Approximating-CVP to within Almost-Polynomial Factors is NP-Hard

I. Dinur
Tel-Aviv University
dinur@math.tau.ac.il

G. Kindler
Tel-Aviv University
puzne@math.tau.ac.il

S. Safra
Tel-Aviv University

Abstract

This paper shows the closest vector in a lattice to be NP-hard to approximate to within any factor up to $2^{(\log n)^{1-\epsilon}}$ *where* $\epsilon = (\log\log n)^{-c}$ *for any constant* $c < \frac{1}{2}$.

Introduction

A lattice $L = L(v_1, .., v_n)$, for vectors $v_1, .., v_n \in R^n$ is the set of all integer linear combinations of $v_1, .., v_n$, that is, $L = \{\sum a_i v_i \mid a_i \in Z\}$. Given a lattice L and an arbitrary vector y, the Closest Vector Problem (CVP) is to find a vector in L closest to y. The Shortest Vector Problem (SVP) is the homogeneous analog of CVP, i.e. finding the shortest non-zero vector in L.

These lattice problems have been introduced in the previous century, and have been studied since. Minkowsky and Dirichlet tried, with little success, to come up with lattice approximation algorithms. It was much later that the lattice reduction algorithm was presented by Lenstra, Lenstra and Lovász [L³82] , achieving a polynomial-time algorithm approximating the Shortest Lattice Vector to within an exponential factor $2^{\frac{dim}{2}}$. Babai [5] applied L³'s methods to present an algorithm that approximates CVP to within a similar factor. Schnorr [13] improved on L³'s technique, reducing the factor of approximation to $(1+\epsilon)^n$, for any constant $\epsilon > 0$, for both CVP and SVP. These positive approximation results are still quite weak, achieving only extremely large (exponential) factors. The question naturally arises: What are the factors of approximation to within which these problems can be approximated in polynomial time?

Interest in lattice problems has been recently renewed due to a result of Ajtai [1], showing a reduction, from the worst-case of a restricted version of SVP, to the average-case of the same problem. Finding a problem whose average case complexity is known to be as hard as the worst-case of some other problem is quite an achievement by itself from complexity theoretic perspective. Yet such a result has significant cryptographic applications, as shown in [3]. Showing NP-hardness for that specific restriction of SVP – although unlikely as discussed below – would imply a cryptosystem whose breaking would imply P=NP.

CVP was shown to be NP-hard for any l_p norm in [14], where it was also conjectured that SVP is NP-hard. Arora et al. [4] utilized the PCP characterization of NP to show that CVP is NP-hard to approximate to within any constant, and quasi-NP-hard to approximate to within $2^{(\log n)^{1-\epsilon}}$ for any constant $\epsilon > 0$.

As to SVP, only recently, Ajtai [2] showed a randomized reduction from the NP-complete problem Subset-Sum to SVP. This has been improved [6], showing approximation hardness for some small factor $(1+\frac{1}{dim^{\epsilon}})$. Very recently Micciancio [12] has significantly strengthened Ajtai's result, showing SVP hard to approximate to within some constant factor. The proof in [12] relies on the PCP characterization of NP and is carried out via a reduction from gap-CVP (shown NP-hard for any constant gap in [4]). Using gap-CVP allows, in addition to the significant improvement in the gap, a major simplification of the main technical lemma from [2]. Better hardness results for gap-CVP may result in hardness results for gap-SVP for larger gaps.

So far there is still a huge gap between the positive results, approximating these problems to within exponential factors, and the above hardness results. Nevertheless, some other results provide a discouraging indication for improving the hardness result beyond a certain factor. Lagarias et al. [10] showed that approximating CVP to within $dim^{1.5}$ is in co-NP, and recently Goldreich and Goldwasser [9] showed that approximating both SVP and CVP to within $\sqrt{dim}$ is in NP∩co-AM. Hence showing NP-hardness for these problems is unlikely.

The strongest hardness result likely to be true for these problems hence, is that they are hard to approximate to within a constant power of the dimension. The proof of [4] utilizes amplification techniques that cause the size of the instance, hence the dimension, to grow faster than the factor for which hardness of approximation is obtained. It is therefore unlikely that using this technique, even if allowing a super-polynomial blow-up, one can obtain such strong results. It seems that it will always be the case that the factor for which hardness of approximation is proven never reaches beyond the barrier of $2^{(\log dim)^{1-\epsilon}}$ for constant $\epsilon > 0$.

Our Results

This paper improves on [4] in two ways. First, it goes beyond the barrier of constant ϵ, proving CVP hard to approximate to within a factor of $2^{(\log dim)^{1-\epsilon}}$ where ϵ, rather than being an arbitrarily small constant, is $(\log\log dim)^{-c}$ for any $c < \frac{1}{2}$. This is the first hardness result for CVP reaching beyond the above mentioned barrier. Furthermore, our result shows approximating CVP is NP-hard for large factors, compared to the previously known *quasi* NP hardness.

