambridge, MA 02139

Abstract

It is well known that two prover proof systems are a convenient tool for establishing hardness of approximation results. In this paper, we show that two prover proof systems are also convenient starting points for establishing easiness of approximation results. Our approach combines the Feige-Lovász (STOC92) semidefinite programming relaxation of one-round two-prover proof systems, together with rounding techniques for the solutions of semidefinite programs, as introduced by Goemans and Williamson (STOC94).

As a consequence of our approach, we present improved approximation algorithms for MAX 2SAT and MAX DICUT. The algorithms are guaranteed to deliver solutions within a factor of 0.931 of the optimum for MAX 2SAT and within a factor of 0.859 for MAX DICUT, improving upon the guarantees of 0.878 and 0.796 of Goemans and Williamson.

1 Introduction

We consider optimization problems defined on Boolean variables $x_1, \ldots, x_n$, in which the objective function $S(x_1, \ldots, x_n)$ to maximize can be expressed as a sum of nonnegative terms involving only two Boolean variables. More precisely, let

$$S(x_1, \ldots, x_n) = \sum_{i<j} w_{ij} f_{ij}(x_i, x_j),$$

where $w_{ij} \geq 0$ and $f_{ij} : \{\texttt{true},\texttt{false}\} \times \{\texttt{true},\texttt{false}\} \longrightarrow \{0,1\}$. Without loss of generality, we can assume that, for every f_{ij}, there exists at least one truth assignment for x_i and x_j which makes f_{ij} equal to 1, and similarly one assignment for which f_{ij} is 0. Thus f_{ij} takes the value 1 for one, two or three out of the four possible truth assignments for x_i and x_j. If f_{ij} depends on both variables, we refer to such functions as and-, xor-, and or-functions respectively, since these cases can be assimilated with the corresponding Boolean operators. The following are the prototypical problems associated with the three types of functions.

1. **MAX DICUT:** Given a directed graph $D = (V, A)$ and a nonnegative weight w_{ij} for each arc (i, j), the goal is to find a set $S \subseteq V$ such that the directed cut $\delta^+(S) = \{(i,j) \in A : i \in S, j \notin S\}$ has maximum total weight. We let x_i be **true** if i belongs to S; all f_{ij}'s are thus and-functions. (In fact, the f_{ij}'s are restricted forms of and-functions, since exactly one of the variables is negated.)

2. **MAX CUT:** Given an undirected graph $G = (V, E)$ and a nonnegative weight w_e for each edge, the goal is to find a set $S \subseteq V$ such that the resulting cut $\delta(S) = \{(i,j) \in E : |\{i,j\} \cap S| = 1\}$ has maximum total weight. Again, we let x_i be **true** if i belongs to S; all f_{ij}'s are thus xor-functions. (In fact, the f_{ij}'s are restricted forms of xor-functions, since variables are never negated.)

3. **MAX 2SAT:** In *proper* MAX 2SAT, we are given a set of clauses of length 2 (i.e. disjunction of two literals, each of the form x_i or $\bar{x}_i$) and a nonnegative weight for each clause, and the goal is to find an assignment maximizing the weight of the satisfied clauses. Thus, for proper MAX 2SAT, all functions f_{ij} are or-functions. In (improper) MAX 2SAT, we allow also unit clauses that contain only one literal.

For the problems above, it is NP-hard to find the optimal solution. Hence the question arises of how well can the optimal solution be approximated in polynomial time. More precisely, for which value of $q \leq 1$

---

*feige@wisdom.weizmann.ac.il. Supported by a Koret Foundation Fellowship (until September 94) and Yigal Alon Fellowship (from October 94).

†goemans@math.mit.edu. Research supported in part by NSF contract 9302476-CCR, DARPA contract N00014-92-J-1799 and an NEC research fund.

0-8186-6915-2/95 $04.00 © 1995 IEEE

is there a polynomial algorithm that is guaranteed to deliver a solution whose weight is at least q times the weight achievable by the optimal solution.

A random assignment of the variables is expected to satisfy 1/4 of the terms in MAX DICUT, 1/2 of the terms in MAX CUT, and 3/4 of the terms in proper MAX 2SAT. Hence one expects to achieve ratios of approximation of at least 1/4, 1/2, and 3/4 respectively (and this can actually be achieved by simple derandomization techniques). For a long time, these were essentially the best approximation ratios known for these problems. (The 3/4 ratio was extended to handle unit clauses in MAX 2SAT [17, 9].)

