Approximation algorithms for MAX 4-SAT and rounding procedures for semidefinite programs

Eran Halperin and *Uri Zwick*

Department of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel.
{heran,zwick}@math.tau.ac.il

Abstract. Karloff and Zwick obtained recently an optimal 7/8-approximation algorithm for MAX 3-SAT. In an attempt to see whether similar methods can be used to obtain a 7/8-approximation algorithm for MAX SAT, we consider the most natural generalization of MAX 3-SAT, namely MAX 4-SAT. We present a semidefinite programming relaxation of MAX 4-SAT and a new family of rounding procedures that try to cope well with clauses of various sizes. We study the potential, and the limitations, of the relaxation and of the proposed family of rounding procedures using a combination of theoretical and experimental means. We select two rounding procedures from the proposed family of rounding procedures. Using the first rounding procedure we seem to obtain an *almost optimal* 0.8721-approximation algorithm for MAX 4-SAT. Using the second rounding procedure we seem to obtain an *optimal* 7/8-approximation algorithm for *satisfiable* instances of MAX 4-SAT. On the other hand, we show that no rounding procedure from the family considered can yield an approximation algorithm for MAX 4-SAT whose performance guarantee on all instances of the problem is greater than 0.8724.

Although most of this paper deals specifically with the MAX 4-SAT problem, we believe that the new family of rounding procedures introduced, and the methodology used in the design and in the analysis of the various rounding procedures considered would have a much wider range of applicability.

1 Introduction

MAX SAT is one of the most natural optimization problems. An instance of MAX SAT in the Boolean variables $x_1, \ldots, x_n$ is composed of a collection of clauses. Each clause is the disjunction of an arbitrary number of literals. Each literal is a variable, x_i, or a negation, $\bar{x}_i$, of a variable. Each clause has a non-negative weight w associated with it. The goal is to find a 0-1 assignment of values to the Boolean variables $x_1, \ldots, x_n$ so that the sum of the weights of the satisfied clauses is maximized.

Following a long line of research by many authors, we now know that MAX SAT is APX-hard (or MAX SNP-hard) [21, 10, 3, 2, 7, 6, 18] . This means that there is a constant $\epsilon > 0$ such that, assuming P$\neq$NP, there is no polynomial time approximation algorithm for MAX SAT with a performance guarantee of at least $1-\epsilon$ on all instances of the problem. Approximation algorithms for MAX SAT were designed by many authors, including [16, 28, 11, 12, 5, 4]. The best performance ratio known for the problem is currently 0.77 [4]. An approximation algorithm for MAX SAT with a *conjectured* performance guarantee of 0.797 is given in [31].

In a major breakthrough, Håstad [14] showed recently that no polynomial time approximation algorithm for MAX SAT can have a performance guarantee of more than 7/8, unless P=NP. Håstad's shows, in fact, that no polynomial time approximation algorithm for satisfiable instances of MAX $\{3\}$-SAT can have a performance guarantee of more than 7/8. MAX $\{3\}$-SAT is the subproblem of MAX SAT in which each clause is of size *exactly* three. An instance is satisfiable if there is an assignment that satisfies all its clauses.

Karloff and Zwick [17] obtained recently an optimal 7/8-approximation algorithm for MAX 3-SAT, the version of MAX SAT in which each clause is of size *at most* three. (This claim appears in [17] as a conjecture. It has since been proved.) Their algorithm uses semidefinite programming. A much simpler approximation algorithm has a performance guarantee of 7/8 if all clauses are of size *at least* three. If all clauses are of size at least three then a *random* assignment satisfies, on the average, at least 7/8 of the clauses.

We thus have a performance guarantee of 7/8 for instances in which all clauses are of size at most three, and for instances in which all clauses are of size at least three. Can we get a performance guarantee of 7/8 for *all* instances of MAX SAT? In an attempt to answer this question, we check the prospects of obtaining a 7/8-approximation algorithm for MAX 4-SAT, the subproblem of MAX SAT in which each clause is of size *at most* four. As it turns out, this is already a challenging problem.

The 7/8-approximation algorithm for MAX 3-SAT starts by solving a semidefinite programming relaxation of the problem. It then *rounds* the solution of this program using a random hyperplane passing through the origin. It is natural to try to obtain a similar approximation algorithm for MAX 4-SAT. It is not difficult, see Section 2, to obtain a semidefinite programming relaxation of MAX 4-SAT. It is again natural to try to round this solution using a random hyperplane. It turns out, however, that the performance guarantee of this algorithm is only 0.845173. Although this is much better than all previous performance guarantees for MAX 4-SAT, this guarantee is, unfortunately, below 7/8.

As the semidefinite programming relaxation of MAX 4-SAT is the strongest relaxation of its kind (see again Section 2), it seems that a different rounding procedure should be used. We describe, in Section 3, a new family of rounding procedures. This family extends all the families of rounding procedures previously suggested for maximum satisfiability problems. The difficulty in developing good rounding procedures for MAX 4-SAT is that rounding procedures that work well for the short clauses, do not work so well for the longer clauses, and vice versa. Rounding procedures from the new family try to work well on all clause sizes simultaneously. We initially hoped that an appropriate rounding procedure from this family could be used to obtain 7/8-approximation algorithms for MAX

4-SAT and perhaps even MAX SAT. It turns out, however, that the new family falls just short of this mission. The experiments that we have made suggest that a rounding procedure from the family, which we explicitly describe, can be used to obtain a 0.8721-approximation algorithm for MAX 4-SAT. Unfortunately, no rounding procedure from the family yields an approximation algorithm for MAX 4-SAT with a performance guarantee larger than 0.8724.

We have more success with MAX $\{2,3,4\}$-SAT, the version of MAX SAT in which the clauses are of size two, three or four. We present a second rounding procedure from the family that seems to yield an optimal 7/8-approximation algorithm for MAX $\{2,3,4\}$-SAT. A 7/8-approximation algorithm for MAX $\{2,3,4\}$-SAT yields immediately an optimal 7/8-approximation algorithm for *satisfiable* instances of MAX 4-SAT, as clauses of size one can be easily eliminated from satisfiable instances.

