# Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint

*Uri Zwick* *

**Abstract**

An instance of MAX 3CSP is a collection of $m$ clauses of the form $f_i(z_{i1}, z_{i2}, z_{i3})$, where the $z_{ij}$'s are literals, or constants, from the set $\{0, 1, x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$, and the $f_i$'s are arbitrary Boolean functions depending on (at most) three variables. Each clause has a non-negative weight $w_i$ associated with it. A solution to the instance is an assignment of 0-1 values to the variables $x_1, \ldots, x_n$ that maximizes $\sum_{i=1}^{n} w_i f_i(z_{i1}, z_{i2}, z_{i3})$, the total weight of the satisfied clauses. The MAX 3CSP problem is clearly a generalization of the MAX 3SAT problem. (In an instance of the MAX 3SAT problem $f_i(z_{i1}, z_{i2}, z_{i3}) = z_{i1} \vee z_{i2} \vee z_{i3}$ for every $1 \le i \le m$.)

Karloff and Zwick have recently obtained a $\frac{7}{8}$-approximation algorithm for MAX 3SAT. Their algorithm is based on a new semidefinite relaxation of the problem. Håstad showed that no polynomial time algorithm can achieve a better performance ratio, unless P=NP. Here we use similar techniques to obtain a $\frac{1}{2}$-approximation algorithm for MAX 3CSP. The performance ratio of this algorithm is also optimal, as follows again from the work of Håstad. We also obtain better performance ratios for several special cases of the problem. Our results include:

- A $\frac{1}{2}$-approximation algorithm for MAX 3AND, the problem in which each clause is of the form $z_{i1} \wedge z_{i2} \wedge z_{i3}$. This result is optimal and it implies the result for MAX 3CSP.
- A $\frac{2}{3}$-approximation algorithm for MAX 3MAJ, the problem in which each clause is of the form $\texttt{MAJ}(z_{i1}, z_{i2}, z_{i3})$, where MAJ is the *majority* function. This result is again optimal.
- A $\frac{3}{2+\sqrt{2}}$-approximation algorithm for MAX 3NAE, the problem in which each clause is of the form $\texttt{NAE}(z_{i1}, z_{i2}, z_{i3})$, where $\texttt{NAE}(x_1, x_2, x_3) = 1$ if and only if $x_1$, $x_2$ and $x_3$ are not all equal. (We assume here that the same literal, or constant, is not allowed to appear twice in the same clause.)

Finally, building on ideas of Trevisan, we get a $\frac{5}{8}$-approximation algorithm for *satisfiable* instances of MAX 3CSP. Our results imply limits on the power of *non-adaptive* verifiers of probabilistically checkable proofs (PCP's) that read only three bits of the proof. We also obtain a $\frac{3}{2\pi}\arccos(-\frac{1}{3})$-approximation algorithm for *satisfiable* instances of MAX 3NAE. (Note that $\frac{3}{2+\sqrt{2}} \simeq 0.87868$ while $\frac{3}{2\pi}\arccos(-\frac{1}{3}) \simeq 0.91226$.)

## 1 Introduction

Recent years have seen two major breakthroughs in the field of approximation algorithms. On one end, an impressive series of works, that includes [FGL+96, AS92, ALM+92, BGLR93, BGS95, Hås97], established, among other things, that there exists a constant $\gamma < 1$ such that MAX E3SAT, the version of MAX SAT in which each clause contains *exactly* three literals, cannot be approximated with a performance ratio of at least $\gamma$, unless P=NP. In the last work of this series, Håstad obtains this result with $\gamma = \frac{7}{8} + \epsilon$, for every $\epsilon > 0$. A random assignment satisfies, on average, $\frac{7}{8}$ of the total weight of the clauses of a MAX E3SAT instance. Håstad's result is therefore tight.

In a second breakthrough, Goemans and Williamson [GW95] used semidefinite programming to obtain a 0.87856-approximation algorithm for MAX CUT. Feige and Goemans [FG95] followed with a 0.93109-approximation algorithm for MAX 2SAT. Karloff and Zwick [KZ97] recently obtained a $\frac{7}{8}$-approximation algorithm for MAX 3SAT, the version of MAX SAT in which each clause is of length *at most* three. Here we continue this line of research and obtain a few more optimal approximation algorithms. In particular, we obtain a $\frac{1}{2}$-approximation algorithm for MAX 3CSP, improving a 0.387-approximation algorithm of Trevisan, Sorkin, Sudan and Williamson [TSSW96], and a $\frac{5}{8}$-approximation algorithm for *satisfiable* instances of MAX 3CSP, improving a 0.514-approximation algorithm of Trevisan [Tre97]. As discussed in [TSSW96]

*Department of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel. Email: zwick@math.tau.ac.il. This work was done while this author was visiting ICSI and UC Berkeley.

and [Tre97], this imply limits on the power of *non-adaptive* verifiers of probabilistically checkable proofs (PCP's) that read only three bits of the proof.

Many other papers deal with issues related to the ones investigated here. Following is a partial list of such papers: [AE97, Asa97, CFZ97, Cre95, CT97, GW94, Joh74, KLP96, Kar96, KMSV94, KST97, KSW97, MR95, PY91, Pet94, Sch78, Yan94].

## 2 Constraints involving three or less variables

In the sequel, we denote by MAX CSP($f$), where $f$ is a 3-variable Boolean function, the subproblem of MAX 3CSP in which all the constraints are of the form $f(z_{i1}, z_{i2}, z_{i3})$.

We say that two Boolean functions $f(x_1, \ldots, x_k)$ and $g(x_1, \ldots, x_k)$ are of the same *type* if and only if there exist a permutation $\pi : \{1, \ldots, k\} \to \{1, \ldots, k\}$ and constants $a_1, \ldots, a_k \in \{0, 1\}$ such that

$$g(x_1, \ldots, x_k) \;=\; f(x_{\pi(1)} \oplus a_1, \ldots, x_{\pi(k)} \oplus a_k)$$

for every $x_1, \ldots, x_k \in \{0, 1\}$. Clearly, if $f$ and $g$ are of the same type, then the problems MAX CSP($f$) and MAX CSP($g$) are equivalent. The three types of Boolean functions depending on two variables are shown in Figure 1. They are represented by the three functions $\mathtt{OR}_2 = x_1 \vee x_2$, $\mathtt{XOR}_2 = x_1 \oplus x_2$ and $\mathtt{AND}_2 = x_1 \wedge x_2$. Goemans and Williamson [GW95] obtained a 0.87856-approximation algorithm for MAX CSP($\mathtt{XOR}_2$), which includes MAX CUT as a subproblem. Feige and Goemans [FG95] obtained a 0.93109-approximation algorithm for MAX CSP($\mathtt{OR}_2$), also known as the MAX 2SAT problem, and a 0.859-approximation algorithm for MAX CSP($\mathtt{AND}_2$), which includes MAX DICUT as a subproblem. The upper bounds on the approximability of these problems were obtained by Trevisan, Sorkin, Sudan and Williamson [TSSW96]. We note here that the inapproximability bounds given here for MAX CSP($\mathtt{XOR}_2$) and MAX CSP($\mathtt{AND}_2$) are stronger, i.e., smaller, than those known for MAX CUT and MAX DICUT. MAX CUT and MAX DICUT are subproblems of these problems and may therefore be easier to approximate. The best inapproximability bounds known for MAX CUT and MAX DICUT are $\frac{16}{17}$ and $\frac{12}{13}$, respectively.

