A $7/8$-Approximation Algorithm for MAX 3SAT?

Howard Karloff *   *Uri Zwick* †

Abstract

We describe a randomized approximation algorithm which takes an instance of MAX 3SAT as input. If the instance—a collection of clauses each of length at most *three—is satisfiable, then the expected weight of the assignment found is at least $7/8$ of optimal. We provide strong evidence (but not a proof) that the algorithm performs equally well on arbitrary MAX 3SAT instances.*

Our algorithm uses semidefinite programming and may be seen as a sequel to the MAX CUT algorithm of Goemans and Williamson and the MAX 2SAT algorithm of Feige and Goemans. Though the algorithm itself is fairly simple, its analysis is quite complicated as it involves the computation of volumes of spherical tetrahedra.

Håstad has recently shown that, assuming $P \neq NP$, no polynomial-time algorithm for MAX 3SAT can achieve a performance ratio exceeding $7/8$, even when restricted to satisfiable instances of the problem. Our algorithm is therefore optimal in this sense.

We also describe a method of obtaining direct semidefinite relaxations of any constraint satisfaction problem of the form MAX CSP($\mathcal{F}$), where $\mathcal{F}$ is a finite family of Boolean functions. Our relaxations are the strongest possible within a natural class of semidefinite relaxations.

1 Introduction

MAX SAT is a central problem in theoretical computer science. As it is NP-hard and, in fact, MAX-SNP complete (Papadimitriou and Yannakakis [30]), much attention has been devoted to approximating it. The first approximation algorithm for MAX SAT was proposed by Johnson [23]. Johnson showed that the performance ratio of his algorithm is at least $1/2$. Chen, Friesen and Zheng [12] recently showed that the performance ratio of his algorithm is actually 2/3. Many years passed before Yannakakis [37] obtained a 3/4-approximation algorithm for the problem. Goemans and Williamson [17] then obtained a different and somewhat simpler 3/4-approximation algorithm. Their algorithm is based on a linear programming relaxation of the problem.

In a major breakthrough, Goemans and Williamson [18] obtained a 0.878-approximation algorithm for MAX CUT and MAX 2SAT, the version of MAX SAT in which each clause is of size at most two. Goemans and Williamson used semidefinite relaxations of these problems. Feige and Goemans [15] then obtained a 0.931-approximation algorithm for MAX 2SAT. Using the MAX 2SAT algorithms of Goemans and Williamson or of Feige and Goemans, slight improvements in the performance ratio for general MAX SAT can be made. Goemans and Williamson [18] obtained a 0.758 bound for MAX SAT. Asano [4] (following [5]) slightly improved this bound to 0.770.

While semidefinite relaxations yield a huge improvement for MAX 2SAT (from 0.75 to 0.931), they give, so far, only a minor improvement for MAX SAT (from 0.75 to 0.770). The reason for this seems to be that the semidefinite relaxations used till now do not directly handle clauses of length three or more.

An attempt to squeeze more from the MAX 2SAT algorithm of Feige and Goemans [15] was made by Trevisan, Sorkin, Sudan and Williamson [34]. They used an optimal gadget, a concept formalized by Bellare, Goldreich and Sudan [7], to reduce MAX 3SAT (the problem in which each clause has length at most three) to MAX 2SAT, thereby obtaining a 0.801-approximation algorithm for MAX 3SAT. Trevisan [33] recently obtained a 0.826-approximation algorithm for *satisfiable* instances of MAX 3SAT, and a 0.8-approximation algorithm for *satisfiable* instances of MAX SAT.

In another major breakthrough, following a long line of research by many authors [16, 3, 2, 8, 7], Håstad [20] recently showed that MAX E3SAT, the version of the MAX SAT problem in which each clause is of length *exactly* three, cannot be approximated in polynomial time to within a ratio greater than 7/8, unless $P = NP$. Håstad shows, in fact, that no polynomial-time algorithm can have a performance

*College of Computing, Georgia Institute of Technology, Atlanta, Georgia 30332-0280, U.S.A. Research supported in part by NSF grant CCR-9319106. E-mail address: howard@cc.gatech.edu.

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

guarantee of more than 7/8 even when restricted to just satisfiable instances of the problem. Håstad's result is easily seen to be tight: a random assignment satisfies, on average, 7/8 of the total weight of a MAX E3SAT instance.

In this paper we present a new approximation algorithm for MAX 3SAT. The algorithm takes as input an instance of MAX 3SAT and runs in time bounded by a polynomial in the length of this instance. If the instance is satisfiable, the expected weight of the assignment the algorithm returns is at least $7/8$ of the weight of any assignment to the instance. We conjecture (but have only strong evidence, not a proof) that the same performance is achieved on arbitrary MAX 3SAT instances. Our algorithm can possibly be derandomized using the techniques of Mahajan and Ramesh [28].

The novelty of our algorithm is that it uses a direct semidefinite relaxation of MAX 3SAT. The algorithm itself is quite simple. The analysis relies, however, on two inequalities involving the volume function of spherical tetrahedra (Lemma 4.4 and Conjecture 4.5 below). Proving these two inequalities seems, at least for now, extremely complicated. We provide a computer-assisted proof of the first inequality. For the second inequality we are able to present only strong numerical evidence. We are currently working on a simpler proof of the first inequality and a complete, and hopefully simple, proof of the second inequality. The first inequality alone implies that the performance ratio of the algorithm for satisfiable instances of MAX 3SAT is at least 7/8.

We also describe a method of obtaining direct semidefinite relaxations of any constraint satisfaction problem of the form MAX CSP($\mathcal{F}$) or MIN CSP($\mathcal{F}$), where $\mathcal{F}$ is a finite family of Boolean functions. Such problems were studied by Khanna, Sudan and Williamson [27] and by Khanna, Sudan and Trevisan [26]. Our relaxations are the strongest possible within a natural class of semidefinite relaxations. This class includes almost all the semidefinite relaxations proposed to date for these problems.

We hope that the results presented here pave the way for similar improvements for MAX SAT.

2 MAX 3SAT

An instance of MAX 3SAT in n variables is an array (w_{ijk}) of nonnegative weights, where $0 \le i,j,k \le 2n$. A valid assignment $x = (x_0, x_1, \ldots, x_{2n})$ is a 0-1 vector such that $x_0 = 0$ and $x_{n+i} = \bar{x}_i$, for $1 \le 1 \le n$. A clause $\langle i,j,k \rangle$ is satisfied by x iff $x_i \vee x_j \vee x_k = 1$, i.e., if at least one of x_i, x_j or x_k is assigned the value 1. The weight of a valid assignment is the sum of the weights of all the satisfied clauses, i.e., $weight(x) = \sum_{i,j,k} w_{ijk}(x_i \vee x_j \vee x_k)$. The optimal solution to the MAX 3SAT instance is an assignment of maximum weight. Note that we always require $x_0 = 0$. This means that some of the clauses are effectively of length two or one (or zero).

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

Figure 1. A direct semidefinite relaxation of a MAX 3SAT instance.

3 The New Approximation Algorithm for MAX 3SAT

A direct semidefinite relaxation of a generic MAX 3SAT instance is presented in Figure 1. In this relaxation, we attach a unit vector v_i to each Boolean variable, $1 \le i \le n$, and a scalar z_{ijk} to each clause. We also have a special vector v_0 that corresponds to FALSE. The vectors $v_0, v_1, \ldots, v_n$ are vectors on the Euclidean unit sphere S^n in R^{n+1}.

