EVERY 2-CSP ALLOWS NONTRIVIAL APPROXIMATION

Johan Håstad

Abstract. We use semidefinite programming to prove that any constraint satisfaction problem in two variables over any domain allows an efficient approximation algorithm that does better than picking a random assignment. Specifically we consider the case when each variable can take values in $[d]$ and that each constraint rejects t out of the d^2 possible input pairs. Then, for some universal constant c, we can, in probabilistic polynomial time, find an assignment whose objective value is, in expectation, within a factor $1 - \frac{t}{d^2} + \frac{ct}{d^4 \log d}$ of optimal, improving on the trivial bound of $1 - \frac{t}{d^2}$.

Keywords. Semi-Definite Programming, Constraint Satisfaction, Approximation Algorithms

Subject classification. 68W25, 68Q25

1. Introduction

As witnessed by the book of Garey & Johnson. (1979), already in the late 1970's most optimization problem were classified as either being NP-hard or solvable in polynomial time. Many interesting and important problems are NP-hard and given that we cannot solve them optimally and efficiently we turn to heuristics and/or approximation algorithms.

For a maximization problem, we say that we have a C-approximation algorithm if the algorithm always, or in expectation over its own internal random choices, returns a solution whose value is at least C times the value of the optimal solution. Determining, for central optimization problems, the best approximation ratio achievable in polynomial time seemed at first a daunting task but progress has been relatively rapid. For many problems it is the case that early approximation algorithms can, by the use of inapproximability techniques originating from the PCP-theorem of Arora *et al.* (1998), be proved to be essentially optimal, see for instance Feige (1998); Feige & Kilian (1998); Håstad (1999, 2001); Khot (2001).

One class of problems containing many interesting and widely studied problems is the set of constraint satisfaction problems, abbreviated as CSPs. We have a set of n variables and we are given a set of constraints each related to only a constant number of the variables and the goal is to find an assignment that satisfies all constraints, or, more generally, the maximal number of constraints. The most basic CSP is 3-Sat, the question of given a Boolean formula in conjunctive normal form with three literals in each clause to find an assignment that satisfies all the clauses. This problem was on the original list of NP-complete problems given by Cook (1971) and has remained a favorite problem to use in reductions.

Let us consider Max-3-Sat, the optimization version of 3-Sat, where we try to satisfy the maximum number of clauses, and let us assume that each clause contains exactly 3 literals. A random assignment satisfies each clause with probability 7/8 and hence just picking a random assignment gives an 7/8-approximation algorithm. One could have thought that there should exist more sophisticated techniques to approximate this problem, but, maybe surprisingly, as proved by Håstad (2001) it is the case that for any $\epsilon > 0$ it is NP-hard to approximate Max-3-Sat within $7/8 + \epsilon$. Furthermore, there are also a number of other CSPs such that the random assignment algorithm is essentially the best we can do and some some examples can be found in Håstad (2001); Samorodnitsky & Trevisan (2000).

At about the same time as we have been able to prove better and better lower bounds on the approximability of some NP-hard optimization problems, progress has also been made on the positive side. An often used tool for designing efficient approximation algorithms has been semi-definite programming introduced in this context by Goemans & Williamson (1995) in the celebrated paper which obtains the $\approx .878$-approximation algorithm for Max-cut as well as strong bounds for the approximability of Max-2-Sat and directed Max-cut. The ratio obtained for Max-cut is still the largest known and recently Khot *et al.* (2007) have given support for the possibility that it might be the largest ratio achievable in polynomial time.

CSPs have been classified in various ways. The classic result by Schaefer (1978) characterized exactly for which Boolean CSPs satisfiability can be decided in polynomial time. Creignou (1995) (see also Khanna *et al.* (2000) and the book Creignou *et al.* (2001)) extended this to the Max-CSP-problems by classifying the optimization problems as either lying in P or being Max-SNP-hard. In the latter case, as a consequence of the PCP-theorem, for each problem there is some constant $c < 1$ such that it is NP-hard to approximate the problem within c. As a curiosity one might note that there are CSPs where the satisfiability problem is easy while the optimization problem is hard. A prime example would be sets of linear equations modulo 2. If all equations can be simultaneously satisfied then it is easy to find a solution by Gaussian elimination, but once the system is inconsistent it is hard to find the optimal solution, and in general one cannot even do much better than finding a random solution, see Håstad (2001).

In either of the two cases mentioned above, the class of problems that is solvable in polynomial time is highly limited. We believe that a more interesting classification of CSPs is given by *approximation resistance*. We say that a predicate is approximation resistant if the trivial algorithm that picks a random assignment is essentially the best possible polynomial time algorithm. We believe that this is a fundamental property as approximation resistance is a very strong indication that nothing useful can be said about this optimization problem in polynomial time. Problems that are NP-hard but not approximation resistant can at least be said to be slightly tractable.