A simple computer program can be used to show that there are 16 types of Boolean functions that depend on 3 variables. They are given in Figure 2. An instance of MAX 3CSP may thus contain constraints of 20 different types: the 16 types shown in Figure 2, the 3 types shown in Figure 1, and constraints of the form $z_i$, where $z_i$ is a literal. In the rest of the paper we describe approximation algorithms for MAX 3CSP, as well as for the problems MAX CSP($f$), where $f$ is one of the 16 functions given in Figure 2.

| constraint type | performance ratio | inapproximability bound |
|---|---|---|
| $\mathtt{OR}_2 = x_1 \vee x_2$ | 0.93109 | $\frac{21}{22}$ |
| $\mathtt{XOR}_2 = x_1 \oplus x_2$ | 0.87856 | $\frac{11}{12}$ |
| $\mathtt{AND}_2 = x_1 \wedge x_2$ | 0.859 | $\frac{9}{10}$ |

Figure 1: The 3 types of 2-variable constraints.

We now try to acquaint ourselves with the functions appearing in Figure 2. The functions AND, XOR and OR need no introduction. The other symmetric functions are $\mathtt{EQU}(x_1, x_2, x_3)$, which returns 1 iff $x_1 = x_2 = x_3$, $\mathtt{TWO}(x_1, x_2, x_3)$, which returns 1 iff exactly *two* of $x_1, x_2$ and $x_3$ are 1, $\mathtt{NAE}(x_1, x_2, x_3)$, which returns 1 iff $x_1, x_2$ and $x_3$ are *not all equal*, and $\mathtt{NTW}(x_1, x_2, x_3)$, which returns 1 iff the number of 1's is *not two*. Note that the function $\mathtt{ONE}(x_1, x_2, x_3)$, which returns 1 iff the number of 1's in exactly *one*, is of the same type as TWO, as $\mathtt{ONE}(x_1, x_2, x_3) = \mathtt{TWO}(\bar{x}_1, \bar{x}_2, \bar{x}_3)$. The functions $\mathtt{AXR} = x_1 \wedge (x_2 \oplus x_3)$, $\mathtt{AOR} = x_1 \wedge (x_2 \vee x_3)$, $\mathtt{XAD} = x_1 \oplus x_2 x_3$, $\mathtt{OAD} = x_1 \vee x_2 x_3$ and $\mathtt{OXR} = x_1 \vee (x_2 \oplus x_3)$ are also fairly natural. Finally, $\mathtt{SEL}(x_1, x_2, x_3)$ is the *selection* function which outputs $x_1$, if $x_3 = 0$, and $x_2$, if $x_3 = 1$. We occasionally attach the subscript 3 to these function names to remind us that they all depend on three variables.

## 3 Semidefinite relaxations

All our approximation algorithms use the *canonical* semidefinite relaxations of constraint satisfaction problems introduced by Karloff and Zwick [KZ97].

For ease of notation, we encode an instance of MAX CSP($f$) in the following way. Let $x_1, \ldots, x_{n-1}$ be the variables appearing in the instance. (Note that the number of Boolean variables is assumed to be $n-1$ and not $n$). We let $x_0$ stand for the Boolean constant 1 and $x_n, \ldots, x_{2n-1}$ stand for the negations $\bar{x}_0, \ldots, \bar{x}_{n-1}$. (Note that $x_n$ stands for the Boolean constant 0.) The instance is then represented as an array $\{w_{ijk}\}$ of non-negative weights, where $w_{ijk}$, for $0 \le i, j, k < 2n$, is the weight of the clause $f(x_i, x_j, x_k)$. Thus, $w_{0,7,n+4}$, for example, is the weight of the clause $f(1, x_7, \bar{x}_4)$, and $w_{5,n,n+5}$ is the weight of the clause $f(x_5, 0, \bar{x}_5)$. Unless specifically prohibited, clauses are allowed to contain constants, and they are allowed to contain more than

| | | |
|---|---|---|
| AND | = | $x_1 \wedge x_2 \wedge x_3$ |
| EQU | = | $x_1 x_2 x_3 \vee \bar{x}_1 \bar{x}_2 \bar{x}_3$ |
| AXR | = | $x_1 \wedge (x_2 \oplus x_3)$ |
| AOR | = | $x_1 \wedge (x_2 \vee x_3)$ |
| AOA | = | $x_2 x_3 \vee x_1 \bar{x}_2 \bar{x}_3$ |
| TWO | = | $x_1 x_2 \bar{x}_3 \vee x_1 \bar{x}_2 x_3 \vee x_1 x_2 \bar{x}_3$ |
| XOR | = | $x_1 \oplus x_2 \oplus x_3$ |
| MAJ | = | $x_1 x_2 \vee x_1 x_3 \vee x_2 x_3$ |

| | | |
|---|---|---|
| XAD | = | $x_1 \oplus x_2 x_3$ |
| SEL | = | $x_1 \bar{x}_3 \vee x_2 x_3$ |
| OAD | = | $x_1 \vee x_2 x_3$ |
| XOA | = | $(x_1 \oplus x_2) \vee x_2 x_3$ |
| NTW | = | $(\bar{x}_1 \vee \bar{x}_2)\bar{x}_3 \vee (x_1 \equiv x_2) x_3$ |
| NAE | = | $(x_1 \oplus x_2) \vee (x_1 \oplus x_3) \vee (x_2 \oplus x_3)$ |
| OXR | = | $x_1 \vee (x_2 \oplus x_3)$ |
| OR | = | $x_1 \vee x_2 \vee x_3$ |

Figure 2: The 16 types of 3-variable constraints.

one occurrence of the same variable, as in the second example above.

The semidefinite program used as a relaxation for an instance $\{w_{ijk}\}$ of MAX 3AND is given in Figure 3. The variables of the program are the $n$ unit vectors $v_0, v_1, \ldots, v_{n-1}$, in $\mathbb{R}^n$, corresponding to the constant 1 and the variables $x_1, \ldots, x_{n-1}$, and scalars $z_{ijk}$ corresponding to the clauses of the instance. For convenience, we let $v_{n+i} = -v_i$, for $0 \leq i < n$. (In the relaxation, $S^{n-1}$ stands for the unit sphere in $\mathbb{R}^n$.) We should note here that while in [KZ97] $v_0$ was used to represent the constant 0, here we find it more convenient to let $v_0$ represent the constant 1.

To verify that this is indeed a relaxation of MAX 3AND, we note that for any truth assignment $a_1, \ldots, a_{n-1}$, we can generate a feasible point of the semidefinite program in the following way: We define $v_i = (-1, 0, \ldots, 0)$, if $a_i = 0$, and $v_i = (1, 0, \ldots, 0)$, if $a_i = 1$. We also let $v_0 = (1, 0, \ldots, 0)$. We then define $z_{ijk} = 1$, if $a_i \wedge a_j \wedge a_k = 1$, and $z_{ijk} = 1$, otherwise. (We use here the convention that $a_0 = 1$ and $a_{n+i} = \bar{a}_i$, for $0 \leq i < n$.) It is not difficult to check that this is indeed a feasible point of the program whose value is equal to the number of clauses satisfied by the assignment.

We may, if we want to, add the additional constraints given in Figure 4. These are the additional constraints used by Feige and Goemans [FG95]. We are allowed to add these constraints as the points constructed in the previous paragraph satisfy them. In the case of MAX 3AND, the addition of the additional constraints does not lead to an improved performance ratio. For some of the other problems, they do help.