To determine the performance guarantee obtained using a given rounding procedure R, or at least a lower bound on this ratio, we have to find the global minimum of a function $ratio_R(v_0, v_1, v_2, v_3, v_4)$, given a set of constraints on the unit vectors $v_0, v_1, \ldots, v_4 \in \mathbb{R}^5$. The function $ratio_R$ is a fairly complicated function determined by the rounding procedure R. As five unit vectors are determined, up to rotations, by the $\binom{5}{2} = 10$ angles between them, the function $ratio_R$ is actually a function of 10 real variables. Finding the global minimum of $ratio_R$ analytically is a formidable task. In the course of our investigation we experimented with hundreds of rounding procedures. Finding these minima 'by hand' was not really an option. We have implemented a set of Matlab functions that use numerical techniques to find these minima.

The discussion so far centered on the quality of the rounding procedures considered. We also consider the quality of the suggested semidefinite programming relaxation itself. The integrality ratio of the MAX 4-SAT relaxation cannot be more than 7/8, as it is also a relaxation of MAX 3-SAT. We also show that the integrality ratio of the relaxation, considered as a relaxation of the problem MAX $\{1,4\}$-SAT, is at most 0.8753. The fact that this ratio is, at best, just above 7/8 is another indication of the difficulty of obtaining optimal 7/8-approximation algorithm for MAX 4-SAT and MAX SAT. It may also indicate that a stronger semidefinite programming relaxation would be needed to accomplish this goal.

The fact that numerical optimization techniques were used to compute the performance guarantees of the algorithms means that we cannot claim the existence of a 0.8721-approximation algorithm for MAX 4-SAT, and of a 7/8-approximation algorithm for MAX $\{2,3,4\}$-SAT as theorems. We believe, however, that it is possible to prove these claims analytically and promote them to the status of theorems, as was eventually done with the optimal 7/8-approximation algorithm for MAX 3-SAT. This would require, however, considerable effort. It may make more sense, therefore, to look for an approximation algorithm that seems to be a 7/8-approximation algorithm for MAX 4-SAT before proceeding to this stage.

In addition to implementing a set of Matlab functions that try to find the performance guarantee of a given rounding procedure from the family considered, we have also implemented a set of functions that *search* for good rounding procedures. The whole project required about 3000 lines of code. The two rounding procedures mentioned above, and several other interesting rounding procedures mentioned in Section 5, were found automatically using this system, with some manual help from the authors. The total running time used in the search for good rounding procedures is measured by months.

We end this section with a short survey of related results. The 7/8-approximation algorithm for MAX 3-SAT is based on the MAX CUT approximation algorithm of Goemans and Williamson [12]. A 0.931-approximation algorithm for MAX 2-SAT was obtained by Feige and Goemans [9]. Asano [4] obtained a 0.770- approximation algorithm for MAX SAT. Trevisan [25] obtained a 0.8-approximation algorithm for *satisfiable* MAX SAT instances. The last two results are also the best published results for MAX 4-SAT.

2 Semidefinite programming relaxation of MAX 4-SAT

Karloff and Zwick [17] describe a canonical way of obtaining semidefinite programming relaxations for any constraint satisfaction problem. We now describe the canonical relaxation of MAX 4-SAT obtained using this approach.

Assume that $x_1, \ldots, x_n$ are the variables of the MAX 4-SAT instance. We let $x_0 = 0$ and $x_{n+i} = \bar{x}_i$, for $1 \le i \le n$. The semidefinite program corresponding to the instance has a variable unit vector v_i, corresponding to each literal x_i, and scalar variables z_i, z_{ij}, z_{ijk} or z_{ijkl} corresponding to the clauses x_i, $x_i \vee x_j$, $x_i \vee x_j \vee x_k$ and $x_i \vee x_j \vee x_k \vee x_l$ of the instance, where $1 \le i, j, k \le 2n$. Note that all clauses, including those that contain negated literals, can be expressed in this form. Clearly, we require $v_{n+i} = -v_i$, or $v_i \cdot v_{n+i} = -1$, for $1 \le i \le n$.

The objective of the semidefinite program is to maximize the function

$$\sum_i w_i z_i + \sum_{i,j} w_{ij} z_{ij} + \sum_{i,j,k} w_{ijk} z_{ijk} + \sum_{i,j,k,l} w_{ijkl} z_{ijkl} \ ,$$

where the w_i's, w_{ij}'s, w_{ijk}'s and w_{ijkl}'s are the non-negative weights of the different clauses, subject to the following collection of constraints. For ease of notation, we write down the constraints that correspond to the clauses x_1, $x_1 \vee x_2$, $x_1 \vee x_2 \vee x_3$ and $x_1 \vee x_2 \vee x_3 \vee x_4$. The constraints corresponding to the other clauses are easily obtained by plugging in the corresponding indices. The constraints corresponding to x_1 and $x_1 \vee x_2$ are quite simple:

$$z_1 = \frac{1 - v_0 \cdot v_1}{2} \quad , \quad z_{12} \le \frac{3 - v_0 \cdot v_1 - v_0 \cdot v_2 - v_1 \cdot v_2}{4} \quad , \quad z_{12} \le 1 \ .$$

The constraints corresponding to $x_1 \vee x_2 \vee x_3$ are slightly more complicated:

$$\begin{array}{ll} z_{123} \le \frac{4 - (v_0 + v_1) \cdot (v_2 + v_3)}{4} \ , & z_{123} \le \frac{4 - (v_0 + v_2) \cdot (v_1 + v_3)}{4} \\ z_{123} \le \frac{4 - (v_0 + v_3) \cdot (v_1 + v_2)}{4} \ , & z_{123} \le 1 \end{array}$$

It is not difficult to check that the first three constraints above are equivalent to the requirement that

$$z_{123} \leq \frac{4-(v_{i_0}\cdot v_{i_1}+v_{i_1}\cdot v_{i_2}+v_{i_2}\cdot v_{i_3}+v_{i_3}\cdot v_{i_0})}{4},$$

for any permutation i_0, i_1, i_2, i_3 on $0, 1, 2, 3$. We will encounter similar constraints for the 4-clauses. The constraints corresponding to $x_1 \vee x_2 \vee x_3 \vee x_4$ are even more complicated. For any permutation i_0, i_1, i_2, i_3, i_4 on $0, 1, 2, 3, 4$ we require:

$$z_{1234} \leq \frac{5-(v_{i_0}\cdot v_{i_1}+v_{i_1}\cdot v_{i_2}+v_{i_2}\cdot v_{i_3}+v_{i_3}\cdot v_{i_4}+v_{i_4}\cdot v_{i_0})}{4},$$

$$z_{1234} \leq \frac{5-(v_{i_0}+v_{i_4})\cdot(v_{i_1}+v_{i_2}+v_{i_3})+v_{i_0}\cdot v_{i_4}}{4} \quad , \quad z_{1234} \leq 1 .$$