The scalar z_{ijk} is meant to get the value 1 if the clause is satisfied and 0 if it is not satisfied. Define

$$
relax(v_0,v_i,v_j,v_k) =
$$
$$
\min \left\{ \begin{array}{cc} \frac{4-(v_0+v_i)\cdot(v_j+v_k)}{4} , & \frac{4-(v_0+v_j)\cdot(v_i+v_k)}{4} \\ \frac{4-(v_0+v_k)\cdot(v_i+v_j)}{4} , & 1 \end{array} \right\} .
$$

The constraints on z_{ijk} in the relaxation are equivalent to the constraint $z_{ijk} \le relax(v_0,v_i,v_j,v_k)$.

It is fairly easy to see that program presented in Figure 1 is equivalent to a semidefinite program. All we have to do is replace each inner product $v_i \cdot v_j$ by a scalar x_{ij}, add the constraints $x_{ii} = 1$, and require that the matrix (x_{ij}) be positive semidefinite. The constraint $v_{n+i} = -v_i$ is equivalent to $v_i \cdot v_{n+i} = -1$. (The z_{ijk}'s can be assumed nonnegative but need not satisfy any semidefiniteness constraints.) Why is it a relaxation of MAX 3SAT? Let $x \in \{0,1\}^{2n+1}$ be a valid assignment. Let $v_i = (-1,0,\ldots,0)$ if $x_i = 0$ and $v_i = (1,0,\ldots,0)$ if $x_i = 1$. Let $v_0 = (-1,0,\ldots,0)$. It is easy to check (see Table 1) that $relax(v_0,v_i,v_j,v_k) = x_i \vee x_j \vee x_k$. The program described in Figure 1 is therefore a relaxation of MAX 3SAT.

It is worthwhile noting that if $i = j = 0$ and $k \ne 0$, i.e., the clause $\langle i,j,k \rangle$ is in fact a clause of length one, then

v_i	v_j	v_k	u_i	u_j	u_k
-1	-1	-1	0	0	0
-1	-1	1	1	1	1
-1	1	-1	1	1	1
-1	1	1	2	1	1
1	-1	-1	1	1	1
1	-1	1	1	2	1
1	1	-1	1	1	2
1	1	1	1	1	1

Table 1. The components of $relax(v_0, v_i, v_j, v_k)$ as a function of $v_i, v_j, v_k \in \{-1, +1\}$, where $v_0 = -1$.

the relaxation of the clause $\langle i,j,k\rangle$ simplifies to the expression $relax(v_0, v_0, v_0, v_k) = \frac{1-v_0\cdot v_k}{2}$, and if $i = 0$ but $j,k \neq 0$ and $j \neq k$, i.e., the clause $\langle i,j,k\rangle$ is of length two, the relaxation simplifies to $relax(v_0, v_0, v_j, v_k) = \min\left\{ \frac{3-v_0\cdot v_j - v_0\cdot v_k - v_j\cdot v_k}{4}, 1 \right\}$. For clauses of length one we get, therefore, exactly the MAX CUT relaxation of Goemans and Williamson [18]. For clauses of length one and two we get, almost exactly, the MAX 2SAT relaxation used by Feige and Goemans [15].

The expression $relax(v_0, v_i, v_j, v_k)$ includes 1 as one of its terms. This is not needed for obtaining a relaxation. It is used, however, in showing that the relaxation obtained is a good one. Where does this relaxation comes from? This is explained in Section 5.

The semidefinite program described above can be solved in polynomial time. To be more precise, an (almost) feasible point whose cost is within an *additive* error of ϵ of optimal can be found in time polynomial in the size of the problem and $\log \frac{1}{\epsilon}$. (See Alizadeh [1], Grötschel *et al.* [19], Nesterov and Nemirovskii [29], Pataki [31] and Vaidya [35], and the survey paper of Vandenberghe and Boyd [36].) It follows easily that if the value of the optimal solution is opt, then a feasible solution, i.e., a sequence of unit vectors $v_0, \ldots, v_n$ and a set of scalars z_{ijk}, that satisfy all the constraints and for which $\sum_{ijk} w_{ijk} z_{ijk} \geq (1-\epsilon)opt$ can also be found in time polynomial in the size of the problem and $\log \frac{1}{\epsilon}$. We need to "round" the vectors $v_0, \ldots, v_n$ to truth values. We use the simple randomized rounding procedure introduced by Goemans and Williamson [18] to round the vectors $v_0, ..., v_n$ to truth values. We pick a random hyperplane that passes through the origin. The Boolean variable x_i is assigned the value 1 if and only if the random hyperplane separates v_i from v_0.

In the next section we analyze the performance ratio of the new approximation algorithm assuming that the semidefinite relaxation given in Figure 1 can be solved *exactly*. Unfortunately, the semidefinite relaxations cannot always be solved exactly, at least because the optimal solution is not always rational. Suppose that the performance ratio of the algorithm when the semidefinite program is solved exactly is β. Let I be an instance of the problem of size m. Let $opt(I)$ be the value of an optimal solution to the semidefinite relaxation of the instance I. If for every instance I, the value of the approximate solution found for the relaxation is at least $(1-\epsilon)opt(I)$, then the performance ratio of the algorithm that uses approximate solutions is at least $(1-\epsilon)\beta$. We can take ϵ to be an arbitrarily small constant, or even $\epsilon = 1/(W\cdot 2^m)$, where W is the sum of the weights in the instance I, and m is the number of clauses in the instance, and still have a polynomial-time algorithm.

There is, however, a simple way of getting a performance ratio of at least β even when the semidefinite relaxation is not solved exactly. Let I be an instance of MAX 3SAT in the variables $x_1, \ldots, x_n$. We may assume, w.l.o.g., that x_1 appears both positively and negatively in the instance. Let I_0 be the instance obtained from I by assigning x_1 the value 0. Let I_1 be the instance obtained from I by assigning x_1 the value 1. Solve *both* instances I_0 and I_1 using the approximation algorithm that achieves a performance ratio of at least $(1-\epsilon)\beta$, with $\epsilon = (1-\beta)/W$, compare the values of the two assignments obtained and return the better one. We claim that the performance ratio of this algorithm is at least β. To see this, assume, w.l.o.g., that there is an optimal assignment of I in which x_1 is assigned the value 0. Let A be the total weight of the clauses of I in which x_1 appears negatively. Then the value of the assignment found by solving I_0 is at least $A + (1-\epsilon)\beta\cdot(opt(I) - A) \geq \beta\cdot opt(I)$, where we have used the fact that $A \geq 1$ and $opt(I) \leq W$.

4 The Performance Ratio of the Algorithm

All that remains is to analyze the performance ratio of the algorithm. We do this in two stages. We first analyze the performance of the algorithm for clauses of size one of two. The analysis in this case is identical to the analysis of Goemans and Williamson [18]. We describe the analysis here as a "warm up" for the much more complicated analysis for clauses of length three.

4.1 Clauses of Length 1 and 2

A clause x_i is relaxed into $(1-v_0\cdot v_i)/2$. The probability that a random hyperplane separates v_0 and v_i is θ/π, where θ is the angle between the two vectors v_0 and v_i, $\cos\theta = v_0\cdot v_i$. The performance ratio of the algorithm for clauses of size one is therefore at least $\alpha_1 = \min_{0\leq\theta\leq\pi} \frac{2}{\pi}\frac{\theta}{1-\cos\theta} > 0.87856 > 7/8$.