Recently, major breakthroughs improved our understanding of these problems considerably. On the negative side, Arora et al. [2] showed NP hardness results for approximating MAX 3SAT, which imply, by the facts that MAX 3SAT is in MAX SNP and that the three problems above are MAX SNP-hard [15], that for some $q < 1$, it is NP-hard to approximate these problems within a ratio of q. An explicit upper bound on q can be computed from the proof in [2] and from the chain of reductions that follows in [15]. The current tightest known upper bound on q is above 0.99.

On the positive side, Goemans and Williamson [10] have derived improved approximation algorithms for MAX DICUT, MAX CUT, and MAX 2SAT. Their algorithm is based on first obtaining a vector for each Boolean variable (or each vertex of the graph), and then randomly partitioning these vectors with a uniformly generated hyperplane. The variables whose corresponding vectors lie on one side of the hyperplane are set to 1, and the other variables are set to 0. The vectors are obtained through the solution of a convex relaxation of the problem, or more precisely a semidefinite programming relaxation. The performance guarantees obtained by Goemans and Williamson are 0.87856 for MAX CUT and MAX 2SAT, and 0.79607 for MAX DICUT. This constitutes the first non-trivial improvement in the approximation of any of these problems.

In this extended abstract, we consider the approach of Goemans and Williamson, and show that further improvements can be obtained for MAX DICUT and MAX 2SAT. The improvements are based on considering a stronger semidefinite program than the one considered in [10] and on rounding the corresponding vectors in a non-uniform way. As we show, there is much freedom in selecting the non-uniform rounding scheme. The improvements in performance guarantee are quite significant. For MAX 2SAT, there exists a rounding scheme for which the improved performance guarantee is 0.931 (rather than 0.87856), while for MAX DICUT we obtain an improved performance guarantee of 0.859 (rather than 0.79607). These values were obtained by numerically solving a constrained minimization problem of 3 variables.

It is interesting that the semidefinite programs that lead to the improvements in approximation ratio are exactly those that were considered by Feige and Lovász [7] in their study of one-round two-prover interactive proofs. One-round two-prover proof systems are exceptionally useful for proving hardness of approximation results [7, 6, 14, 1]. Our current work shows that they are also a useful framework for designing approximation algorithms with a strong performance guarantee. We elaborate a bit more on this point.

As part of [7], Feige and Lovász represented one-round two-prover proof systems as a quadratic programming problem. The value of this quadratic program was equal to the acceptance probability of the verifier, under the optimal strategy for the provers. The quadratic program was relaxed to a semidefinite program. The question that Feige and Lovász asked was: under which circumstances, the value of the semidefinite program is 1 iff the value of the original quadratic program is 1? This question had a two-fold motivation. The first motivation concerned a certain parallel repetition conjecture, and is beyond the scope of the current paper. The second motivation was in the design of polynomial time algorithms for the exact solution of certain problems (e.g., it leads to an algorithm of computing the chromatic number of perfect graphs, as in [13, 11]). This second motivation will resurface in Section 3. In the current paper, we are concerned with approximate solutions rather than exact solutions, and hence we ask what is the worst-case ratio between the value of the semidefinite program and the value of the original quadratic program. The answers we give are based on the techniques developed by Goemans and Williamson [10].

Feige and Lovász observe that many NP-hard optimization problems have simple representations as one-round two-prover interactive proofs, and this automatically leads to relaxed positive semidefinite formulations for these problems (see [7] for examples). Hence there is the potential of obtaining improved approximation algorithms for a wide range of NP-hard optimization problems. In the current paper, we concentrate only on a small subset of these problems, which corresponds to one-round two-prover proof systems in which the answer of each prover is only one bit long. The reason we concentrate on this subcase, is that this is the case that is most easily handled by the rounding techniques of Goemans and Williamson.

In the rest of this paper, we no longer refer to interactive proofs and the general Feige-Lovász formulation. We specialize our presentation to the actual NP-hard optimization problems that we study, namely MAX 2SAT and MAX DICUT.

2 Formulations

For the definitions of the three main problems studied in Goemans and Williamson [10] — MAX CUT, MAX 2SAT and MAX DICUT — see the introduction. In this section, we present relaxed semidefinite programs for these problems. The semidefinite programs that are used by our approximation algorithms are equivalent to those used in Feige and Lovász [7]. These are presented in Section 2.2. But first, in Section 2.1, we show that by including a subset of the constraints of [7], we obtain semidefinite relaxations which are equivalent to those used by Goemans and Williamson [10]. In Section 2.3, we discuss additional constraints that can be added to obtain even stronger relaxations. We suspect that these constraints give further improvements to the ratio of approximation for MAX 2SAT, MAX CUT and MAX DICUT, but we have not been able to prove this.