The first line above contributes 12 different constraints, the second line contributes 10 different constraints. Together with the constraint $z_{1234} \leq 1$ we get a total of 23 constraints per 4-clause. In addition, for every distinct $0 \leq i_1, i_2, i_3, i_4, i_5 \leq 2n$, we require

$$\sum_{1\leq j<k\leq 3} v_{i_j}\cdot v_{i_k} \geq -1 \quad \text{and} \quad \sum_{1\leq j<k\leq 5} v_{i_j}\cdot v_{i_k} \geq -2 .$$

Although we do not consider here clauses of size larger than 4, we remark that for any k, and for any permutation $i_0, i_1, \ldots, i_k$ on $0, 1, \ldots, k$, $z_{12\ldots k} \leq ((k+1) - \sum_{j=0}^{k} v_{i_j}\cdot v_{i_{j+1}})/4$, where the index $j+1$ is interpreted modulo $k+1$, is a valid constraint in the semidefinite programming relaxation of MAX k-SAT.

It is not difficult to verify that all these constraints are satisfied by any valid 'integral' assignment to the vectors v_i and the scalars z_i, z_{ij}, z_{ijk} and z_{ijkl}, i.e., an assignment in which $v_i = (1, 0, \ldots, 0)$ if $x_i = 1$, and $v_i = (-1, 0, \ldots, 0)$ if $x_i = 0$, and in which every z variable is set to 1 if its corresponding clause is satisfied by the assignment $x_1, x_2, \ldots, x_n$, and to 0 otherwise. Thus, the presented semidefinite program is indeed a relaxation of the MAX 4-SAT instance.

The constraints of the above semidefinite program correspond to the *facets* of a polyhedron corresponding to the Boolean function $x_1 \vee x_2 \vee x_3 \vee x_4$. As explained in [17], it is therefore the strongest semidefinite relaxation that considers the clauses of the instance one by one. Stronger relaxations may be obtained by considering several clauses at once.

The semidefinite programming relaxation of a MAX 4-SAT instance has $n+1$ unknown unit vectors $v_0, v_1, \ldots, v_n$ (the vectors v_{n+i} are just used as shorthands for $-v_i$), $O(n^4)$ scalar variables and $O(n^5)$ constraints. An almost optimal solution in which all unit vectors $v_0, v_1, \ldots, v_n$ lie in $\mathbb{R}^{n+1}$ can be found in polynomial time ([1],[13],[19]).

3 Rounding procedures

In this section we consider various procedures that can be used to round solutions of semidefinite programming relaxations and examine the performance guarantees that we get for MAX 4-SAT using them. We start with simple rounding procedures and then move on to more complicated ones. The new family of rounding procedures is then presented in Section 3.5.

3.1 Rounding using a random hyperplane

Following Goemans and Williamson [12], many semidefinite programming based approximation algorithms round the solution to the semidefinite program using a *random hyperplane* passing through the origin. A random hyperplane that passes through the origin is chosen by choosing its normal vector r as a uniformly distributed vector on the unit sphere (in $\mathbb{R}^{n+1}$). A vector v_i is then rounded to 0 if v_i and v_0 fall on the same side of the random hyperplane, i.e., if $sgn(r \cdot v_i) = sgn(r \cdot v_0)$, and to 1, otherwise. Note that the rounded values of the variables are usually not independent. More specifically, if v_i and v_j are not perpendicular, i.e., if $v_i \cdot v_j \neq 0$, then the rounded values of x_i and x_j are dependent.

Given a set of unit vector V, we let $prob_H(V)$ denote the probability that not all the vectors of V fall on the same side of a random hyperplane that passes through the origin. It is not difficult to see that

$$prob_H(v_0, v_1) = \frac{\theta_{01}}{\pi} \quad , \quad prob_H(v_0, v_1, v_2) = \frac{\theta_{01} + \theta_{02} + \theta_{12}}{2\pi} \ ,$$

where $\theta_{ij} = \arccos(v_i \cdot v_j)$ is the angle between v_i and v_j. Evaluating $prob_H(v_0, v_1, v_2, v_3)$ is more difficult. As noted in [17],

$$prob_H(v_0, v_1, v_2, v_3) = 1 - \frac{Vol(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})}{\pi^2} \ ,$$

where $(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}) = \pi - (\theta_{23}, \theta_{13}, \theta_{03}, \theta_{12}, \theta_{02}, \theta_{01})$ and $Vol(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})$ is the volume of a spherical tetrahedron with dihedral angles $\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}$. Unfortunately, the volume function of spherical tetrahedra seems to be a non-elementary function and numerical integration should be used to evaluate it ([24],[8],[15],[27]).

It is not difficult to verify, using inclusion-exclusion, that

$$prob_H(v_0, v_1, v_2, v_3, v_4) = \frac{1}{2} \sum_{i<j<k<l} prob_H(v_i, v_j, v_k, v_l) - \frac{1}{4} \sum_{i<j} prob_H(v_i, v_j) \ .$$

Thus, the probability that certain five unit vectors are separated by a random hyperplane can be expressed as a combination of probabilities of events that involve at most four vectors and no further numerical integration is needed. More generally, $prob_H(v_0, \ldots, v_k)$, for any *even* k, can be expressed as combinations of probabilities involving at most k vectors. The same does not hold, unfortunately, when k is odd.

We let $relax(v_0, v_1, \ldots, v_i)$, where $1 \leq i \leq 4$, denote the 'relaxed' value of the clause $x_1 \vee \ldots \vee x_i$, i.e., the maximal value to which the scalar $z_{12\ldots i}$ can be set, while still satisfying all the relevant constraints, given the unit vectors $v_0, v_1, \ldots, v_i$. We let

$$ratio_H(v_0, v_1, \ldots, v_i) = prob_H(v_0, v_1, \ldots, v_i) / relax(v_0, v_1, \ldots, v_i) \ .$$

Finally, for every $1 \leq i \leq 4$, we let $\alpha_i = \min ratio_H(v_0, v_1, \ldots, v_i)$, where the minimum is over all configurations of unit vectors that satisfy the constraints described in the previous section, and $\alpha = \min\{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$. As follows from straightforward arguments (see [12] or [17]), α is a lower bound on the performance guarantee of the approximation algorithm for MAX 4-SAT that uses the semidefinite relaxation and then rounds the solution using a random hyperplane.

