The Hardness of Approximate Optima in Lattices, Codes, and Systems of Linear Equations

Sanjeev Arora*
Computer Science
Princeton University

László Babai†
University of Chicago and
Eötvös University, Hungary

Jacques Stern
Laboratoire d'Informatique
Ecole Normale Supérieure

Z Sweedyk‡
CS Division
U. C. Berkeley

Abstract

We prove the following about the Nearest Lattice Vector Problem (in any ℓ_p norm), the Nearest Codeword Problem for binary codes, the problem of learning a halfspace in the presence of errors, and some other problems.

1. *Approximating the optimum within any constant factor is NP-hard.*
2. *If for some $\epsilon > 0$ there exists a polynomial-time algorithm that approximates the optimum within a factor of $2^{\log^{0.5-\epsilon} n}$, then every NP language can be decided in quasi-polynomial deterministic time, i.e., $NP \subseteq DTIME(n^{poly(\log n)})$.*

Moreover, we show that result 2 also holds for the Shortest Lattice Vector Problem in the ℓ_∞ norm. Also, for some of these problems we can prove the same result as above, but for a larger factor such as $2^{\log^{1-\epsilon} n}$ or n^ϵ.

Improving the factor $2^{\log^{0.5-\epsilon} n}$ to $\sqrt{dimension}$ for either of the lattice problems would imply the hardness of the Shortest Vector Problem in ℓ_2 norm; an old open problem.

Our proofs use reductions from few-prover, one-round interactive proof systems [FL], [BG+], either directly, or through a set-cover problem.

1 Introduction

Many important optimization problems are NP-hard [GJ]. Recent work has shown that computing even approximate solutions to many of these problems is NP-hard (cf. [Con], [FG+], [AS], [AL+], [LY], [Z], [BG+]). This paper continues that line

*Research done while the author was at UC Berkeley, and was supported by an IBM Graduate Fellowship.

†Partially supported by NSF grant CCR-9014562.

‡Partially supported by NSF grant CCR-9201092 and Sandia National Laboratories

of work by demonstrating the hardness of computing approximate solutions to well-known minimum distance problems for integral lattices and linear codes, as well as the problem of finding a largest feasible subsystem of a system of linear equations (or inequalities) over $\mathbf{Q}$.

In fact, just as in all the previous work on this subject, we do not require the computation of an approximately optimal solution, merely an estimation of the optimum value. Even this task is often hard.

We adopt the following terminology.

Approximating a minimization problem *within a factor* c, where $c \geq 1$, means to find a value v guarateed to satisfy the inequalities $w \leq v \leq wc$, where w is the actual optimum. Similarly for a maximization problem, we need to find v such that $w/c \leq v \leq w$ (again, $c > 1$). The number c is called the *approximation ratio* of the approximation algorithm. Note that the closer c is to 1, the better the approximation.

Some results in this paper show that achieving certain approximation ratios for certain problems is NP-hard. Other results show only that the task is *almost* NP-hard. A function f is *almost-NP-hard* if, using f as an oracle, we could recognize any NP language in deterministic *quasi-polynomial* time, i.e. in time $O(n^{poly(\log n)})$. Here and elsewhere in this section, n denotes the length of the input. Further, m denotes the dimension of the lattice, code etc. under consideration.

Now we define the optimization problems considered in this paper. An *integral lattice* $\mathcal{L}(b_1, \ldots, b_m)$ in $\mathbf{R}^k$ is the set of all points in the set $\{\sum_{i=1}^{m} \alpha_i b_i : \alpha_i \in \mathbf{Z}\}$, where $\{b_1, \ldots, b_m\}$ is a set of independent vectors in $\mathbf{Z}^k$, called the *basis* of the lattice. If p is a positive integer, the ℓ_p *norm* of a vector $(a_1, \ldots, a_k) \in \mathbf{R}^k$ is the number $(a_1^p + a_2^p + \cdots + a_k^p)^{1/p}$. The ℓ_∞ *norm* of this same vector is the number $\max \{|a_1|, \ldots, |a_k|\}$.

Definition 1 (Shortest Vector Problem in ℓ_p norm, SV_p)*: Given a basis* $\{b_1, \ldots, b_m\}$*, find the shortest non-zero vector (in* ℓ_p *norm) in* $\mathcal{L}(b_1, \ldots, b_m)$ *.*

Definition 2 (Nearest Vector Problem in ℓ_p norm, NV_p)*: Given a basis* $\{b_1, \ldots, b_m\}$*, and a point* $b_0 \in \mathbf{Q}^k$*, find the nearest vector (in* ℓ_p *norm) in* $\mathcal{L}(b_1, \ldots, b_m)$ *to* b_0*.*

The SV problem is particularly important because even relatively poor polynomial-time approximation algorithms for it ([L^3]) have been used in a host of applications, including integer programming, solving low-density subset-sum problems and breaking knapsack-based codes [LO], simultaneous diophantine approximation and factoring polynomials over the rationals [L^3], and strongly polynomial-time algorithms in combinatorial optimization [FT]. For details and more applications, especially to classical problems in the "geometry of numbers", see the surveys by Lovász [Lov] or Kannan [K2].

Lovász's celebrated lattice transformation algorithm [L^3] runs in polynomial time and approximates SV_p ($p \geq 1$) within a factor 2^m. A modification of this algorithm [Bab] allows the same approximation for NV_p. Finally, Schnorr has shown how to modify Lovász's algorithm to approximate both these problems within a factor $O(2^{\epsilon m})$ in polynomial time, for every $\epsilon > 0$ [Sch].

On the other hand, Van Emde Boas showed that NV_p is NP-hard for all $p \geq 1$ ([vEB]; see [K] for a simpler proof). Lagarias showed that the shortest vector problem is NP-hard under the ℓ_∞ (i.e. max) norm. But it is still an open problem whether SV_p is NP-hard for any other p, and specifically for $p = 2$.

While we do not solve this open problem, we obtain hardness results for the approximate solutions of the known NP-hard cases.

We follow the example of recently established links between the theory of interactive proofs and the hardness of approximation (cf. [Con], [FG+], [AS], [AL+], [LY], [Z], [BG+]), and add some new links . Specifically, we introduce a new combinatorial problem, which we call label cover. This problem is proved hard to approximate (up to LARGE factors) using ideas from interactive proofs. We propose that label cover can be used as a new canonical problem for showing the hardness of approximations, just as 3SAT is the canonical problem for showing hardness of finding exact solutions.

In the following theorems, LARGE factors mean factors $2^{\log^{0.5-\epsilon} n}$ where $\epsilon > 0$ is fixed and n = input size.

Theorem 1 *1. Approximating NV_p (i) within any constant factor $c \geq 1$ is NP-hard; (ii) within any LARGE factor is almost-NP-hard.*

2. Approximating SV_∞ within any LARGE factor is almost-NP-hard.

Our reductions use only $(0, \pm 1)$-vectors. Hence the problems remain hard in that subcase.

We mention that improving the LARGE factors in either of the above results to $\sqrt{m}$ (m = dimension) would prove hardness of SV_2, a long standing open question. The reason is that approximating either SV_∞ or NV_2 to within a factor $\sqrt{m}$ is reducible to SV_2. To see this for SV_∞, notice that the solutions in SV_∞ and SV_2 are always within a factor $\sqrt{m}$ of each other. For NV_2 the implication follows from Kannan's result [K] that approximating NV_2 within a factor $\Omega(\sqrt{d})$ is reducible to SV_2.

We also note that approximating NV_2 within any factor greater than $m^{1.5}$ is unlikely to be NP-complete, since Lagarias et al. [LLS] have shown that this problem lies in NP $\cap$ co-NP.

Our results, like several other recent "inapproximability" results, use ideas involving few-prover single-round interactive membership proofs for NP. Specifically, we use a theorem of Feige and Lovász [FL].

We also derive hardness results for a number of other problems which in one way or another involve distances of vectors. Next, we define these problems.

Nearest-Codeword: INPUT : An $m \times k$ matrix A over $GF(2)$ and a vector $y \in GF(2)^k$.
OUTPUT : A vector $x \in GF(2)^m$.
MINIMIZE: the Hamming distance between xA and y.

Max-Satisfy: INPUT : A system of k equations in m variables over $\mathbf{Q}$.
OUTPUT : subset of equations that is a feasible system.
MAXIMIZE: size of feasible subset (number of equations)

Min-Unsatisfy: INPUT : A system of k equations in m variables over $\mathbf{Q}$.
OUTPUT : A set of equations whose removal makes the system feasible.
MINIMIZE the number of equations to be omitted.

Observe that a solution to MAX-SATISFY is exactly the complement of a solution to MIN-UNSATISFY, and therefore the two problems have the same complexity. (Indeed, it is known that both are NP-hard; this is implicit for example in [JP]). However, the same need not be true for approximate solutions. For instance, vertex cover and independent set are another "complementary" pair of problems, and seem to differ greatly in how well they can be approximated in polynomial time. (Vertex cover can be approximated within a factor 2, and independent set is NP-hard up to a factor n^c for some $c > 0$ [FG+, AS, AL+].)

Theorem 2 *Approximating MIN-UNSATISFY and NEAREST-CODEWORD within any constant factor is NP-hard and within any LARGE factor is almost-NP-hard.*

Theorem 3 *Approximating MAX-SATISFY within a factor of n^δ is NP-hard for some $\delta > 0$.*

We know of no good approximation algorithms for any of these problems. Kannan has shown us a simple polynomial time algorithm that uses Helly's theorem to approximate MIN-UNSATISFY within a factor of $m+1$.

Finally, we consider a well-known problem in *learning theory:* learning a halfspace in the presence of malicious errors. (The context in which the problem arises is that of training a perceptron, a learning model first studied by Minsky and Papert [MP].) Rather than describing the learning problem in the usual PAC setting, we merely present the underlying combinatorial problem.

The input to the learner consists of a set of k points in $\mathbf{R}^m$, each labelled with $+$ or $-$. (These should be considered as positive and negative examples of a concept.) The learner's output is a hyperplane, $\sum_{i=1}^m a_i \cdot x_i = b$ ($a_i, b \in \mathbf{R}$). The hypothesis is said to *correctly classify* a point marked $+$ (resp. $-$) if that point, say y satisfies $a \cdot y > b$ ($a \cdot y < b$, resp.). Otherwise it is said to *misclassify* the point.

Finding a hypothesis that minimizes the number of misclassifications is the *open hemispheres* problem, which is NP-hard [GJ]. Define the *error* of the algorithm as the number of misclassifications by its hypothesis, and the *noise* of the sample as the error of the best possible algorithm. Let the *failure ratio* of the algorithm be the ratio of the error to noise.

Theorem 4 *Achieving any constant failure ratio in the LEARNING_HALFSPACES problem is NP-hard and achieving any LARGE failure ratio is almost-NP-hard.*

This confirms what has been conjectured in learning theory [HS]. A failure ratio $\leq m$ can be achieved by Kannan's idea, mentioned above.