2.1 Relaxations of Goemans and Williamson

Optimization problems of the type that was presented in the introduction can be formulated as quadratic integer programs. We first follow the approach of Feige and Lovász [7].

For every index i, let t_i be equal to 1 if x_i =**true** and 0 otherwise, and let f_i be 1 if x_i =**false** and 0 otherwise. As a result, we have that $t_i + f_i = 1$ and $t_i f_i = 0$. These two constraints guarantee that t_i and f_i are 0-1 and distinct. By including one, two or three of the terms $t_i t_j$, $t_i f_j$, $f_i t_j$ or $f_i f_j$ in the objective function, one can model all three problems.

We can obtain an upper bound on the optimum value by relaxing the t_i's and f_i's to be vectors (say of dimension k) instead of scalars. For example, in the case of MAX DICUT, we obtain the following relaxation (DI_1):

$$\begin{array}{lll} \text{Maximize} & \sum_{(i,j)\in A} w_{ij}(t_i \cdot f_j) & \\ \text{subject to:} & t_i \cdot f_i = 0 & i \in V \\ & t_i + f_i = v_0 & i \in V \\ & t_i, f_i \in R^k & i \in V, \end{array}$$

where v_0 is any unit vector (either given or a variable) and "$\cdot$" denotes the inner product. The constraints $t_i \cdot f_i = 0$ and $t_i + f_i = v_0$ correspond to their scalar counterparts and express the desire of setting variable i to either **true** or **false** but not both (notice that together they imply that $(v_0 - t_i)\cdot(v_0 - f_i) = 0$). One could eliminate the variables f_i by using $f_i = v_0 - t_i$:

$$\begin{array}{lll} \text{Maximize} & \sum_{(i,j)\in A} w_{ij}(t_i \cdot v_0 - t_i \cdot t_j) & \\ \text{subject to:} & t_i \cdot v_0 - t_i \cdot t_i = 0 & i \in V \\ & v_0 \cdot v_0 = 1 & \\ & t_i \in R^k & i \in V, \\ & v_0 \in R^k. & \end{array}$$

Alternatively, the constraints $t_i + f_i = v_0$ can be expressed in terms of inner products in several ways. We can either eliminate v_0 and impose that $(t_i + f_i)\cdot(t_j + f_j) = 1$ for all i and j, or we can impose that $v_0 \cdot (t_i + f_i) = v_0 \cdot v_0 = 1$ and $(t_i + f_i)\cdot(t_i + f_i) = v_0 \cdot (t_i + f_i)$ for all i. As in [10], if the dimension k is large enough, the relaxation can be formulated as a semidefinite program, and is thus polynomially solvable (or, more precisely, can be approximated within an additive ϵ in time polynomial in the input and $\log \frac{1}{\epsilon}$). To ensure this, one needs to set k to be equal to $2n+1$ (where $n = |V|$, if we use the formulation in terms of the $2n+1$ vectors t_i, f_i and v_0) or $n+1$ (since the f_i's linearly depend on the t_i's), or even just $\sqrt{2(n+1)}$ by using a result of Barvinok [5] and Pataki [16] (since the relaxation can be expressed by imposing $n+1$ linear constraints on inner products, see [10]). Both MAX 2SAT and MAX CUT can be similarly upper bounded, by just modifying the objective function.

The formulation above contains only a subset of the constraints that are used in Feige and Lovász [7]. However, in this weakened form, these relaxations can be seen to be equivalent to the relaxations considered in Goemans and Williamson [10]. Indeed, letting $v_i = t_i - f_i$, or $t_i = \frac{1}{2}(v_0 + v_i)$ and $f_i = \frac{1}{2}(v_0 - v_i)$, we can reformulate (DI_1) for $k = n+1$ as:

Maximize

$$\frac{1}{4}\sum_{(i,j)\in A} w_{ij}(1 + v_0 \cdot v_i - v_0 \cdot v_j - v_i \cdot v_j)$$

$$\begin{array}{lll} \text{subject to:} & v_i \cdot v_i = 1 & i \in V \cup \{0\} \\ & v_i \in R^{n+1} & i \in V \cup \{0\}. \end{array}$$

which we call (DI_2). The reduction is identical for MAX CUT and MAX 2SAT, resulting in the relaxations (CUT_2) and (SAT_2). For MAX CUT, the objective function of (CUT_2) does not depend on v_0, since the contribution of any edge (i,j) is of the form $t_i \cdot f_j + f_i \cdot t_j = \frac{1}{4}[(v_0+v_i)\cdot(v_0-v_j)+(v_0-v_i)\cdot(v_0+v_j)] = \frac{1}{2}(1 - v_i \cdot v_j)$.