By the results mentioned above we know that Max-3-Sat is approximation resistant while Max-Cut is not. Although our information is far from complete we do have many partial results. We know, by Goemans & Williamson (1995), that we have no approximation resistant constraints over Boolean variables that are binary, i.e., which depends on exactly two Boolean variables. If we widen the constraint to depend on three variables a complete classification was given by Zwick (1998): a constraint is approximation resistant if and only if it is implied by a parity constraint, i.e. if all rejected inputs have the same parity. The classification of constraints on four variables does not seem to follow such simple rules, but recently a systematic study has begun by Hast (2005) and over three quarters of the predicates are already proved either to be approximation resistant or to allow non-trivial approximation. Also, many predicates that depend on more variables have been classified in the same paper.

The approximation algorithms for binary Boolean predicates of Goemans & Williamson (1995) build on semi-definite programming and as the technique is very general one should expect it to extend to other situations. If we keep the width of each constraint at two but increase the size of the domain, other algorithms that rely on semi-definite programming have been successfully designed.

Andersson *et al.* (2001) showed that if we allow only two variables in each equation then, for the problem of linear systems of equations mod m, it is possible to construct an efficient algorithm giving a non-trivial approximation ratio. Engebretsen & Guruswami (2004) extended the result of Andersson *et al.* (2001) to establish non-trivial approximability for any binary constraint over any domain with the restriction that for each value of one variable we have the same number of values of the other variable that fulfill the constraint. Engebretsen and Guruswami also conjectured that their result would extend to the more general situation and in fact that any binary predicate over any domain does allow non-trivial approximability in polynomial time. We prove this conjecture and indeed show that semi-definite programming is, in this sense, universal for binary constraints.

Our approach is based on the approach of Engebretsen & Guruswami (2004) and we use essentially the same formulation of the semi-definite program and also the main procedure to obtain a solution to the CSP is the same. The novelty that enables us to obtain the new result is simple but still powerful: When rounding the solution for a semi-definite program to a solution to the underlying combinatorial problem one usually relies on a completely local analysis (and this is the case also in the paper Engebretsen & Guruswami (2004)). In a local analysis one compares the probability that a constraint is satisfied with the contribution to the objective function of the semi-definite program. We go a small step beyond this by looking at the linear terms in the objective function separately, and analyzing them *globally*. If the linear terms give a large contribution to the objective function, then it is simple to find a solution that is better than a random solution. If the linear terms are small then in fact they can, essentially, be discarded in the local analysis and this enables us to get the result.

The approach of looking at the linear terms in a global way seems to be useful and it is one main technique used, in a slightly different form, by Hast (2005) to prove that many Boolean predicates on 4 or more variables are not approximation resistant.

An outline of this paper is as follows. We start in Section 2 by giving some preliminaries. In Section 3 we give our main result. The approximation ratio obtained, although non-trivial, is still not that far from the trivial ratio and thus the approximation guarantee gives something better than a random assignment only for almost satisfiable instances. Our algorithm can be tuned to have much stronger properties and in Section 4 we prove that whenever the optimal solution is significantly better than a random solution we can set the parameters to find an assignment that does significantly better than random. We end by some final remarks in Section 5.

This is the full version of the results described in the conference paper given by Håstad (2005).

2. Preliminaries

We have n variables $(x_i)_{i=1}^n$ each taking values in $[d] = \{1, 2, \ldots d\}$ for some integer $d \geq 2$.

We have a constraint satisfaction problem given by m binary constraints $(C_i)_{i=1}^m$. One could ask for the constraints to be of the same "kind" , e.g., to be a linear equation, but as this is not needed for our algorithm, let us formulate the problem in as much generality as possible.

The i'th constraint is over the variables $x_{a_1^i}$ and $x_{a_2^i}$ and the constraint is satisfied iff this pair of variables does not take one of t_i specified values given as $(b_1^{i,j}, b_2^{i,j})_{j=1}^{t_i}$. To write our objective function in a convenient form let us introduce the indicator variable I_i^j which takes the value 1 if x_i takes the value j and is 0 otherwise. With this notation the number of satisfied constraints is exactly

$$m - \sum_{i=1}^{m} \sum_{j=1}^{t_i} I_{a_1^i}^{b_1^{i,j}} I_{a_2^i}^{b_2^{i,j}}. \tag{2.1}$$

To eliminate one index from this rather cumbersome expression let us change notation slightly. The total number of terms in the sum is $\sum_{i=1}^m t_i$ and let us write this number as tm. This is a natural notation as in many applications each constraint rejects equally many pairs and then $t = t_i$ for $1 \leq i \leq m$, but in general t is a rational number in the interval $[1, d^2 - 1]$.