Consider now a clause of length two, e.g., $x_1 \vee x_2$. The clause $x_1 \vee x_2$ is relaxed into $relax(v_0, v_1, v_2) = \frac{3-v_0\cdot v_1 - v_0\cdot v_2 - v_1\cdot v_2}{4}$. (We ignore here the minimum with 1 in the definition of $relax(v_0, v_1, v_2)$.) Given three vectors v_0, v_1 and v_2, what is the probability that the rounded assignment satisfies the clause $x_1 \vee x_2$? This is exactly the probability that a random hyperplane separates at least one of v_1 and v_2 from v_0, which equals $1 - prob(v_0, v_1, v_2)$, where $prob(v_0, v_1, \ldots, v_k)$

$$= \Pr\left[\begin{array}{l} v_0, v_1, \ldots, v_k \text{ lie on the same} \\ \text{side of a random hyperplane} \end{array}\right].$$

The performance ratio of the algorithm for clauses of length two is at least $\alpha_2 = \min_{v_0,v_1,v_2 \in S^n} \frac{1-prob(v_0,v_1,v_2)}{relax(v_0,v_1,v_2)}$. What is $1 - prob(v_0, v_1, v_2)$? There is a simple way of working out this probability: $1 - prob(v_0, v_1, v_2) = \frac{1}{2} \cdot (prob(v_0|v_1) + prob(v_0|v_2) + prob(v_1|v_2))$, where $prob(v_i|v_j) = 1 - prob(v_i, v_j)$ is the probability that a random hyperplane separates v_i and v_j. We saw already that $prob(v_i|v_j) = \frac{\theta_{ij}}{\pi}$, where $\theta_{ij} = \arccos(v_i \cdot v_j)$ is the angle between the vectors v_i and v_j. Thus

$$\alpha_2 = \min_{0 \le \theta_{ij} \le \pi} \frac{\frac{1}{2\pi}(\theta_{01} + \theta_{02} + \theta_{12})}{\frac{3 - \cos\theta_{01} - \cos\theta_{02} - \cos\theta_{12}}{4}}$$

$$= \frac{2}{\pi} \min_{0 \le \theta_{ij} \le \pi} \frac{\theta_{01} + \theta_{02} + \theta_{12}}{(1 - \cos\theta_{01}) + (1 - \cos\theta_{02}) + (1 - \cos\theta_{12})},$$

and it is easy to see, therefore, that $\alpha_2 = \alpha_1 > 0.87856$.

4.2 Clauses of Length 3

Consider now a clause of length 3, say $x_1 \vee x_2 \vee x_3$. The performance ratio of the algorithm for clauses of length 3 is at least $\alpha_3 = \min_{v_0,v_1,v_2,v_3 \in S^n} \frac{1-prob(v_0,v_1,v_2,v_3)}{relax(v_0,v_1,v_2,v_3)}$. The performance ratio of the algorithm for *satisfied* clauses of length 3 is at least $\alpha'_3 = \min\,[1 - prob(v_0, v_1, v_2, v_3)]$, where the minimum is over $v_0, v_1, v_2, v_3 \in S^n$ such that $relax(v_0, v_1, v_2, v_3) = 1$. The simple way of evaluating $1 - prob(v_0, v_1, v_2)$ cannot be used, unfortunately, for evaluating $1 - prob(v_0, v_1, v_2, v_3)$. We have to use, therefore, a different way that relies more heavily on spherical geometry.

A random hyperplane that passes through the origin is conveniently chosen by choosing its normal vector r uniformly at random in S^n. Note that

$$\begin{array}{c} prob(v_0, v_1, v_2, v_3) = \\ \Pr[\, sgn(v_0 \cdot r) = sgn(v_1 \cdot r) = sgn(v_2 \cdot r) = sgn(v_3 \cdot r) \,] \\ = 2 \cdot \Pr[\, v_0 \cdot r \ge 0\ ,\ v_1 \cdot r \ge 0\ ,\ v_2 \cdot r \ge 0\ ,\ v_3 \cdot r \ge 0 \,]\,. \end{array}$$

As we are only interested in the inner products $v_0 \cdot r, v_1 \cdot r, v_2 \cdot r$ and $v_3 \cdot r$, we are only interested in the projection of r into the 4-space spanned by v_0, v_1, v_2 and v_3. It is not difficult to see that r', the normalized projection of r into this 4-space, is uniformly distributed on the unit sphere of this 4-space. We may assume, therefore, that $v_0, v_1, v_2, v_3, r \in S^3$, the unit sphere in R^4.

Let $v_0, v_1, v_2, v_3 \in S^3$. The *spherical tetrahedron* $tetra(v_0, v_1, v_2, v_3)$ is defined as follows: $tetra(v_0, v_1, v_2, v_3) = \{ \sum_{i=0}^{3} \alpha_i v_i \mid \alpha_i \ge 0, \sum_{i=0}^{3} \alpha_i v_i \in S^3 \}$. A spherical tetrahedron is said to be *nondegenerate* if v_0, v_1, v_2, v_3 are linearly independent. In this case, the vectors v_0, v_1, v_2, v_3 are said to be the *vertices* of $tetra(v_0, v_1, v_2, v_3)$. If $tetra(v_0, v_1, v_2, v_3)$ is a nondegenerate spherical tetrahedron, then its *polar* tetrahedron, denoted by $tetra'(v_0, v_1, v_2, v_3)$, is defined to be $tetra(u_0, u_1, u_2, u_3)$, where $u_0, u_1, u_2, u_3 \in S^3$ satisfy $u_i \cdot v_i > 0$, for $0 \le i \le 3$, and $u_i \cdot v_j = 0$, for $0 \le i, j \le 3$, $i \ne j$. As v_0, v_1, v_2, v_3 are linearly independent, this determines u_0, u_1, u_2, u_3 uniquely. It is easy to see that the polar tetrahedron can be defined alternatively as $tetra'(v_0, v_1, v_2, v_3) = \{ r \in S^3 \mid v_i \cdot r \ge 0 \}$. Thus, $prob(v_0, v_1, v_2, v_3)$ is simply twice the probability that a random unit vector $r \in S^3$ falls into the polar tetrahedron $tetra'(v_0, v_1, v_2, v_3)$. This probability is proportional to the volume of the tetrahedron. As the volume of S^3 is $2\pi^2$ (see Berger [9], p. 261), we get

$$prob(v_0, v_1, v_2, v_3) = volume(tetra'(v_0, v_1, v_2, v_3))/\pi^2\,.$$

While computing *areas* of spherical *triangles* in S^2 is a relatively simple matter, Girard's formula (see [10, p. 278]) stating that the area of a spherical triangle with angles α, β and γ (on S^2) is $\alpha + \beta + \gamma - \pi$, computing volumes of spherical tetrahedra is a much more complicated matter. This subject was investigated in the previous century by Schläfli [32] and in the present one by Coxeter [13], Böhm and Hertel [11] and Hsiang [21]. Unfortunately, no closed-form formula for the volume is known. As the volume function is related to the dilogarithm function (see Kellerhals [25]), possibly no closed-form formula exists.