Better factors. For NEAREST-CODEWORD and NV_p for all $p \geq 1$, in Section 4 we shall actually prove hardness up to a factor of $2^{\log^{1-\epsilon} n}$ instead of $2^{\log^{0.5-\epsilon} n}$. Also, in our reductions the number of variables, dimensions, input size etc. are polynomially related, so n could be any of these.

Previous or independent work. Bruck and Naor ([BN]) have shown the hardness of approximating the NEAREST-CODEWORD problem to within some $1+\epsilon$ factor. Amaldi and Kann [AK93] have independently obtained the same result as ours for MAX-SATISFY. They have also extended our results on MIN-UNSATISFY and the learning problem to a class of related problems [AK95].

2 Organization of the Paper

Although the various problems mentioned in the introduction may appear to be very different, the reductions used to show their nonapproximability are very similar (MAX-SATISFY requires a different reduction, however). Section 3 shows the NP-hardness of approximating these problems within any constant factor. Although the reductions in that section are fairly simple, they will help illustrate some of the ideas used subsequently in Section 5, in which we will show the hardness of approximating the same problems up to LARGE factors. Specifically, Section 3 shows how the reductions to the different problems are connected to one another; these connections continue to hold in Section 5.

The hardness result for SV_∞ uses a lemma about an intermediate problem called Label Cover. The lemma is proved (using existing constructions of interactive proofs and some new geometric arguments) in Section 6.

We have also discovered simpler versions of some of the above reductions. These are described in Section 4. We chose to describe both the original reductions and the simpler reductions because they involve two different paradigms, and both might be useful in future work.

Finally, Section 7 contains the hardness result for MAX-SATISFY.

3 Constant Factors

In this section we prove the following theorem.

Theorem 5 *For any constant $c > 1$, approximating the following problems within a factor c is NP-hard: (i) NV_p for $p \geq 1$ (ii) NEAREST-CODEWORD (iii) MIN-UNSATISFY (iv) Minimum Failure Ratio while learning a halfspace in presence of errors.*

We use the reduction to set-cover given by Bellare et al. [BG+], which improves the reduction due to Lund and Yannakakis [LY]. Recall that an instance of *set-cover* consists of a ground-set $\mathcal{U}$ and a collection of subsets $S_1, S_2, \ldots, S_m$ of $\mathcal{U}$. A *cover* is a subcollection of the S_i's whose union is $\mathcal{U}$. The cover is said to be *exact* if the sets in the cover are pairwise disjoint. The following result appears in [BG+].

Proposition 6 [BG+] *For every $c > 1$ there is a polynomial time reduction that, given an instance φ of SAT, produces an instance of set-cover and integer K with the following property: If φ is satisfiable, there is an exact cover of size K, and otherwise no set cover has size less than $c \cdot K$.* $\square$

Now we prove Theorem 5.

Proof:(*of Theorem 5*) We prove the result for NV_1 first, then modify it to prove the other results.

Let $(\mathcal{U}, S_1, \ldots, S_m), K$ be the instance of set-cover obtained in Proposition 6. We transform it to an instance of NV_1, b_0 and $\mathcal{L}(b_1, \ldots, b_m)$, such that the distance of b_0 to the nearest lattice point is either K or $\geq c \cdot K$.

The vectors have $|\mathcal{U}| + m$ coordinates. Let $L = c \cdot K$. The point b_0 is the vector having L in the first $|\mathcal{U}|$ coordinates, followed by m 0's. The i'th basis vector b_i is $(L \cdot \chi_{\mathrm{S_i}}, 0, \ldots, 0, 1, 0, \ldots, 0)$ (the 1 appears in position $|\mathcal{U}| + i$). Here $\chi_{\mathrm{S_i}}$ is the characteristic vector of the set S_i, i.e., a vector with $|\mathcal{U}|$ coordinates that has a 1 in each coordinate corresponding to an element in S_i and 0 otherwise.

Let OPT $=$ min $\{|| - b_0 + \sum_i \alpha_i \cdot b_i||_1 : \alpha_i \in \mathbf{Z}\}$. Just the contribution from the last m coordinates shows $|| - b_0 + \sum_i \alpha_i \cdot b_i||_1 \geq \sum_i |\alpha_i|$, with equality holding iff the first $|\mathcal{U}|$ coordinates are 0. Furthermore, if any of the first $|\mathcal{U}|$ coordinates is not 0, then it is at least L, in which case $|| - b_0 + \sum_i \alpha_i \cdot b_i||_1 \geq L + \sum_i |\alpha_i| \geq L$.

If there is an exact cover of size K, say $(S_{i_1}, S_{i_2}, \ldots, S_{i_K})$, then the vector $\sum_{j=1}^{K} b_{i_j}$ has ℓ_1 distance K from b_0. (Note that the exactness of the cover is crucial.)

Conversely, suppose no cover has size less than L. Let us show that OPT $\geq L$. Consider, for any integer assignment to the α_i's, the collection of subsets $\{S_i : \alpha_i \neq 0\}$. If these sets form a set-cover, then $\sum_i |\alpha_i| \geq cL$. If they do not, then one of the coordinates in $-b_0 + \sum_i \alpha_i \cdot b_i$ is at least L. In any case, $|| - b_0 + \sum_i \alpha_i \cdot b_i||_1 \geq L$. This proves our result for NV_1.

NV_1 with 0/1 vectors. Replace each of the first $|\mathcal{U}|$ coordinates by a set of L new coordinates. If a vector had an L in the original coordinate, it has a 1 in each of the new L coordinates, and 0 otherwise. Now, in the argument above, whenever a coordinate contributes L to the ℓ_1 norm of a vector, so do the coordinates corresponding to it in the new vectors. The rest of the argument then goes through.

Other finite norms. In the above construction, changing the norm from ℓ_1 to ℓ_p changes the minimum distances in the two cases to $\sqrt[p]{K}$ and $\sqrt[p]{cK}$ respectively. Hence if we wish to prove the hardness of approximating NV_p within a factor t, we just use the reduction described in Proposition 6 with $c = t^p$. (Note that since t and p are constants, so is t^p.) Thus the result for the ℓ_p norm follows.

l_∞ norm. The result for this norm uses some other ideas (related to the construction in Section 6). The result appears in [A96].

Nearest-Codeword. View the vectors $b_1', \ldots, b_m'$ obtained from the reduction to "NV_1 with 0/1 vectors" as vectors over $GF(2)$, in other words, as the generators of a binary code. Let the received message be the vector b_0' described there. In the case there is an exact cover $S_{i_1}, \ldots, S_{i_K}$ of size K, we showed that a closest point to b_0' is $\sum_{j=1}^K b_{i_j}'$, and the vector $b_0' - \sum_{j=1}^K b_{i_j}'$ is a 0/1 vector. Since interpreting the $+$ and $-$ above as operations over $GF(2)$ cannot change a result of 0 into a result of 1, we conclude that there is a codeword whose hamming distance to b_0' is at most K. Conversely, suppose every set cover has size $\geq cK$. Note that a codeword $\sum_{i \in I} b_i'$ can be only of one of the following two types: either I is a set cover, or it is not. In both cases, the same analysis as before shows that no codeword has distance $< cK$ to b_0'.

Min-Unsatisfy. Consider the instance $b_0', b_1', \ldots, b_m'$ again. Each vector has $m + L \cdot |\mathcal{U}|$ coordinates. This instance implicitly defines the following system of $m + L \cdot |\mathcal{U}|$ equations in m variables :

$$-b_0' + \sum \alpha_i \cdot b_i' = \vec{0},$$

where the α_i's are the variables and $\vec{0}$ is the vector whose all coordinates are 0. For clarity, we restate this system by using a variable x_S for each set S in the Set Cover instance. For each element $u \in \mathcal{U}$, the system of equations contains L copies of

$$\sum_{S \ni u} x_S = 1, \tag{1}$$

and for every set S it contains one copy of

$$x_S = 0 \tag{2}$$

Now, if there is an exact cover of size K, we can assign every variable corresponding to those sets to 1 and all other variables to 0, thus satisfying all equations of the type in (1), but failing to satisfy K equations of the type in (2).

Conversely, suppose every set cover has size $\geq cK$. There are two types of assignments to the variables, and both fail to satisfy at least cK equations: *(i)* Those in which the non-zero variables give rise to a set cover. Such an assignment fails to satisfy at least cK equations of the type in (2). *(ii)* Those in which the non-zero variables do not form a set cover. Such an assignment fails to satisfy, for some $u \in \mathcal{U}$, all L equations of type in (1) corresponding to it.

Learning Halfspaces. First we convert the instances of MIN-UNSATISFY described above into a system of strict inequalites. Introduce a new variable δ and replace each equation of the type in (1) by

$$\sum_{S \ni u} x_S + \delta \quad > \quad 0 \tag{3}$$

$$\sum_{S \ni u} x_S - \delta \quad < \quad 0 \tag{4}$$

and the equation in (2) by

$$x_S + \delta \quad > \quad 0 \tag{5}$$

$$x_S - \delta \quad < \quad 0. \tag{6}$$

Add to the system L copies each of the inequalities

$$-\frac{1}{2m} < \delta < \frac{1}{2m}, \tag{7}$$

where m is the number of sets in the set system.

Clearly, if there is an exact cover of size K, then there is an assignment to the variables that fails to satisfy K inequalities – just make $\delta = 0$.

But when every set cover has size $\geq cK$, we show that every assignment fails to satisfy at least cK inequalities. First, note that if the assignment does not satisfy Inequality (7), it already doesn't satisfy L inequalities. So assume it satisfies Inequality (7). Then consider the set of variables to which it assigns values $\geq 1/m$. If they do not form a set cover, then L inequalities corresponding to some $u \in \mathcal{U}$ described in (3) and (4) are not satisfied. If the variables do form a set cover, then $\geq cK$ inequalities of the type in (5) and (6) are not satisfied.

Finally, to go from the system of strict inequalites to an instance of the learning problem, notice that the learning problem also involves a system of strict inequalities, except that the system is homogeneous and the coeeficients of one of the variables is always 1. Recall that if $\sum_i a_i y_i = b$ is the unknown hyperplane, then the variables are the a_i's and b, and the coeeficient of b is 1 in all the constraints.

Let us transform our system of inequalities into this special form. Since the coefficient of δ in all our equations is ± 1, we can use it as b. To achieve homogeneity, use the familiar device of adding a new variable y and replacing every appearance of a constant c (which makes the constraint inhomogenous) by $c \cdot y$. By adding L copies of the constraint $y > 0$, we ensure that the optimum assignment makes $y > 0$, and then we can divide out every assignment by the value of y to get an assignment for the old (inhomogenous) system.