It is not important for us whether the same pair of variables appear together in many constraints and hence by a redefinition of the a and b-variables it is possible to write (2.1) in the form

$$m - \sum_{k=1}^{tm} I_{a_1^k}^{b_1^k} I_{a_2^k}^{b_2^k} \tag{2.2}$$

and this is the formulation that we use. Note that from this formulation it is clear that if we can give a non-trivial approximation ratio for the case $t_i = 1$ for $1 \leq i \leq m$ then we can get a non-trivial approximation ratio for any other case. This fact has been observed many times and is stated explicitly by Engebretsen & Guruswami (2004).

For an instance φ of this problem let $OPT(\varphi)$ denote the number of constraints satisfied by the optimal assignment. Furthermore for an assignment α let $Val(\varphi, \alpha)$ be the number of constraints of φ satisfied by α. We have the following basic definition.

Definition 2.3. *A probabilistic algorithm A is a C-approximation algorithm if for any φ, we have*

$$E[Val(\varphi, A(\varphi))] \geq C \cdot OPT(\varphi)$$

We remark that the expectation is only over the internal random choices of A. The simplistic A just giving random independent value to the variables satisfy

$$E[Val(\varphi, A(\varphi))] = m - \frac{tm}{d^2} \geq (1 - \frac{t}{d^2}) \cdot OPT(\varphi),$$

and our goal is to design an algorithm with a better approximation ratio.

2.1. Semi-definite programming. We need very little from semi-definite programming and refer to the paper by Goemans & Williamson (1995) for more information. In a semi-definite program we have, for some integer s, variables $(Y_{i,j})_{i,j=1}^{s}$ to be thought of as an $s \times s$ matrix. Apart from linear conditions on these variables we have the constraint that the matrix Y is symmetric and positive semi-definite. Under these constraints we can, by the ellipsoid algorithm of Grötschel *et al.* (1981) or by the result of Alizadeh (1995), find the optimum of any linear function in the Y-variables to any desired accuracy.

For notational convenience we assume that we actually find the exact optimum. This "cheating" only results in an arbitrarily small factor gain in the approximation ratios obtained, and can be absorbed by changing the non-explicit constant c in each of Theorem 3.1 and Theorem 4.1 below.

As Y is symmetric and positive semi-definite there are vectors $(v_i)_{i=1}^{s}$ such that the elements of Y are the pairwise inner products of these vectors, i.e. $Y_{i,j} = (v_i, v_j)$. We use this point of view and formulate our semi-definite program in terms of the vectors v_i and their inner products.

3. The main result

We are now ready for our main result.

Theorem 3.1. *There is an absolute constant $c > 0$ such that for any d the following is true: There is a probabilistic polynomial time algorithm A such that for an instance φ of the constraint satisfaction problem over $[d]$ formalized as (2.2)*

$$E[Val(\varphi, A(\varphi))] \geq (1 - \frac{t}{d^2} + \frac{ct}{d^4 \log d}) \cdot OPT(\varphi).$$

Remark: The exact quantitative statement of this theorem improves over the results of Engebretsen & Guruswami (2004) for their class of problems. That paper obtained results when $t = d$ and derived an approximation ratio of $1 - \frac{1}{d} + \Omega(d^{-4})$ while we get $1 - \frac{1}{d} + \Omega(d^{-3}(\log d)^{-1})$. This minor improvement is due to a slightly more streamlined analysis rather than any real difference in the algorithms when applied in this special case.

We now turn to the proof of the theorem.

Proof. We model each variable x_i as a set of d vectors $(v_i^j)_{j=1}^d$. We also have a special vector v_0. Consider the following semi-definite program.

$$(v_0, v_0) = 1 \tag{3.2}$$

$$(v_0, v_i^j) \geq 0, \quad 1 \leq i \leq n, 1 \leq j \leq d \tag{3.3}$$

$$\sum_{j=1}^{d} (v_0, v_i^j) = 1, \quad 1 \leq i \leq n \tag{3.4}$$

$$\sum_{j=1}^{d} (v_i^j, v_i^j) \leq 1, \quad 1 \leq i \leq n \tag{3.5}$$

$$\sum_{j_1, j_2} (v_{i_1}^{j_1}, v_{i_2}^{j_2}) = 1, \quad 1 \leq i_1, i_2 \leq n \tag{3.6}$$

where the objective function is to maximize

$$m - \sum_{k=1}^{tm} (v_{a_1^k}^{b_1^k}, v_{a_2^k}^{b_2^k}). \tag{3.7}$$

Our semi-definite program is very close to the corresponding program of Engebretsen & Guruswami (2004). One slight difference is that we have an inequality in (3.5). This little detail is however of no real importance and is only there to allow us to more easily construct a feasible solution in the proof of Lemma 3.13 below.