The semidefinite relaxations that we use for problems MAX CSP($f$), with different choices of $f$, are all similar to the one shown in Figure 3. The only difference is that different bounds are used on the $z_{ijk}$'s and that sometimes we also use the additional constraints of Figure 4. The relaxations used for all other types of Boolean constraints involving at most three variables

$$
\begin{array}{ll}
\text{Maximize} & \sum_{i,j,k} w_{ijk} \cdot z_{ijk} \\
\text{subject to} & \\
z_{ijk} \leq \frac{1+v_0\cdot v_i+v_0\cdot v_j+v_i\cdot v_j}{4} & \forall\, 0 \leq i,j,k < 2n \\
z_{ijk} \leq \frac{1+v_0\cdot v_i+v_0\cdot v_k+v_i\cdot v_k}{4} & \forall\, 0 \leq i,j,k < 2n \\
z_{ijk} \leq \frac{1+v_0\cdot v_j+v_0\cdot v_k+v_j\cdot v_k}{4} & \forall\, 0 \leq i,j,k < 2n \\
z_{ijk} \leq \frac{1+v_i\cdot v_j+v_i\cdot v_k+v_j\cdot v_k}{4} & \forall\, 0 \leq i,j,k < 2n \\
v_i \in S^{n-1} & \forall\, 0 \leq i < 2n \\
v_{n+i} = -v_i & \forall\, 0 \leq i < n
\end{array}
$$

Figure 3: A semidefinite relaxation of MAX 3AND.

$$
v_i \cdot v_j + v_i \cdot v_k + v_j \cdot v_k \;\geq\; -1 \quad \forall\, 0 \leq i,j,k < 2n
$$

Figure 4: Possible additional constraints.

will appear in the full version of the paper.

How are these relaxations obtained? This is described in [KZ97]. Very briefly, to any Boolean function $f : \{-1,1\}^k \rightarrow \{0,1\}$ we attach a polytope *polytope*($f$) in $\mathbb{R}^{\frac{k(k+1)}{2}+1}$. (For convenience we use $\{-1,1\}^k$, instead of $\{0,1\}^k$, as the domain of $f$, with $-1$ representing FALSE and 1 representing TRUE.) The constraints of the semidefinite relaxation correspond to the *facets* of *polytope*($f$). (We assume here that *polytope*($f$) is non-degenerate). If $\sum_{ij} \alpha_{ij} y_{ij} + \beta z \leq \gamma$ is a facet of the polytope, the semidefinite relaxation will include constraints of the form $z \leq \frac{\gamma - \sum_{ij} \alpha_{ij} v_i \cdot v_j}{\beta}$.

Given a vector $x = (x_1, \ldots, x_k) \in \{-1,1\}^k$, we let $\text{prod}(x) = (x_0 x_1, x_0 x_2, \ldots, x_{k-1} x_k)$, where $x_0 = 1$. Note that $\text{prod}(x) \in \{-1,1\}^{\frac{k(k+1)}{2}}$. The polytope

*polytope*$(f)$ is defined as follows:

$$polytope(f) = \mathrm{conv}(\{(\mathrm{prod}(x), f(x)) \mid x \in \{-1,1\}^k\}),$$

where $(\mathrm{prod}(x), f(x))$ denotes a vector of length $\frac{k(k+1)}{2}+1$ obtained by appending $f(x)$ to $\mathrm{prod}(x)$, and $\mathrm{conv}(\{v_1,\ldots,v_\ell\})$ is the *convex hull* of the set $\{v_1,\ldots,v_\ell\}$. The points $(\mathrm{prod}(x), f(x))$, for $x \in \{-1,1\}^k$, are therefore the vertices of *polytope*$(f)$.

It is not difficult to see (and it is shown in [KZ97]) that the obtained semidefinite programs are indeed relaxations of the Boolean constraint satisfaction problems. Furthermore, they are the *strongest* semidefinite relaxations within a natural class of relaxations that are allowed to 'consider' the Boolean constraints only one at a time. For details, see [KZ97].

## 4 Rounding using a random hyperplane

A good relaxation is only the first component of a good approximation algorithm. The second component is a good *rounding* procedure. In our case, a rounding procedure should convert the vectors $v_0,\ldots,v_{n-1}$ obtained as an optimal solution to the semidefinite program into a truth assignment $a_1,\ldots,a_{n-1}$ to the variables $x_1,\ldots,x_{n-1}$.

The first rounding procedure that comes to mind, in connection with semidefinite programs, is the rounding procedure suggested by Goemans and Williamson [GW95]. This rounding procedure chooses a random *hyperplane* that passes through the origin. A variable $x_i$ is assigned the value 1 iff $v_i$ and $v_0$ lie on the same side of the hyperplane.

This rounding procedure performs well for MAX CUT (no better rounding procedure is known for MAX CUT). This is also the rounding procedure used in [KZ97] to obtain the $\frac{7}{8}$-approximation algorithm for MAX 3SAT. Slightly more complicated rounding procedures, involving *rotations*, are used by Feige and Goemans [FG95] in their MAX 2SAT and MAX DICUT algorithms.

An even simpler, seemingly stupid, rounding procedure, that should nonetheless be considered, is the 'rounding' procedure that does not even look at the vectors $v_0,\ldots,v_{n-1}$ and simply chooses the truth values assigned to the variables $x_1,\ldots,x_{n-1}$ independently at random.

The performance ratio obtained using these two rounding procedures, for the 16 different types of constraints, are given in the second and third columns of Figure 5. The way in which these performance ratios were obtained is shortly described in the rest of this section.

| constraint type | random assignment | random hyperplane | improved rounding | inapprox. bound |
|---|---|---|---|---|
| AND | $\frac{1}{8}$ | 0 | $\frac{1}{2}$ | $\frac{1}{2}$ |
| EQU | $\frac{1}{4}$ | 0.796070 | | $\frac{7}{8}$ |
| AXR | $\frac{1}{4}$ | 0.404973 | $\frac{1}{2}$ | $\frac{1}{2}$ |
| AOR | $\frac{3}{8}$ | 0.733368 | | $\frac{3}{4}$ |
| AOA | $\frac{3}{8}$ | 0.733368 | | $\frac{3}{4}$ |
| TWO | $\frac{3}{8}$ | 0.488862 | $\frac{1}{2}$ | $\frac{1}{2}$ |
| XOR | $\frac{1}{2}$ | $\frac{1}{2}$ | | $\frac{1}{2}$ |
| MAJ | $\frac{1}{2}$ | 0.666158 | $\frac{2}{3}$ | $\frac{2}{3}$ |
| XAD | $\frac{1}{2}$ | 0.666158 | | $\frac{2}{3}$ |
| SEL | $\frac{1}{2}$ | 0.796070 | 0.859 | $\frac{9}{10}\Big/\frac{11}{12}$ |
| OAD | $\frac{5}{8}$ | 0.810686 | | $\frac{5}{6}$ |
| XOA | $\frac{5}{8}$ | 0.810686 | | $\frac{5}{6}$ |
| NTW | $\frac{5}{8}$ | $\frac{5}{8}$ | | $\frac{5}{8}$ |
| NAE | $\frac{3}{4}$ | 0.878567 | $\frac{3}{2+\sqrt{2}}$ | $\frac{11}{12}\Big/\frac{15}{16}$ |
| OXR | $\frac{3}{4}$ | $\frac{3}{4}$ | | $\frac{3}{4}$ |
| OR | $\frac{7}{8}$ | $\frac{7}{8}$ | | $\frac{7}{8}$ |

Figure 5: The performance ratios obtained using the different rounding procedures and upper bounds on the achievable performance ratios for the 16 types of 3-variable constraints.