$\square$

A comment. We note that the reduction depends crucially upon the exactness of the cover when φ is satisfiable. Such peculiarities seem inherent in recent non-approximability results. For instance, the hardness result for Chromatic Number in [LY] depends upon very specific properties of the graph obtained in the clique reductions of [FG+, AS, AL+].

4 Larger factors: The First Reduction

This section contains a proof of the following result.

Theorem 7 *If there are polynomial time algorithms that approximate any of the following problems within a factor of $2^{\log^{1-\epsilon} n}$ for any $\epsilon > 0$: (i) NV_p, for $p \geq 1$ (ii) NEAREST CODEWORD (iii) MIN UNSATISFY, then NP $\subseteq$ Dtime($n^{poly(\log n)}$).*

Our proof will be based on an iterative construction that starts from instances with a (fixed) constant gap and gradually increases the gap. We will use the setting of lattices in order to describe the method and we will further restrict ourselves to the case $p = 1$. Hardness results for the other problems will follow by suitable modifications as indicated in the previous section.

We showed in section 3 that one could transform an instance ϕ of SAT into an instance of NV_1 consisting of a lattice $\mathcal{L}(b_1, \cdots, b_m)$ together with a vector b_0 and an integer K such that, for some constant $c > 1$ the following statements hold:
i) if ϕ is satisfiable then the minimum distance of $\mathcal{L}$ to b_0 under the ℓ_1-norm is K
ii) if ϕ is not satisfiable then the minimum distance of $\mathcal{L}$ to b_0 under the ℓ_1-norm is $\geq cK$.

We have also shown how to modify our basic reduction in order to deal with the NV_1 problem with 0/1 vectors. The instances we built for this specific problem actually enjoy some further properties that we will need:
iii) if ϕ is satisfiable, then there is a vector v achieving minimum distance to b_0, with all coordinates 0 or 1
iv) if ϕ is not satisfiable then, for any non zero integer α and any vector w of the lattice, $\alpha b_0 + w$ has at least cK non zero coordinates.

In order to prove theorem 7, we will build a polynomial-time reduction which transforms instances of NV_1 enjoying the above properties into instances enjoying similar properties with c replaced by c^2. Iterating a large number of times will then yield the result.

We start from the matrix M whose rows are the vectors $b_1, \cdots, b_m$ generating the lattice $\mathcal{L}(b_1, \cdots, b_m)$, considered above. M has m rows and say p columns. We let $\tilde{M}$ be the (m, p^2) matrix obtained from M by replacing each coefficient μ by a block of p coordinates exactly equal to the row vector μb_0. The new lattice is generated by the rows of a matrix of the following form

$$M' = \left(\begin{array}{|c|} \hline \tilde{M} \\ \hline \overline{M} \\ \hline \end{array} \right)$$

where $\overline{M}$ is built as follows:

$$\overline{M} = \begin{pmatrix} \boxed{P_1} & \cdots & \boxed{0} & \boxed{0} \\ \boxed{0} & \boxed{P_2} & \cdots & \boxed{0} \\ \cdots & \cdots & \cdots & \cdots \\ \boxed{0} & \boxed{0} & \cdots & \boxed{P_p} \end{pmatrix}$$

In the above diagram, each matrix P_k has m rows and p columns and is simply a copy of M. Thus, we have built a matrix M' with $m(p+1)$ rows and p^2 columns through a construction that can clearly be achieved in polynomial time. Finally, we consider the lattice $\mathcal{L}'$ generated by the rows of M' and the vector b_0' obtained from b_0 by replacing each coefficient μ by a block of p coordinates exactly equal to the row vector μb_0. We also set $K' = K^2$. We then make the following remarks:

1. If ϕ (the given instance of SAT) is satisfiable, then there is a lattice vector w built as a linear combination of the first m rows of M' such that $b_0' + w$ has exactly K non-zero blocks, each consisting of a copy of the row vector b_0. In each of these blocks, coordinates can be further cancelled by using the corresponding matrix P_k so as to leave exactly K ones. Thus, the distance of b_0' to $\mathcal{L}'$ is exactly K^2. This proves properties *i* and *iii* for $\mathcal{L}'$, b_0', K' with c replaced by c^2.

2. If Φ is not satisfiable, then any vector w built as a linear combination of the first m rows of M' will leave at least cK non-zero blocks in $b_0' + w$. In each of these blocks, use of the corresponding matrix will cancel more coordinates but, using property *iv* of the pair $\mathcal{L}', b_0'$, we can see that at least cK non zero coordinates are left. Therefore the distance of b_0' to $\mathcal{L}'$ is at least $(cK)^2$. The same argument holds with b_0' replaced by $\alpha b_0'$, thus showing properties *ii* and and *iv* for $\mathcal{L}'$, b_0', K'.

Thus, as announced, we have built a reduction with c replaced by c^2. We now turn to the proof of theorem 7. This proof is based on repeating the above construction $k(n)$ times, where $k(n)$ is a function of the size n of the original SAT instance we started with. During this iterative construction, the size of the matrix used to define the lattice grows: the number of columns reaches $p^{2^{k(n)}}$ and the number of rows $m \prod_{i=0}^{k(n)} (p^{2^i} + 1)$, which is bounded by $m(1+p)^{2^{k(n)}+1}$. In any case, the construction cannot be achieved in time polynomial in n but rather in time $n^{O(2^{k(n)})}$. If we let $2^{k(n)}$ be equal to $\log^\beta n$, then the construction is in $\mathrm{Dtime}(n^{poly(\log n)})$

and the size of the resulting lattice is $N = n^{O(\log^{\beta} n)} = 2^{O(\log^{\beta+1} n)}$. Note that $\log N = O(\log^{\beta+1} n)$. Also, the gap for the resulting instances of NV_1 has become $c^{2^{k(n)}} = c^{\log^{\beta} n} = 2^{O((\log N)^{\frac{\beta}{\beta+1}})}$.

Assume that some polynomial time algorithm approximates the problem NV_1 within a factor $2^{\log^{1-\epsilon} n}$ for any $\epsilon > 1 - \frac{\beta}{\beta+1}$ then, applying this algorithm to the output of the above iterated construction, one can decide the satisfiability problem for a given instance ϕ of SAT. Since our construction is in Dtime$(n^{poly(\log n)})$ the resulting decision algorithm is in Dtime$(n^{poly(\log n)})$ as well. This finishes the proof of theorem 7.

5 Large Factors: The Second Reduction

In this section we prove the following theorem. Note that the result about NV$_p$, NEAREST CODEWORD, and MIN-UNSATISFY has already been proved in the previous section; we reprove it here to demonstrate the power of the approach in this section.

Theorem 8 *Approximating each of the following problems up to LARGE factors (as defined in the introduction) is almost-NP-hard: (i) NV_p for any $p \geq 1$. (ii) SV_∞ (iii) NEAREST CODEWORD (iv) Minimum failure ratio while learning a halfspace in the presence of errors (v) MIN-UNSATISFY.*

We will show the result only for NV$_1$ and SV$_\infty$; the hardness results for the other problems will follow from the NV$_1$ result exactly as in Theorem 5.

The reductions in this section are from certain problems involving label covers for bipartite graphs. The hardness of approximating these covering problems is proved in Lemmas 9 (used in the reduction to NV$_1$) and 10 (used in the reduction to SV$_\infty$). These hardness results use a theorem of Feige and Lovász [FL] about the existence of efficient 2-prover, 1-round interactive proofs for NP. In this respect, our reductions are similar to previous reductions in [Bel, BR, LY, BG+], and are closest in spirit to the reduction to set-cover given by Lund and Yannakakis [LY]. However, proving the correctness of our reduction to SV$_\infty$ involves delving into the geometric structure of the Feige-Lovász proof-system, and it is an open problem whether there are reductions based on simpler principles.

5.1 The Label Cover Problem

The input to the label cover problem is a bipartite graph $G = (V_1, V_2, E)$ (where $E \subseteq V_1 \times V_2$), and a set of possible labels B_1 and B_2 for vertices in V_1 and V_2 respectively. Also included in the input is a relation $\Pi \subseteq E \times B_1 \times B_2$ that consists of admissible pairs of labels for each edge. A *labelling* of the graph is a pair of functions $(\mathcal{P}_1, \mathcal{P}_2)$ where $\mathcal{P}_i : V_i \to 2^{B_i}$ for $i = 1, 2$; in other words, $(\mathcal{P}_1, \mathcal{P}_2)$ assigns a set of labels – possibly empty – to each vertex of the graph. The l_1-*cost* of the labelling is $\sum_{v \in V_1} |\mathcal{P}_1(v)|$ and its ℓ_∞-*cost* is $\max_{v \in V_1} |\mathcal{P}_1(v)|$. A labelling is said to *cover* an edge (v_1, v_2) if both $\mathcal{P}_1(v_1)$ and $\mathcal{P}_2(v_2)$ are nonempty and for every label

$b_2 \in \mathcal{P}_2(v_2)$ there exists a $b_1 \in \mathcal{P}_1(v_1)$ such that $(e, b_1, b_2) \in \Pi$. A *total-cover* of G is a labelling that covers every edge.

It is a simple exercise to see that finding the minimum cost (under any ℓ_p norm) of these covers is NP-hard. Furthermore, it is implicit in the calculations of [LY] that it is almost-NP-hard to obtain even weak approximations to these costs. Here we restate this result in a more exact form.

Lemma 9 *For each fixed $\epsilon > 0$ there is a quasi-polynomial time reduction that reduces an instance φ of SAT of size n to an instance of label cover (G, B_1, B_2, Π) of size N ($N \leq 2^{poly(\log n)}$) such that*

- *If φ is satisfiable, there is a total-cover with l_1-cost $= |V_1|$ and ℓ_∞-cost $= 1$.*
- *If φ is not satisfiable then any labelling that covers more than $\frac{1}{2}$ the edges (in particular, every total-cover) has l_1-cost $\geq 2^{\log^{0.5-\epsilon} N} \cdot |V_1|$ and ℓ_∞-cost $\geq 2^{\log^{0.5-\epsilon} N}$.*

The reduction has the property that every vertex in V_1 has the same degree. Furthermore, for each $e \in E$ and $b_1 \in B_1$ there is at most one $b_2 \in B_2$ such that $(e, b_1, b_2) \in \Pi$.

The proof of Lemma 9 appears in Section 6. For the reduction to SV_∞ we'll need to prove the hardness of a related (and not very natural) covering problem.