Let us start with an observation.

Lemma 3.8. *The optimal value of the semi-definite program defined by (3.2) to (3.7) is at least $OPT(\varphi)$.*

Proof. Take any assignment α to the variables of φ. Set $v_i^j = v_0$ if $\alpha_i = j$ and $v_i^j = 0$ otherwise. It is a feasible solution to the semi-definite program and the value of (3.7) is $Val(\varphi, \alpha)$. The optimum over a wider class of possible vectors can only make the optimum larger. □

We turn to the question of given a good solution to the semidefinite program how to construct a good solution to the constraint satisfaction problem.

In the semidefinite program it is not difficult to see that for any feasible solution

$$\sum_{j=1}^{d} v_i^j = v_0 \tag{3.9}$$

for any i. Indeed, by (3.6) with $i_1 = i_2 = i$ we know that $\sum_{j=1}^{d} v_i^j$ is a unit vector and by (3.2) and (3.4) it must equal v_0. Conversely, note that (3.9) clearly implies (3.6) and hence it is sufficient to check this condition.

Our algorithm starts by finding an (almost) optimal solution to the semidefinite program and suppose the solution is given by

$$v_i^j = (\alpha_i^j + \frac{1}{d})v_0 + w_i^j$$

where

$$\sum_{j=1}^{d} \alpha_i^j = 0 \tag{3.10}$$

and $(v_0, w_i^j) = 0$, and furthermore, by (3.9), it follows that

$$\sum_{j=1}^{d} w_i^j = 0.$$

Suppose the obtained maximum is $m(1-\delta)$. If $\delta \geq \frac{t}{4d^2}$ then we proceed by picking a random solution, ignoring the solution to the semi-definite program, obtaining an assignment with expected objective value

$$(1 - \frac{t}{d^2})m \geq \frac{(1 - \frac{t}{d^2})}{(1 - \frac{t}{4d^2})} \cdot OPT(\varphi). \tag{3.11}$$

Using

$$\frac{1-x}{1-\frac{x}{4}} \geq 1 - \frac{3x}{4} - \frac{x^2}{4}$$

which is valid for any $0 \leq x \leq 1$ we see that the bound (3.11) is at least

$$(1 - \frac{3t}{4d^2} - \frac{t^2}{4d^4}) \cdot OPT(\varphi) \geq (1 - \frac{t}{d^2} + \frac{t}{4d^4}) \cdot OPT(\varphi)$$

establishing the theorem in this case. In view of this we may from now on assume that $\delta \le \frac{t}{4d^2}$.

We proceed to find a good solution to the variables of φ.

In the sum (2.2) we say that b_j^k is a *non-desired* value for $x_{a_j^k}$, and we let f_i^j be the number of times that j is a non-desired value for x_i and consider the quantity

$$\sum_{i,j} f_i^j \alpha_i^j. \tag{3.12}$$

Note first that if the non-desired values for x_i are uniformly distributed, i.e. if f_i^j only depends on i, then in fact (3.12) equals 0 due to the fact that $\sum_j \alpha_i^j = 0$. It is not difficult to see that if the non-desired values are not uniformly distributed then one can find values of x_i that satisfy more than a fraction $(1 - \frac{t}{d^2})$ of the constraints and the larger distance from uniformity the better assignment is possible. This is essentially the basis for the first approximation algorithm. Let us first make a minor observation.

Lemma 3.13. *We have*

$$\sum_{k=1}^{tm} \alpha_{a_1^k}^{b_1^k} \alpha_{a_2^k}^{b_2^k} \le -\frac{1}{d} \sum_{i,j} f_i^j \alpha_i^j.$$

Proof. The sum, $\sum_{k=1}^{tm} (v_{a_1^k}^{b_1^k}, v_{a_2^k}^{b_2^k})$, in (3.7) equals

$$\frac{tm}{d^2} + \frac{1}{d} \sum f_i^j \alpha_i^j + \sum_{k=1}^{tm} \alpha_{a_1^k}^{b_1^k} \alpha_{a_2^k}^{b_2^k} + \sum_{k=1}^{tm} (w_{a_1^k}^{b_1^k}, w_{a_2^k}^{b_2^k}). \tag{3.14}$$