A spherical tetrahedron can be characterized by either the six angles $\theta_{ij} = \arccos(v_i \cdot v_j)$ between the unit vectors that correspond to its vertices—note that θ_{ij} is also the distance between v_i and v_j on the sphere—or by its six dihedral angles. The dihedral angle along an edge of the tetrahedron is the angle between the two "faces" that meet at the edge; see Hsiang [22] for a formal definition. Let $\mathrm{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})$ be the volume of a spherical tetrahedron with dihedral angles $\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}$. While the volume itself may be nonelementary, its partial derivatives have a surprisingly simple form (see Appendix A). We need the following relation between the lengths of the sides of a spherical tetrahedron and the dihedral angles of the corresponding polar tetrahedron.

Lemma 4.1 *If $\lambda_{01}, \ldots, \lambda_{23}$ are the dihedral angles and $\theta_{01}, \ldots, \theta_{23}$ are the side lengths of a spherical tetrahedron*

T, and $\lambda'_{01}, ..., \lambda'_{23}$ and $\theta'_{01}, ..., \theta'_{23}$ are the dihedral angles and side lengths of the spherical tetrahedron T' polar to T, then $\lambda'_{01} = \pi - \theta_{23}$ and $\theta'_{01} = \pi - \lambda_{23}$. (Other equalities such as $\lambda'_{13} = \pi - \theta_{02}$ follow by symmetry.)

This is a classical result in spherical geometry. For a proof, see Hsiang [22, eqn. (41)]. We now have

Theorem 4.2 $\alpha'_3 = 7/8$.

Conjecture 4.3 $\alpha_3 = 7/8$.

It is easy to see that $\alpha_3 \le \alpha'_3 \le \frac{7}{8}$. The upper bound on α'_3 is obtained by considering the case in which v_0, v_1, v_2 and v_3 are all perpendicular (this is possible as we are in R^4). It is easy to verify that in that case $relax(v_0, v_1, v_2, v_3) = 1$ and that $prob(v_0, v_1, v_2, v_3) = 1/8$.

Proof of Theorem 4.2: We have to show that for every v_0, $v_1, v_2, v_3 \in S^3$ with $relax(v_0, v_1, v_2, v_3) = 1$, $1 - prob(v_0, v_1, v_2, v_3) \ge \frac{7}{8}$, or equivalently $prob(v_0, v_1, v_2, v_3) \le \frac{1}{8}$. Note that $relax(v_0, v_1, v_2, v_3) = 1$ implies that $(v_0 + v_1) \cdot (v_2+v_3) \le 0$, $(v_0+v_2)\cdot(v_1+v_3) \le 0$, $(v_1+v_2)\cdot(v_0+v_3) \le 0$. Let $\theta'_{ij} = \arccos(v_i \cdot v_j)$. Let λ_{ij} be the dihedral angles of the *polar* spherical tetrahedron. Now

$$prob(v_0, v_1, v_2, v_3) = \mathrm{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})/\pi^2 .$$

Thus, using the relation between the θ'_{ij}'s (in the primal) and λ_{ij}'s (in the polar) given by Lemma 4.1, we infer that it is enough to prove the following:

Lemma 4.4 *Let $\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}$ be the six dihedral angles of a spherical tetrahedron. If*

$$\begin{aligned}
\cos\lambda_{02} + \cos\lambda_{13} + \cos\lambda_{03} + \cos\lambda_{12} &\ge 0 , \\
\cos\lambda_{01} + \cos\lambda_{23} + \cos\lambda_{03} + \cos\lambda_{12} &\ge 0 , \\
\cos\lambda_{01} + \cos\lambda_{23} + \cos\lambda_{02} + \cos\lambda_{13} &\ge 0 ,
\end{aligned}$$

then $\mathrm{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}) \le \frac{\pi^2}{8}$.

A sketch of the computer-assisted proof of Lemma 4.4 appears in Appendix B. □

Beginning of a possible proof of Conjecture 4.3: As we have proved Theorem 4.2, we may assume here that $relax(v_0, v_1, v_2, v_3) < 1$. By symmetry, we may assume w.l.o.g. that $relax(v_0, v_1, v_2, v_3) = \frac{4-(v_0+v_2)\cdot(v_1+v_3)}{4}$. Simple algebra and the fact that $relax(v_0, v_1, v_2, v_3) < 1$ imply that to prove Conjecture 4.3, it suffices to prove

Conjecture 4.5 *Let $\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}$ be the six dihedral angles of a spherical tetrahedron. If*

$$\begin{aligned}
\cos\lambda_{01} + \cos\lambda_{23} + \cos\lambda_{03} + \cos\lambda_{12} &\le 0 , \\
\cos\lambda_{03} + \cos\lambda_{12} - \cos\lambda_{02} - \cos\lambda_{13} &\le 0 , \\
\cos\lambda_{01} + \cos\lambda_{23} - \cos\lambda_{02} - \cos\lambda_{13} &\le 0 ,
\end{aligned}$$

then

$$\begin{aligned}
&\mathrm{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}) \\
&+ \tfrac{7\pi^2}{32} \cdot (\cos\lambda_{01} + \cos\lambda_{12} + \cos\lambda_{23} + \cos\lambda_{03}) \le \tfrac{\pi^2}{8} .
\end{aligned}$$

The evidence that we have in support of Conjecture 4.5 appears in Appendix C. □

5 Semidefinite Relaxations of Constraint Satisfaction Problems

Let $f : \{-1, 1\}^k \to \{0, 1\}$ be a Boolean function. For convenience we use different encodings of the truth values at the input and the output of f. At the input, we use -1 to represent FALSE and 1 to represent TRUE. At the output, we use 0 to represent FALSE and 1 to represent TRUE.

An instance of the problem MAX CSP(f) is a collection $\{ (f(y_{i1}, \ldots, y_{ik}) , w_i) \}_{i=1}^m$ of weighted constraints, where $y_{ij} \in \{0, 1, x_1, \ldots, x_n, -x_1, \ldots, -x_n\}$. The goal is to find an assignment of ± 1 values to the variables $x_1, \ldots, x_n$ that maximizes the total weight of the clauses that evaluate to 1. The MAX CSP(f) problem is therefore similar to the MAX SAT problem except that the constraints are now of the form $f(y_{i1}, \ldots, y_{ik})$ and not of the form $y_{i1} \vee \cdots \vee y_{ik}$. The problem MAX CSP($\mathcal{F}$), where $\mathcal{F}$ is a finite collection of Boolean functions, can also be defined.

There is no precise definition of the term "relaxation." Here we propose a definition of a class of semidefinite relaxations which we call "standard semidefinite relaxations."