It is shown in [12] that $\alpha_1 = \alpha_2 \simeq 0.87856$. It is shown in [17] that $\alpha_3 = 7/8$. Unfortunately, it turns out that $\alpha_4 \simeq 0.845173$. The minimum is attained when the angle between each pair of vectors among $v_0, v_1, \ldots, v_4$ is exactly $\arccos(1/5) \simeq 1.369438$. It is not difficult to check that when $v_i \cdot v_j = 1/5$, for every $0 \leq i < j \leq 4$, all the inequalities on z_{1234} simplify to $z_{1234} \leq 1$ and therefore $\mathit{relax}(v_0, v_1, v_2, v_3, v_4) = 1$.

3.2 Pre-rounding rotations

Feige and Goemans [9] introduced the following variation of random hyperplane rounding. Let $f : [0, \pi] \to [0, \pi]$ be a continuous function satisfying $f(0) = 0$ and $f(\pi - \theta) = \pi - f(\theta)$, for $0 \leq \theta \leq \pi$. Before rounding the vectors using a random hyperplane, the vector v_i is rotated into a new vector v_i', in the plane spanned by v_0 and v_i, so that the angle θ_{0i}' between v_0 and v_i' would be $\theta_{0i}' = f(\theta_{0i})$. The rotations of the vectors $v_1, \ldots, v_n$ affects, of course, the angles between these vectors. Let θ_{ij}' be the angle between v_i' and v_j'. It is not difficult to see (see [9]), that for $i, j > 0$, $i \neq j$, we have

$$\cos\theta_{ij}' \;=\; \cos\theta_{0i}' \cdot \cos\theta_{0j}' + \frac{\cos\theta_{ij} - \cos\theta_{0i} \cdot \cos\theta_{0j}}{\sin\theta_{0i} \cdot \sin\theta_{0j}} \cdot \sin\theta_{0i}' \cdot \sin\theta_{0j}' \;.$$

The vectors $v_0, v_1', \ldots, v_n'$ are then rounded using a random hyperplane. The condition $f(0) = 0$ is required to ensure the continuity of the transformation $v_i \to v_i'$. The condition $f(\pi - \theta) = \pi - f(\theta)$ ensures that unnegated and negated literals are treated in the same manner.

Feige and Goemans [9] use rotations to obtain a 0.931-approximation algorithm for MAX 2-SAT. Rotations are also used in [29] and [30]. Can we use rotations to get a better approximation algorithm for MAX 4-SAT? The answer is that rotations on their own help, but very little. Consider the configuration v_0, v_1, v_2, v_3, v_4 in which $\theta_{0i} = \pi/2$, for $1 \leq i \leq 4$, and $\theta_{ij} = \arccos(1/3)$, for $1 \leq i < j \leq 4$. For this configuration we get $\mathit{relax} = 1$ and $\mathit{ratio}_H \simeq 0.8503$. As every rotation function f must satisfy $f(\pi/2) = \pi/2$, rotations have no effect on this configuration.

A different type of rotations was recently used by Nesterov [20] and Zwick [31]. These *outer-rotations* are used in [31] to obtain some improved approximation algorithms for MAX SAT and MAX NAE-SAT. We were not able to use them, however, to improve the results that we get in this paper for MAX 4-SAT.

3.3 Rounding the vectors independently

The semidefinite programming relaxation of MAX 4-SAT that we are using here is stronger than the linear programming relaxation suggested by Goemans and

Williamson [11]. It is nonetheless interesting to consider an adaptation of the rounding procedures used in [11] to the present context. The rounding procedures of [11] are based on the *randomized rounding* technique of Raghavan and Thompson [23],[22].

Let $g : [0, \pi] \to [0, \pi]$ be a continuous function such that $g(\pi - \theta) = \pi - g(\theta)$, for $0 \le \theta \le \pi$. Note again that we must have $g(\pi/2) = \pi/2$. We do not require $g(0) = 0$ this time. The rounding procedure described here rounds each vector *independently*. The variable x_i is assigned the value 1 with probability $g(\theta_{0i})/\pi$, and the value 0 with the complementary probability. The probability $prob_I(v_0, v_1, \ldots, v_i)$ that a clause $x_1 \vee \ldots \vee x_i$ is satisfied is now

$$prob_I(v_0, v_1, \ldots, v_i) = 1 - \prod_{j=1}^{i} \left(1 - \frac{g(\theta_{0j})}{\pi}\right) .$$

Note that as each vector is rounded independently, the angles θ_{ij}, where $i, j > 0$, between the vectors, have no effect this time. It may be worthwhile to note that the choice $g(\theta) = \pi/2$, for every $0 \le \theta \le \pi$, corresponds to choosing the assignment to the variables $x_1, x_2, \ldots, x_n$ uniformly at random, a 'rounding procedure' that yields a ratio of 7/8 for clauses of size 3 and 15/16 for clauses of size 4. Goemans and Williamson [11] describe several functions g using which a 3/4-approximation algorithm for MAX SAT may be obtained.

Independent rounding performs well for long clauses. It cannot yield a ratio larger than 3/4, however, for clauses of size 2. To see this, consider the configuration v_0, v_1, v_2 in which $\theta_{01} = \theta_{02} = \pi/2$ and $\theta_{12} = \pi$. We have $relax(v_0, v_1, v_2) = 1$ and $prob_I(v_0, v_1, v_2) = 3/4$, for any function g.

3.4 Simple combinations

We have seen that hyperplane rounding works well for short clauses and that independent rounding works well for long clauses. It is therefore natural to consider a combination of the two.

Perhaps the most natural combination of hyperplane rounding and independent rounding is the following. Let $0 \le \epsilon \le 1$. With probability $1 - \epsilon$ round the vectors using a random hyperplane. With probability ϵ choose a random assignment. It turns out that the best choice of ϵ here is $\epsilon \simeq 0.086553$. With this value of ϵ, we get $\alpha_1 = \alpha_2 = \alpha_4 \simeq 0.853150$ while $\alpha_3 = 7/8$. Thus, we again get a small improvement but we are still far from 7/8.