The performance ratio achieved for MAX CSP$(f)$ using random assignments is simply the probability $r_f = \Pr[\, f(x_1,x_2,x_3) = 1\,]$, where $x_1$, $x_2$ and $x_3$ are assigned independent random bits. For certain functions $f$, we have to assume here that the three literals appearing in each clause belong to different variables, and are therefore independent. For such functions, clauses like $f(x_1,x_2,0)$, $f(x_1,x_1,x_2)$ or $f(x_1,x_2,\bar{x}_1)$ are not allowed. For some other functions $f$, there is no harm in allowing such clauses, as their satisfiability probability is either 0, in which case the clauses can be ignored, or at least as high as $r_f$.

Finding the performance ratio achieved using random hyperplane rounding is harder. A lower bound $\alpha_f$ on this performance ratio can be obtained as follows:

$$\alpha_f \;=\; \inf_{relax_f(v_0,v_1,v_2,v_3)>0} \frac{prob_f(v_0,v_1,v_2,v_3)}{relax_f(v_0,v_1,v_2,v_3)}\,,$$

where the infimum is taken over all the feasible configurations of the vectors $v_0, v_1, v_2, v_3$ for which

$relax_f(v_0, v_1, v_2, v_3) > 0$. Here $prob_f(v_0, v_1, v_2, v_3)$ is the probability that rounding the vectors $v_0, v_1, v_2, v_3$ using a random hyperplane yields an assignment $a_1, a_2, a_3$ to $x_1, x_2, x_3$ for which $f(a_1, a_2, a_3) = 1$. As for $relax_f(v_0, v_1, v_2, v_3)$, this is the maximal feasible value of $z_{123}$, given the vectors $v_0, v_1, v_2, v_3$. For $f = \texttt{AND}_3$, for example, we get

$$relax_{\texttt{AND}_3}(v_0, v_1, v_2, v_3) = \min \left\{ \begin{array}{cc} \frac{1+v_0\cdot v_1+v_0\cdot v_2+v_1\cdot v_2}{4}, & \frac{1+v_0\cdot v_1+v_0\cdot v_3+v_1\cdot v_3}{4} \\ \frac{1+v_0\cdot v_2+v_0\cdot v_3+v_2\cdot v_3}{4}, & \frac{1+v_1\cdot v_2+v_1\cdot v_3+v_2\cdot v_3}{4} \end{array} \right\} .$$

For some functions $f$, computing $prob_f(v_0, v_1, v_2, v_3)$ is fairly easy. For example

$$prob_{\texttt{NAE}_3}(v_0, v_1, v_2, v_3) = \frac{\theta_{12} + \theta_{13} + \theta_{23}}{2\pi} ,$$

$$prob_{\texttt{SEL}_3}(v_0, v_1, v_2, v_3) = 1 - \frac{\theta_{01} + \theta_{02} - \theta_{13} + \theta_{23}}{2\pi} .$$

where $\theta_{ij} = \arccos(v_i \cdot v_j)$ is the angle between the vectors $v_i$ and $v_j$. For the other functions $f$, computing $prob_f(v_0, v_1, v_2, v_3)$ is much harder. It follows from [KZ97] that

$$prob_{\texttt{AND}_3}(v_0, v_1, v_2, v_3) = \frac{\text{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})}{\pi^2} ,$$

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 $\text{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})$ is the volume of a *spherical tetrahedron* with *dihedral angles* $\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}$, $\lambda_{13}$ and $\lambda_{23}$. Unfortunately, $\text{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})$ seems to be a non-elementary function and no closed-form formula, in terms of the basic elementary functions, is known for it. Things are not hopeless, however, as the partial derivatives $\partial\text{Vol}/\partial\lambda_{ij}$ are elementary functions. Schläfli [Sch58] showed, in 1858, that $\partial\text{Vol}/\partial\lambda_{ij} = \theta'_{ij}/2$, where $\theta'_{ij}$ is the length of the edge connecting $i$ and $j$ in a spherical tetrahedron whose dihedral angles are $(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})$. (See also Coxeter [Cox57]) and Vinberg [Vin93].

All other $prob_f(v_0, v_1, v_2, v_3)$ can be expressed using the volume function. For example

$$prob_{\texttt{MAJ}_3}(v_0, v_1, v_2, v_3) = 3 - \frac{2\theta_{01} + 2\theta_{02} + 2\theta_{03} + \theta_{12} + \theta_{13} + \theta_{23}}{2\pi} - \frac{2\text{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})}{\pi^2} .$$

Computing $prob_f(v_0, v_1, v_2, v_3)$, for given $v_0, v_1, v_2, v_3$ is not easy. Solving the optimization problem used to define $\alpha_f$ is even harder. In most cases, we use numerical methods to compute $\alpha_f$. We are now in the process of converting this numerical evidence into proofs, as was done in [KZ97].

It was shown in [KZ97], using a combination of analytical and numerical methods, that $\alpha_{\texttt{OR}_3} = \frac{7}{8}$, resulting in a $\frac{7}{8}$-approximation algorithm for MAX 3SAT. We should remark here that while in the analysis of the performance of random assignments we had to assume, in some cases, that two literals of the same variable are not allowed to appear in the same clause, such a restriction is *not* required when analyzing the performance ratio of the random hyperplane rounding procedure. In the case of MAX 3SAT, if all clauses are of length 3, then a random assignment achieves a performance ratio of $\frac{7}{8}$. If clauses of length 1 and 2 are allowed, then the performance ratio of random assignments drops to $\frac{1}{2}$. A clause $x_1 \vee x_2$ of length 2 can be represented, however, as a clause $x_1 \vee x_1 \vee x_2$ of length 3. This *is* allowed when considering the random hyperplane rounding procedure. The algorithm of [KZ97] achieves, therefore, a performance ratio of $\frac{7}{8}$ even when clauses of length 1 and 2 are allowed.

A random assignment satisfies, on average, $\frac{1}{8}$ of the total weight of the clauses of a MAX 3AND instance. What performance ratio do we get using random hyperplanes? The disappointing answer is 0 ! This does not look very promising. In the next section we show, however, that this can be easily fixed. We show there that if we generate two assignments, one of them being a completely random one while the other is obtained by rounding the vectors $v_0, \ldots, v_{n-1}$ using a random hyperplane, and then take the better one, we obtain a $\frac{1}{2}$-approximation algorithm for MAX 3AND.