Now consider the set of vectors $\tilde{v}_i^j$ defined by

$$\tilde{v}_i^j = \frac{1}{d} v_0 + w_i^j.$$

We claim that these are feasible vectors for the semi-definite program. Note first that conditions (3.2) and (3.3) are obvious. Furthermore, since

$$\sum_{j=1}^{d} \tilde{v}_i^j = \sum_{j=1}^{d} v_i^j = v_0$$

conditions (3.4) and (3.9) (which as noted implies (3.6)) are also true and the condition that remains to be checked is (3.5). Because of (3.10) we have

$$1 \geq \sum_{j=1}^{d} (v_i^j, v_i^j) \quad = \tag{3.15}$$

$$\sum_{j=1}^{d} ((\alpha_i^j)^2 + \frac{1}{d^2} + (w_i^j, w_i^j)) \quad = \tag{3.16}$$

$$\sum_{j=1}^{d} ((\alpha_i^j)^2 + (\tilde{v}_i^j, \tilde{v}_i^j)) \quad \geq \tag{3.17}$$

$$\sum_{j=1}^{d} (\tilde{v}_i^j, \tilde{v}_i^j) \tag{3.18}$$

and thus also (3.5) holds for the vectors $\tilde{v}_i^j$. The sum in the objective function (3.7) is, for this set of vectors,

$$\frac{tm}{d^2} + \sum_{k=1}^{tm} (w_{a_1^k}^{b_1^k}, w_{a_2^k}^{b_2^k}) \tag{3.19}$$

and as the v_i^j give an optimal solution this value must be larger than (3.14) and we conclude that the lemma holds. □

We now turn to finding a good assignment and we have two cases depending on whether

$$\sum_{i,j} f_i^j \alpha_i^j \leq -\frac{tm}{2d}. \tag{3.20}$$

Let us denote by Case 1 when this inequality holds and by Case 2 when it does not. In Case 1 we have the the following strategy for obtaining a good solution.

Set x_i to the value j with probability $p_i^j = \frac{1}{d} + \frac{1}{2}\alpha_i^j$.

By (3.10) if follows that $\sum_j p_i^j = 1$ for any i and by (3.3) we have $p_i^j \geq 0$ for any i and j and hence we have a correct set of probabilities.

The expected number of falsified constraints under this strategy is

$$\begin{aligned}
\sum_k \left(\frac{1}{d}+\frac{1}{2}\alpha_{a_1^k}^{b_1^k}\right)\left(\frac{1}{d}+\frac{1}{2}\alpha_{a_2^k}^{b_2^k}\right) &= \\
\frac{tm}{d^2}+\frac{1}{2d}\sum_{i,j} f_i^j\alpha_i^j+\frac{1}{4}\sum_k \alpha_{a_1^k}^{b_1^k}\alpha_{a_2^k}^{b_2^k} &\le \\
\frac{tm}{d^2}+\frac{1}{4d}\sum_{i,j} f_i^j\alpha_i^j &\le \\
\frac{tm}{d^2}-\frac{tm}{8d^2}, & \qquad (3.21)
\end{aligned}$$

where the first inequality follows from Lemma 3.13 and the second inequality is a consequence of the assumption (3.20). We conclude that under the assumption (3.20) we have an

$$(1+\frac{t}{8d^2}-\frac{t}{d^2})$$

approximation ratio, independently of δ. This establishes the theorem in Case 1 and we now turn to Case 2 when (3.20) is not true.

Let us define

$$u_i^j = v_i^j - \frac{1}{d}v_0 = \alpha_i^j v_0 + w_i^j$$

to be shifted variants of the vectors. Note that as v_i^j are at most unit length and $\alpha_i^j \ge -\frac{1}{d}$ we have

$$\|u_i^j\|^2 \le 1+\frac{1}{d^2}. \qquad (3.22)$$

Let r be a random vector where each coordinate is picked independently from the normal distribution with mean 0 and standard deviation 1, a distribution we call $N(0,1)$. It is well known, and important for us, that the distribution of r is spherically symmetric and in fact the inner product of r with any unit length vector has the normal distribution with mean 0 and standard deviation 1. Define

$$s_i^j = (u_i^j, r).$$

Note intuitively as the u_i^j are vectors of length at most marginally larger than one we would expect most s_i^j to be bounded by a constant in size. Let $D = \Theta(\sqrt{\log d})$ be such that

$$\frac{4}{\sqrt{\pi}}De^{-D^2/4} = \frac{1}{8d^3}. \qquad (3.23)$$

and proceed as follows. If $|s_i^j| \le D$ for $1 \le j \le d$ then set $t_i^j = s_i^j$ for all j and otherwise set $t_i^j = 0$ for all j. Note that in either case we have

$$\sum_j t_i^j = 0 \tag{3.24}$$

as $\sum_j u_i^j$ is the zero vector. Consider the following strategy.

Set $x_i = j$ with probability $q_i^j = \frac{1}{d}(1 + t_i^j/D)$.