Instead of rounding all vectors using a random hyperplane, or choosing random values to all variables, we can round some of the vectors using a random hyperplane, and assign some of the variables random values. More precisely, we choose one random hyperplane. Each vector is now rounded using this random hyperplane with probability $1 - \epsilon$, or is assigned a random value with probability ϵ. The decisions for the different variables made independently. Letting $\epsilon \simeq 0.073609$, we get $\alpha_1 = \alpha_2 = \alpha_4 \simeq 0.856994$, while $\alpha_3 \simeq 0.874496$. This is again slightly better but still far from 7/8.

3.5 More complicated combinations

Simple combinations of hyperplane rounding and independent rounding yield modest improvements. Can we get more substantial improvements by using more sophisticated combinations? To answer this question we introduce the following family of rounding procedures. The new family seems to include all the natural combinations of the rounding procedures mentioned above.

Each rounding procedure in the new family is characterized by three continuous functions $f, g : [0,\pi] \to [0,\pi]$ and $\epsilon : [0,\pi] \to [0,1]$. The function f is used for rotating the vectors before rounding them using a random hyperplane, as described in Section 3.2. The function g is used to round the vectors independently, as described in Section 3.3. The function ϵ is used to decide which of the two roundings should be used. The decision is made *independently* for each vector, depending on the angle between it and v_0. The function $\epsilon : [0,\pi] \to [0,1]$ is a continuous function satisfying $\epsilon(\pi-\theta) = \epsilon(\theta)$, a condition that ensures that negated and unnegated literals are treated in the same manner. The vector v_i is rounded using a random hyperplane, shared by all the vectors rounded using a random hyperplane, with probability $1-\epsilon(\theta_{0i})$, and is rounded independently, with probability $\epsilon(\theta_{0i})$. Vectors rounded using the shared hyperplane are rotated before the rounding. Let $v_1', v_2', \ldots, v_n'$ be the vectors obtained by rotating the vectors $v_1, v_2, \ldots, v_n$, as specified by the rotation function f. The probability that a clause $x_1 \vee x_2 \vee \ldots \vee x_i$ is satisfied by the assignment produced by this combined rounding procedure is given by the following expression:

$$prob_C(v_0, v_1, \ldots, v_i) = 1 - \sum_S pr(S) \cdot (1 - prob_H(v'(S))) \cdot (1 - prob_I(v(\bar{S})))$$

where

$$pr(S) = \prod_{i\in S}(1-\epsilon(\theta_{0i})) \cdot \prod_{i\notin S}\epsilon(\theta_{0i}) \;,$$

$$v'(S) = \{v_0\} \cup \{v_i' \mid i \in S\} \quad , \quad v(\bar{S}) = \{v_0\} \cup \{v_i \mid i \notin S\} \;,$$

and where S ranges over all subsets of $\{1, 2, \ldots, i\}$. Recall that $prob_H(u_1, u_2, \ldots, u_k)$ is the probability that the set of vectors $u_1, u_2, \ldots, u_k$ is separated by a random hyperplane, and that $prob_I(v_0, u_1, \ldots, u_k)$ is the probability that at least one of the vectors $u_1, u_2, \ldots, u_k$ is assigned the value 1 when all these vectors are rounded independently using the function g.

We have made some experiments with an even wider family of rounding procedures but we were not able to improve on the results obtained using rounding procedures selected from the family described here. More details will be given in the full version of the paper.

Can we select a rounding procedure from the proposed family of rounding procedures using which we can get an optimal, or an almost optimal, approximation algorithm for MAX 4-SAT?

4 The search for good rounding procedures

The new family of rounding procedures defined in the previous section is huge. How can we expect to select the best, or almost the best, rounding procedure from this family? As it turns out, although each rounding procedure is defined by three continuous functions f, g and ϵ, most of the values of these functions do not matter much. What really matter are the values of these functions at several 'important' angles. We therefore restrict ourselves to rounding procedures defined by *piecewise linear* functions f, g and ϵ with a relatively small number of bends. By placing these bends at the 'important' angles, we can find, as we shall see, a rounding procedure which is close to being the best rounding procedure from this family.

More specifically, we consider functions obtained by connecting k given points $(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)$ by straight line segments, where $x_1 = 0$ and $x_k = \pi/2$. For f we also require $y_1 = 0$ and $y_k = \pi/2$. For g we also require $y_k = \pi/2$. The values of the functions f, g and ϵ for $\pi/2 < \theta \le \pi$ are determined by the conditions $f(\pi - \theta) = \pi - f(\theta)$, $g(\pi - \theta) = \pi - g(\theta)$ and $\epsilon(\pi - \theta) = \epsilon(\theta)$. We usually worked with $k \le 5$.

For a given value of k we are now faced with a very difficult optimization problem in $6k-9$ real variables, the variables being the x and y coordinates of the points through which the functions f, g and ϵ are required to pass. The objective is to maximize $\alpha(C(f, g, \epsilon))$, the performance guarantee obtained by using the rounding procedure defined by the functions f, g and ϵ that pass through the points. Recall that evaluating $\alpha(C(f, g, \epsilon))$ for a given set of functions f, g and ϵ is already a difficult task that requires finding the global minimum of a rather complicated function of 10 real variables.

We have written a Matlab program, called `opt_fun`, that tries to find a close to optimal rounding procedure that uses functions specified using at most k points. This is quite a non-trivial task and, as mentioned in the introduction, it required about 3000 lines of code, in addition to the sophisticated numerical optimization routines of Matlab's optimization toolbox.

Although numerical methods were used to evaluate the performance guarantees of the different rounding procedures, we believe that the 0.8721 and 7/8 performances ratio claimed for the two rounding procedures that will be described shortly are the correct performance ratios. There is, in fact, a completely mechanical way of generating a (long and tedious) rigorous proof of these claims. As mentioned in the introduction, we believe that it would be more fruitful to look for an algorithm that seems to achieve a performance ratio of 7/8 before taking on the task of producing rigorous proofs. We believe that the use of numerical methods would be inevitable in the search for optimal algorithms for MAX 4-SAT and MAX SAT, at least using the current techniques.