Definition 3 *Let (G, B_1, B_2, Π) be an instance of label cover and $(\mathcal{P}_1, \mathcal{P}_2)$ be a labelling. An edge $e = (v_1, v_2)$ is* untouched *by the labelling if $\mathcal{P}_1(v_1)$ and $\mathcal{P}_2(v_2)$ are empty sets. It is* cancelled *if $\mathcal{P}_2(v_2)$ is empty, $\mathcal{P}_1(v_1)$ is not empty, and for every $b_1 \in \mathcal{P}_1(v_1)$ there is an $b_1' \in \mathcal{P}_1(v_1)$ such that both (e, b_1, b_2) and (e, b_1', b_2) are in Π for some $b_2 \in B_2$.*

Definition 4 *A labelling $(\mathcal{P}_1, \mathcal{P}_2)$ is a* pseudo-cover *if it assigns a label to some vertex, and every edge is either untouched, cancelled or covered.*

Note that in a pseudo-cover the only case not allowed is that for some edge (v_1, v_2), $\mathcal{P}_1(v_1)$ is empty but $\mathcal{P}_2(v_2)$ is not.

One of our main theorems is that approximating the minimum ℓ_∞ cost of a pseudo-cover is hard.

Lemma 10 *In the label cover instances constructed in Lemma 9, if φ is satisfiable, there is a pseudo-cover with ℓ_∞ cost 1. If φ is not satisfiable every pseudo-cover has ℓ_∞ cost $2^{\log^{0.5-\epsilon} N}$.*

The first part of the lemma follows from Lemma 9, since a total-cover is also a pseudo-cover. The more difficult second half is left to Section 6.

5.2 Vectors for $\mathbf{NV}_1$ and $\mathbf{SV}_\infty$

Given an instance of label cover obtained from the previous section, we construct a corresponding set of vectors. Certain linear combinations of these vectors will correspond to total-covers or pseudo-covers.

First we simplify the structure of the relation Π in the instance of label cover of Lemma 9 by restricting the labels admissible at vertex $v_1 \in V_1$ to only *valid* labels. A label b_1 is *valid* for v_1 if, for every edge e incident to v_1, there is a label b_2 such that $(e, b_1, b_2) \in \Pi$; in other words, b_1 can be used to cover all edges incident to v_1. The reason for restricting attention to valid labels is that when φ is satisfiable, the label cover uses only 1 label for a vertex, so the label used must be valid. And if φ is not satisfiable, restricting the set of possible (vertex, label) pairs to be valid can only increase the minimum cost of a label cover.

The basis set contains a vector $V_{[v_i,b_i]}$ for each *valid pair* (v_i, b_i), i.e. where $v_i \in V_i$ and $b_i \in B_i$ $(i = 1, 2)$, and further if $i = 1$, b_1 is valid for v_1. Any linear combination of these vectors implicitly defines a labelling of the graph G, as follows.

Definition 5 *Let $x = \sum c_{[v_i,b_i]} \cdot V_{[v_i,b_i]}$ be a linear combination of the vectors in the set, where the coefficients $c_{[v_i,b_i]}$ are integers. The* labelling defined by the vector *x, denoted $(\mathcal{P}_1^x, \mathcal{P}_2^x)$, is $\mathcal{P}_i^x(v_i) = \{b_i \mid c_{[v_i,b_i]} \neq 0\}$, for $i = 1, 2$.*

Recall from Lemma 9 that for a fixed pair e, b_1, there is a unique label b_2 such that $(e, b_1, b_2) \in \Pi$. We denote such a b_2 by $b_2[e, b_1]$.

Each vector in our set has $|E|(1 + |B_2|)$ coordinates; $1 + |B_2|$ coordinates for each $e \in E$. The coordinates corresponding to e in a vector is referred to as its *e-projection*.

For $j = 1, 2, \ldots, |B_2|$, let u_j be a vector with $1 + |B_2|$ coordinates, in which the j'th entry is 1 and all the other entries are 0. With some abuse of notation we'll associate the vector u_{b_2} with the label $b_2 \in B_2$.

For $v_2 \in V_2, b_2 \in B_2$, the e-projection of the vector $V_{[v_2,b_2]}$ is u_{b_2} if e is incident to v_2; and $\vec{0}$ otherwise.

For each valid pair v_1, b_1, the e-projection of the vector $V_{[v_1,b_1]}$ is $\vec{1} - u_{b_2[e,b_1]}$ if e is incident to v_1; and $\vec{0}$ otherwise.

Here $\vec{0}$ and $\vec{1}$ are the all-zero vector and the all-one vector, resp. Notice that the e-projections of the vectors form a multi-set comprised of exactly one copy of the vector u_{b_2} for each $b_2 \in B_2$, and zero or more copies of the vector $\vec{1} - u_{b_2}$, plus multiple copies of $\vec{0}$.

In the reductions that follows we shall be interested in linear combinations of the vectors $\{V_{[v_i,b_i]}\}$ that sum to a multiple of $\vec{1}$ or to $\vec{0}$. In particular we will look at the e-projections of such sums, whose behavior is described by the following simple lemma.

Lemma 11 *Let X be the set of vectors $\{u_{b_2} | b_2 \in B_2\}$ and Y a multiset over $\left\{\vec{1} - u_{b_2} | b_2 \in B_2\right\}$. Let z be a linear combination of these vectors such that $z = \alpha\vec{1}$. Then for each $b_2 \in B_2$*

i. *If the coefficient of u_{b_2} is nonzero, there is some vector in Y of the form $\vec{1} - u_{b_2}$ with nonzero coefficient.*

ii. *If the coefficient of u_{b_2} is 0 then the number of vectors in Y of the form $\vec{1} - u_{b_2}$ with nonzero coefficients is either 0 or ≥ 2.*

Further, if $\alpha \neq 0$, then some vector in X has a nonzero coefficient in the combination.

Proof: The vectors $\left\{u_1, \ldots, u_{|B_2|}\right\}$ are linearly independent and do not span $\vec{1}$. Therefore if
$\sum_{b_2}(c_{b_2} \cdot u_{b_2} + d_{b_2} \cdot (\vec{1} - u_{b_2})) = \alpha\vec{1}$, then $c_{b_2} = d_{b_2}$ for all b_2. The claims follow. □

Corollary 12 *If x is a nontrivial linear combination of the vectors $\{V_{[v_i,b_i]}\}$ and $x = \alpha\vec{1}$, then $(\mathcal{P}_1^x, \mathcal{P}_2^x)$ is a pseudo-cover of G. If $\alpha \neq 0$, this pseudo-cover is a total-cover.*

Proof: For any edge e, the e-projections of the vectors $\{V_{[v_i,b_2]}\}$ form a system described in the hypothesis of Lemma 11. Then case (i) of the lemma corresponds to e being covered, and case (ii) to e being either cancelled or untouched. Thus $(\mathcal{P}_1^x, \mathcal{P}_2^x)$ is a pseudo-label cover. When $\alpha \neq 0$, then case (i.) holds for each edge (for some b_2), so each edge is covered. □

5.3 Hardness of Approximating NV_1

The set of vectors $\left\{V_{[v_i,b_i]}\right\}$ defined in Section 5.2 has the property that any linear combination of them that sums to $\vec{1}$ defines a total-cover. This fact is used in the following reduction.

Theorem 13 *There is a polynomial time transformation from the instance of label cover in Lemma 9 to an instance of NV_1 such that the optimum in the NV_1 instance has cost $\geq$ minimum ℓ_1-cost of a total-cover. Further, if there is a total-cover of ℓ_1-cost $|V_1|$, then the optimum is also $|V_1|$.*

Proof: Let L be the integer $|E| \cdot (1 + |B_2|)$. The vectors in the instance of NV_1 have $|E| \cdot (1 + |B_2|) + |V_1| \cdot |B_1|$ coordinates. The fixed point, W_0, has an L in each of the first $|E|(1 + |B_2|)$ coordinates and 0 elsewhere.

The basis of the lattice consists of a vector, $W_{[v_i,b_i]}$ for every valid (vertex,label) pair (v_i, b_i). In the first $|E| \cdot (1 + |B_2|)$ coordinates, the vector $W_{[v_i,b_i]}$ equals $L \cdot V_{[v_i,b_i]}$. We think of the last $|V_1| \cdot |B_1|$ coordinates as being identified one-to-one with a pair (v_1, b_1). Then the coordinate identified with (v_1, b_1) contains 1 in $W_{[v_1,b_1]}$ and 0 in all other vectors. This makes the last $|V_1| \cdot |B_1|$ coordinates suitable for counting, as shown in the next observation.

Claim: *Let $x = \sum c_{[v_i,b_i]} \cdot W_{[v_i,b_i]}$ be a vector in the lattice. Then $||-W_0 + x||_1 \geq$ ℓ_1-cost of $(\mathcal{P}_1^x, \mathcal{P}_2^x)$.*

The claim follows from the observation that any of last $|V_1| \cdot |B_1|$ coordinates of x that is not 0 corresponds to a label assigned by $\mathcal{P}_1^x$ to some node in V_1.

Now let OPT be the minimum ℓ_1 cost of a total-cover. We show that every vector x in the lattice satisfies $||-W_0+x||_1 \geq \min\{L, \mathrm{OPT}\}$. Notice that each entry of $W_0 - x$ in the first $|E|(1+|B_2|)$ dimensions is a sum of integer multiples of L. If it isn't 0, its magnitude is $\geq L$, and so $||-W_0+x||_1 \geq L$. On the other hand, if all those entries are 0 then, by Corollary 12, $(\mathcal{P}_1^x, \mathcal{P}_2^x)$ is a total-cover, and so by the above claim $||-W_0+x||_1 \geq \mathrm{OPT}$.

Finally, if there is a total-cover of ℓ_1-cost $|V_1|$, then the following vector has length $|V_1|$.
$x = -W_0 + \sum_{v_1 \in V_1} W_{[v_1, \mathcal{P}_1(v_1)]} + \sum_{v_2 \in V_2} W_{[v_2, \mathcal{P}_2(v_2)]}$. □

5.4 Hardness of Approximating $\mathbf{SV}_\infty$

The set of vectors $\left\{V_{[v_i,b_i]}\right\}$ defined in Section 5.2 has the property that any non-trivial linear combination of them that sums to $\vec{0}$ defines a pseudo-cover. This fact is used in the following reduction.

Theorem 14 *There is a polynomial time transformation from the instance of label cover in Lemma 10 to an instance of SV_∞ such that the solution to the SV_∞ instance is $\geq$ minimum ℓ_∞-cost of a pseudo-cover. Further, if there is a total-cover of ℓ_∞-cost 1, then the solution is 1.*

The reduction uses an $\ell \times \ell$ Hadamard matrix i.e. a (± 1) matrix such that $H_\ell^t H_\ell = \ell I_\ell$. ($H_\ell$ exists e.g. when ℓ is a power of 2, cf. [Bol, p.74]).