Because of (3.24) we have $\sum_j q_i^j = 1$ and since $|t_i^j| \le D$ the defined probabilities are non-negative.

We get that the expected number of falsified constraints under this strategy is

$$\begin{aligned} \frac{1}{d^2}\sum_{k=1}^{tm}(1 + t_{a_1^k}^{b_1^k}/D)(1 + t_{a_2^k}^{b_2^k}/D) &= \\ \frac{tm}{d^2} + \frac{1}{d^2 D}\sum_{k=1}^{tm}(t_{a_1^k}^{b_1^k} + t_{a_2^k}^{b_2^k}) + \frac{1}{d^2D^2}\sum_{k=1}^{tm} t_{a_1^k}^{b_1^k} t_{a_2^k}^{b_2^k}. \end{aligned} \tag{3.25}$$

As r and $-r$ are equally likely we see that $E[t_i^j] = 0$ for any i and j and thus we only need to consider the second sum. It is easier to analyze the s-values and the following lemma is useful.

Lemma 3.26. *For any i_1, i_2, j_1, j_2 we have*

$$|E[t_{i_1}^{j_1} t_{i_2}^{j_2}] - E[s_{i_1}^{j_1} s_{i_2}^{j_2}]| \le \frac{1}{8d^2}.$$

Proof. Because of (3.22) each s_i^j is normally distributed with mean 0 and standard deviation at most $\sqrt{2}$. Let

$$s = \max_{i \in i_1, i_2, j \in [d]} (|s_i^j|).$$

The difference of $t_{i_1}^{j_1} t_{i_2}^{j_2}$ and $s_{i_1}^{j_1} s_{i_2}^{j_2}$ is 0 if $s \le D$ and otherwise it is at most s^2. The density function of the maximum of $2d$ random variables is at most the sum of the density functions of the variables and as the density of $N(0, \sigma)$ at s is increasing in σ for $\sigma \le s$ we conclude that the difference in expectation is at most

$$2d \cdot \frac{1}{\sqrt{2} \cdot \sqrt{2\pi}} \int_{|s| \ge D} s^2 e^{-s^2/4} ds = \frac{2d}{\sqrt{\pi}} \int_D^\infty s^2 e^{-s^2/4} ds.$$

Integrating by parts and we see that this is bounded by

$$\frac{4d}{\sqrt{\pi}}De^{-D^2/4} - \frac{4d}{\sqrt{\pi}}\int_D^\infty e^{-s^2/4}ds \le \frac{4d}{\sqrt{\pi}}De^{-D^2/4}.$$

The lemma now follows by the property (3.23) defining D. □

In view of the lemma we can now study the s-values and we have the following lemma:

LEMMA 3.27. *Let x_1 and x_2 be any vectors and let r be a random vector in which each coordinate is picked independently from $N(0,1)$. Then $E[(x_1,r)(x_2,r)] = (x_1,x_2)$.*

PROOF. Suppose that $(x_1,x_2) = \beta$ and that $\|x_k\| = \gamma_k$ for $k = 1,2$. As the distribution of r is spherically symmetric we can pick any coordinate system and in particular we can assume that all vectors lie in R^2 and

$$x_1 = (\gamma_1, 0)$$

and

$$x_2 = (\frac{\beta}{\gamma_1}, \sqrt{\gamma_2^2 - \frac{\beta^2}{\gamma_1^2}}).$$

Writing $r = (r_0, r_1)$, and as $E[r_0 r_1] = 0$ we have

$$E[(x_1,r)(x_2,r)] = E[\beta r_0^2] = \beta.$$

□

We conclude that

$$E\left[\sum_k s_{a_1^k}^{b_1^k} s_{a_2^k}^{b_2^k}\right] = \sum_k (u_{a_1^k}^{b_1^k}, u_{a_2^k}^{b_2^k}) \quad = \tag{3.28}$$

$$\sum_k (v_{a_1^k}^{b_1^k} - \frac{1}{d}v_0, v_{a_2^k}^{b_2^k} - \frac{1}{d}v_0) \quad = \tag{3.29}$$

$$\sum_k (v_{a_1^k}^{b_1^k}, v_{a_2^k}^{b_2^k}) - \frac{1}{d}\sum_k (\alpha_{a_1^k}^{b_1^k} + \alpha_{a_2^k}^{b_2^k}) - \frac{tm}{d^2} \quad = \tag{3.30}$$

$$\delta m - \frac{tm}{d^2} - \frac{1}{d}\sum_{i,j} f_i^j \alpha_i^j \quad \le \tag{3.31}$$