2.2 Relaxations of Feige and Lovász

The relaxations considered in the previous section can be strengthened by adding valid inequalities. Because of their equivalence, we can either use the Feige-Lovász type of formulation (like (DI_1)), or the Goemans-Williamson one (like (DI_2)). For simplicity, we will be using the latter.

The Feige-Lovász [7] formulation contains the additional constraints $t_i \cdot t_j \geq 0$, $t_i \cdot f_j \geq 0$, $f_i \cdot f_j \geq 0$. Transforming these constraints to the Goemans-Williamson formulation, we obtain

$$v_0 \cdot v_i + v_0 \cdot v_j + v_i \cdot v_j \geq -1 \quad (1)$$

$$-v_0 \cdot v_i - v_0 \cdot v_j + v_i \cdot v_j \geq -1 \quad (2)$$

$$-v_0 \cdot v_i + v_0 \cdot v_j - v_i \cdot v_j \geq -1. \quad (3)$$

The relaxations we consider are obtained by adding the inequalities (1)–(3) for any i and j. The resulting relaxations will be denoted by (DI_3), (CUT_3) and (SAT_3). They can still be formulated as semidefinite programs and can thus be solved arbitrarily closely to optimum in polynomial time. These relaxations are equivalent to the relaxations in [7]. We notice that the inequalities (1)–(3) are not satisfied by any three vectors; for example, taking three coplanar vectors separated by angles of $2\pi/3$ would violate (1) since its left-hand-side is $-3/2$. We also remark that (CUT_3) is identical to (CUT_2). Indeed, given a solution $v_1, \ldots, v_n$ for (CUT_2), one can always select v_0 to be orthogonal to all v_i's resulting in a solution satisfying (1)–(3) and of the same objective value.

Adding the inequalities (1)–(3) is most important for MAX 2SAT. Indeed, consider for example a MAX 2SAT instance with only one clause, say $x_1 \vee x_2$. The objective function in (SAT_2) or (SAT_3) is $\frac{1}{4}(3 + v_0 \cdot v_1 + v_0 \cdot v_2 - v_1 \cdot v_2)$. In (SAT_3), this value cannot be more than 1, but in (SAT_2) it can be made equal to $\frac{9}{8}$ by letting v_0, v_1 and v_2 be coplanar and $v_0 \cdot v_1 = v_0 \cdot v_2 = -v_1 \cdot v_2 = \frac{1}{2}$. By relating the value of a heuristic algorithm to the value of (SAT_2), one can thus not improve beyond a performance guarantee of $\frac{8}{9} = 0.8888\ldots$. On the other hand, we describe an algorithm which delivers a solution within a factor of 0.931 of the value of (SAT_3).

2.3 Additional valid inequalities

It is possible to obtain even stronger relaxations, by allowing any vector v_k to take the role of v_0 in the above constraints. This results in the following valid constraints:

$$v_i \cdot v_j + v_i \cdot v_k + v_j \cdot v_k \geq -1 \quad (4)$$

$$-v_i \cdot v_j - v_i \cdot v_k + v_j \cdot v_k \geq -1 \quad (5)$$

$$-v_i \cdot v_j + v_i \cdot v_k - v_j \cdot v_k \geq -1 \quad (6)$$

$$v_i \cdot v_j - v_i \cdot v_k - v_j \cdot v_k \geq -1. \quad (7)$$

Their validity follows from the fact that these constraints hold for Boolean variables. The relaxations that are obtained by adding the inequalities (4)–(7) for all distinct indices i, j and k will be denoted by (DI_4), (CUT_4) and (SAT_4). (They can still be formulated as semidefinite programs and can thus be solved arbitrarily closely to optimum in polynomial time.)