5 Almost optimal or optimal approximation algorithms

We now present some optimal or close to optimal approximation algorithms obtained using rounding procedures from the new family of rounding procedures.

f	g	ϵ
(0 , 0)	(0 , 0)	(0 , 0.250000)
(0.777843 , 1.210627)	(0.750000 , 0)	(0.744611 , 0.357201)
(1.038994 , 1.445975)	(1.072646 , 0)	(1.039987 , 0.255183)
(1.248362 , 1.394099)	(1.248697 , 0.872552)	(1.072689 , 0.222928)
($\pi/2$, $\pi/2$)	($\pi/2$, $\pi/2$)	($\pi/2$, 0.131681)

Fig. 1. The rounding procedure that seems to yield a 0.8721-approximation algorithm for MAX 4-SAT

f	g	ϵ
(0 , 0)	(0 , 0.550000)	(0 , 0.650000)
(1.394245 , 1.544705)	(1.155432 , 1.154866)	(0.413021 , 0.163085)
($\pi/2$, $\pi/2$)	(1.394111 , 0.931661)	($\pi/2$, 0.160924)
	($\pi/2$, $\pi/2$)	

Fig. 2. The rounding procedure that seems to yield an optimal 7/8-approximation algorithm for MAX $\{2,3,4\}$-SAT.

5.1 MAX 4-SAT

Using the semidefinite programming relaxation of Section 2 and the rounding procedure defined by the three piecewise linear functions passing through the points given in Figure 1 we seem to obtain a 0.8721-approximation algorithm for MAX 4-SAT, or more specifically, an algorithm with $\alpha_1 \simeq \alpha_2 \simeq \alpha_3 \simeq \alpha_4 \simeq 0.8721$. As we shall see in Section 6, this is essentially the best approximation ratio that we can obtain using a rounding procedure from the family considered.

It is interesting to note that $g(\theta) = 0$ for $0 \le \theta \le 1.072646$ and that $0.13 \le \epsilon(\theta) \le 0.36$ for $0 \le \theta \le \pi$. This means that if the angle θ_{0i} between v_i and v_0 is less than about $\pi/3$, then with a probability of about $1/4$, the variable x_i is assigned the value 0, without any further consideration of the angle θ_{0i}. It is also interesting to note that the function $f(\theta)$ is not monotone.

5.2 MAX {2,3,4}-SAT

Using the semidefinite programming relaxation of Section 2 and the rounding procedure defined by the three piecewise linear functions passing through the points given in Figure 2 we believe we obtain a 7/8-approximation algorithm for

MAX $\{2,3,4\}$-SAT. We get in fact, an approximation algorithm for MAX 4-SAT with $\alpha_2 \simeq 0.8751$, $\alpha_3 = 7/8$, $\alpha_4 \simeq 0.8755$ but with $\alpha_1 \simeq 0.8352$. It is interesting to note the non-monotonicity of the function $g(\theta)$ and the fact that only one intermediate point is needed for $f(\theta)$ and $\epsilon(\theta)$ and only two intermediate points are needed for $g(\theta)$.

A $7/8$-approximation algorithm for MAX $\{2,3,4\}$-SAT is of course optimal as a ratio better than $7/8$ cannot be obtained even for MAX $\{3\}$-SAT, which is a subproblem of MAX $\{2,3,4\}$-SAT.

5.3 MAX 3-SAT

The optimal $7/8$-approximation algorithm for MAX 3-SAT presented in [17] has $\alpha_1 = \alpha_2 \simeq 0.87856$ and $\alpha_3 = 7/8$. Using pre-rounding rotations we can obtain an approximation algorithm for MAX 3-SAT with $\alpha_1 = \alpha_2 \simeq 0.9197$ and $\alpha_3 = 7/8$. This algorithm would perform better than the algorithm of [17] on instances in which some of the contribution to the optimal value of their semidefinite programming relaxation comes from clauses of size one or two. The details of this algorithm will be given in the full version of the paper.

5.4 MAX 2-SAT

Feige and Goemans [9] obtained an approximation algorithm for MAX 2-SAT with $\alpha_1 \simeq 0.976$ and $\alpha_2 \simeq 0.931$. Although we cannot improve α_2, the performance ratio on clauses of size two, we can obtain, using pre-rounding rotations, an approximation algorithm for MAX 2-SAT with $\alpha_1 \simeq 0.983$ and $\alpha_2 \simeq 0.931$. The details of this algorithm will be given in the full version of the paper.

6 Limitations of current rounding procedures

We presented above a rounding procedure using which we seem to get a 0.8721-approximation algorithm for MAX 4-SAT. This is extremely close to $7/8$. Could it be that by searching a little bit harder, or perhaps allowing more bends, we could find a rounding procedure from the family defined in Section 3.5 using which we could obtain an optimal $7/8$-approximation algorithm for MAX 4-SAT? Unfortunately, the answer is no. We show in this section that the rounding procedure described in Section 5.1 is close to being the best rounding procedure of the family considered.

Let θ_{ij}, for $0 \le i < j \le 4$, be the angles between the five unit vectors v_0, v_1, v_2, v_3, v_4. Let $c_{ij} = \cos\theta_{ij}$. It is not difficult to check that if

$$\begin{array}{lcl}
c_{12} = \frac{1+c_{01}+c_{02}-2c_{03}-2c_{04}}{3} & , & c_{13} = \frac{1+c_{01}-2c_{02}+c_{03}-2c_{04}}{3} \\
c_{23} = \frac{1-2c_{01}+c_{02}+c_{03}-2c_{04}}{3} & , & c_{14} = \frac{1+c_{01}-2c_{02}-2c_{03}+c_{04}}{3} \\
c_{24} = \frac{1-2c_{01}+c_{02}-2c_{03}+c_{04}}{3} & , & c_{34} = \frac{1-2c_{01}-2c_{02}+c_{03}+c_{04}}{3}
\end{array}$$

then $\mathit{relax}(v_0, v_1, v_2, v_3, v_4) = 1$.