The best known $\mathcal{PCP}$ characterization of NP (and even the conjectured one) seems inappropriate in order to show hardness of approximating CVP to within large factors. We introduce, for that purpose, a new characterization of NP, $\mathcal{SSAT}$, and prove the hardness of gap-CVP using this new characterization. The $\mathcal{SSAT}$ characterization is different from the $\mathcal{PCP}$ characterization, despite relying on similar techniques in its proof.

We define the SAT[$\mathcal{F}$] problem, and the gap version of it, $\mathcal{SSAT}$. We show (theorem 1) that solving this problem is NP-hard for any gap $g \leq 2^{(\log n)^{1-\epsilon}}$ where $\epsilon = (\log\log n)^{-c}$ and c is a positive constant $c < \frac{1}{2}$. (n denotes, as usual, the size of the instance).

Improving the hardness of approximation factor to a constant power of n, namely where $g = n^{\epsilon}$ for some constant ϵ (conjecture 2), would imply CVP to be hard to approximate to within a constant power of the dimension.

We also show that our proof works for lattices over finite-fields (instead of Z). This in particular implies NP-hardness for approximating the nearest-codeword [4] within factor g.

Structure of the Paper

Section 1 presents the new characterization of NP, $\mathcal{SSAT}$. It starts by formally defining $\mathcal{SSAT}$ and then states theorem 1, which asserts that it is NP-hard for large factors of approximation. Section 2 shows a simplistic reduction from $\mathcal{PCP}$ instances to $\mathcal{SSAT}$ (via a super-polynomial construction), which will be used as a basis for the complete proof. We describe the full construction of the $\mathcal{SSAT}$ reduction in section 3, and give most of the reduction correctness proof - the proof of the consistency lemma used in the correctness proof, only appears in the full version of the paper. In section 4 we show the simple reduction from $\mathcal{SSAT}$ to CVP.

1 Super-SAT - $\mathcal{SSAT}$

In this section we define a new characterization of NP, referred to as $\mathcal{SSAT}$. Let us begin by defining SAT[$\mathcal{F}$], which is actually SAT from a consistency point of view. An instance of SAT[$\mathcal{F}$] is a set of tests (Boolean functions) over a common set of variables that range over a field $\mathcal{F}$ of $\leq$ polynomial size. An assignment to a *test* maps to each test one of the test's satisfying values. The assignments to each test ψ have a different range, denoted $\mathcal{R}_\psi$. We denote $\mathcal{R} \stackrel{def}{=} \bigcup \mathcal{R}_\psi$. An instance is accepted iff there is an assignment to the tests that is everywhere consistent, that is, each variable is given the same value by the assignments to all tests that depend on it.

$\mathcal{SSAT}$ is a gap variant of this problem, obtained by allowing certain non-satisfiable instances to be accepted. The gap of $\mathcal{SSAT}$ is no longer the fractional gap of the $\mathcal{PCP}$ (i.e. finding the maximal fraction of satisfiable tests) but of a different nature.

We will introduce a new notion of super-assignment[1] to the tests, that is, a formal linear combination of assignments. We will allow acceptance of non-satisfiable instances that have 'short' and 'consistent' super-assignments.

Definition 1 (Super-Assignment to Tests) *A super-assignment is a function* $\mathbf{S}$ *mapping to each* $\psi \in \Psi$ *a value from* $Z^{\mathcal{R}_\psi}$. $\mathbf{S}(\psi)$ *is a vector of integer coefficients, one for each possible value* $r \in \mathcal{R}_\psi$. *Denote by* $\mathbf{S}(\psi)[r]$ *the* r^{th} *coordinate of* $\mathbf{S}(\psi)$.

$\mathbf{S}$ is said to be *non-trivial* if $\forall\psi \in \Psi$, $||\mathbf{S}(\psi)|| > 0$, where $||\cdot||$ denotes l_1 norm. Note that $||\mathbf{S}(\psi)|| > 0$ means $||\mathbf{S}(\psi)|| \geq 1$ since all the entries are integers. Also note that the use of l_1 norm throughout the proof is arbitrary, and can be replaced by any l_p norm, $p > 1$. For a test ψ we think of all the values receiving non-zero coefficients in $\mathbf{S}(\psi)$ as being simultaneously 'assigned' to ψ. The non-triviality requirement means that each test must be assigned at least one value.

A *natural assignment* (an assignment in the usual sense) is identified with a super-assignment where ψ is assigned a unit vector with a 1 in the corresponding coordinate.