Let us check now how does random hyperplane rounding perform for the other types of constraints. A quick look at Figure 5 shows that in all other 15 cases, random hyperplane rounding does at least as well as selecting a random assignment. This does not sound very impressive. It should be remembered, however, that the performance ratios given for random hyperplane rounding are valid even when clauses are allowed to contain constants and more than one occurrence of the same variable. For the problems MAX CSP($f$), where $f = \texttt{XOR}_3$, $\texttt{NTW}_3$, $\texttt{OXR}_3$ and $\texttt{OR}_3$ we get optimal approximation algorithms. For the functions $\texttt{NTW}_3$, $\texttt{OXR}_3$ and $\texttt{OR}_3$, the approximation algorithm that uses hyperplane rounding is stronger than the naive algorithm that simply picks a random assignments. In the case of $\texttt{NTW}_3$, for example, we are allowed to use clauses of the form $\mathtt{NTW}(x_1, \bar{x}_1, \bar{x}_2) = x_2$ and $\mathtt{NTW}(x_1, \bar{x}_2, 1) = x_1 \oplus x_2$. In the case of $\mathtt{OXR}_3$, we are allowed, for example, to use clauses of the form $\mathtt{OXR}(0, x_1, x_2) = x_1 \oplus x_2$. The fact that random hyperplane rounding achieves performance ratios $\frac{1}{2}$ for $\mathtt{XOR}_3$ and $\frac{5}{8}$ for $\mathtt{NTW}_3$ will be important in Section 9, where an improved approximation algorithm for *satisfiable* instances of MAX 3CSP is presented.

In all the other 11 cases, random hyperplane rounding performs better than naive random choice, even if only simple clauses are allowed. While none of the obtained results is optimal, it should be noted that in most cases, the obtained performance ratio is very close to optimality. In the case of $\mathtt{MAJ}_3$, for example, we get a ratio of 0.666158 which is extremely close to the inapproximability bound of $\frac{2}{3}$. In Section 6 we show how to obtain an optimal $\frac{2}{3}$-approximation algorithm for MAX CSP($\mathtt{MAJ}_3$) using a modified rounding procedure. It should also be noted that the performance ratio obtained for $\mathtt{NAE}_3$ is exactly the performance ratio of the MAX CUT algorithm of Goemans and Williamson [GW95] (see also [KLP96]), and that the performance ratio that we get for $\mathtt{EQU}$ and $\mathtt{SEL}$ is exactly the performance ratio of the MAX DICUT algorithm of Goemans and Williamson [GW95]. In Section 7 we obtain an improved approximation algorithm for MAX CSP($\mathtt{NAE}_3$). This algorithm is our only algorithm in which we cannot allow a variable to appear twice in a clause. Improved approximation algorithms for all other problems are mentioned in Section 8.

The $\frac{1}{2}$-approximation algorithm for MAX 3AND yields immediately a $\frac{1}{2}$-approximation algorithm for MAX 3CSP. This follows from the fact that any constraint of the form $f(z_1, z_2, z_3)$ can be translated into a set of $\mathtt{AND}_3$ constraints of which no two are simultaneously satisfiable. (In the language of Section 10, there is a 1-gadget from $f$ to $\mathtt{AND}_3$.) If, for example, the satisfying assignments of $f$ are $(1,0,0)$, $(0,1,1)$ and $(0,0,0)$, then the clause $f(z_1, z_2, z_3)$ can be replaced by the clauses $z_1 \wedge \bar{z}_2 \wedge \bar{z}_3$, $\bar{z}_1 \wedge z_2 \wedge z_3$ and $z_1 \wedge z_2 \wedge z_3$.

## 5 A $\frac{1}{2}$-approx. algorithm for MAX 3AND

As mentioned in the previous section, the performance ratio obtained by rounding an optimal solution of the semidefinite relaxation given in Figure 3 using a random hyperplane is 0. To see this, we consider a configuration $v_0, v_1, v_2, v_3$ in which $\theta_{ij} = \arccos(-\frac{1-\epsilon}{3})$, for every $i < j$, where $\epsilon$ is a small constant. It is not difficult to check that such a configuration is realizable. (A given set of angles can be realized as a set of angles between a set of vectors if and only if the matrix $\{\cos\theta_{ij}\}$ is positive semidefinite.) It is easy to check that $relax_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3) = \frac{\epsilon}{4}$ and that $prob_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3) = \mathrm{Vol}(\lambda, \lambda, \lambda, \lambda, \lambda, \lambda)$, where $\lambda = \arccos(\frac{1-\epsilon}{3})$. Using the formula

$$\mathrm{Vol}(\lambda,\lambda,\lambda,\lambda,\lambda,\lambda) \;=\; 3\int_{\cos\lambda}^{\frac{1}{3}} \frac{\arccos\left(\frac{x}{1-2x}\right)}{\sqrt{1-x^2}}\,dx \;=$$

$$9(\frac{1}{3}-\cos\lambda)^{3/2} - \frac{567}{40}(\frac{1}{3}-\cos\lambda)^{5/2} + \ldots\,,$$

obtained by integrating the partial derivatives $\partial\mathrm{Vol}/\partial\lambda_{ij}$ of the volume function, we get that $prob_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3) \sim 9(\frac{\epsilon}{3})^{3/2}$ and therefore $prob_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3)/relax_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3) \sim \frac{3}{4}(\frac{\epsilon}{3})^{1/2}$. When $\epsilon$ tends to 0, the ratio tends to 0.

A ratio approaching 0 is also attained at other points. In all of them, however, $relax_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3)$ also approaches 0 and the contribution of the corresponding clause to the objective function of the semidefinite program is negligible. For such clauses, a random assignment performs exceptionally well. This suggests the strategy outlined in the previous section. We use a rounding procedure that generates *two* assignments and then chooses the best one among them. The first assignment is simply a random assignment. The second assignment is obtained by rounding the vectors $v_0, \ldots, v_{n-1}$ using a random hyperplane. We claim that this algorithm achieves an optimal performance ratio of $\frac{1}{2}$.

Analyzing this algorithm directly is difficult. We analyze instead the following rounding procedure:

- With probability $\frac{\pi}{4}$ round the vectors $v_0, \ldots, v_{n-1}$ using a random hyperplane.
- With probability $1-\frac{\pi}{4}$ choose a random assignment.

Clearly, the algorithm that does both things and chooses the better assignment behaves at least as well as this algorithm. A lower bound $\alpha'_{\mathtt{AND}_3}$ on the performance ratio of the algorithm that uses this combined rounding procedure can be obtained as follows:

$$\alpha'_{\mathtt{AND}_3} = \inf \frac{\frac{\pi}{4}\cdot prob_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3) + (1-\frac{\pi}{4})\cdot\frac{1}{8}}{relax_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3)}\,,$$

where the infimum is again taken over all the feasible configurations of the vectors $v_0, v_1, v_2, v_3$ for which $relax_f(v_0, v_1, v_2, v_3) > 0$.

It is easy to see that $\alpha'_{\mathtt{AND}_3} \le \frac{1}{2}$. To see this we consider the configuration in which $v_0, v_1, v_2, v_3$ are all perpendicular to one other, i.e., $\theta_{ij} = \frac{\pi}{2}$, for every $i < j$. For this configuration we get $prob_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3) = \frac{1}{8}$ and $relax_{\mathtt{AND}_3}(v_0, v_1, v_2, v_3) = \frac{1}{4}$. It follows therefore that $\alpha'_{\mathtt{AND}_3} \le \frac{1}{2}$.

Showing that $\alpha'_{\mathtt{AND}_3} = \frac{1}{2}$ is much harder. A relatively simple manipulation shows that it is equivalent to the following inequality involving the volume function:

$$\mathrm{Vol}(\lambda_{01}, \ldots, \lambda_{23}) + \frac{\pi}{2} \cdot (\cos\lambda_{01} + \cos\lambda_{02} + \cos\lambda_{03}) \ \ge\ \frac{\pi^2}{8} \ ,$$

for every 6-tuple of dihedral angles satisfying:

$$\begin{aligned} \cos\lambda_{01} + \cos\lambda_{02} &\ \ge\ \cos\lambda_{13} + \cos\lambda_{23} \ , \\ \cos\lambda_{01} + \cos\lambda_{03} &\ \ge\ \cos\lambda_{12} + \cos\lambda_{23} \ , \\ \cos\lambda_{02} + \cos\lambda_{03} &\ \ge\ \cos\lambda_{12} + \cos\lambda_{13} \ . \end{aligned}$$

We have a computer assisted proof of this inequality.