Claim: *Let $z \in \mathbf{Z}^\ell$. If z has at least k nonzero entries then $||H_\ell z||_\infty \geq \sqrt{k}$.*
Proof: The columns of $\frac{1}{\sqrt{\ell}} H_\ell$ form an orthonormal basis. Hence $||\frac{1}{\sqrt{\ell}} H_\ell z||_2 = ||z||_2 \geq \sqrt{k}$. □

Proof:(*of Theorem 14*) Let L be the integer $|E|\,|B_2|$. The vectors in the lattice have $|E| \cdot (1+|B_2|) + |V_1| \cdot |B_1|$ coordinates. The basis of the lattice consists of a vector $W_{[v_i,b_i]}$ for each each valid (vertex, label) pair v_i, b_i, and an additional vector, W_0, that has L in each of the first $|E| \cdot (1+|B_2|)$ coordinates and 0 elsewhere.

As in the last reduction, $W_{[v_i,b_i]}$ will equal $L \cdot V_{[v_i,b_i]}$ in the first $|E| \cdot (1+|B_2|)$ coordinates. The remaining $|V_1| \cdot |B_1|$ coordinates will be viewed as blocks of $|B_1|$ coordinates, each associated with a $v_1 \in V_1$. We refer to entries in the block associated with v_1 as v_1-*projection* of the vector.

We may assume there exists a Hadamard matrix H_ℓ for $\ell = |B_1|$. With each label $b_1 \in B_1$ we identify a unique column vector of H_l, denoted h_{b_1}. Then the v_1-projection of $W_{[v,b]}$ is h_b if $v = v_1$ and $\vec{0}$ if $v \neq v_1$.

Let OPT be the minimum ℓ_∞-cost of a pseudo-cover. For any vector x in the lattice, the entry in any of the first $|E|(1+|B_2|)$ coordinates is a sum of integer multiples of L, so if it is not 0, its magnitude is $\geq L$, and hence $\geq$ OPT. So all these entries must be 0. But then by Lemma 12, we conclude that the labelling defined by x is a pseudo-cover, and must therefore assign $\geq$ OPT labels to some $v_1 \in V_1$. But then $||x||_\infty \geq \sqrt{\mathrm{OPT}}$ by the Claim.

Now suppose OPT = 1 and $(\mathcal{P}_1, \mathcal{P}_2)$ achieves it. Then the following vector has ℓ_∞ norm 1.

$$-W_0 + \sum_{v_1 \in V_1} W_{[v_1, \mathcal{P}_1(v_1)]} + \sum_{v_2 \in V_2} W_{[v_2, \mathcal{P}_2(v_2)]}.$$

$\square$

6 Hardness of Label Covers

In this section we prove Lemmas 9 and 10. We use the fact ([FL]) that every language in NP has an efficient 2-Prover 1-Round Interactive proof system (such systems are defined next). We use such a proof system to construct an instance of label cover. Every low cost solution to that instance yields a strategy (i.e., a way to answer questions) for provers with a high chance of satisfying the verifier. The large "gap" in acceptance probability present in the definition of interactive proofs then translates into a large gap in the minimum cost solution to label cover. For exhibiting the hardness of l_∞ cost, we have to modify the protocol of [FL] a little, and use some new geometric arguments.

6.1 2-Prover (and Multi-Prover) 1-Round Interactive Proofs

A 2-Prover 1-Round interactive proof system for a language L consists of a probabilistic polynomial-time verifier V. The verifier interacts with two nontrustworthy provers, which are deterministic turing machines with unlimited computational power whose task is to convince the verifier that the input, x, belongs to L.

For a fixed input, there are five sets associated with the verifier: Q_1, Q_2, R, A_1, A_2. The verifier itself consists of three polynomial-time functions q_1, q_2, and Ψ. For $i = 1, 2$, the function q_i maps R to Q_i and describes the verifier's *query generation*. The predicate $\Psi : R \times A_1 \times A_2 \rightarrow \{0, 1\}$ describes the verifier's *acceptance conditions*. The provers are arbitrary functions $P_i : Q_i \rightarrow A_i$.

The verifier begins the protocol by picking an $r \in R$ uniformly at random (this is its "random seed"). Then it computes the query $q_i[r] \in Q_i$ and sends it to prover P_i $(i = 1, 2)$. Prover P_i responds with $a_i = P_i(q_i[r])$. Finally, the verifier decides whether to accept or reject according to the value $\Psi(r, a_1, a_2)$.

It is implicit in [FL] that for every polynomial-time computable function $k(n) < poly(n)$, there is a 2-prover 1-round proof-system for 3SAT with the following properties:

- If $\varphi \in$ 3SAT then there exist provers such that the verifier always accepts (i.e., for every choice of r).
- If $\varphi \notin$ 3SAT then for any pair of provers the verifier accepts with probability at most $2^{-k(n)}$ (where n = size of φ).
- $|R|, |Q_i|, |A_i| \leq 2^{f(k(n), \log(n))}$, where f is a suitable bivariate polynomial.

We give a specific protocol in section 6.4 with $f(k(n), \log(n)) = k^2(n) \log^2(n)$. Further, it is a feature of this protocol that for a fixed $r \in R$ and a fixed $a_1 \in A_1$, there is at most one $a_2 \in A_2$ such that the verifier accepts, i.e. $\Psi(r, a_1, a_2) = 1$.

Multi-Prover 1-Round proof systems. It is easy to generalize the above definitions, by allowing the verifier to interact with more than two provers, while still restricting it to one round of queries. The systems thus obtained are called Multi-Prover 1-Round proof systems; this concept was first defined in [BGKW88].

6.2 From Interactive Proofs to Label Cover

We give a generic way to construct instances of label cover using 2-Prover 1-Round interactive proof system. The running time of the construction is polynomial in the running time of the verifier and $|R| + |A_1| + |A_2|$. Since 3SAT has efficient 2-Prover 1-Round interactive proof systems, this construction provides a way to reduce 3SAT to label cover. Both Lemmas 9 and 10 use this reduction, and we use properties of the our very specific proof system to argue about the reduction's correctness in each case.

For a 3SAT instance, φ, let R, Q_i, A_i, $i = 1, 2$ be the sets associated with the interative proof protocol. Then the graph of the label cover instance is (V_1, V_2, E) where $V_i = Q_i$, for $i = 1, 2$ and $E = \{(q_1[r], q_2[r]) :\ r \in R\}$. For $i = 1, 2$, the set of labels B_i is A_i. The relation Π is exactly the predicate Ψ computed by the verifier, i.e.

$$\Pi = \{(e, b_1, b_2) : (\exists r)(e = (q_1[r], q_2[r]) \& \Psi(r, b_1, b_2) = 1\}).$$

6.3 Hardness of Total Covers

Now we prove Lemma 9.

Proof:(*Of Lemma 9*) We know that 3SAT has a 2-Prover 1-Round proof system of the type described in Section 6.1. (We will specify the parameter $k(n)$ for the proof system later.) Now let φ be any 3CNF formula. Using the reduction in Section 6.2, transform it into an instance of Label Cover.

If $\varphi \in$ 3SAT, there exist provers P_1 and P_2 that make the verifier accept with probability 1 (i.e. for every choice of r). Since a prover P_i is an assignment of *one* answer to each query, and this assignment makes the verifier accept for every random integer r (equivalently, for every edge in the above graph), we conclude that (P_1, P_2) form a total-cover with l_1-cost of $|Q_1|\ \ (= |V_1|)$ and ℓ_∞-cost 1.

To prove the other half of the lemma, assume $\varphi \notin$ 3SAT. So no provers can satisfy the verifier with probability more than $2^{-k(n)}$. In terms of the label cover instance, this means that no labelling using at most 1 label per vertex can cover more than a fraction $2^{-k(n)}$ of the edges. Now we show that no labelling that uses less than $2^{k(n)-1} \cdot |V_1|$ labels (i.e., less than $2^{k(n)-1}$ labels per vertex on average) can cover more than $\frac{1}{2}$ the edges. For, suppose $(\mathcal{P}_1, \mathcal{P}_2)$ is such a labelling. Let G' be the subgraph of G consisting of all the edges covered by $(\mathcal{P}_1, \mathcal{P}_2)$.

Pick a new labelling $(\mathcal{P}_1', \mathcal{P}_2')$ randomly by choosing, for each vertex $v_i \in V_i$, a label uniformly at random from $\mathcal{P}_i(v_i)$. Note that $(\mathcal{P}_1', \mathcal{P}_2')$ uses only one label per vertex. Furthermore, if $(v_1, v_2) \in G'$, then the probability that $(\mathcal{P}_1'(v_1), \mathcal{P}_2'(v_2))$ is a matching pair of labels is $\geq \frac{1}{|\mathcal{P}(v_1)|}$. Therefore the expected number of edges covered by $(\mathcal{P}_1', \mathcal{P}_2')$ is

$$\geq \sum_{v_1 \in V_1} \frac{\text{degree}_{G'}(v_1)}{|\mathcal{P}(v_1)|}.$$

It is easily seen that this expression is minimized when $\text{degree}_{G'}(v_1) = A \cdot |\mathcal{P}(v_1)|$, where

$$\begin{aligned} A &= \frac{\sum_{v_1 \in V_1} \text{degree}_{G'}(v_1)}{\sum_{v_1 \in V_1} |\mathcal{P}(v_1)|} \\ &\geq \frac{|E|/2}{2^{k(n)-1}|V_1|}. \end{aligned}$$

Thus the expected number of covered edges is at least $A \cdot |V_1|$, which is at least $(|E|/2)/2^{k(n)-1}$. In particular, this implies the *existence* of a labelling $(\mathcal{P}_1', \mathcal{P}_2')$ that uses 1 label per vertex and covers at least $|E|/2^{k(n)}$ edges. This is a contradiction. Hence our assumption that there is a labelling that covers 1/2 the edges and has ℓ_1-cost $< 2^{k(n)-1} \cdot |V_1|$ must be wrong.

Finally, note that every labelling of ℓ_1-cost $2^{k(n)-1} \cdot |V_1|$ must assign at least $2^{k(n)-1}$ labels to some vertex in V_1, which means that the l_∞-cost is $\geq 2^{k(n)-1}$.

To finish the proof of Lemma 9, we decide upon the value of the parameter $k(n)$. Note that the size of the label cover instance N, is $2^{(k(n))^2 \log^2 n}$. The gap between the costs of the label cover in the two cases is $2^{k(n)-1}$. Hence choosing $k(n) = \log^{\frac{3}{4\epsilon}} n$ yields the desired gap of $2^{\log^{0.5-\epsilon} N}$.

□

6.4 A Specific 2-Prover, 1-Round Proof Protocol for SAT

The proof of Lemma 10 will rely upon the properties of a specific 2-Prover, 1-Round proof system for 3SAT. For ease of exposition, we develop the verifier in three stages. Each stage consists in constructing a Multi-Prover 1-Round proof system for SAT. These proof systems are called PP1, PP2 and PP3 respectively; out of these PP3 will be used to construct label cover instances.

PP1:
This is a 2-Prover 1-Round proof system, and its error probability is some fixed constant less than 1.