Definition 2 (Norm of a Super-Assignment) *The norm of a super-assignment* $\mathbf{S}$ *is* $||\mathbf{S}|| = \frac{1}{|\Psi|}\sum_{\psi\in\Psi}||\mathbf{S}(\psi)||$.

The norm of a natural super-assignment is 1. The gap of $\mathcal{SSAT}$ will be formulated in terms of the norm of the minimal super-assignment that maintains consistency. In the SAT[$\mathcal{F}$] problem a satisfying assignment is one that is everywhere consistent: For every pair of tests with a mutual variable, the assignments to the tests, restricted to the variable, are equal. We extend this notion to super-assignments by defining the projection of a super-assignment to a test onto each of its variables. Consistency between tests will amount to equality of projections on mutual variables.

[1]The name is derived from the notion of a super-position of assignments.

A natural assignment r to a test induces an assignment to each variable x, denoted $r|_x$. Similarly, a super-assignment, induces a super-assignment on a variable by taking the formal linear combination of the assignments' restrictions.

Definition 3 (Projection) *Let* $\mathbf{S}$ *be a super-assignment to the tests. We define the projection of* $\mathbf{S}(\psi)$ *on a variable* x *of* ψ, $\pi_x(\mathbf{S}(\psi)) \in Z^{|\mathcal{F}|}$, *as follows:*

$$\forall f \in \mathcal{F} : \qquad \pi_x(\mathbf{S}(\psi))[f] \stackrel{def}{=} \sum_{r \in \mathcal{R},\ r|_x = f} \mathbf{S}(\psi)[r]$$

We shall now proceed to define the notion of consistency between tests. If the projections of two tests on a mutual variable x are equal (in other words, they both give x the same super-assignment), we say that the super-assignments of the tests are consistent (match).

Definition 4 (Consistency) *Let* $\mathbf{S}$ *be a super-assignment to the tests in* Ψ. $\mathbf{S}$ *is* consistent *if for every pair of tests* ψ_i *and* ψ_j *with a mutual variable* x,

$$\pi_x(\mathbf{S}(\psi_i)) = \pi_x(\mathbf{S}(\psi_j))$$

We can now define the $\mathcal{SSAT}$ problem.

Definition 5 ($\mathcal{SSAT}$) *An instance of* $\mathcal{SSAT}$ *with parameter* g *consists of a system of tests (Boolean functions)* $\Psi = \{\psi_1, ..., \psi_m\}$ *over common variables from* $\mathcal{V} = \{x_1, .., x_n\}$ *ranging over a polynomial range* $\mathcal{F}$. *Each test depends on exactly two variables. The problem is to determine if the instance falls into one of the following two cases,*

Yes: There is a consistent natural assignment for Ψ.

No: No non-trivial consistent super-assignment is of norm $\leq g$ *(i.e. with an average of* $\leq g$ *assignments per function).*

Theorem 1 ($\mathcal{SSAT}$ Theorem) $\mathcal{SSAT}$ *is NP-hard for* $g = 2^{(\log n)^{1-\epsilon}}$ *where* $\epsilon = (\log \log n)^{-c}$, *for any* $c < \frac{1}{2}$.

We suggest a stronger conjecture, which, if true, would imply that CVP is hard to approximate to within a *constant power* of the dimension.

Conjecture 2 $\mathcal{SSAT}$ *is hard for* $g = n^c$ *for some constant* $c < \frac{1}{2}$.

The $\mathcal{SSAT}$ theorem (theorem 1) can be viewed as an extension of Cook's theorem [7] in the following way. An algorithm solving $\mathcal{SSAT}$ is required to accept if the test system is satisfiable. However, the algorithm is allowed to accept non-satisfiable instances that have a consistent super-assignment of norm $\leq g$. We are, in fact, adding slackness between the acceptance and rejection cases.

The Depend Parameter. In the above formulation, the $\mathcal{SSAT}$ tests depend on exactly two variables. Consider the following modification. Let each test depend on a *polynomial* number of variables, as long as the number of satisfying values per each test is *polynomially bounded*. The reduction from this modification to the above formulation is simple:

- Add one new variable for every test. The variable for ψ will range over the satisfying values of ψ, $\mathcal{R}_\psi$.
- Replace the tests with a test for every pair of (test,variable) verifying that the values match.

The range of the new variables is still polynomial because of the restriction on the number of satisfying assignments to each test. The $\mathcal{SSAT}$ gap property is maintained, and every new test depends on exactly two variables. Note that this simple transformation in a $\mathcal{PCP}$ test system will severely increase the error probability.

Proving the NP-hardness of $\mathcal{SSAT}$, we construct a test system where the depend is much larger than 2. We show, in exchange, that the number of satisfying assignments for every test is polynomial. Such a test system can then be translated to a $\mathcal{SSAT}$ test system by the above transformation.