Definition 5.1 (standard semidefinite relaxations)
A standard semidefinite relaxation *of an instance I of MAX CSP(f) is a semidefinite program that has the following properties:*

1. *The variables of the program are: a unit vector v_i corresponding to each variable x_i, and a scalar z_j corresponding to each clause, and a unit vector v_0 (representing FALSE).*
2. *The objective function to be maximized is $\sum_{j=1}^m w_j z_j$.*
3. *For every clause $f(y_{i1}, \ldots, y_{ik})$ of I, the program contains a set of linear inequalities involving inner products $v_i \cdot v_j$ of the vectors that correspond to the variables $y_{i1}, \ldots, y_{ik}$, and the scalar z that corresponds to the clause. All the constraints of the program are of this form.*
4. *For every $(x_1, \ldots, x_n) \in \{-1, 1\}^n$, assign v_i the vector $(x_i, 0, \ldots, 0)$, assign v_0 the vector $(-1, 0, \ldots, 0)$, and assign z_j the value 1 if the jth clause is satisfied by this assignment and the value 0, otherwise. Then this assignment is a feasible point of the program.*

Condition (4) insures that the program is indeed a relaxation and that the value of the program is at least the value of the instance I. Condition (3) says, in effect, that the program "considers" the Boolean constraints one at a time. (Note the similarity of this to the gadgets of Bellare *et al.* [7] and Trevisan *et al.* [34].) Almost all semidefinite relaxations of constraint satisfaction problems proposed to date are standard, or can be made standard with only minor modification. Exceptions are the relaxations of MAX CUT proposed by Feige and Goemans (See Section 5 of [15]), that impose constraints involving up to k vertices, for varying values of k. The definition of standard relaxations may be generalized appropriately.

Consider the Boolean functions $\mathrm{OR}_2(x_1, x_2) = x_1 \vee x_2$, $\mathrm{AND}_2(x_1, x_2) = x_1 \wedge x_2$, $\mathrm{XOR}_2(x_1, x_2) = x_1 \oplus x_2$, and $\mathrm{NAE}_3(x_1, x_2, x_3) = (x_1 \oplus x_2) \vee (x_1 \oplus x_3)$. With our input and output conventions,

$$\mathrm{OR}_2 = \tfrac{3+x_1+x_2-x_1x_2}{4}\ ,\ \mathrm{AND}_2 = \tfrac{1+x_1+x_2+x_1x_2}{4}$$

$$\mathrm{XOR}_2 = \tfrac{1-x_1x_2}{2}\ ,\ \mathrm{NAE}_3 = \tfrac{3-x_1x_2-x_1x_3-x_2x_3}{4}$$

All these functions can be represented as degree-2 polynomials over the real numbers. Obtaining (standard) semidefinite relaxations for these functions is therefore relatively straightforward. To each variable x_i, we attach a unit vector v_i. Every product $x_i x_j$ is now replaced by the inner product $v_i \cdot v_j$. To handle linear terms, we introduce another unit vector, v_0, that represents FALSE. A linear term x_i is now replaced by the inner product $-v_0 \cdot v_i$. By doing so, we obtain the semidefinite relaxations of MAX CUT and MAX 2SAT used by Goemans and Williamson [18]. Are these the best standard relaxations that we can obtain? We answer this question shortly.

To obtain a semidefinite relaxation of MAX 3SAT, we need a semidefinite relaxation of the function $\mathrm{OR}_3(x_1, x_2, x_3)$. Unfortunately, this function cannot be represented as a degree-2 polynomial and the simple approach described above cannot be used.

Let $f : \{-1,1\}^k \to \{0,1\}$ be any Boolean function. We now describe a way of obtaining the strongest standard semidefinite relaxations for instances of MAX CSP(f).

Given a vector $x = (x_1, \ldots, x_k) \in \{-1,1\}^k$, we let $\mathrm{prod}(x) = (x_0x_1, x_0x_2, \ldots, x_{k-1}x_k) \in \{-1,1\}^{\frac{k(k+1)}{2}}$, where $x_0 = -1$. Define

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

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

$$\mathrm{conv}(\{v_1, \ldots, v_\ell\}) = \{\sum_{i=1}^{\ell} \alpha_i v_i \mid \alpha_i \geq 0\ ,\ \sum_{i=1}^{\ell} \alpha_i = 1\}$$

Note that $polytope(f)$ is a polytope in $\mathbb{R}^{\frac{k(k+1)}{2}+1}$. We defined $polytope(f)$ by giving its 2^k vertices. Alternatively, $polytope(f)$, like any other polytope, can be defined as the intersection of a finite number of halfspaces. In other words, there exists an $m \times (k(k+1)/2+1)$ matrix A and a vector b such that $polytope(f) = \{x \in R^{\frac{k(k+1)}{2}+1} \mid Ax \leq b\}$. Let (A, b) be such a pair with m minimal. (If $polytope(f)$ is full-dimensional, we can take the rows of A and entries of b to correspond to the facets of the polytope.) We refer to the pair (A, b) as a set of *defining hyperplanes* of $polytope(f)$. Each defining hyperplane is an inequality of the form $\sum_{i<j} \alpha_{ij} x_{ij} + \beta z \leq \gamma$, where x_{ij} is a variable corresponding to the position occupied by the product $x_i x_j$, and z is a variable that corresponds to the last position in $(\mathrm{prod}(x), f(x))$.

As an example, the eight facets of $polytope(\mathrm{OR}_3)$ are given in Figure 2. The last four facets are exactly the four constraints we used in the relaxation of MAX 3SAT given in Figure 1. The first four facets give lower bounds, rather than upper bounds, on z and they can therefore be ignored. The facets that give lower bounds on z can be eliminated by considering the polytope

$$polytope'(\mathrm{OR}_3) = \left\{ (x_{01}, \ldots, x_{23}, z') \,\middle|\, \begin{array}{c} (x_{01}, \ldots, x_{23}, z) \in polytope(\mathrm{OR}_3) \\ z' \leq z \end{array} \right\}$$

whose 20 facets are given in Figure 3. The first 16 facets are just the "triangle inequalities" used in the MAX 2SAT algorithm Feige and Goemans [15]. They are in fact facets of the cut polytope (see [6]).

We now define canonical semidefinite relaxations for all instances of MAX CSP(f). We later show that they are the strongest standard semidefinite relaxations possible.

Definition 5.2 (Canonical semidefinite relaxations) *A* canonical semidefinite relaxation *of an instance I of MAX CSP(f) is a semidefinite program obtained in the following way.*

1. *The variables of the program are: a unit vector v_i corresponding to each variable x_i, a scalar z_j corresponding to each clause, and a unit vector v_0 representing FALSE.*
2. *The objective function to be maximized is $\sum_{j=1}^{m} w_j z_j$.*
3. *For every clause $f(y_1, \ldots, y_k)$ of I, and for every defining hyperplane $\sum_{i<j} \alpha_{ij} x_{ij} + \beta z \leq \gamma$ of $polytope(f)$, from a fixed family of defining hyperplanes, the program contains the inequality*

$$\sum_{i<j} \alpha_{ij} (u_i \cdot u_j) + \beta\, z \leq \gamma\ .$$

$$
\begin{array}{rrrrrrrcr}
-x_{01} & -x_{02} & -x_{12} & & & & -4z & \leq & -3 \\
-x_{01} & & & -x_{03} & -x_{13} & & -4z & \leq & -3 \\
& -x_{02} & & -x_{03} & & -x_{23} & -4z & \leq & -3 \\
& & -x_{12} & & -x_{13} & -x_{23} & -4z & \leq & -3 \\
& & & & & & +z & \leq & 1 \\
+x_{01} & +x_{02} & & & +x_{13} & +x_{23} & +4z & \leq & 4 \\
+x_{01} & & +x_{12} & +x_{03} & & +x_{23} & +4z & \leq & 4 \\
& +x_{02} & +x_{12} & +x_{03} & +x_{13} & & +4z & \leq & 4
\end{array}
$$