In the case of MAX CUT, by a result of Barahona and Mahjoub [4], the resulting semidefinite relaxation (CUT_4) has value equal to the maximum cut value for any planar graph (their result in fact holds even if one gets rid of the semidefinite constraints, by considering only the linear constraints obtained by replacing $v_i \cdot v_j$ by a scalar y_{ij} in (4)–(7)). It is interesting to note that the worst instance known for (CUT_2) is the 5-cycle which happens to be planar, and is thus solved optimally by (CUT_4).

One could try to go a step further and add additional valid inequalities to the relaxations. For example, one could consider the so-called cycle inequalities. For any "cycle" C and any even subset F of C, one could add the inequality

$$\sum_{(i,j)\in F} v_i \cdot v_j - \sum_{(i,j)\in C \setminus F} v_i \cdot v_j \geq 2 - |C|. \quad (8)$$

However, as shown by Barahona [3], the inequalities (8) are implied by (4)–(7).

3 Satisfiable formulas

The question of whether there exists a (directed) cut that contains all the edges, or a satisfying assignment for a 2CNF formula, is solvable in polynomial time. Hence we would like our relaxed versions of these problems to capture this fact. For example, for a 2CNF formula in which the sum of the weights of the clauses is W, we would like our semidefinite relaxation of the problem to have value W if and only if the formula is satisfiable. This property did not hold for (SAT_2). A qualitative improvement of the (SAT_3) relaxation over the (SAT_2) relaxation is that this property does hold for (SAT_3). Similarly, for a graph (digraph) in which the sum of the weights of the edges is W, (CUT_3) (or (DI_3)) gives value W if and only if the graph is bipartite (or there is a directed cut with all the edges). The case of (CUT_3) and (DI_3) is actually a special case of Theorem 4.18 in [7]. The case of (SAT_3) was not proved in [7], and can either be proved by extending the techniques of [7], or by using rounding techniques in the spirit of [10]. More specifically, instead of picking a random vector r relative to which to do the rounding, we let $r = v_0$. Only if there remain variables whose value is not resolved by this specific choice of r, we handle them by choosing a random (or any) vector r (as often as needed to fix all variables). The proof rests on the fact that, for any clause say $x_i \vee x_j$, the fact that $v_0 \cdot v_i + v_0 \cdot v_j - v_i \cdot v_j = 1$ implies that at least one of the following three conditions must hold: either $v_0 \cdot v_i > 0$, or $v_0 \cdot v_j > 0$ or $v_i \cdot v_j = -1$. In all three cases, the algorithm produces a satisfying assignment. In fact, the proof works for any formula that is a conjunction of terms, where in each term there is an arbitrary function of two variables (2SAT, CUT, and DICUT are special cases of this).

4 The algorithm

The randomized algorithm of Goemans and Williamson proceeds as follows. First solve the semidefinite relaxation (CUT_2), (SAT_2) or (DI_2) to obtain an optimum set of vectors $v_0, v_1, \ldots, v_n$. In the case of (CUT_2), v_0 is not necessary, since the objective function does not depend on it. Then consider a uniformly selected hyperplane (i.e. its normal r is uniformly distributed on the unit sphere); the hyperplane separates the vectors into two sets. In the case of MAX CUT, the vectors on either side defines the set S output by the algorithm. For MAX DICUT, the set S output corresponds to those vectors on the same side as v_0 (i.e. $S = \{i : \operatorname{sgn}(v_i \cdot r) = \operatorname{sgn}(v_0 \cdot r)\}$). Finally, for MAX 2SAT, a variable x_i is set to **true** if v_i is on the same side as v_0. Goemans and Williamson show that the expected value of the solution produced is at least 0.87856 times the value of the semidefinite relaxation in the case of MAX CUT and MAX 2SAT; the bound is 0.79607 for MAX DICUT. These algorithms can be derandomized.

For MAX 2SAT and MAX DICUT, we show that improvements can be obtained by taking advantage of the special role of v_0. Since our analysis relies heavily on numerical computations, we first explain why an improvement is at all possible. For this purpose, consider the algorithm of Goemans and Williamson for MAX 2SAT. They have shown that the probability that any clause, say $\bar{x}_i \vee \bar{x}_j$, is satisfied is equal to $(\arccos(v_0 \cdot v_i) + \arccos(v_0 \cdot v_j) + \arccos(v_i \cdot v_j))/(2\pi)$ and is at least 0.87856... times its contribution in the objective function $((3 - v_0 \cdot v_i - v_0 \cdot v_j - v_i \cdot v_j)/4)$. Moreover, the worst case of 0.87856... is attained only if two of the inner products are equal to $-0.689\ldots$ and the third one is equal to 1. This observation suggests the following algorithm. First solve (SAT_2) and then, with probability $1 - \epsilon$, use the approximation algorithm of Goemans and Williamson while, with probability ϵ, simply let $x_i =$ **true** if $v_i \cdot v_0 \geq 0$ and **false** otherwise. Observe that the probability that the clause $\bar{x}_i \vee \bar{x}_j$ is satisfied increases with ϵ for any configuration of vectors close to a worst-case configuration. As a result, if we choose ϵ small enough, the worst-case ratio will improve.