2 The Simplistic Construction

In this section we give a 'simplistic' hardness proof for $\mathcal{SSAT}$, via a super-polynomial construction. This is done by a reduction from $\mathcal{PCP}$ to $\mathcal{SSAT}$, that has super-polynomial variable range. The final $\mathcal{SSAT}$ reduction, described on the next section, is based on this construction. Our starting point is the $\mathcal{PCP}$ characterization of NP. For this purpose, any of the known $\mathcal{PCP}$ theorems would do, since the only property we need is a constant error probability.

Theorem 3 (PCP Theorem [8]) *Let* $\Phi = \{\phi_1, ..., \phi_n\}$ *be a system of tests over variables* $\mathcal{V} = \{x_1, .., x_{n'}\}$ *such that each test depends on* $O(1)$ *variables, and each variable ranges over a field* $\mathcal{F}$ *where* $|\mathcal{F}| = O(2^{(\log n)^{1-\epsilon}})$ *for any constant* $\epsilon > 0$. *It is NP-hard to distinguish between the following two cases:*

Yes: There is an assignment to the variables such that all $\phi_1, ..., \phi_n$ *are satisfied.*

No: No assignment can satisfy more than $O(\frac{1}{|\mathcal{F}|})$ *fraction of the* ϕ_i*'s.*

We shall construct a new test system Ψ with an $\mathcal{SSAT}$ gap, based on Φ, and show a reduction from the $\mathcal{PCP}$ instance to the $\mathcal{SSAT}$ instance.

Cancellations

Suppose that for every 'no' instance of Φ, all of the consistent super-assignments are of norm $> g$ - we could then take Φ for our final construction. Unfortunately, this is not necessarily the case. Although a natural assignment cannot satisfy Φ, a *super-assignment* may somehow locally cancel values on each variable and hence yield false consistency. (For example, let $(1,3),(3,3),(3,1)$ be the satisfying assignments of $\varphi(x,y)$; then the super-assignment $1\cdot(1,3)-1\cdot(3,3)+1\cdot(3,1)$ projects to the natural assignment 1 on both x and y, although $(1,1)$ doesn't satisfy φ).

Since we do not know the exact structure of the assignments to the tests, the cancellation problem cannot be ruled out. To overcome the cancellation problem, we add auxiliary variables that serve as an error correcting code. We will show that for every test, only a negligible fraction of its variables can be canceled in the above sense, and deduce that a consistent super-assignment must in fact be globally consistent.

Low Degree Functions

We begin with a few basic definitions relating to low degree functions.

Definition 6 **($[r,d]$-LDF)** *Let $\mathcal{F}=\mathcal{Z}_p$ for some prime p, and let $\mathcal{D}=\mathcal{F}^d$ be a domain. A low-degree function (LDF for short) with parameters $[r,d]$ is a polynomial function over $\mathcal{F}^d$ whose degree in each variable is no more than r.*

Denote by $LDF_{r,d}$ the set of all $[r,d]$-LDFs. We frequently use the following property of LDFs,

Proposition 1 *Let f and g be two distinct $[r,d]$-LDFs. The fraction of points $x\in\mathcal{D}$ on which $f(x)=g(x)$ is $\le\frac{rd}{|\mathcal{F}|}$.*

The Low Degree Extension

Let $x_1,..,x_n$ be Φ's variables. We embed them in a larger domain (as done in numerous $\mathcal{PCP}$ papers): We view the variables as points of a set $\mathcal{H}^d$ (where $\mathcal{H}$ and d are chosen so that $|\mathcal{H}|^d=n$). We then extend the set $\mathcal{H}^d$ to a domain $\mathcal{F}^d\supset\mathcal{H}^d$ by taking $\mathcal{F}\supset\mathcal{H}$ to be a field of size $|\mathcal{F}|=|\mathcal{H}|^{O(1)}$. We have a variable for each point in the domain $\mathcal{F}^d$. The points of the extended domain $\mathcal{F}^d$ serve as the auxiliary variables that help eliminate the cancellation problem.

Satisfying assignments to the new variables would be *extensions* of satisfying assignments to the original variables in the following sense. Let $A:\mathcal{H}^d\to\mathcal{F}$ be an assignment to the original variables. There exists exactly one polynomial $\hat{A}:\mathcal{F}^d\to\mathcal{F}$ such that $\hat{A}$ *extends* A, and $\hat{A}$ has degree h in each of its variables. $\hat{A}$ is called the h degree extension of A in $\mathcal{F}$.

2.1 The Construction

We now proceed to describe the simplistic construction. This construction possesses the desired $\mathcal{SSAT}$ gap but it inflates the size of the generated instance.