Figure 2. The facets of $polytope(\mathrm{OR}_3)$**.**

$$
\begin{array}{rrrrrrrcr}
-x_{01} & +x_{02} & +x_{12} & & & & & \leq & 1 \\
+x_{01} & -x_{02} & +x_{12} & & & & & \leq & 1 \\
+x_{01} & +x_{02} & -x_{12} & & & & & \leq & 1 \\
-x_{01} & -x_{02} & -x_{12} & & & & & \leq & 1 \\
-x_{01} & & & +x_{03} & +x_{13} & & & \leq & 1 \\
+x_{01} & & & -x_{03} & +x_{13} & & & \leq & 1 \\
+x_{01} & & & +x_{03} & -x_{13} & & & \leq & 1 \\
-x_{01} & & & -x_{03} & -x_{13} & & & \leq & 1 \\
& -x_{02} & & +x_{03} & & +x_{23} & & \leq & 1 \\
& +x_{02} & & -x_{03} & & +x_{23} & & \leq & 1 \\
& +x_{02} & & +x_{03} & & -x_{23} & & \leq & 1 \\
& -x_{02} & & -x_{03} & & -x_{23} & & \leq & 1 \\
& & -x_{12} & & +x_{13} & +x_{23} & & \leq & 1 \\
& & +x_{12} & & -x_{13} & +x_{23} & & \leq & 1 \\
& & +x_{12} & & +x_{13} & -x_{23} & & \leq & 1 \\
& & -x_{12} & & -x_{13} & -x_{23} & & \leq & 1 \\
& & & & & & +z & \leq & 1 \\
+x_{01} & +x_{02} & & & +x_{13} & +x_{23} & +4z & \leq & 4 \\
+x_{01} & & +x_{12} & +x_{03} & & +x_{23} & +4z & \leq & 4 \\
& +x_{02} & +x_{12} & +x_{03} & +x_{13} & & +4z & \leq & 4
\end{array}
$$

Figure 3. The facets of $polytope'(\mathrm{OR}_3)$**.**

> *Here $u_0 = v_0$, u_i is the vector that corresponds to the literal y_i (if $y_i = x_j$, then $u_i = v_j$, if $y_i = -x_j$, then $u_i = -v_j$, if $y_i = 0$, then $u_i = v_0$, and $u_i = -v_0$ if $y_i = 1$, and z is the scalar corresponding to the clause. All the constraints of the program are of this form.*

The simple proof of the following lemma is omitted.

Lemma 5.3 *A canonical semidefinite relaxation of an instance I of MAX CSP(f) is a standard semidefinite relaxation.*

The number of vertices of $polytope(f)$ is exponential in k, the number of variables of f. The number of facets, or defining hyperplanes, of $polytope(f)$ may be even larger. However, for every fixed constraint satisfaction problem MAX CSP(f), k is fixed, and therefore the size of a canonical semidefinite relaxation of an instance I is linear in the size of the instance.

We now show that canonical semidefinite relaxations are the strongest standard semidefinite relaxations possible. Given an instance I, we let $opt(I)$ be the value of an optimal solution of I. Given a semidefinite program P, we let $opt(P)$ be the value of an optimal solution of P.

Theorem 5.4 *Let I be an instance of MAX CSP(f). Let P be a canonical semidefinite relaxation of I and let Q be any standard semidefinite relaxation of I. Then, any feasible point of P is also a feasible point of Q. In particular, $opt(I) \leq opt(P) \leq opt(Q)$.*

Proof: Let $p = (v_0, \ldots, v_n, z_1, \ldots, z_m)$ be a feasible point of P. Suppose, for the sake of contradiction, that the point p is not a feasible point of Q, so p violates an inequality of Q. Assume w.l.o.g. that this inequality is one of the inequalities attached to the Boolean constraint $f(x_1, \ldots, x_k)$. Let $p' = (v_0, \ldots, v_k, z)$ be the restriction of p to the vectors and the scalar that appear in the inequalities attached to this constraint. Let $p'' = (x_{0,1}, \ldots, x_{k-1,k}, z) = (v_0 \cdot v_1, v_0 \cdot v_2, \ldots, v_{k-1} \cdot v_k, z)$. As p is a feasible point of P, we infer that $p'' \in polytope(f)$. Thus p'' is a convex combination of the vertices of $polytope(f)$. On the other hand, p'' violates at least one of the constraints of Q. Thus at least one of the vertices of $polytope(f)$ violates this constraint of Q and this contradicts condition 4 of the definition of standard semidefinite relaxations. $\square$

By computing $polytope(\mathrm{XOR}_2)$ and $polytope(\mathrm{OR}_2)$, we can infer that the semidefinite relaxation of MAX CUT given by Goemans and Williams [18] is a canonical semidefinite relaxation of MAX CUT. Their relaxation of MAX 2SAT, however, is not canonical, as it does not include the triangle inequalities. Feige and Goemans [15] include these inequalities in their relaxation and obtain a canonical relaxation. It is interesting to note that while the triangle inequalities help for MAX 2SAT, it seems that they are not required for getting an optimal 7/8 approximation algorithm for MAX 3SAT.

The facets of $polytope(\mathrm{MAJ}_3)$ can apparently be used to obtain a 2/3-approximation algorithm for MAX CSP(MAJ_3). For this and some other results, see [38].

6 Concluding Remarks

While we described a way of getting the strongest semidefinite relaxations—at least in some natural sense—for all constraint satisfaction problems of the form MAX CSP(f), we do not automatically get good approximation algorithms for all of them. Rounding the optimal solutions of the semidefinite programs using a random hyperplane does not work well for all problems.

7 Acknowledgment

We would like to thank Wu-Yi Hsiang and Michael Loss for many helpful discussions on spherical tetrahedra. We would also like to thank UC Berkeley and ICSI for their hospitality.

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 hardness of approximation problems. In *Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania*, pages 14–23, 1992.

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

[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. In *Proceedings of the 5th Scandinavian workshop on algorithm theory*, pages 100–111, 1996.

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

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

[8] 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.

[9] M. Berger. *Geometry I*. Springer-Verlag, 1987.

[10] M. Berger. *Geometry II*. Springer-Verlag, 1987.

[11] J. Böhm and H. Hertel. *Polyedergeometrie in n-dimensionalen räumen konstanter krümmung*. Birkhäuser, 1981.

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

[13] H. Coxeter. The functions of Schläfli and Lobatschefsky. *Quarterly Journal of of Mathematics (Oxford)* 6:13–29, 1935.

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

[15] U. Feige and M. 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.

[16] 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.

[17] M. Goemans and D. Williamson. New 3/4-approximation algorithms for the maximum satisfiability problem. *SIAM Journal on Discrete Mathematics* 7:656–666, 1994.

[18] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM* 42:1115–1145, 1995.

[19] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*. Springer Verlag, 1988.

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

[21] W. Hsiang. On infinitesimal symmetrization and volume formula for spherical or hyperbolic tetrahedrons. *Quarterly Journal of Mathematics (Oxford)* 39:463–468, 1988.

[22] W. Hsiang. On the development of quantitative geometry from Pythagoras to Grassmann. In D.-Z. Du and F. Hwang, editors, *Computing in Euclidean Geometry*, pages 1–21. World Scientific Publishing Co., 1992.

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

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

[25] R. Kellerhals. The dilogarithm and volumes of hyperpolic polytopes. In L. Lewin, editor, *Structural Properties of Polylogarithms*, Mathematical Surveys and Monographs, Volume 37, pages 301–336. American Mathematical Society, 1991.