In order to obtain more substantial improvements, our approximation algorithms first solve (SAT_3) or (DI_3), instead of (SAT_2) or (DI_2). The special role of v_0 can be exploited in many ways as illustrated by the following rounding schemes.

1. Select r according to a distribution which is skewed towards v_0 but is uniform in any direction orthogonal to v_0. Then, as in Goemans and Williamson, let $i \in S$ or x_i be **true** if $\operatorname{sgn}(v_i \cdot r) = \operatorname{sgn}(v_0 \cdot r)$.
2. Decide whether i is in S or x_i is set **true** depending on the values $v_i \cdot r$, $v_0 \cdot r$ and $v_0 \cdot v_i$, where r is uniformly distributed on the unit sphere. More precisely, decide the fate of i depending on the sign of a certain function $f(v_i \cdot r, v_0 \cdot r, v_0 \cdot v_i)$.
3. Map v_i to a vector w_i depending on v_0 (and v_i). Then proceed with the Goemans and Williamson's algorithm.

For any given scheme, because of linearity of expectations, the performance guarantee will be at least γ if one can show that, for any three vectors v_0, v_i and v_j satisfying (1)–(3), the probability that a given clause involving x_i and x_j is satisfied (or the arc (i, j) is in the directed cut) is at least γ times its contribution to the objective function of the semidefinite program (see Goemans and Williamson [10] for details). We would like to stress again the importance of imposing inequalities (1)–(3); without them, one cannot hope for performance guarantees better than 9/8 for MAX 2SAT. Even for fairly simple schemes, an analytical derivation of the best γ does not seem to be an easy task. We have, however, evaluated numerically the performance of several schemes by discretizing the set of all possible angles between the three vectors and by evaluating, for each triple, the corresponding ratio.

For MAX 2SAT, we have obtained a scheme of type 3 which attains a performance guarantee of 0.931. We consider type 3 schemes of the following form. Map any vector v_i to a vector w_i, coplanar with v_0, on the same side of v_0 as v_i is, and which forms an angle with v_0 equal to $f(\theta_i)$ for some function f, where θ_i is the angle between v_0 and v_i. We impose that $f(\pi - \theta) = \pi - f(\theta)$ to guarantee that unnegated literals are treated in the same manner as negated literals. The original Goemans and Williamson scheme corresponds to $f_0(\theta) = \theta$. If we set $f_1(\theta) = \frac{\pi}{2}(1 - \cos(\theta))$ then it is easy to show that the probability that a given 1-clause is satisfied is equal to its value in the objective function of the semidefinite program. In other words, the expected weight of satisfied 1-clauses is precisely equal to the contribution of 1-clauses to the objective function. A scheme attaining a bound of 0.9249 can be obtained by taking the average between the above two functions:

$$f_{1/2}(\theta) = \frac{1}{2}\theta + \frac{1}{2}\left\{\frac{\pi}{2}(1 - \cos(\theta))\right\}.$$

The best scheme we have obtained is based on another convex combination. More precisely, letting

$$f(\theta) = \theta + 0.806765\left[\frac{\pi}{2}(1 - \cos\theta) - \theta\right],$$

we have shown numerically that the probability that a clause, say $\bar{x}_i \vee \bar{x}_j$, is satisfied is at least 0.93109 times the contribution of the clause to the objective function. The worst-case ratio of approximately 0.93109 is attained when $\theta_i = \theta_j = \pi - 1.32238$ and $v_i \cdot v_j = 2\cos(1.32238) - 1$, so that inequality (1) is satisfied at equality.