## 6 A $\frac{2}{3}$-approx. algorithm for MAX 3MAJ

As mentioned, the performance ratio obtained for MAX 3MAJ using random hyperplane rounding is $\alpha_{\mathtt{MAJ}_3} \simeq 0.666158$. This is extremely close to the inapproximability bound of $\frac{2}{3}$ that will be presented in Section 10. The ratio $\alpha_{\mathtt{MAJ}_3} \simeq 0.666158$ is attained when $\theta_{01} = \theta_{02} = \theta_{03} \simeq 1.537234 \simeq 0.489316\pi$ and $\theta_{12} = \theta_{13} = \theta_{23} \simeq 1.604359 \simeq 0.510684\pi$.

To obtain an optimal $\frac{2}{3}$-approximation algorithm for MAX 3MAJ we use a rounding procedure similar to one used by Feige and Goemans [FG95] for MAX 2SAT and for MAX DICUT. Let $f : [0, \pi] \to [0, \pi]$ be a function satisfying $f(\pi - \theta) = \pi - f(\theta)$, for every $0 \le \theta \le \pi$, and $f(0) = 0$. Before rounding the vectors $v_0, \ldots, v_{n-1}$, we *rotate* them with respect to $v_0$. The vector $v_i$ is rotated in the plane containing $v_0$ and $v_i$ so that it would now form an angle of $f(\theta_{0i})$ with $v_0$, where $\theta_{0i}$ is the original angle between $v_0$ and $v_i$. We denote the resulting set of vectors by $v'_0, \ldots, v'_{n-1}$. The vectors $v'_0, \ldots, v'_{n-1}$, and not $v_0, \ldots, v_{n-1}$, are then rounded by choosing a random hyperplane. The Boolean variable $x_i$ is assigned the value 1 if and only if $v'_i$ and $v'_0 = v_0$ lie on the same side of the random hyperplane chosen.

Feige and Goemans [FG95] use rotation functions of the form:

$$f_\lambda(\theta) = (1 - \lambda) \cdot \theta + \lambda \cdot \frac{\pi}{2}(1 - \cos\theta) \ .$$

Their 0.93109-approximation algorithm for MAX 2SAT is obtained using $\lambda = 0.806765$. A $\frac{2}{3}$-approximation algorithm for MAX 3MAJ can be obtained by using $\lambda = \frac{2}{3} \cdot \frac{\pi - 3}{\pi - 2} \simeq 0.0826872$. This value of $\lambda$ was chosen so that $f'_\lambda(\frac{\pi}{2}) = \frac{\pi}{3}$. The derivative of the function $f(\theta)$ at $\theta = \frac{\pi}{2}$ is the important factor here. The exact shape of the function $f(\theta)$ makes little difference. We can also use, for example, the function

$$g_{\lambda'}(\theta) = \theta - \lambda' \cdot \sin 2\theta \ ,$$

with $\lambda' = \frac{1}{2}(\frac{\pi}{3} - 1) \simeq 0.0235988$. We note that the values used for $\lambda$ and $\lambda'$ are fairly small and the rotation used here is 'gentle', compared to the rotation used by Feige and Goemans [FG95] for MAX 2SAT.

## 7 A $\frac{3}{2+\sqrt{2}}$-approx. algorithm for MAX 3NAE

An instance of MAX 3NAE can be easily converted into an equivalent instance of MAX CSP($\mathtt{XOR}_2$). Simply replace each clause $\mathtt{NAE}(z_1, z_2, z_3)$ with weight $w$ with the three clauses $z_1 \oplus z_2$, $z_1 \oplus z_3$ and $z_2 \oplus z_3$, each with a weight of $w/2$. Thus MAX 3NAE is at least as easy to approximate as MAX CSP($\mathtt{XOR}_2$).

An instance of MAX CSP($\mathtt{XOR}_2$) can be easily transformed into an equivalent instance of MAX 3NAE. Simply replace each clause $z_1 \oplus z_2$ with the clause $\mathtt{NAE}(z_1, z_1, z_2)$. MAX 3NAE and MAX CSP($\mathtt{XOR}_2$) are therefore equivalent problems. No wonder, therefore, that the performance ratio that we get for MAX 3NAE was exactly the performance ratio that [GW95] got for MAX CUT and MAX CSP($\mathtt{XOR}_2$).

Note, however, that while casting MAX CSP($\mathtt{XOR}_2$) instances as MAX 3NAE instances, we used clauses of the form $\mathtt{NAE}(z_1, z_1, z_2)$ in which the same literal appears twice in the same clause. We now show that if clauses of the form $\mathtt{NAE}(z_1, z_1, z_2)$, where $z_1$ is either a literal or a constant, are not allowed, then the problem becomes slightly easier to approximate. We present a $\frac{3}{2+\sqrt{2}}$-approximation algorithm for this version. Note that $\frac{3}{2+\sqrt{2}} \simeq 0.878679$ is marginally larger that $\alpha_{\mathtt{XOR}_2} = \min \frac{2}{\pi} \frac{\theta}{1-\cos\theta} \simeq 0.878567$. Nonetheless, we think that this result is important as it may eventually be used to obtain an improved approximation algorithm for the MAX CUT problem. It immediately implies, for example, a slightly improved performance ratio for MAX CUT in graphs that contain many triangles.

To obtain the improved performance ratio, we impose the additional constraints of Figure 4 and then use the rounding procedure used in Section 5, i.e., we consider a random assignment and an assignment obtained using random hyperplane rounding and take the better one. We analyze the variant of this rounding procedure in which random hyperplane rounding is used with probability $\frac{3}{4} \cdot \frac{\pi}{\sqrt{2}+1} \simeq 0.975968$, and a random assignment is used with probability $1 - \frac{3}{4} \cdot \frac{\pi}{\sqrt{2}+1} \simeq 0.024032$. For brevity, we let $\epsilon = 1 - \frac{3}{4} \frac{\pi}{\sqrt{2}+1}$. A lower bound on the performance ratio obtained using this combined rounding procedure is

$$\alpha'_{\mathtt{NAE}_3} \;=\; \inf \frac{(1-\epsilon)\cdot\dfrac{\theta_{12}+\theta_{13}+\theta_{23}}{2\pi}+\epsilon\cdot\dfrac{3}{4}}{\dfrac{3-\cos\theta_{12}-\cos\theta_{13}-\cos\theta_{23}}{4}}\,,$$

where the infimum is over all $0 \le \theta_{01}, \theta_{02}, \theta_{12} \le \pi$ such that $\cos\theta_{12}+\cos\theta_{13}+\cos\theta_{23} \ge -1$, $\cos\theta_{12}-\cos\theta_{13}-\cos\theta_{23} \ge -1$, $-\cos\theta_{12}+\cos\theta_{13}-\cos\theta_{23} \ge -1$ and $-\cos\theta_{12}-\cos\theta_{13}+\cos\theta_{23} \ge -1$. As this expression does not involve volumes, it is not difficult to show analytically that $\alpha'_{\mathtt{NAE}_3} = \frac{3}{2+\sqrt{2}}$. The proof will appear in the full version of the paper. The worst ratio is attained when $\theta_{12} = \theta_{13} = \frac{3\pi}{4}$ and $\theta_{23} = 0$.