[26] S. Khanna, M. Sudan, and L. Trevisan. Constraint satisfaction: the approximability of minimization problems. In *Proceedings of the 12th Annual IEEE Conference on Computational Complexity, Ulm, Germany*, pages 11–20, 1997. Full version available as E-CCC Technical Report number TR96-064.

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

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

[29] Y. Nesterov and A. Nemirovskii. *Interior Point Polynomial Methods in Convex Programming*. SIAM, 1994.

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

[31] G. Pataki. On cone-LP's and semidefinite programming: facial structure, basic solutions and the simplex method. Technical report, GSIA Carnegie Mellon University, Pittsburgh, PA, 1994.

[32] L. Schläfli. On the multiple integral $\int^n dx\, dy \cdots dz$, whose limits are $p_1 = a_1x + b_1y + \cdots + h_1z > 0$, $p_2 > 0, \ldots, p_n > 0$, and $x^2 + y^2 + \cdots + z^2 < 1$. *Quarterly Journal of Mathematics (Oxford)* 2:269–300, 1858. Continued in Vol. 3 (1860), pp. 54–68 and pp. 97-108.

[33] L. Trevisan. Approximating satisfiable satisfiability problems. In *Proceedings of the 5th European Symposium on Algorithms, Graz, Austria*, 1997.

[34] L. Trevisan, G. Sorkin, M. Sudan, and D. 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.

[35] P. Vaidya. A new algorithm for minimizing convex functions over convex sets (extended abstract). In *Proceedings of the 20th Annual IEEE Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina*, pages 338–343, 1989.

[36] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review* 38:49–95, 1996.

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

[38] U. Zwick. Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. Submitted for publication.

A Volumes of Spherical Tetrahedra

Let $v_0, v_1, v_2, v_3 \in S^3$ be the vertices of a nondegenerate spherical tetrahedron. Let $\theta_{ij} = \arccos(v_i \cdot v_j)$ be the angle between v_i and v_j, or, equivalently, the *length* of the edge ij, as measured on the sphere. The *dihedral* angle λ_{ij} is the angle between the two "faces" that meet at the edge ij.

Definition A.1 *(high-dimensional inner product; see Hsiang [22, eqn. (1)]) Let $a_1, \ldots, a_k$ and $b_1, \ldots, b_k$ be two sequences of vectors all of the same length. Define $\langle\, a_1 \wedge \cdots \wedge a_k\, ,\, b_1 \wedge \cdots \wedge b_k\, \rangle = \det(\{a_i \cdot b_j\})$, and $|a_1 \wedge \cdots \wedge a_k| = \langle\, a_1 \wedge \cdots \wedge a_k\, ,\, a_1 \wedge \cdots \wedge a_k\, \rangle^{1/2}$.*

Lemma A.2 *Let $v_0, v_1, v_2, v_3 \in S^3$ be the vertices of a nondegenerate spherical tetrahedron with dihedral angles $\lambda_{01}, \ldots, \lambda_{23}$. Let (i, j, k, ℓ) be a permutation of $(0, 1, 2, 3)$. Then*

$$\cos\lambda_{ij} = \frac{\langle\, v_i \wedge v_j \wedge v_k\, ,\, v_i \wedge v_j \wedge v_\ell\, \rangle}{|v_i \wedge v_j \wedge v_k|\; |v_i \wedge v_j \wedge v_\ell|} \,.$$

Although the function $\mathrm{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23})$, giving the volume of a spherical tetrahedron as a function of its six dihedral angles, is complicated, we have

Theorem A.3 (Schläfli (1858) [32]) $$\frac{\partial \mathrm{Vol}}{\partial \lambda_{ij}} = \frac{\theta_{ij}}{2}\,.$$

B Proof of Lemma 4.4

Here we provide a computer-assisted proof of Lemma 4.4. We have to prove that $\mathrm{Vol}(\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}) \le \pi^2/8$ subject to

$$\begin{array}{rcl}
\cos\lambda_{01} + \cos\lambda_{23} + \cos\lambda_{03} + \cos\lambda_{12} & \geq & 0, \\
\cos\lambda_{01} + \cos\lambda_{23} + \cos\lambda_{02} + \cos\lambda_{13} & \geq & 0, \\
\cos\lambda_{02} + \cos\lambda_{13} + \cos\lambda_{03} + \cos\lambda_{12} & \geq & 0,
\end{array}$$

and subject to the condition that $\lambda_{01}, \ldots, \lambda_{23}$ is a valid sequence of dihedral angles. It is easy to verify that $\lambda_{01}, \ldots, \lambda_{23}$ is a valid sequence if and only if the following matrix is positive semidefinite:

$$\begin{bmatrix}
1 & -\cos\lambda_{23} & -\cos\lambda_{13} & -\cos\lambda_{12} \\
-\cos\lambda_{23} & 1 & -\cos\lambda_{03} & -\cos\lambda_{02} \\
-\cos\lambda_{13} & -\cos\lambda_{03} & 1 & -\cos\lambda_{01} \\
-\cos\lambda_{12} & -\cos\lambda_{02} & -\cos\lambda_{01} & 1
\end{bmatrix}.$$

If the minimal eigenvalue of this 4×4 matrix is 0, then the matrix is singular, and the vectors v_0, v_1, v_2 and v_3 are linearly dependent. They may assumed therefore to lie in R^3. Instead of computing volumes in S^3, we then have to compute areas in S^2. In this case, which is much easier than the general case, it is not difficult to show that $\mathrm{Vol} \le (1 - \alpha_1)\pi^2$, where $\alpha_1 \simeq 0.87856$ is the performance ratio of the Goemans-Williamson MAX CUT algorithm. We will henceforth assume that the 4×4 matrix is positive-definite. By Theorem A.3, the volume is an increasing function of the dihedral angles. Simple perturbation arguments allow us, w.l.o.g., to reduce to the problem of proving that $\mathrm{Vol}(\lambda_{01}, ..., \lambda_{23}) \le \pi^2/8$ if

$$\begin{array}{rcl}
\cos\lambda_{01} + \cos\lambda_{23} + \cos\lambda_{03} + \cos\lambda_{12} & = & 0\,, \\
\cos\lambda_{01} + \cos\lambda_{23} + \cos\lambda_{02} + \cos\lambda_{13} & = & 0\,, \\
\cos\lambda_{02} + \cos\lambda_{13} + \cos\lambda_{03} + \cos\lambda_{12} & \geq & 0\,.
\end{array}$$

Symmetry arguments allow us to assume w.l.o.g. that $\lambda_{02} \le \lambda_{12} \le \lambda_{03} \le \lambda_{13}$.

We define the following sequence of three points:

$$\begin{array}{rcl}
\Lambda_0 & = & (\lambda_{01}, \lambda_{02}, \lambda_{12}, \lambda_{03}, \lambda_{13}, \lambda_{23}), \\
\Lambda_1 & = & (\lambda_{01}, \lambda_{12}, \lambda_{12}, \lambda_{03}, \lambda_{03}, \lambda_{23}), \\
\Lambda_2 & = & (\lambda_{01}, \lambda, \lambda, \lambda, \lambda, \lambda_{23}),
\end{array}$$

where $\cos\lambda = (\cos\lambda_{02} + \cos\lambda_{13})/2 = (\cos\lambda_{12} + \cos\lambda_{03})/2 = -(\cos\lambda_{01} + \cos\lambda_{23})/2 \ge 0$. One can show that Λ_1 and Λ_2 are valid sequences of dihedral angles and that they both satisfy the hypotheses of Lemma 4.4.