Parameters. Denote the size of the original $\mathcal{PCP}$ instance by n. Let $c<\frac{1}{2}$ be arbitrary. We choose $\epsilon=(\log\log n)^{-c}$, $|\mathcal{H}|=2^{O((\log n)^{1-2\epsilon})}$, $|\mathcal{F}|=|\mathcal{H}|^{O(1)}$ and $d\approx(\log n)^{2\epsilon}$. These parameters will be fixed throughout the rest of the paper.

Variables. We shall have one variable for every point $x\in\mathcal{D}\stackrel{def}{=}\mathcal{F}^d$. The original variables of Φ are identified with the subset $\mathcal{H}^d\subset\mathcal{F}^d$ of the new variables.

Tests. The tests of Ψ will correspond to affine subspaces (cubes) of $\mathcal{D}$. We define a *t-cube* of $\mathcal{D}$ to be an affine subspace of $\mathcal{D}$ of dimension t. Assume w.l.o.g. that all of the tests in Φ depend on exactly $D=O(1)$ variables. For a test $\varphi\in\Phi$ that depends on $x_{i_1},..,x_{i_D}$, define the D-tuple $\tau_\varphi\stackrel{def}{=}(x_{i_1},..,x_{i_D})$.

Denote by $\mathcal{A}_{\tau_\varphi}$ the set of all $(D+3)$-cubes that contain the points of τ_φ. Let $\mathcal{T}=\{\tau_\varphi\}_{\varphi\in\Phi}$, and define $\mathcal{A}_\mathcal{T}=\bigcup_{\tau\in\mathcal{T}}\mathcal{A}_\tau$.

For every cube $\mathcal{C}\in\mathcal{A}_{\tau_\varphi}$, Ψ has a test that depends on the variables corresponding to the points of $\mathcal{C}$. This test accepts only $[dh,D+3]$-LDFs whose restriction to the tuple points of τ_φ satisfy $\varphi\in\Phi$. We call $\mathcal{T}$ the tuple-set of the test system. Note that the tests of Ψ are determined by the tuples.

Super-Assignments. A super-assignment to a test (a cube) is, by definition, a formal linear combination of LDFs. We shall call such an object a super-polynomial. We include the explicit definition of a super-polynomial for clarity,

Definition 7 **($[r,t]$-Super-Polynomial)** *An $[r,t]$-super-polynomial is a function $\mathcal{G}:LDF_{r,t}\to Z$ that assigns each $[r,t]$-LDF an integer coefficient. One may think of $\mathcal{G}$ as a vector with $|LDF_{r,t}|$ integer coordinates.*

Denote by $SLDF_{r,t}$ the set of all $[r,t]$-super-polynomials. The norm and projection of a super-polynomial are defined as in the general case for super-assignments. Consistency of super-polynomials amounts to equality of projections on each mutual point. The projection $\pi_\mathcal{C}(\mathcal{G})$ of a super-polynomial $\mathcal{G}$ on a cube $\mathcal{C}$ is naturally defined as the formal linear combination of the restrictions of the LDFs in $\mathcal{G}$ to the cube.

We shall now describe a property of super-polynomials that will help us overcome the cancellation problem: low-ambiguity. A point x_0 is called *ambiguous* for a super-polynomial $\mathcal{G}$, if there are two LDFs $P_1\neq P_2$ that each have a non-zero coefficient in $\mathcal{G}$, and $P_1(x_0)=P_2(x_0)$. The ambiguous points are the only points that are candidates for cancellation. Only a negligible fraction of the points are ambiguous.

Proposition 2 (Low Ambiguity) *Let $\mathcal{G}$ be an $[r,t]$-super-polynomial of norm $\leq g$. The fraction of ambiguous points in $\mathcal{D}$ is $\leq amb(r,t,g) \stackrel{def}{=} \binom{g}{2}\frac{rt}{|\mathcal{F}|}$.*

We omit the simple proof of this proposition. We now know that no more than $amb(r,t,g)$ fraction of the variables of a test (points of a cube) can be canceled. Using the low-ambiguity property we can deduce global consistency from consistency of super-assignments, as seen in the following lemma. This lemma is a special case of a more general consistency lemma, that is proven in the full version of the paper.

Lemma 1 (The Simplistic Consistency Lemma) *Let $\mathbf{S} : \mathcal{A}_{\mathcal{T}} \rightarrow SLDF_{r,t}$ be a super-assignment of norm $\leq g < |\mathcal{F}|^{\frac{1}{100}}$ and assume that $amb(r,t,g) \ll |\mathcal{F}|^{-\frac{1}{2}}$. If $\mathbf{S}$ is consistent (i.e. for every pair of cubes $\mathcal{C}_1, \mathcal{C}_2$ with a mutual point x – the projections of $\mathbf{S}(\mathcal{C}_1)$ and $\mathbf{S}(\mathcal{C}_2)$ on x are equal); then there exists a global super-polynomial $\mathcal{G}$ of degree h on $\mathcal{D}$ such that $||\mathcal{G}|| \leq 2g$ and*

$$\Pr_{\mathcal{C}\in\mathcal{A}_{\mathcal{T}}}[\mathbf{S}(\mathcal{C}) = \pi_{\mathcal{C}}(\mathcal{G})] > 1 - \frac{g^{c_1}}{|\mathcal{F}|^{c_2}}$$

For an appropriate choice of $g = |\mathcal{F}|^{c_3}$, we obtain a global super-polynomial that agrees with all but a negligible fraction of the cubes in $\mathcal{A}_{\mathcal{T}}$. This is the aforementioned global consistency.