In [AL+] it is shown that there is some constant $0 < b < 1$, such that for any 3CNF formula, φ, there is a "robust" 3CNF formula, φ', constructible in polynomial time, such that:

- φ is satisfiable if and only if φ' is satisfiable.
- If φ' is not satisfiable, then any truth assignment satisfies at most a fraction b of its clauses.

The PP1 protocol proceeds as follows on a 3CNF formula φ. The verifier transforms φ to its robust analogue φ'. Then it selects a clause, say c, at random from among all clauses in φ' and a variable, say x, at random from all variables in c. It passes c to prover P_1 and x to prover P_2. Prover P_1 returns at most 3 bits that represent a truth assignment for the variables occurring in c. Prover P_2 returns a truth value for x. The verifier accepts iff the truth assignments are consistent (agree on x) and satisfy the clause c. Notice that prover P_2 is a deterministic function from variables to boolean values; hence it can be viewed as a truth assignment.

If φ is satisfiable then the provers can convince the verifier to accept with probability 1, by answering according to the same satisfying assignment of φ', e.g. the first in lexicographic order. If φ is not satisfiable, the truth assignment according to which prover P_2 answers can satisfy at most a b fraction of clauses in φ'. Each clause corresponds to 3 queries to $\mathcal{P}_1$, and if the clause is no satisfied, at least one of those queries will result in a reject. So the verifier will reject with probability at least $\frac{1-b}{3}$.

PP2:

This protocol uses many provers and 1 round, but has an error probability of $2^{-k(n)}$. Also, the verifier's set of queries has been suitably extended with dummy questions so that some geometric properties (defined later) are satisfied.

First we modify the verifier in PP1. Assume, by padding queries with unnecessary bits if need be, that each query in PP1 is $d = O(\log(n))$ bits long. For $i = 1, 2$, let Q_i be the queries for prover P_i, and let $Q \subseteq Q_1 \times Q_2$ contain all pairs of queries that the verifier could actually make. Let D denote the following set of "dummy" queries: $\left\{w \in \left\{0,1\right\}^d : w \text{ has exactly one } 1\right\}$. The property of D relevant later is it is a basis for $\{0,1\}^d$ over $GF(2)$.

Now modify PP1 so that the verifier selects a query at random from the set:

$$Q \quad \cup \quad Q_1 \times D \quad \cup \quad D \times Q_2 \quad \cup \quad D \times D$$

If the pair of queries is not in Q, the verifier accepts regardless of the provers' responses. If the pair is in Q, the verifier follows the PP1 protocol. Note that the verifier chooses a query from Q with probability $\geq \frac{1}{d^2}$. Therefore, the probability it rejects an unsatisfiable formula is at least $\frac{1-b}{3d^2}$.

Finally, the Multi-Prover proof system PP2 is obtained by repeating the verifier's interaction with m independent pairs of provers in parallel, where $m = O(k(n)d^2)$. The probability of accepting an unsatisfiable formula goes down to $2^{-2k(n)}$.

PP3:

This is a 2-Prover 1-Round protocol which is a parallelization of PP2 using the techniques of [FL].

Let F be a finite field, $|F| \geq 2^{O(k(n))}$. (Here we assume $k(n) = \mathrm{poly}(\log(n))$). For any input φ, and fixed strategy of the m provers, let $p_i : F^d \to F$ be the unique multilinear function in d variables such that for any $q \in \{0,1\}^d$, $p_i(q)$ is the answer provided by prover i on query q.

The verifier, V, will simulate the verifier of PP2 and construct queries for each of the m provers, $q_1, \ldots, q_m$. Note that each q_i can be viewed as a point in F^d. Next, for each i, the verifier, V, chooses a random line, l_i, in F^d that passes through the point q_i. These m lines are provided to prover P_1. The verifier also selects a random point, t_i, on each line, l_i, and sends them to prover P_2.

Prover P_1 returns m degree-d polynomials which are supposed to represent, for each i, the function p_i parameterized along the line l_i. Prover P_2 responds with m values from F which are supposed to represent, for each i, the function p_i evaluated at the point t_i.

The verifier, V, evaluates each of the i polynomials at the corresponding query point, q_i, and checks that these answers would have been accepted by the verifier in PP2. V also checks that the value of the ith polynomial at the random point t_i agrees with the value provided by prover P_2. If all of these checks pass, the verifier accepts.

If φ is satisfiable, then there is a way for the m provers of the PP2 protocol to answer queries such that the verifier accepts with probability 1. Clearly, the same remains true for PP3. In [FL] the converse is also shown: If for every strategy of the m provers of PP2, the verifier accepts with probability less than $2^{-2k(n)}$, then for every strategy of the 2 provers of PP3, the verifier accepts with probability less than $2^{-k(n)}$.

6.5 Hardness of Pseudo-Covers

Now we prove Lemma 10. Using the protocol PP3 in Section 6.4, we reduce instances of 3SAT to label cover as described before. As argued before, if the underlying 3SAT formula is satisfiable, there exists a labelling using only 1 label per vertex and which covers all edges. The proof of Lemma 10 is completed by the next claim.

Claim: *In the instances of label cover defined using PP3, let OPT be the minimum ℓ_∞-cost of a labelling that covers half of the edges and F be the field used by the verifier. Then the minimum ℓ_∞-cost of a pseudo-cover is* $\min\left\{\sqrt{|F|}, OPT\right\}$.

The proof of the Claim divides into two parts. The first part considers pseudo-covers that, for some node in V_1, cancel a "large" fraction of edges incident to that node. Lemma 15 shows that such pseudo-covers have ℓ_∞ cost at least $\sqrt{|F|}$. The second part, Lemma 16, considers all other pseudo-covers, and shows that they must cover at least 1/2 of all edges. Hence their ℓ_∞ cost is at least OPT.

Lemma 15 *In the instances of label cover defined using PP3, a pseudo-cover of l_∞-cost at most $\sqrt{|F|}$ cancels no more than $\frac{d}{\sqrt{|F|}}$ fraction of the edges incident to any vertex in V_1. Here d is the upperbound on the degree of the univariate polynomials returned by the provers.*

Proof: Let $\hat{p} = (p_1(x_1), \ldots, p_m(x_m))$ be a label in A_1 that is assigned by a pseudo-cover to a node q_1. If the edge, $e = (q_1[r], q_2[r])$, incident to q_1 is cancelled by the labelling then there must be a distinct label, $\hat{p}' = (p'_1(x_1), \ldots, p'_m(x_m))$, assigned to q_1 such that $\hat{p}'(q_2[r]) = \hat{p}(q_2[r])$; i.e. these m-tuples of polynomials agree on the m-tuple of points $q_2[r]$. But $\hat{p}'(x)$ and $\hat{p}(x)$ are distinct m-tuples of polynomials of degree at most d and thus agree at no more than $d/|F|$ fraction of the points in F^m. Consequently, two labels can cancel at most a $d/|F|$ fraction of edges incident to q_1 and $\sqrt{|F|}$ are necessary to cancel a $d/\sqrt{|F|}$ fraction. $\square$

Lemma 16 *In the label cover instance obtained from protocol PP3, a pseudo-cover that cancels at most a $\frac{d}{\sqrt{|F|}}$ fraction of the edges at any vertex in V_1 must cover at least $\frac{1}{2}$ of the edges.*

To prove Lemma 16, we prove an expansion-like property of the bipartite graph in the label cover instance. Consider a pseudo-cover that cancels at most a fraction $\alpha \leq \frac{d}{\sqrt{F}}$ fraction of edges of any node in V_1. For a node v, let $\Gamma(v)$ denote the neighbors of v that are connected to it by a covered edge. Notice, if the pseudo-cover cancels no edge, then it is a total cover and so has ℓ_∞ cost at least OPT. So assume w.l.o.g. that it cancels at least one edge, incident to, say, node $v_1 \in V_1$. But then $\Gamma(v_1)$ contains at least $1-\alpha$ of the neighbors of v_1 (since by definition of pseudo-cover, once it has assigned a label to v_1, it has to either cancel or cover every edge incident to v_1, and it can cancel only a fraction α). So the pseudo-cover needs to assign labels to all vertices in $\Gamma(v_1)$, as well as to vertices in $\cup_{v_2 \in \Gamma(v_1)} \Gamma(v_2)$. The next lemma shows that for every "small" set $S \subseteq V_1$, the set $\Gamma(V_1)$ is "big," so the pseudo-cover must actually assign labels to many vertices.

Lemma 17 *(An Expansion Lemma) In the label cover instances obtained from PP3, let B be a proper subset of V_2 s.t. every vertex $v_1 \in V_1$ either has all its neighbors in B or at most an α fraction of its neighbors in B. Then $\geq (1-\beta)$ fraction of the vertices in V_1 have a neighbor in $V_2 \backslash B$, where $\beta = m/|F| + 2(d-1)\alpha$.*

Lemma 17 relies on geometric properties of PP3 protocol for 3SAT. It is restated in geometric terms and proven in Section 6.6. Now we prove Lemma 16

Proof: (*of Lemma 16*). Let B be the set of nodes in V_2 that are *not* assigned labels by the pseudo-cover. Note that if $v_1 \in V_1$ is adjacent to a node in $V_2 \setminus B$, it must be assigned a label. Further, at most an α fraction of its edges can be incident to B.

Lemma 17 implies that at most $(1-\beta)(1-\alpha)$ edges must be covered by any pseudo-cover that cancels at most an α fraction of edges of any V_1 node. Hence Lemma 16 follows by letting $\alpha = \frac{d}{\sqrt{F}}$, and noticing that both α and β are $o(1)$, since $d = O(\log(n))$, $m = O(k(n)\log n)$, $|F| = 2^{O(k(n))}$ and $k(n) = \mathrm{poly}(\log n)$. $\square$

6.6 A Proof of the Expansion Lemma

We prove Lemma 17 in this section by stating and proving an equivalent geometric fact, Lemma 19. But first we formalize the geometric structure of the protocol PP3.

6.6.1 A Geometric View of PP3

This section deals with two kinds of spaces. The first, denoted W_0, is F^d where field F and integer d are the same as in the description of PP3. The second space, denoted W, is $W_1 \times \cdots \times W_m$, where $W_i \cong W_0$. We will define points, lines, and hyperplanes for both spaces; to avoid confusion, names of objects belonging to W are written with a capitalized letter. For example, a *point* is an element of W_0, and a *Point* is an element of W.

An *n-dimensional affine subspace of* W_0 is a set of $|F|^{n-1}$ points that can be described as

$$\left\{ \sum_{i=1}^{n} \lambda_i u_i : \forall i\ \lambda_i \in F \text{ and } \sum_{i=1}^{n} \lambda_i = 1 \right\},$$

for some $\{u_1, \ldots, u_n\} \subseteq W_0$. This set $\{u_1, \ldots, u_n\}$ is called a *basis* of the subspace. Note that W_0 is a $(d+1)$-dimensional subspace of itself.