In the numerical computations, given θ_i, θ_j and $v_i \cdot v_j$, we need to be able to compute the angle between w_i and w_j in order to compute the probability that the clause is satisfied. This can be done by using the cosine rule for spherical triangles. In particular, we have that

$$\begin{aligned} v_i \cdot v_j &= \cos\theta_i \cos\theta_j + \cos\alpha \sin\theta_i \sin\theta_j \\ w_i \cdot w_j &= \cos f(\theta_i) \cos f(\theta_j) \\ &\quad + \cos\alpha \sin f(\theta_i) \sin f(\theta_j), \end{aligned}$$

where α denotes the angle between the planes defined by (v_0, v_i) and (v_0, v_j). This allows the determination of the angle between w_i and w_j.

For MAX DICUT, the scheme based on the function $f_{1/2}(\theta)$ can be shown numerically to give a performance guarantee of 0.857. A slightly better performance guarantee of 0.859 can be obtained through a much more complicated function.

Our algorithms for MAX 2SAT and MAX DICUT can be derandomized, as they are based on the algorithm of Goemans and Williamson [10].

5 Discussion

Our results are a bit more general than may appear from considering only the problems MAX DICUT, MAX CUT, and MAX 2SAT. In the introduction we presented three types of terms, **and**, **xor**, and **or**. In addition, we had unit terms (in MAX 2SAT). The approximation ratio that we can achieve for each type of term is 0.859, 0.878, 0.931, and 1.0, respectively. Moreover, if the objective function is mixed, in the sense that it contains more than one type of term (as in the case of MAX 2SAT), then the approximation ratio guaranteed by the rounding scheme for the most difficult type of term holds simultaneously for all terms. In particular, if the objective function is the sum of arbitrary nonnegative terms, each involving at most two Boolean variables, then we achieve an approximation ratio of 0.859. Observe that this is a nontrivial phenomenon, that is not known to hold for other problems. For example, for any k, *proper* MAX k-SAT can be approximated within a ratio of at least 0.875, but the best ratio of approximation known to be achievable for MAX SAT (which contains clauses of varying length) is roughly 0.7584 [10]. (This bound can be slightly improved using results in the current paper).

Many questions remain open. We discuss a few of them.

1. It is fairly easy do obtain semidefinite relaxations for a large variety of NP-hard optimization problems. Some examples are presented in [7]. The more difficult part is to analyze the performance guarantee of these approximation algorithms (for an illustration, see the recent result of Frieze and Jerrum [8]). Any new techniques for doing so are welcome. A notable result in this context is the use of semidefinite programming by Karger, Motwani, and Sudan [12] in order to color 3-colorable graphs with $O(n^{1/4})$ colors (which is an improvement over the previously best algorithm).

2. For the problems studied in the current paper, there is hope that the stronger formulations presented in Section 2.3 may lead to ratios of approximation that are significantly better than those achieved using the formulations of Section 2.2. Again, this would require new techniques of analysis.

3. How large can the ratio be between the true optimum and the optimum of the semidefinite relaxation? For the relaxations (CUT_3), (SAT_3) and (DI_3), we are aware of fairly bad instances. For MAX CUT, the 5-cycle gives a ratio between the optimum cut value and the optimum value of (CUT_3) of $\frac{32}{25+5\sqrt{5}} = 0.88445\ldots$ (see Goemans and Williamson [10]). Since any MAX DICUT instance can be formulated as a MAX CUT instance, the bidirected 5-cycle also gives a ratio of 0.88445 for (DI_3). For MAX 2SAT, the instance consisting of the following 10 clauses with unit weights can be seen to lead to a ratio of $\frac{72}{65+5\sqrt{5}} = 0.94512\ldots$ for (SAT_3):

$$x_1 \vee x_2,\ x_2 \vee x_3,\ x_3 \vee x_4,\ x_4 \vee x_5,\ x_5 \vee x_1,$$

$$\bar{x}_1 \vee \bar{x}_2,\ \bar{x}_2 \vee \bar{x}_3,\ \bar{x}_3 \vee \bar{x}_4,\ \bar{x}_4 \vee \bar{x}_5,\ \bar{x}_5 \vee \bar{x}_1.$$

However, for all these instances, the optimum solution of the semidefinite program includes triples of vectors (all different from v_0) that do not satisfy constraints (4)–(7). For (CUT_4), the clique on 5 vertices is a fairly bad instance (and is in fact the smallest instance for which the value of (CUT_4) differs from the maximum cut value by the result of Barahona and Mahjoub [4]). The maximum cut has size 6. However, selecting vectors for the vertices such that the inner product between any two of them is $-1/4$ (i.e. they form a regular simplex) satisfies all the constraints, and leads to a semidefinite solution of value 25/4. Hence the ratio between optimal and semidefinite solutions is $24/25 = 0.96$. As above, this example can be adapted to (DI_4) and (SAT_4) as well, giving ratios of 24/25 and 64/65 respectively.