The Construction is Correct - a Sketch

Completeness. If Φ is totally satisfiable, take an LDF f over $\mathcal{F}^d$ that extends the satisfying assignment to the x_i's (the low degree extension of these values). Its restrictions to the cubes of Ψ will supply the consistent natural assignment for Ψ.

Soundness. We take g such that $amb(h,t,g) = \binom{g}{2}\frac{ht}{|\mathcal{F}|}$ is negligible. Given a non-trivial consistent super-assignment to the cubes with norm $\leq g$, we use the simplistic consistency lemma (lemma 1) and the low-ambiguity property to come up with a polynomial P whose restrictions appear in $\mathbf{S}(\mathcal{C})$ for most $\mathcal{C} \in \mathcal{A}_{\mathcal{T}}$.

Since $\mathbf{S}(\mathcal{C})$ contains LDFs whose restrictions are satisfying, taking P's value on the points of $\mathcal{H}^d$ produces a satisfying assignment for most of the tests in Φ. This shows that if the $\mathcal{PCP}$ instance was a 'no' instance, then any consistent super-assignment for the $\mathcal{SSAT}$ instance must be of norm $> g$.

Size. The size of the construction is easily shown to be polynomial. The problem is the range of the assignments to the tests. The range of satisfying assignments for the tests behaves like the number of $[h,t]$-LDFs, i.e. it is super-polynomial. We overcome this problem by an iterative substitution of the cubes, as shown in the final construction, in the next section.

3 The Final Construction

In the previous section we constructed a test system that had the $\mathcal{SSAT}$ property (a consistent super-assignment of norm $\leq g$ for it, implies that the original test system was a 'yes' instance). We shall maintain this property while decreasing the range of the tests. Since every cube ranged over too many LDFs, we represent each cube by new variables that have a smaller range. This replacement procedure will be repeated several times, until the final variables have a polynomial range.

Cube-Systems An $[r,t]$-*cube-system* is a specific form of a test system. There is an underlying set of domains (copies of $\mathcal{F}^d$). The variables correspond to the points in these domains. Some points are *mutual* to several domains, i.e. the *same* variable represents these points in each of the domains. The tests in the cube-system correspond to cubes in these domains defined by a set of tuples. The satisfying assignments to the tests are $[r,t]$-LDFs. The simplistic construction is an example of a $[dh, D+3]$-cube-system with one domain.

3.1 The b-transformation of Ψ to Ψ'

In this section we show a transformation procedure from an $[r,t]$-cube-system into an $[r',t']$-cube-system, where $r' \ll r$ and $t' \approx t$. Starting from the simplistic construction and making a series of these transformations, will preserve (most of) the gap of the initial construction, as shown in the soundness theorem, theorem 4.

We shall repeat this b-transformation iteratively, starting from the simplistic construction, until we reach super-polynomials with small enough degree and dimension (enough so that the range is polynomial). The b-transformation is thus the key step in the reduction. It is the tool that enables us to keep the entire construction polynomial in size, while retaining the large gap factor.

Proposition 3 *Let $b > 1$. Let Ψ be an $[r,t]$-cube system. There exists (polynomially constructible) a $[bt\log_b r, t+4]$-cube-system Ψ' that "represents" Ψ.*

By "represents" we mean that consistent super-assignments to Ψ naturally translate to almost consistent super-assignments to Ψ' (with roughly the same norm) and vice versa. The exact meaning of this representation will become clear in the end of this section.

The Embedding Extension

We will replace each cube by a new set of cubes such that the super-assignments to the new cubes have a smaller range. We first extend every cube to a larger domain. Sub-cubes of this domain will become the new tests of the new cube system Ψ'.

Let $\mathcal{C}$ be a t-cube, and let f be an $[r,t]$-LDF on $\mathcal{C}$. We map $\mathcal{C}$ to an extension domain $ext(\mathcal{C})$, and f to an extension LDF f_{ext} using an embedding technique from [8, Lemma 4], as described below.