$$\delta m - \frac{tm}{d^2} + \frac{tm}{2d^2} = \delta m - \frac{tm}{2d^2} \quad \le \tag{3.32}$$

$$-\frac{tm}{4d^2}, \tag{3.33}$$

where the first inequality follows from the fact that (3.20) is false and the second inequality follows from the assumption that $\delta \leq \frac{t}{4d^2}$. Substituting the result of Lemma 3.26 into (3.25), and remembering that $E[t_i^j] = 0$ for any i and j, we see that the rounding procedure falsifies on the average, over r and the choices of x_i, at most

$$\frac{tm}{d^2} + \frac{1}{d^2D^2}\left(\frac{tm}{8d^2} - \frac{tm}{4d^2}\right) = \frac{tm}{d^2} - \frac{tm}{8d^4D^2} \tag{3.34}$$

constraints and hence the theorem is established also in Case 2.

□

4. Maintaining an advantage

The guaranteed approximation ratio of the algorithm described in the previous section is useful only for instances that are almost satisfiable. In this section we prove the stronger statement that, as long as the optimal solution beats the random solution by a constant factor, we can efficiently find an assignment that beats the random assignment by another, smaller constant factor. Similar theorems for the Boolean case have previously been established by Zwick (1999) and a similar quantitative result for the case $d = 2$ is established using similar methods to ours by Charikar & Wirth (2004).

THEOREM 4.1. *There is a universal constant $c > 0$ such that the following is true. Suppose $OPT(\varphi) \geq m(1-\frac{t}{d^2}+\epsilon)$ then there is an probabilistic polynomial time algorithm A such that*

$$E[Val(\varphi, A(\varphi))] \geq m(1 - \frac{t}{d^2} + \frac{\epsilon c}{d^2 \log{(dt/\epsilon)}}).$$

PROOF. We follow the proof of Theorem 3.1 only adjusting the parameters. In particular we start with the same semi-definite program defined by (3.2)-(3.7). By the assumption on $OPT(\varphi)$ we find a set of vectors v_i^j with objective value $m(1 - \delta)$ where

$$\delta \leq \frac{t}{d^2} - \epsilon.$$

We again split the analysis into two cases with the difference that the inequality (3.20) is replaced by

$$\sum_{i,j} f_i^j \alpha_i^j \leq -\frac{d\epsilon m}{2}. \tag{4.2}$$

The probabilities p_i^j in Case 1 are defined as in the proof of Theorem 3.1 and the calculation for the expected number of falsified constraints is the same except for the final step of (3.21) that uses (4.2) giving the final result

$$\frac{tm}{d^2} - \frac{\epsilon m}{8},$$

and this completes the analysis of Case 1.

The analysis of Case 2 is again almost identical but we redefine D to satisfy

$$\frac{4}{\sqrt{\pi}} D e^{-D^2/4} = \frac{\epsilon}{4dt}.$$

This implies that Lemma 3.26 remains true with the constant $\frac{1}{8d^2}$ replaced by $\epsilon/(4t)$. In the calculation replacing (3.28)-(3.33) we use $\delta \le \frac{t}{d^2} - \epsilon$ and $\frac{1}{d}\sum_{i,j} f_i^j \alpha_i^j \ge -\frac{\epsilon m}{2}$ giving the total bound $-\epsilon m/2$. This implies that the final calculation (3.34) is now replaced by

$$\frac{tm}{d^2} + \frac{1}{d^2 D^2}\left(\frac{\epsilon m}{4} - \frac{\epsilon m}{2}\right) = \frac{tm}{d^2} - \frac{\epsilon m}{4d^2 D^2}$$

and the proof is complete, as $D^2 = \Theta(\log(dt/\epsilon))$.

□

5. Final remarks

We have proved that semi-definite programming is a universal tool for establishing non-trivial approximation results for binary predicates over any domain. The technique has also been used with good results for other problems such as coloring of three-colorable graphs (see Arora *et al.* (2006) and the references in this paper) and approximation of Max-4-Sat instances that also contain shorter clauses given by Halperin & Zwick (2001). We will probably see many good uses of semi-definite programming also in the future.

We have mainly addressed the question on whether our predicates allow non-trivial approximation ratios and the quantitative results are not very good. Clearly this could be improved, or one could try to prove (close to) matching lower bounds. We note that, as d increases and for $t = d^2 - d$, by a result of Feige & Reichman (2004) the approximation ratio must turn to 0. Looking at the new proof of the PCP-theorem by Dinur (2006) we see that this also applies to satisfiable instances and a slightly smaller value of t.