Let $0 < \theta_1 < \theta_2 \leq \pi/2$ be two angles. Consider the configuration (v_0, v_1) in which $\theta_{01} = \pi - \theta_1$, and the two configurations $(v_0, v_1^1, v_2^1, v_3^1, v_4^1)$ and $(v_0, v_1^2, v_2^2, v_3^2, v_4^2)$ in which

$$(\theta_{01}^1, \theta_{02}^1, \theta_{03}^1, \theta_{04}^1) \;=\; (\theta_1, \theta_1, \theta_1, \pi - \theta_2) \quad , \quad (\theta_{01}^2, \theta_{02}^2, \theta_{03}^2, \theta_{04}^2) \;=\; (\theta_2, \theta_2, \theta_2, \theta_2)$$

and in which the angles θ_{jk}^i, for $1 \leq j < k \leq 4$ are determined according to the relations above so that $relax(v_0, v_1^i, v_2^i, v_3^i, v_4^i) = 1$, for $i = 1, 2$. It is not difficult to check that $\theta_{12}^1 = \theta_{13}^1 = \theta_{23}^1 = \arccos(\frac{1+2\cos\theta_2}{3})$, $\theta_{14}^1 = \theta_{24}^1 = \theta_{34}^1 = \arccos(\frac{1-3\cos\theta_1-\cos\theta_2}{3})$ and that $\theta_{ij}^2 = \arccos(\frac{1-2\cos\theta_2}{3})$, for $1 \leq j \leq k \leq 4$. Assume that the configurations $(v_0, v_1^i, v_2^i, v_3^i, v_4^i)$, for $i = 1, 2$, are feasible. For every rounding procedure C we have

$$\alpha(C) \leq \min\{\; ratio_C(v_0, v_1), ratio_C(v_0, v_1^1, v_2^1, v_3^1, v_4^1), ratio_C(v_0, v_1^2, v_2^2, v_3^2, v_4^2)\; \}.$$

As the only angles between v_0 and and other vectors in these three configurations are $\theta_1, \theta_2, \pi - \theta_1$ and $\pi - \theta_2$, and as $f(\pi - \theta) = \pi - f(\theta)$, $g(\pi - \theta) = \pi - g(\theta)$ and $\epsilon(\pi - \theta) = \epsilon(\theta)$, we get that for every rounding procedure from our family, the three ratios $ratio_C(v_0, v_1)$, $ratio_C(v_0, v_1^1, v_2^1, v_3^1, v_4^1)$ and $ratio_C(v_0, v_1^2, v_2^2, v_3^2, v_4^2)$ depend only on the six parameters $f(\theta_1), f(\theta_2), g(\theta_1), g(\theta_2), \epsilon(\theta_1)$ and $\epsilon(\theta_2)$.

Take $\theta_1 = 0.95$ and $\theta_2 = \arccos(1/5) \simeq 1.369438$. It is possible to check that the resulting two configurations $(v_0, v_1^1, v_2^1, v_3^1, v_4^1)$ and $(v_0, v_1^2, v_2^2, v_3^2, v_4^2)$ are feasible. The choice of the six parameters that maximizes the minimum ratio of the three configurations, found again using numerical optimization, is:

$$\begin{array}{lclcl} f(\theta_1) \simeq 1.410756 & , & g(\theta_1) \simeq 0 & , & \epsilon(\theta_1) \simeq 0.309376 \\ f(\theta_2) \simeq 1.448494 & , & g(\theta_2) \simeq 1.233821 & , & \epsilon(\theta_2) \simeq 0.122906 \end{array}$$

With this choice of parameters, the three ratios evaluate to about 0.8724. No rounding procedure from the family can therefore attain a ratio of more than 0.8724 simultaneously on these three specific configurations. No rounding procedure from the family can therefore yield a performance ratio greater than 0.8724 for MAX 4-SAT, even if the functions f, g and ϵ are not piecewise linear.

7 The quality of the semidefinite programming relaxation

Let I be an instance of MAX 4-SAT. Let $opt(I)$ be the value of the optimal assignment for this instance. Let $opt^*(I)$ be the value of the optimal solution of the canonical semidefinite programming relaxation of the instance given in Section 2. Clearly $opt(I) \leq opt^*(I)$ for every instance I. The *integrality ratio* of the relaxation is defined to be $\gamma = \inf_I opt(I)/opt^*(I)$, where the infimum is taken over all the instances.

In Section 3, when we analyzed the performance of different rounding procedures, we compared the value, or rather the expected value, of the assignment produced by a rounding procedure to $opt^*(I)$, the optimal value of the semidefinite programming relaxation. It is not difficult to see that any lower bound α

on the performance ratio of a rounding procedure obtained in this way would satisfy $\alpha \le \gamma$. Thus, the rounding procedure of Section 5.1 seems to imply that $\gamma \ge 0.8721$. In this section we describe *upper bounds* on the integrality ratio γ, thereby obtaining upper bounds on the performance ratios that can be obtained by *any* approximation algorithm that uses the relaxation of Section 2, at least using the type of analysis used in Section 3.

It is shown in [17] that the integrality ratio of the canonical semidefinite programming relaxation of MAX 3-SAT is exactly $\gamma_3 = 7/8$. As the canonical relaxations of MAX 3-SAT and MAX 4-SAT coincide on instances of MAX 3-SAT, we get that $\gamma = \gamma_4 \le 7/8$.

We can show, that the integrality ratio of the canonical relaxation of MAX 4-SAT, given in Section 2, is at most 0.8753, even when restricted to instances of MAX $\{1,4\}$-SAT, i.e., to instances of MAX 4-SAT in which all clauses are of size 1 or 4. Though this upper bound does not preclude the possibility of obtaining an optimal 7/8-approximation algorithm for MAX 4-SAT using the canonical semidefinite programming relaxation of the problem, the closeness of this upper bound to 7/8 does indicate that it will not be easy, even if clauses of length 3 are not present. It may be necessary to consider stronger relaxations of MAX 4-SAT, e.g., relaxations obtained by considering several clauses of the instance at once.

8 Concluding remarks

We have come frustratingly close to obtaining an optimal 7/8-approximation algorithm for MAX 4-SAT. We have seen that devising a 7/8-approximation algorithm for MAX $\{1,4\}$-SAT is already a challenging problem. Note that Håstad's 7/8 upper bound for MAX 3-SAT and MAX 4-SAT does not apply to MAX $\{1,4\}$-SAT, as clauses of length three are not allowed in this problem. A gadget (see [26]) supplied by Greg Sorkin shows that no polynomial time approximation algorithm for MAX $\{1,4\}$-SAT can have a performance ratio greater that 9/10, unless P=NP.