We map the points of $\mathcal{C}$ to a manifold in $\mathcal{D}_{\mathcal{C}} \stackrel{def}{=} ext(\mathcal{C})$ by $E : \mathcal{C} \to \mathcal{D}_{\mathcal{C}}$ as follows. E maps an arbitrary point $x = (\xi_1, .., \xi_t) \in \mathcal{C}$ to $y \in \mathcal{D}_{\mathcal{C}}$, $y = E(x) = (\eta_1, .., \eta_{kt})$ by replacing each axis ξ_i with k axes $\eta_{i,1}, .., \eta_{i,k}$ such that the following equations hold,

$$\forall i, m \quad \eta_{i,m} = \xi_i^{b^m} \qquad (*)$$

Now let f be an $[r,t]$-LDF. We map f to a polynomial over $\mathcal{D}_{\mathcal{C}}$ by mapping each of the monomials,

$$\forall i, r \quad \xi_i^r \longrightarrow \eta_{i,0}^{b_0}\eta_{i,1}^{b_1}...\eta_{i,k}^{b_k} \qquad (**)$$

where $b_0 b_1 ... b_k$ is the base b representation of r.

f_{ext} is an LDF of degree b in each variable, and dimension $t \log_b r$. Taking the restriction of f_{ext} to the manifold defined by equations $(*)$, will give f. (This can be easily seen by substituting the manifold equations into each of the monomials). In other words, there is a natural 1-1 mapping of the original cube $\mathcal{C}$ to the manifold of $ext(\mathcal{C})$ defined by equations $(*)$. Computing f on a point $x \in \mathcal{C}$ is equivalent to computing $E(f) = f_{ext}$ on the point $E(x)$.

The initial cube-system, Ψ.

Let $\mathcal{D}_1, .., \mathcal{D}_k$ be domains that may have some mutual points (It may be helpful to think, at first, of $k = 1$ and of Ψ as being the test system from the simplistic construction). Let $\mathcal{T} = \mathcal{T}_1 \cup \cdots \cup \mathcal{T}_k$, where $\mathcal{T}_i$ is a set of t-tuples in $\mathcal{D}_i$. Define, as before, $\mathcal{A}_{\mathcal{T}_i}$ to be the set of $(t+3)$-cubes of $\mathcal{D}_i$ that contain at least one tuple from $\mathcal{T}_i$. Let Ψ be a cube-system with a test for each cube in $\mathcal{A}_{\mathcal{T}} = \bigcup \mathcal{A}_{\mathcal{T}_i}$, and a variable for each point $x \in \bigcup_i \mathcal{D}_i$.
We now construct the transformation of Ψ.

The new cube-system Ψ'.

We shall replace every cube $\mathcal{C} \in \mathcal{A}_{\mathcal{T}}$ by a set of cubes (defined by the tuples below) of its extension domain $ext(\mathcal{C})$. Assignments to these new cubes will range over $[b, t \log_b r]$-LDFs.

Domains. For every cube $\mathcal{C} \in \mathcal{A}_{\mathcal{T}}$ we have a new domain $\mathcal{D}_{\mathcal{C}} \stackrel{def}{=} ext(\mathcal{C})$.

Tuples. We now define a new collection of tuples (that will generate our new cubes). For every $\tau \in \mathcal{T}$ and $\mathcal{C} \in \mathcal{A}_\tau$, let

$$\mathcal{T}_{\mathcal{C}} = \{(x, \tau) \mid x \in \mathcal{C}\}$$

be a set of $(t+1)$-tuples. We consider the tuples in $\mathcal{T}_{\mathcal{C}}$ as belonging to the domain $\mathcal{D}_{\mathcal{C}}$. The tuple collection of Ψ' is $\mathcal{T}' \stackrel{def}{=} \bigcup_{\mathcal{C}} \mathcal{T}_{\mathcal{C}}$.

Variables. For every point in the the extension $\mathcal{D}_{\mathcal{C}}$ of $\mathcal{C}$, Ψ' will have a variable. The extensions of a point $x \in \mathcal{C}_1 \cap \mathcal{C}_2$, in $\mathcal{D}_{\mathcal{C}_1}$ and $\mathcal{D}_{\mathcal{C}_2}$ are considered mutual, and therefore will be represented by *the same* variable in Ψ'.

Tests. There will be a test for every $(t+4)$-cube in $\mathcal{A}' \stackrel{def}{=} \mathcal{A}_{\mathcal{T}'}$. Before we describe the range of the tests, let us dwell for a minute on the meaning of this replacement transformation.