A *line* in W_0 is a 2-dimensional affine subspace, i.e., a set of points of the type $\{\lambda \hat{u} + (1-\lambda)\hat{v} : \lambda \in F\}$, for some $\hat{u}, \hat{v} \in W_0$. A *Line* is an ordered m-tuple of lines. Thus a Line has $|F|^m$ Points. Likewise, we define an *n-dimensional hyperplane* as a n-dimensional affine subspace of W, and am *n-Hyperplane* as an ordered m-tuple of n-dimensional hyperplanes.

The geometric structure of the PP3 protocol is as follows. There is a set S of Points that are *special* for the verifier (these correspond to the questions picked uniformly at random in the PP2 protocol). A Line that contains a special Point will be called a *special Line*.

The set of queries Q_1 to prover P_1 are exactly the special Lines. The set of queries Q_2 to prover P_2 are exactly the Points.

The set of random seeds is $R = \{(q_1, q_2) : q_1 \text{ is a special Line containing the Point } q_2\}$.

Thus, in the label cover instance obtained from PP3, nodes in V_1 and V_2 correspond, respectively, to special Lines and Points. Further, adjacency in this bipartite graph corresponds to incidence in the affine space.

We say that a set $N \subseteq W_i$ is *full-dimensional* if it contains an affine basis of W_i. A *box* is a set $N = \prod_i N_i$ where each $N_i \subseteq W_i$. The box N is said to be *Full-Dimensional* in W if each N_i is full-dimensional in the corresponding W_i. Finally, a set $S \subseteq W$ is *Totally Full-Dimensional* if for every $s \in S$ there is some Full-Dimensional box $T \subseteq S$ such that $s \in T$.

Since the set of questions in protocol PP2 was extended (with "dummy" questions) to contain a basis of F^d, and protocol PP3 is merely its parallelization, the following lemma is immediate.

Lemma 18 *The special Points of the PP3 protocol are Totally Full-Dimensional.*

□

6.6.2 A Geometric Lemma

Using the observation in Lemma 18, we restate Lemma 17, thus making its geometric nature explicit.

Let the Points of W be colored white and blue. We'll say a Line is all-blue if every Point on it is blue.

Lemma 19 *Let $S \subset W$ be a set of special Points that is Totally Full-Dimensional. Suppose every special Line is either all-blue or has at most an α fraction of blue points. If there is at least one white Point, then at least a $(1-\beta)$ fraction of the special Lines must have a white Point, where $\beta = m/|F| + 2(d-1)\alpha$.*

The proof of the lemma is easier to understand when $m = 1$. In this case, $W = W_0$ so Points are points, Lines are lines and Hyperplanes are hyperplanes. Further, the condition that S is Totally Full-Dimensional is equivalent to requiring that S contain an affine basis of W. We will first prove the lemma when $m = 1$ and then generalize the result.

6.6.3 Proof of the Geometric Lemma: $m = 1$

Proof of Lemma 19: $m = 1$

Let B be the set of blue points. Assume that less than a $(1-\beta)$-fraction of special Lines have a white Point. We'll show that $B = W$.

There must be a Point $u \in S$ such that the proportion of all-blue Lines through u is at least β. Let $x \neq u$ be a Point in W. Now S contains an affine basis of W, so it contains an affine basis of a d-dimensional Hyperplane U that includes u but not x. From now on, we abbreviate d-dimensional Hyperplane as just Hyperplane.

Call a Line *very special* if it passes through u but not through any other Point of U. Then the proportion of all-blue lines among the very special Lines is at least $\beta - m/|F| = 2(d-1)\alpha$.

By the following Claim, $W \setminus U \subseteq B$. But the point x was arbitrary. Thus $W \setminus \{u\} \subseteq B$. But every Line through u must be all-blue, thus $u \in B$. □

Claim. *Assume S contains an affine basis of a Hyperplane U of W and $u \in U$. Assume further that the proportion of all-blue Lines among the very special Lines is greater than $2(d-1)\alpha$. Then $B \supseteq W \setminus U$.*

Proof of the Claim: Use induction on d. When $d = 2$, W is a single Line and the claim holds.

Now assume $d \geq 3$. Let V be a Hyperplane of U and let $s \in S \cap U$ such that $s \notin V$. An example for the case when $d = 3$ is shown in Figure 1; i.e. U is a Line and V is a Point. The reader may refer to this figure to get the intuition for what follows.

Let $\mathcal{Q}$ be the set of Hyperplanes that contain V (one of these is U). Then the sets $Q \setminus V$, where $Q \in \mathcal{Q}$, partition $W \setminus V$. On the other hand, a refinement of this partition is provided by the sets $\lambda \cap (W \setminus V)$ for Lines λ joining u to points in $W \setminus V$.

Call a $Q \in \mathcal{Q}$ almost-blue if $Q \setminus V \subseteq B$. Note that by the inductive hypothesis, if $Q \in \mathcal{Q}$ is not almost-blue then the proportion of all-blue Lines among the very special Lines in $\mathcal{Q}$ is $\leq 2(d-2)\alpha$.

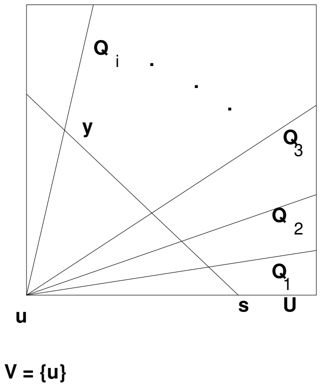

Figure 1: The case d= 3.

Let η denote the proportion of almost-blue Hyperplanes in $\mathcal{Q}$. If $\eta \leq 2\alpha$, then the proportion of all-blue Lines among the very special lines is $\leq 2(d-2)\alpha + 2\alpha = 2(d-1)\alpha$, contradicting the assumption in the Sublemma. Thus $\eta > 2\alpha$.

Now let $y \in W \setminus U$ and consider the (unique) Line λ joining y to s. Each $Q \in \mathcal{Q}$ intersects λ in exactly one point; therefore more than a 2α fraction of $\lambda \cap W \setminus U$ is blue. But more than $1 - m/|F|$ fraction of λ lies in $W \setminus U$; so at least a $(1 - m/|F|)2\alpha > \alpha$ fraction of λ is blue. Consequently λ is all-blue; in particular, $x \in B$. □

6.6.4 Proof of the Geometric Lemma: $m > 1$

To prove the result for general m we want, simply, to carry out the previous proof in parallel in each of the m dimensions.

As before, there is a Point $u = (u_1, \ldots, u_m) \in S$, (where $u_i \in W_i$) such that the proportion of all-blue Lines through u is at least β. Let $x = (x_1, \ldots, x_m) \neq u$ be a Point in W. Now S is Totally Full Dimensional, so it contains a Full-Dimensional Box, $T = \prod T_i$ containing u; in other words, for each i, T_i is an affine basis for W_i and $u_i \in T_i$. Then clearly, for each i there is a $T_i' \subset T_i$ such that $u_i \in T_i'$, T_i' is the affine basis of a hyperplane U_i, and $\prod T_i' \subseteq T \subseteq S$. Therefore, S contains a Full Dimensional box of the Hyperplane $U = \prod U_i$ and $u \in U$.

The proof follows exactly as before from the generalized Claim.

General Claim. *Assume S contains a Full-Dimensional box of a Hyperplane U of W and $u \in U$. Assume further that the proportion of all-blue Lines among the very special Lines is greater than $2(d-1)\alpha$. Then $B \supseteq W \setminus U$.*

The proof, again, follows by induction on d. When $d = 2$ the Space is a single Line and the claim hold.

Now assume $d \geq 3$. Since S contains a Full Dimensional Box of U, we can find a Hyperplane V of U that contains u and a Point $s \in S \cap U$ such that $s \notin V$. (Using the same kind of argument as used above.)

The remainder of the proof is identical to the case for $m = 1$.

7 Hardness of MAX-SATISFY

This section proves Theorem 3 about the hardness of approximating MAX-SATISFY.
Proof: (*of Theorem 3*) We first show that approximating the problem within a factor $(1+\epsilon)$ is NP-hard, and then prove the stronger result.

We use the following result, a joint consequence of a theorem of Arora et. al. [AL+] and a reduction due to [PY]. A *vertex cover* in a graph $G = (V, E)$ is a set $V' \subseteq V$ such that for every edge $\{u, v\}$, either $u \in V'$ or $v \in V'$. Let VC_{min} be the size of the smallest vertex cover.

Theorem 20 *There exists fixed constants c, ϵ and a polynomial time reduction from a SAT instance φ to a graph with n vertices, $m = O(n)$ edges and degree 5, such that if φ is satisfiable then $VC_{min} = cn$, and if φ is not satisfiable then $VC_{min} \geq (1+\epsilon)cn$.*

We will reduce the vertex cover instance to a system of $N = n + 3m$ linear equations, of which at most $2m + n - VC_{min}$ can be simultaneously satisfied. This implies a gap of $\Theta(N)$ in the optimum in the two cases, since $m = O(n)$.

For each vertex i we'll construct a variable x_i and an equation $x_i = 0$. For each edge, $\{i, j\}$, we'll construct 3 equations:

$$\begin{aligned} x_i + x_j &= 1 \\ x_i &= 1 \\ x_j &= 1 \end{aligned}$$

Notice, at most 2 of the 3 equations for each edge can be satisfied simultaneously. Further, to satisfy 2 of these equations, x_i and x_j must take on values from $\{0, 1\}$ and at least one must take the value 1.

We claim that the maximum number of equations are satisfied when all the x_i's are 0/1. Suppose, under some assignment, x_i is not 0/1. We can strictly increase the number of equations satified by resetting the value of x_i as follows: If $x_i < \frac{1}{2}$ then set $x_i = 0$. If $x_i \geq \frac{1}{2}$ then set $x_i = 1$.

Now notice that under any optimal assignment, the set of vertices $\{i : x_i = 1\}$ constitutes a vertex cover. If not, then there must be be an edge $\{i, j\}$, such that both x_i and x_j are 0. Thus all three equations associated with this edge are unsatisfied. Resetting x_i to 1 will satisfy 2 equations associated with this edge, and violate one equation, $x_i = 0$, which was previously satisfied. Thus there is a a net gain of 1 equation satisfied, which contradicts the optimality of the original assignment. It follows that the optimum assignment satisfies $2m + n - VC_{min}$ equations.

Boosting the gap. We have a system of N equations, in which the answer to MAX-SATISFY is either OPT or OPT$\cdot(1-\delta)$ for some $\delta > 0$. Let the equations be written as $p_1 = 0, p_2 = 0, \ldots, p_N = 0$. First we introduce a technique to increase this gap to any arbitrary constant, and then to n^a for some $a > 0$.