Acknowledgements

I am grateful to Lars Engebretsen both for giving me a reprint of the article Engebretsen & Guruswami (2004) which inspired me to think about this problem and for discussions that simplified and streamlined the proof. I am also grateful to Venkatesan Guruswami for suggesting that it would be possible to obtain better results for predicates rejecting more than one input. Finally I am grateful to an anonymous referee for pointing out an inaccuracy in the final remarks section of a previous version of the paper.

References

F. Alizadeh (1995). Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization* **5**, 13–51.

G. Andersson, L. Engebretsen & J. Håstad (2001). A New Way to Use Semidefinite Programming with Applications to Linear Equations mod *p*. *Journal of Algorithms* **39**, 162–204.

S. Arora, E. Chlamtac & M. Charikar (2006). New Approximation Guarantee for Chromatic Number. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computating*, 215–224.

S. Arora, C. Lund, R. Motwani, M. Sudan & M.Szegedy (1998). Proof verification and intractability of approximation problems. *Journal of the ACM* **45**, 501–555.

M. Charikar & A. Wirth (2004). Maximizing quadratic programs: extending Grothendieck's inequality. In *Proceedings of 45th Annual IEEE Symposium of Foundations of Computer Science*, 54–60.

S. Cook (1971). The complexity of theorem proving procedures. In *3rd Annual ACM Symposium on Theory of Computing*, 151–158.

N. Creignou (1995). A dichotomy theorem for maximum generlized satisfiability problems. *Journal of Computer and System Science* **52**, 511–522.

N. Creignou, S. Khanna & M. Sudan (2001). Complexity Classifications of Boolean Constraint Satisfaction Problems .

I. Dinur (2006). The PCP Theorem by gap amplification. In *Proceedings of 38th Annual ACM symposium on Theory of Computing*, 241–250.

L. ENGEBRETSEN & V. GURUSWAMI (2004). Is constraint satisfaction over two variables always easy? *Random Structures and Algorithms* **25**, 150–178.

U. FEIGE (1998). A threshold of $\ln n$ for approximating set cover. *Journal of the ACM* **45**, 634–652.

U. FEIGE & J. KILIAN (1998). Zero-knowledge and the chromatic number. *Journal of Computer and System Sciences* **57**, 187–200.

U. FEIGE & D. REICHMAN (2004). On Systems of Linear Equations with Two Variables per Equation. In *Approximation, Randomization and Combinatorial Optimization*, K. JANSEN, S. KHANNA, J. ROLIM & D. RON, editors, 117–127. Proceedings of Approx 2004 and Random 2004, LNCS 3122.

M.R. GAREY & D.S. JOHNSON. (1979). *Computers and Intractability.* W.H. Freeman and Company.

M. GOEMANS & D. WILLIAMSON (1995). Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM* **42**, 1115–1145.

M. GRÖTSCHEL, L. LOVÁSZ & A. SCHRIJVER (1981). The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica* **1**, 169–197.

E. HALPERIN & URI ZWICK (2001). Approximation algorithms for Max 4-Sat and rounding procedures for semidefinite programs. *Journal of Algorithms* **40**, 184–211.

G. HAST (2005). *Beating a random assignment.* KTH, Stockholm. Ph.D Thesis.

J. HÅSTAD (1999). Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica* **182**, 105–142.

J. HÅSTAD (2001). Some optimal inapproximability results. *Journal of ACM* **48**, 798–859.

J. HÅSTAD (2005). Every 2-CSP Allows Nontrivial Approximation. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computating*, 740–746.

S. KHANNA, M. SUDAN, L. TREVISAN & D.P. WILLIAMSON (2000). A approximability of constraint satisfaction problems. *SIAM Journal on Computing* **30**, 1863–1920.

S. KHOT (2001). Improved inapproximability results for maxclique and chromatic number. In *Proceedings of 42nd Annual IEEE Symposium of Foundations of Computer Science*, 600–609.

S. Khot, E. Mossel G. Kindler & R. O'Donnell (2007). Optimal inapproximability results for Max-Cut and other 2-variable CSPs? *Siam Journal on Computing* **37**, 319–357.

A. Samorodnitsky & L. Trevisan (2000). A PCP characterization of NP with optimal amortized query complexity. In *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*, 191–199.

T. Schaefer (1978). The complexity of satisfiability problems. In *Conference record of the Tenth annual ACM Symposium on Theory of Computing*, 216–226.

U. Zwick (1998). Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In *Proceedings 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, 201–210. ACM.

U. Zwick (1999). Outward rotations: a tool for rounding solutions of semidefinite programming relaxations, with applications to MAX CUT and other problems. In *Proceedings 31st ACM Symposium on Theory of Computing*, 679–687. ACM.

Manuscript received October 11, 2007

Johan Håstad
Royal Institute of Technology
Stockholm Sweden
johanh@kth.se
http://www.csc.kth.se/~johanh/