We next show that $\mathrm{Vol}(\Lambda_0) \leq \mathrm{Vol}(\Lambda_1) \leq \mathrm{Vol}(\Lambda_2)$ by defining

$$f_{\Lambda',\Lambda''}(t) = \mathrm{Vol}(\arccos(\cos\Lambda' + t(\cos\Lambda'' - \cos\Lambda')))$$

for two feasible points Λ' and Λ'', by showing that for $f_i(t) = f_{\Lambda_{i-1},\Lambda_i}(t)$, $f_i'(t) \geq 0$ for $0 \leq t \leq 1$ and $i = 1, 2$. In fact, it is enough to prove that $f_i'(0) \geq 0$ for $i = 1, 2$, as Λ_{i-1} is just like any other point on the path from Λ_{i-1} to Λ_i. (This part of the proof does not rely on a computer.) We calculate the derivative of $f_i(t)$ explicitly using Theorem A.3 and Lemma A.2. Some calculus and lots of ugly computations eventually lead us to $f_i'(0) \geq 0$ for $i = 1, 2$. This completes the proof that $\mathrm{Vol}(\Lambda_0) \leq \mathrm{Vol}(\Lambda_2)$.

Now let $\mathrm{Vol}_2(\lambda_{01}, \lambda_{23}) = \mathrm{Vol}(\lambda_{01}, \lambda, \lambda, \lambda, \lambda, \lambda_{23})$ where $\lambda = \arccos(-\frac{\cos\lambda_{01}+\cos\lambda_{23}}{2})$. It is easy to see that if $\cos\lambda_{01} + \cos\lambda_{23} \leq 0$ then $(\lambda_{01}, \lambda, \lambda, \lambda, \lambda, \lambda_{23})$ is a valid sequence of dihedral angles. It is hence enough to prove

Lemma B.1 *For every $\lambda_{01}, \lambda_{23}$ such that $\cos\lambda_{01} + \cos\lambda_{23} \leq 0$, $\mathrm{Vol}_2(\lambda_{01}, \lambda_{23}) \leq \pi^2/8$.*

Proof: Let $\lambda^* = \arccos(1 - 2\cos\frac{\pi}{8}) \simeq 2.58254$. We break the proof into the following three cases:

Case 1 $\cos\lambda_{01} + \cos\lambda_{23} = 0$. This one is easy and the proof is omitted.

Case 2 $\cos\lambda_{01} + \cos\lambda_{23} < 0$ and $\lambda_{01}, \lambda_{23} \leq \lambda^*$. The hard case. Sketch below.

Case 3 $\lambda_{01} \geq \lambda^*$ or $\lambda_{23} \geq \lambda^*$. Easy, based on the fact that $\mathrm{Vol}_2(\lambda_{01}, \lambda_{23})/\pi^2$ is just $prob(v_0, v_1, v_2, v_3)$, the probability that v_0, v_1, v_2 and v_3 lie on the same side of a random hyperplane, and the probability that v_0, v_1, v_2, v_3 are not separated is at most the probability that v_1, v_2, v_3 are not separated. We then assume that $\lambda_{01} \geq \lambda^*$ or $\lambda_{23} \geq \lambda^*$. Let us assume, w.l.o.g., that $\lambda_{01} \geq \lambda^*$. It turns out to be enough to prove that if $\lambda_{01} \geq \lambda^*$ and $\cos\lambda_{01} + \cos\lambda_{23} \leq 0$, then $2\lambda + \lambda_{23} \leq 5\pi/4$, which is proved by simple calculus.

We return to Case 2. Here it is sufficient to prove that $\mathrm{Vol}_2(\lambda_{01}, \lambda_{23})$ has no critical point with $\lambda_{01}, \lambda_{23} \leq \lambda^*$. It is enough to prove

Claim B.2 *For every $\lambda_{23} \leq \lambda_{01} \leq \lambda^*$ such that $\cos\lambda_{01} + \cos\lambda_{23} \leq 0$, $\frac{\partial \mathrm{Vol}_2}{\partial \lambda_{01}} = \frac{1}{2}\theta_{01} - \frac{\sin\lambda_{01}}{\sin\lambda}\theta < 0$.*

Note that $\lambda_{01} = \lambda_{12} = \lambda_{03} = \lambda_{13}$ implies that $\theta_{01} = \theta_{12} = \theta_{03} = \theta_{13}$. In the statement of the claim, θ refers to this common value.

To prove Claim B.2, we partition the feasible region into small squares, explicitly bound the derivative and, using Mathematica with 50 digits of precision, show that the derivative is bounded above by -0.11. □

C Evidence in Support of Conjecture 4.5

A tuple $(\lambda_{01}, \ldots, \lambda_{23})$ satisfying the hypotheses of Lemma 4.4 is said to belong to *case a*. A tuple satisfying the hypotheses of Conjecture 4.5 is said to belong to *case b*. The evidence for Conjecture 4.5 is a sequence of computer runs testing whether $\frac{1-prob(v_0,v_1,v_2,v_3)}{relax(v_0,v_1,v_2,v_3)} \geq \frac{7}{8}$. The conclusion of Conjecture 4.5 is equivalent to the statement $\frac{1-\mathrm{Vol}(\lambda_{01},\ldots,\lambda_{23})/\pi^2}{1+(\cos\lambda_{01}+\cos\lambda_{12}+\cos\lambda_{23}+\cos\lambda_{03})/4} \geq \frac{7}{8}$.

1. A systematic search over the space of case-b possibilities with each of five variables running from 0 to π, the sixth determined by the other five, with a step size of $\frac{\pi}{100}$, yielded no ratio less that $7/8$. (A simple perturbation argument shows that w.l.o.g. the minimum value of the ratio is attained at a point with $\cos\lambda_{03}+\cos\lambda_{12}-\cos\lambda_{02} - \cos\lambda_{13} = 0$.) The only point in which $7/8$ was attained was $(\frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2})$.
2. A systematic search of the space of 6-tuples $(\lambda_{01}, \ldots, \lambda_{23})$ of both case a and case b using Matlab, with each λ_{ij} running from 0 to π in steps of $\frac{\pi}{56}$, found no ratio less than $7/8$ and the only point in which a ratio of $7/8$ was attained was $(\frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2})$.
3. A systematic search using Mathematica over all 6-tuples and with a step size of $\frac{\pi}{32}$ found no counterexample.

Both the Matlab and Mathematica runs pruned the search space by considering only tuples $(\lambda_{01}, \ldots, \lambda_{23})$ satisfying the triangle inequality constraints (the first 16 in Figure 2).

Each of the three runs numerically integrated Hsiang's formula for the volume of a spherical tetrahedron. While Mathematica and Matlab have built-in numerical integration, the first run, written in C, used 20-point Gaussian quadrature.

As the first systematic search suggests that the worst ratio is obtained only near the point $(\frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2})$, we performed another systematic search in the neighborhood of this point. We enumerated again on five angles, ranging from $\frac{\pi}{2} - 0.1$ to $\frac{\pi}{2} + 0.1$ in steps of $1/300$. Again, no ratio less than 7/8 was found.

We also tried to find the minimal ratio numerically using Matlab's constrained minimization function `constr`. No counterexample was found.