Let k, T be integers (to be specified later). Construct a new set of equations containing, for every k-tuple of old equations, $p_{i_1}, \ldots, p_{i_k}$, a set of T equations $\sum_{j=1}^{k} p_{i_j} y^j = 0$, where $y = 1, 2, \ldots, T$. Thus the number of equations becomes $\binom{N}{k} \cdot T$.

Whenever an assignment to the variables does not satisfy the equations $p_{i_1}, \ldots, p_{i_k}$, the polynomial $\sum_{j \leq k} p_{i_j} y^j$ is non-zero. Since a polynomial of degree k has at most k roots, it follows that any such $p_{i_1}, \ldots, p_{i_k}$ gives rise to at least $T - k$ unsatisfied equations in the new system. It follows that the number of equations that can be satisfied in the two cases is either $\geq \binom{OPT}{k} \cdot T$ or $\leq \binom{N}{k} \cdot k + \binom{OPT(1-\delta)}{k} \cdot T$. If we choose $T \geq N^{k+1}$, we see that the gap between the optima in the two cases is approximately $(1-\delta)^k$. Fixing k to be a large enough constant, we get the result that any constant factor approximation is NP-hard. When we fix k to be nonconstant, the size of the new system becomes superpolynomial, and the reduction no longer runs in polynomial time.

So use an idea due to [AF+]. Instead of constructing the new system by using the set of k-tuples of $\{1, \ldots, N\}$ for $k = \theta(1)$, use the set of all random walks of length $k = O(\log N)$ on a constant-degree Ramanujan graph with N vertices. (That is, if $i_1, \ldots, i_k$ is the sequence of vertices encountered on the walk, then construct T equations for this tuple just as above.) Let M be the number of such walks. Since $M = poly(N)$, the reduction still runs is polynomial time. Further, as shown in [AF+], for every set $A \subseteq \{1, \ldots, N\}$, there is a constant $\alpha > 0$ (which can be made arbitrarily small by taking a good enough expander) such that the fraction of walks lying entirely in A is between $(|A|/n - \alpha)^k$ and $(|A|/n + \alpha)^k$.

Thus in the new system of equations, the maximum number of satisfiable equations is $\geq M(OPT/N - \alpha)^k \cdot T$ or $\leq M(OPT(1-\delta)/N + \alpha)^k \cdot T$. Since $OPT/N = 1/3$ and α can be made arbitrarily small, say $\delta/100$, we can choose $T = M^2$ and get a system with M^3 equations and a gap of approximately $(\frac{1-\delta}{3})^k$. Since $k = O(\log n) = O(\log M)$, we see that the the gap is M^b for some small $b > 0$.

$\square$

8 Conclusions

The Label Cover problem introduced by us has subsequently been used in other papers such as [AK95]. It also appears as a "canonical" inapproximable problem in the survey [AL96].

This paper leaves some interesting problems open. As mentioned earlier, hardness for NV_p or SV_2 within a factor of $\sqrt{\text{dimension}}$ would prove the hardness of SV_2, which is a major open problem. It seems that the best conceivable factor achievable using interactive proofs/PCP may be n^c for some small $c > 0$. But we feel that the geometric facts used in our reduction to SV_∞ might be indicative of the nature of relevant techniques.

A related open problem is to improve our results by proving the problems NP-hard rather than almost-NP-hard. More efficient 2-prover, 1-round interactive proofs for NP (as conjectured in [BG+]), or a direct reduction from [AL+] might help.

Acknowledgements

Thanks to Madhu Sudan for suggesting that techniques from interactive proofs might be helpful in proving the hardness of lattice vector problems, and to Ravi Kannan for his prompt responses to questions on lattices. We also thank Manuel Blum, Tomas Feder, Dick Karp, Mike Kearns, Mike Luby, Ron Rivest and Umesh Vazirani for helpful discussions.

References

[AF+] N. Alon, U. Feige, A. Wigderson, D. Zuckerman. *Derandomized Graph Products Manuscript, 1993.*

[AK93] E. Amaldi and V. Kann. The complexity and approximability of finding maximum feasible subsystems of linear relations. *Theoretical Comp. Sc.* 147:187–210, 1995. An earlier version is in Proc. of STACS, 1994.

[AK95] E. Amaldi and V. Kann. On the approximability of some NP-hard minimization problems for linear systems. Technical Report 95-7, Cornell Computational Optimization Project, Cornell University, Ithaca, NY.

[A96] S. Arora. Inapproximability result for Nearest Vector Problem in ℓ_∞ norm. Unpublished note, 1996.

[AL96] S. Arora and C. Lund. Hardness of Approximations. Survey chapter in *Approximation Algorithms for NP-hard Problems*, editor D. Hochbaum. PWS Publishing, Boston, 1996.

[AL+] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy. Proof verification and intractability of approximaion problems. In *Proc. 33rd IEEE Symp. on Foundations of Computer Science*, pages 13–22, 1992.

[AS] S. Arora, S. Safra. Probabilistic Checking of Proofs: A New Characterization of NP. In *Proc. 33rd IEEE Symp. on Foundations of Computer Science*, pages 2–13, 1992.

[Bab] L. Babai. On Lovász's lattice reduction and the nearest lattice point problem. In *Combinatorica 6:1-14*, 1986.

[BF+] L. Babai, L. Fortnow, L. Levin, M. Szegedy. Checking Computations in Polylogarithmic Time. *Proc. 23rd ACM Symp. on Theory of Computing*, pages 21-31, 1991.

[BFL] L. Babai, L. Fortnow, C. Lund. Non-deterministic exponential time has two-prover interactive protocols. *Comput. Complexity* **1** (1991), 16-25.

[Bel] M. Bellare. Interactive Proofs and Approximation. IBM Res. Rep. RC 17969 (1992).

[BG+] M. Bellare, S. Goldwasser, C. Lund, A. Russell. Efficient Multi-Prover Interactive Proofs with Applications to Approximation Problems. In *Proc. 25th ACM Symp. on Theory of Computing*, pages 113–131, 1993.

[BR] M. Bellare, P. Rogaway. The Complexity of Approximating Non-Linear Programs. In P.M. Pardalos, editor, *Complexity of Numerical Optimization.* World Scientific, 1993. Preliminary version: IBM Research Report RC 17831 (March 1992).

[BGKW88] M. Ben-or, S. Goldwasser, J. Kilian, and A. Wigderson. Multi prover interactive proofs: How to remove intractability assumptions. In *Proc. 20th ACM Symp. on Theory of Computing*, pages 113–121, 1988.

[BMV] E. R. Berlekamp, R. J. Mc Eliece, H. C. A. Van Tilborg. On the inherent intractability of certain coding problems, *Trans. Inform. Theory* (1978) 384-386.

[BK] M. Blum, S. Kannan. Designing programs that check their work. In *Proc. 21st ACM Symp. on Theory of Computing*, pages 86–97, 1989. 1989.

[BLR] M. Blum, M. Luby, R. Rubinfeld. Self-testing/correcting with applications to numerical problems. In *Proc. 22nd ACM Symp. on Theory of Computing*, pages 73–83, 1990.

[Bol] B. Bollobás. *Combinatorics.* Cambridge Univ Press 1986.

[BN] J. Bruck and M. Naor. The hardness of decoding linear codes with preprocessing. *IEEE Transactions on Inform. Theory 1990* 381-385.

[Con] A. Condon. The complexity of the max-word problem and the power of one-way interactive proof systems. *Comput. Complexity* **3** (1993), 292–305. Preliminary version appeared in *Proc. 8th Symp. on Theor. Aspects of Comp. Sci.*, Springer L.N.C.S. 1991, pp. 456–465.

[FG+] U. Feige, S. Goldwasser, L. Lovász, S. Safra, M. Szegedy. Approximating clique is almost NP-complete. In *Proc. 32nd IEEE Symp. on Foundations of Computer Science*, pages 2–12, 1991.

[FL] U. Feige, L. Lovász. Two-Prover One-Round Proof Systems: Their Power and Their Problems. In *Proc. 24th ACM Symp. on Theory of Computing*, pages 733–741, 1992.

[FT] A. Frank, É. Tardos. An application of simultaneous approximation in combinatorial optimization. In *Proc. 26th IEEE Symp. on Foundations of Computer Science*, pages 459–463, 1985.

[GJ] M. R. Garey, D. S. Johnson. *Computers and Intractability: A Guide to the theory of NP-Completeness.* W. H. Freeman, 1979.

[HS] K-U. Hoeffgen, H-U. Simon. Robust Trainability of Single Neurons. In *Proceedings of the Conference of Learning Theory*, pages 428–439, 1992.

[JP] D. S. Johnson, F. P. Preparata. The Densest Hemisphere Problem. *Theor. Comput. Sci.* **6**(1978), 93-107.

[K] R. Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, **12/3**, 1987.

[K2] R. Kannan. Algorithmic Geometry of Numbers. In *Annual Reviews of Computer Science*, eds. Traub, Grosz, Lampson, and Nilsson, Vol. 2(1987), pages 231–267. Publ. Annual Reviews Inc.

[Kar] R. M. Karp. Reducibility among combinatorial problems. Miller and Thatcher, eds., *Complexity of Computer Computations*, 85-103. Plenum Press, 1972.

[LLS] J. Lagarias, H.W. Lenstra, C.P. Schnorr. Korkine-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica* **10** (1990), 333-348.

[LO] J.C. Lagarias and A.M. Odlyzko. Solving low-density subset-sum problems. *J. ACM* **32** (1985), 229-246.

[L^3] A.K. Lenstra, H.W. Lenstra, L. Lovász. Factoring Polynomials with rational coefficients. *Math. Ann.* **261** (1982), 513-534.

[Lov] L. Lovász. An Algorithmic Theory of Numbers, Graphs and Convexity. *NSF-CBMS Reg. Conference Series.* SIAM, 1986.

[LY] C. Lund, M. Yannakakis. On the Hardness of Approximating Minimization Problems. *Journal of the ACM*, 41(5):960–981, 1994. Prelim. version in *25th STOC*, 1993, 286-293.

[MP] M. Minsky, S. Papert. *Perceptrons,* 1968.

[PY] C. Papadimitriou and M. Yannakakis. Optimization, approximation and complexity classes *Journal of Computer and System Sciences*, 43:425–440, 1991. Prelim. version in *20th ACM STOC*, 229-234, 1988.

[Sch] C.P. Schnorr. A hierarchy of polynomial-time basis reduction algorithms. *Proc. Conf. on Algorithms* , Pécs (Hungary), Lovász, Szemerédi, eds., North-Holland 1985, 375-386.

[S] J. Stern. Approximating the number of error locations is NP-complete. *Proc AAECC-10*, LNCS (1993), to appear.

[vEB] P. van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. *Tech. Report 81-04, Math. Inst. Univ. Amsterdam, 1981.*

[Z] D. Zuckerman. NP-complete problems have a version that's hard to approximate. *8th Structure in Complexity Theory Conf.*, 1993, 305-312.