## 8 Other approximation algorithms

As MAX CSP($\mathtt{AXR}_3$) and MAX CSP($\mathtt{TWO}_3$) are subproblems of MAX 3CSP, we get an optimal performance ratio of $\frac{1}{2}$ for these two problems. An improved performance ratio of 0.859 for MAX CSP($\mathtt{SEL}_3$) is obtained by reducing an instance of this problem into an equivalent instance of MAX 2CSP. A clause $\mathtt{SEL}(z_1, z_2, z_3) = z_1\bar{z}_3 \vee z_1 z_2$ is simply replaced by the two clauses $z_1 \wedge \bar{z}_3$ and $z_2 \wedge z_3$. Note that these two clauses cannot be satisfied simultaneously.

Slightly improved performance ratios for $\mathtt{AOR}_3$, $\mathtt{AOA}_3$, $\mathtt{XAD}_3$, $\mathtt{OAD}_3$ and $\mathtt{XOA}_3$ can be obtained using rotations, as described in Section 6. The exact details will appear in the full version of the paper.

It is interesting to note that the performance ratio obtained for $\mathtt{XAD}_3$ using random hyperplane rounding is the same as that obtained for $\mathtt{MAJ}_3$. While for $\mathtt{MAJ}_3$, we were able to improve the performance ratio to the optimal value of $\frac{2}{3}$ using rotations, this does not seem to work for $\mathtt{XAD}_3$.

Another interesting observation is that the performance ratios obtained using random hyperplane rounding for $\mathtt{AOR}_3$ and $\mathtt{AOA}_3$, and for $\mathtt{OAD}_3$ and $\mathtt{XOA}_3$ are the same, although we have not been able to show the equivalence of the corresponding problems. Slightly different performance ratios are obtained for these problems when rotations are considered.

## 9 Approximating satisfiable instances

An instance of MAX CSP($f$) is said to be *satisfiable* if there is an assignment that satisfies all the clauses. Satisfiable instances are sometimes easier to solve. Given a satisfiable instance of MAX 2CSP, for example, we can find a satisfiable assignment in polynomial time. For some other problems, like MAX 3SAT, finding a satisfiable assignment remains NP-hard. In fact, satisfiable instances of MAX 3SAT are as hard to approximate as general MAX 3SAT instances. Sometimes, the problem remains NP-hard but becomes easier to approximate.

In this section we build on nice ideas of Trevisan [Tre97] and obtain an improved approximation algorithm for *satisfiable* instances of MAX 3CSP. It is not difficult to see that clauses involving $\mathtt{AND}_3$, $\mathtt{EQU}_3$, $\mathtt{AXR}_3$, $\mathtt{AOR}_3$ and $\mathtt{AOA}_3$, as well as those involving $\mathtt{AND}_2$ and $\mathtt{XOR}_2$, can be easily removed from satisfiable instances. If a satisfiable instance includes, for example, a clause $\mathtt{AXR}(z_1, z_2, z_3)$, we can immediately deduce that $z_1 = 1$ and that $z_3 = \bar{z}_2$. We can therefore replace all the occurrences of the literal $z_1$ by 1, and all the occurrences of the literal $\bar{z}_1$ by 0. Similarly, we can replace all the occurrences of $z_3$ by $\bar{z}_2$ and all the occurrences of $\bar{z}_3$ by $z_2$. We may therefore concentrate on the other types of constraints.

A lower bound $\beta_f$ on the performance ratio of the algorithm obtained by solving the semidefinite relaxation of MAX CSP($f$) and then rounding the vectors using a random hyperplane can be obtained as follows:

$$\beta_f \;=\; \inf_{relax_f(v_0,v_1,v_2,v_3)=1} prob_f(v_0, v_1, v_2, v_3)\;.$$

The values of $\beta_f$ for the functions $f$ that we are interested in appear in the SDP row of Figure 6. All these bounds have closed forms that will appear in the full version of the paper. We mention here in passing that many of these bounds can be improved using more sophisticated rounding procedures. We do not have space to elaborate on this here. It is important to note that the performance ratios given in the SDP row are attained *simultaneously*, using the same rounding procedure.

A quick look at Figure 6 reveals that for any constraint type, except $\mathtt{XOR}_3$, we obtain a ratio of at least $\frac{5}{8}$. If all the constraints are of this type, the problem is easy, as we can find a satisfying assignment using *Gaussian elimination*. What happens, however, if we have a mixture of constraint types? Trevisan [Tre97] suggests the following approach: Use Gaussian elimination to find a basis of the afine linear subspace containing all the solutions to the $\mathtt{XOR}_3$-type constraints. Here we supplement this idea with the observation that for any clause of the form $\mathtt{TWO}(z_1, z_2, z_3)$, we can add to the linear system the constraint $z_1 \oplus z_2 \oplus z_3 = 0$. We now consider each clause $f(z_1, z_2, z_3)$ of the instance. If there exists a satisfying assignment of $f(z_1, z_2, z_3)$ which is inconsistent with the linear subspace of solutions, i.e., if no point in the subspace yields this assignment to

| algorithm | TWO | $\mathtt{XOR}_3$ | MAJ | XAD | SEL | OAD | XOA | NTW | $\mathtt{OR}_2$ | NAE | OXR | $\mathtt{OR}_3$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SDP | 0.649 | $\frac{1}{2}$ | 0.736 | 0.736 | $\frac{5}{6}$ | 0.824 | 0.824 | $\frac{5}{8}$ | 0.912 | 0.912 | $\frac{3}{4}$ | $\frac{7}{8}$ |
| RGE | $\frac{3}{4}$ | 1 | $\frac{1}{2}$ | $\frac{1}{2}$ | $\frac{1}{2}$ | $\frac{5}{8}$ | $\frac{5}{8}$ | $\frac{5}{8}$ | $\frac{3}{4}$ | $\frac{3}{4}$ | $\frac{3}{4}$ | $\frac{7}{8}$ |

Figure 6: Approximating satisfiable instances of MAX 3CSP.

$z_1, z_2$ and $z_3$, then the clause $f(z_1, z_2, z_3)$ can be simplified. For example, if the clause is $\mathtt{NTW}(z_1, z_2, z_3)$ and in no point of the solution subspace we have $(z_1, z_2, z_3) = (0, 0, 0)$, we can replace the constraint $\mathtt{NTW}(z_1, z_2, z_3)$ with the constraint $\mathtt{XOR}_3(z_1, z_2, z_3)$. We can now add a new equation to the linear system and obtain, perhaps, a slightly smaller space of solutions. We repeat this process until all satisfiable assignments of all the clauses of the instance are consistent with the afine subspace of solutions. We then pick a *random* point of this solution subspace. It is not difficult to verify that the different types of constraints are now satisfied with the probabilities given in the RGE row of Figure 6. The important things to note is that all the $\mathtt{XOR}_3$-type clauses are now satisfied, and that each $\mathtt{TWO}(z_1, z_2, z_3)$ clause is satisfied with probability $\frac{3}{4}$, as in each assignment generated we have $z_1 \oplus z_2 \oplus z_3 = 0$. All other clauses are satisfied with exactly the same probability that they would have been satisfied, had the assignment been chosen completely at random, and not from the afine subspace.