We believe that optimal 7/8-approximation algorithms for MAX 4-SAT and MAX SAT do exist. The fact that we have come so close to obtaining such algorithms may in fact be seen as cause for optimism. There is still a possibility that simple extensions of ideas laid out here could be used to achieve this goal. If this fails, it may be necessary to attack the problems from a more global point of view. Note that the analysis carried out here was very local in nature. We only considered one clause of the instance at a time. As a result we only obtained lower bounds on the performance ratios of the algorithms considered. It may even be the case that the algorithms from the family of algorithms considered here do give a performance ratio of 7/8 for MAX 4-SAT although a more global analysis is required to show it.

We also hope that MAX 4-SAT would turn out to be the last barrier on the road to an optimal approximation algorithm for MAX SAT. The almost optimal algorithms for MAX 4-SAT presented here may be used to obtain an almost optimal algorithm for MAX SAT. We have not worked out yet the exact bounds that we can get for MAX SAT as we still hope to get an optimal algorithm for MAX 4-SAT before proceeding with MAX SAT.

Finally, a word on our methodology. Our work is a bit unusual as we use experimental and numerical means to obtain *theoretical* results. We think that the nature of the problems that we are trying to solve calls for this approach. No one can rule out, of course, the possibility that some clever new ideas would dispense with most of the technical difficulties that we are facing here. Until that happens, however, we see no alternative to the current techniques. The use of experimental and numerical means does not mean that we have to give up the rigorousity of the results. Once we obtain the 'right' result, we can devote efforts to proving it rigorously, possibly using automated means.

References

1. F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5:13–51, 1995.
2. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45:501–555, 1998.
3. S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45:70–122, 1998.
4. T. Asano. Approximation algorithms for MAX SAT: Yannakakis vs. Goemans-Williamson. In *Proceedings of the 3nd Israel Symposium on Theory and Computing Systems, Ramat Gan, Israel*, pages 24–37, 1997.
5. T. Asano, T. Ono, and T. Hirata. Approximation algorithms for the maximum satisfiability problem. *Nordic Journal of Computing*, 3:388–404, 1996.
6. M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCPs, and nonapproximability—towards tight results. *SIAM Journal on Computing*, 27:804–915, 1998.
7. M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient probabilistically checkable proofs and applications to approximation. In *Proceedings of the 25rd Annual ACM Symposium on Theory of Computing, San Diego, California*, pages 294–304, 1993. See Errata in STOC'94.
8. H.S.M. Coxeter. The functions of Schläfli and Lobatschefsky. *Quarterly Journal of of Mathematics (Oxford)*, 6:13–29, 1935.
9. U. Feige and M.X. Goemans. Approximating the value of two prover proof systems, with applications to MAX-2SAT and MAX-DICUT. In *Proceedings of the 3nd Israel Symposium on Theory and Computing Systems, Tel Aviv, Israel*, pages 182–189, 1995.
10. U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43:268–292, 1996.
11. M.X. Goemans and D.P. Williamson. New 3/4-approximation algorithms for the maximum satisfiability problem. *SIAM Journal on Discrete Mathematics*, 7:656–666, 1994.
12. M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995.
13. M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer Verlag, 1993. Second corrected edition.

14. J. Håstad. Some optimal inapproximability results. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing, El Paso, Texas*, pages 1–10, 1997. Full version available as E-CCC Report number TR97-037.
15. W.Y. Hsiang. On infinitesimal symmetrization and volume formula for spherical or hyperbolic tetrahedrons. *Quarterly Journal of Mathematics (Oxford)*, 39:463–468, 1988.
16. D.S. Johnson. Approximation algorithms for combinatorical problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.
17. H. Karloff and U. Zwick. A 7/8-approximation algorithm for MAX 3SAT? In *Proceedings of the 38rd Annual IEEE Symposium on Foundations of Computer Science, Miami Beach, Florida*, pages 406–415, 1997.
18. S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. In *Proceedings of the 35rd Annual IEEE Symposium on Foundations of Computer Science, Santa Fe, New Mexico*, pages 819–830, 1994.
19. Y. Nesterov and A. Nemirovskii. *Interior Point Polynomial Methods in Convex Programming.* SIAM, 1994.
20. Y. E. Nesterov. Semidefinite relaxation and nonconvex quadratic optimization. *Optimization Methods and Software*, 9:141–160, 1998.
21. C.H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.
22. P. Raghavan. Probabilistic construction of deterministic algorithms: Approximating packing integer programs. *Journal of Computer and System Sciences*, 37:130–143, 1988.
23. P. Raghavan and C. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7:365–374, 1987.
24. L. Schläfli. On the multiple integral $\int^n dx\, dy \ldots dz$, whose limits are $p_1 = a_1x + b_1y + \ldots + h_1z > 0, p_2 > 0, \ldots, p_n > 0$, and $x^2 + y^2 + \ldots + z^2 < 1$. *Quarterly Journal of Mathematics (Oxford)*, 2:269–300, 1858. Continued in Vol. 3 (1860), pp. 54–68 and pp. 97-108.
25. L. Trevisan. Approximating satisfiable satisfiability problems. In *Proceedings of the 5th European Symposium on Algorithms, Graz, Austria*, 1997. 472–485.
26. L. Trevisan, G.B. Sorkin, M. Sudan, and D.P. Williamson. Gadgets, approximation, and linear programming (extended abstract). In *Proceedings of the 37rd Annual IEEE Symposium on Foundations of Computer Science, Burlington, Vermont*, pages 617–626, 1996.
27. E.B. Vinberg. Volumes of non-Euclidean polyhedra. *Russian Math. Surveys*, 48:15–45, 1993.
28. M. Yannakakis. On the approximation of maximum satisfiability. *Journal of Algorithms*, 17:475–502, 1994.
29. U. Zwick. Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, California*, pages 201–210, 1998.
30. U. Zwick. Finding almost-satisfying assignments. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing, Dallas, Texas*, pages 551–560, 1998.
31. U. Zwick. Outward rotations: a tool for rounding solutions of semidefinite programming relaxations, with applications to max cut and other problems. In *Proceedings of the 31th Annual ACM Symposium on Theory of Computing, Atlanta, Georgia*, 1999. To appear.