This suggests a formulation (CUT_5) with additional inequalities that rules out this bad instance (that is, for any 5 vertices, the sum of all 10 possible inner products is at least -2). This is, of course, an endless game. It suggests the following question. Let q_k denote the ratio of approximation obtained for MAX CUT by a semidefinite program that includes all necessary valid inner product constraints that involve up to k vertices. Let $q_{\text{hard}} < 1$ denote a ratio that is NP-hard to achieve (as guaranteed to exist from [2]). What is the minimal k for which $q_k \geq q_{\text{hard}}$? Observe that $1 = q_n > q_{\text{hard}}$, and that for any fixed k (independent of n), $q_k < q_{\text{hard}}$, unless P=NP.

In Table 1, we summarize our current understanding of the tightness of the different semidefinite relaxations for the three basic problems.

Relaxation		CUT	DICUT	2SAT
1 or 2	lower	0.87856	0.79607	0.87856
	upper	0.88446	0.88446	0.88889
3	lower	0.87856	0.859	0.93109
	upper	0.88446	0.88446	0.94513
4	lower	0.87856	0.859	0.93109
	upper	0.96	0.96	0.98462

Table 1: Summary of known upper and lower bounds on the worst-case ratio between the value of the optimum solution and the value of semidefinite relaxations.

4. In [10], a ratio of 0.878 was achieved both for approximating MAX CUT and for approximating MAX 2SAT. In the current paper, we "separate" these two problems by showing that for a given set of constraints (defined in Section 2.2), the ratio between the true optimum and the optimum of the semidefinite relaxation is at least 0.931 for MAX 2SAT, but at most 0.885 for MAX CUT. However, we could not obtain such a separation between MAX CUT and MAX DICUT. Does (DI_3) approximate MAX DICUT as well as (CUT_3) approximates MAX CUT?

References

[1] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. In *Proc. 34th FOCS*, pages 724–733, 1993.

[2] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, pages 14–23, 1992.

[3] F. Barahona. On cuts and matchings in planar graphs. Report 87454-OR, Institut für Operations Research, Universität Bonn, 1988.

[4] F. Barahona and A. R. Mahjoub. On the cut polytope. *Mathematical Programming*, 36:157–173, 1986.

[5] A. I. Barvinok. Problems of distance geometry and convex properties of quadratic maps. Submitted to *Discrete and Computational Geometry*. A preliminary version appeared as "Topology and

convex geometry of quadratic equations", Mathematical Sciences Institute, Cornell University, Technical Report 94-2, 1994.

[6] M. Bellare. Interactive proofs and approximation: reductions from two provers in one round. In *Proc. of 2nd Israel Symposium on Theory of Computing and Systems, 266–274, 1993.*

[7] U. Feige and L. Lovász. Two-prover one-round proof systems: Their power and their problems. In *Proceedings of the 24th Annual ACM Symposium on the Theory of Computing*, pages 733–744, 1992.

[8] A. Frieze and M. Jerrum. Improved approximation algorithms for MAX k-CUT and MAX BISECTION. Manuscript, 1994.

[9] M. X. Goemans and D. P. Williamson. New 3/4-approximation algorithm for MAX SAT. In *Proc. 3rd MPS Conference on Integer Programming and Combinatorial Optimization, 313–321, 1993.* Journal version to appear in *SIAM J. Disc. Math.*, 7, 1994.

[10] M. X. Goemans and D. P. Williamson. .878-approximation algorithms for MAXCUT and MAX2SAT. In *Proceedings of the 26th Annual ACM Symposium on the Theory of Computing*, pages 422–431, 1994. Journal version submitted to *J. ACM.*

[11] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.

[12] D. Karger, R. Motwani, M. Sudan. Approximate graph coloring by semidefinite programming. *Proc. FOCS 1994*, to appear.

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

[14] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. In *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*, pages 286–293, 1993.

[15] C. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *JCSS, 43 (1991), 425–440.*

[16] G. Pataki. On the multiplicity of optimal eigenvalues. Management Science Research Report #MSRR-604, Carnegie-Mellon University, 1994.

[17] M. Yannakakis. On the approximation of maximum satisfiability. In *proc. 3rd Symp. on Discrete Algorithms, ACM-SIAM 1992, pp. 1-9.*