We thus have two approximation algorithms whose performance ratios are given in Figure 6. We obtain a combined algorithm by running both algorithms, comparing the quality of the two assignments produced and choosing the better one. We claim that the performance ratio of this combined algorithm, for satisfied clauses of MAX 3CSP, is $\frac{5}{8}$. This follows from the fact that the performance ratio of this algorithm is at least as good as the performance ratio of the algorithm that runs SDP, our first algorithm, with probability $\frac{3}{4}$, and algorithm RGE, our second algorithm, with probability $\frac{1}{4}$. It is interesting to note that the hardest satisfiable instances of MAX 3CSP turn out to be instances in which all clauses are $\mathtt{NTW}_3$ clauses.

## 10 Hardness results

The inapproximability bounds given in Figure 1 and 5 are obtained using *gadgets* (see [BGS95] and [TSSW96]). Most of these gadgets are new. They will appear in the full version of the paper. The larger inapproximability bounds given for $\mathtt{SEL}_3$ and $\mathtt{NAE}_3$ in Figure 5 correspond to the case in which only simple clauses are allowed.

## 11 Concluding remarks

We find it remarkable that semidefinite programming yields optimal approximation algorithms for so many constraint satisfaction problems that involve at most three variables per constraint. Many interesting and challenging open problems still remain, however. The most important ones are perhaps improving either the performance guarantees, or the inapproximability bounds, for MAX CUT and MAX 2SAT.


## Acknowledgment

We would like to thank Greg Sorkin for supplying some of the gadgets used to obtained the hardness results. We would like to thank Daniel Lewin for bringing reference [Vin93] to our attention and for sharing his ideas regarding the proofs of inequalities involving the volume function of spherical tetrahedra.


## References


[AE97] G. Andersson and L. Engebretsen. Better approximation algorithms and tighter analysis for Set Splitting and Not-All-Equal Sat. E-CCC Report TR97-022, 1997.

[ALM+92] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proc. of the 33rd Annual IEEE Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania*, pages 14–23, 1992.

[AS92] S. Arora and M. Safra. Probabilistic checking of proofs: a new characterization of NP. In *Proc. of the 33rd Annual IEEE Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania*, pages 2–13, 1992.

[Asa97] T. Asano. Approximation algorithms for MAX SAT: Yannakakis vs. Goemans-Williamson. In *Proc. of the 3nd Israel Symposium on Theory and Computing Systems, Ramat Gan, Israel*, pages 24–37, 1997.

[BGLR93] M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient probabilistically checkable proofs and applications to approximation. In *Proc. of the 25rd Annual ACM Symposium on Theory of Computing, San Diego, California*, pages 294–304, 1993. See Errata in STOC'94.

[BGS95] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCPs and non-approximability—towards tight results. In *Proc. of the 36rd Annual IEEE Symposium on Foundations of Computer Science, Milwaukee, Wisconsin*, pages 422–431, 1995. Full version available as E-CCC Report number TR95-024.

[CFZ97] J. Chen, D. Friesen, and H. Zheng. Tight bound on Johnson's algorithm for max-SAT. In *Proc. of the 12th Annual IEEE Conference on Computational Complexity, Ulm, Germany*, pages 274–281, 1997.

[Cox57] H.S.M. Coxeter. *Non-Euclidean Geometry.* The University of Toronto Press, 1957.

[Cre95] N. Creignou. A dichotomy theorem for maximum generalized satisfiability problems. *Journal of Computer and System Sciences*, 51:511–522, 1995.

[CT97] P. Crescenzi and L. Trevisan. Max NP-completeness made easy. Technical Report TR97-039, Electronic Colloquium on Computational Complexity, 1997.

[FG95] U. Feige and M.X. Goemans. Approximating the value of two prover proof systems, with applications to MAX-2SAT and MAX-DICUT. In *Proc. of the 3nd Israel Symposium on Theory and Computing Systems, Tel Aviv, Israel*, pages 182–189, 1995.

[FGL+96] 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.

[GW94] 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.

[GW95] 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.

[Hås97] J. Håstad. Some optimal inapproximability results. In *Proc. of the 28th Annual ACM Symposium on Theory of Computing, El Paso, Texas*, pages 1–10, 1997.

[Joh74] D.S. Johnson. Approximation algorithms for combinatorical problems. *Journal of Computer and System Sciences*, 9:256–278, 1974.

[Kar96] H. Karloff. How good is the Goemans-Williamson MAX CUT algorithm? In *Proc. of the 28th Annual ACM Symposium on Theory of Computing, Philadelphia, Pennsylvania*, pages 427–434, 1996.

[KLP96] V. Kann, J. Lagergren, and A. Panconesi. Approximability of maximum splitting of $k$-sets and some other APX-complete problems. *Information Processing Letters*, 58:105–110, 1996.

[KMSV94] S. Khanna, R. Motwani, M. Sudan, and U. Vazirani. On syntactic versus computational views of approximability. In *Proc. of the 35rd Annual IEEE Symposium on Foundations of Computer Science, Santa Fe, New Mexico*, pages 819–830, 1994.

[KST97] S. Khanna, M. Sudan, and L. Trevisan. Constraint satisfaction: The approximability of minimization problems. In *Proc. of the 12th Annual IEEE Conference on Computational Complexity, Ulm, Germany*, pages 282–296, 1997. Full version available as E-CCC Report number TR96-064.

[KSW97] S. Khanna, M. Sudan, and D.P. Williamson. A complete classification of the approximability of maximization problems derived from Boolean constraint satisfaction. In *Proc. of the 28th Annual ACM Symposium on Theory of Computing, El Paso, Texas*, pages 11–20, 1997. Full version available as E-CCC Report number TR96-062.

[KZ97] H. Karloff and U. Zwick. A 7/8-approximation algorithm for MAX 3SAT? In *Proc. of the 38rd Annual IEEE Symposium on Foundations of Computer Science, Miami Beach, Florida*, 1997. 406–415.

[MR95] S. Mahajan and H. Ramesh. Derandomizing semidefinite programming based approximation algorithms. In *Proc. of the 36rd Annual IEEE Symposium on Foundations of Computer Science, Milwaukee, Wisconsin*, pages 162–169, 1995.

[Pet94] E. Petrank. The hardness of approximation: Gap location. *Computational Complexity*, 4:133–157, 1994.

[PY91] C.H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.

[Sch58] 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.

[Sch78] T.J. Schaefer. The complexity of satisfiability problems. In *Proc. of the 10th Annual ACM Symposium on Theory of Computing, San Diego, California*, pages 216–226, 1978.

[Tre97] L. Trevisan. Approximating satisfiable satisfiability problems. In *Proc. of the 5th European Symposium on Algorithms, Graz, Austria*, pages 472–485, 1997.

[TSSW96] L. Trevisan, G.B. Sorkin, M. Sudan, and D.P. Williamson. Gadgets, approximation, and linear programming (extended abstract). In *Proc. of the 37rd Annual IEEE Symposium on Foundations of Computer Science, Burlington, Vermont*, pages 617–626, 1996.

[Vin93] E.B. Vinberg. Volumes of non-Euclidean polyhedra. *Russian Math. Surveys*, 48:15–45, 1993.

[Yan94] M. Yannakakis. On the approximation of maximum satisfiability. *Journal of Algorithms*, 17:475–502, 1994.