Let $\mathcal{C}$ be a cube in the previous cube-system, Ψ, ranging over a subset of $LDF_{r,t}$. $\mathcal{C}$ was mapped to $\mathcal{D}_{\mathcal{C}}$, its extension domain, where an $[r,t]$-LDF f is naturally mapped to f_{ext}, a $[b, t \log_b r]$-LDF. We have new cubes that allegedly represent $(t+4)$-cubes of $\mathcal{D}_{\mathcal{C}}$. We would like their values to represent the value of the original variable $\mathcal{C}$, i.e. we would like them to represent restrictions of f_{ext} to the sub-cubes of $\mathcal{D}_{\mathcal{C}}$ (these are $[bt \log_b r, t+4]$-LDFs). The range of our new tests, will therefore be a subset of $LDF_{bt \log_b r, t+4}$. This completes the description of the b-transformation.

3.2 The Entire Construction

Let us step back to examine the bigger picture. We begin with the simplistic cube-system Ψ_0. We transform it (by a b-transformation) into Ψ_1, and then transform Ψ_1 into Ψ_2 etc. In the end, the resulting cube-system will be shown to have the $\mathcal{SSAT}$ gap property. Note that the transformations only depend on our choice of the b parameter.

Tree Structure. The recursive structure of the construction is easily depicted as a tree of cubes. The root of the tree is the domain of Φ, the original $\mathcal{PCP}$ test system. Directly beneath the root are all of the cubes of Ψ_0, the simplistic construction. In the b-transformation, every cube was replaced by a set of cubes in its extension domain. These will be placed directly beneath the cube in the tree. Except for the root, every node in the tree can be dually viewed either as a cube, or as a domain that is the embedding extension of that cube.

Alternatively, the nodes of the tree each correspond to a tuple. The first level tuples (directly beneath the root) are the tuples defined by the tests of Φ plus three random points. the offspring of any node $\mathcal{C}$ in the tree, are the tuples that contain the node's tuple, plus a random point from the manifold $E(\mathcal{C}) \subset ext(\mathcal{C})$, plus three random points in the extension domain $ext(\mathcal{C})$.

We shall return to this tree structure for the soundness proof. Let us first complete the description of the construction by giving the exact sequence of 'b's used in the b-transformation sequence; the last cube-system in the sequence being the final construction.

Notation. It will be convenient to use the following parameters for a cube-system Ψ. $t = t(\Psi)$ is the cube dimension of Ψ, $D = D(\Psi)$ is the dimension of Ψ, and $r = r(\Psi)$ is the overall degree of the assignments for Ψ.
We can construct, (proposition 3), the b-transformation Ψ' of Ψ with the following parameters: $D(\Psi') = t\log_b r$, $t(\Psi') = t + 4$, $r(\Psi') = tb\log_b r$.

The variable range of the new cube-system is, for the right choice of b, considerably smaller than that of the old system. However, one such transformation is not enough - we need to repeat the transformation several times ($O(\frac{1}{\epsilon})$) to get our desired polynomial range.

The Initial Cube-System Let Ψ_2 denote the simplistic $\mathcal{SSAT}$ system (we start with Ψ_2 for notational convenience). Let us recall some of its parameters: $D(\Psi_2) = \log^{2\epsilon} n$, $t(\Psi_2) = const$, $r(\Psi_2) = 2^{O(\log^{1-2\epsilon} n)}$.

The Values Chosen for b. The following are the values of b that are used in the series of b-transformations:

- *The b reduction phase.* The *first* 'b' used will be denoted b_3 for convenience, and is defined as $b_3 = 2^{\log^{1-3\epsilon} n}$. The second '$b$', b_4, is defined $b_4 = 2^{\log^{1-4\epsilon} n}$. We continue to take $b_k = 2^{\log^{1-k\epsilon} n}$, for $k \le K$, where $K \stackrel{def}{=} \lfloor \frac{1}{\epsilon} \rfloor$. After that we can no longer proceed, because the power of the logarithm would become negative.
- *The sub-logarithmic phase.* We use $b = 2$ for three additional iterations.

These values of b *define* the resulting cube-system. We need to show the completeness and soundness of the cube-system, as well as to show that its size is indeed polynomial (the computation of the size will only appear in the full version of the paper). We now cite the soundness theorem, and show that it indeed yields the desired $\mathcal{SSAT}$ gap property.

Theorem 4 (Soundness) *Let $g \stackrel{def}{=} |\mathcal{F}|^{\frac{1}{100}\cdot\frac{1}{a}^K}$ for some constant $a > 1$. If there exists a satisfying super-assignment of size $\le g$ for Ψ_K, then Φ (the $\mathcal{PCP}$ system we began with) is satisfiable.*

The gap parameter in the soundness theorem is indeed enough for the g-$\mathcal{SSAT}$ theorem (theorem 1), as the following proposition states. We omit its simple proof.

Proposition 4 $\forall c < \frac{1}{2}$, $\exists c < c' < \frac{1}{2}$ *such that*

$$|\mathcal{F}| = 2^{\log^{1-\epsilon(c')} n}$$

$$g = |\mathcal{F}|^{\frac{1}{100}\cdot\frac{1}{a}^K} = 2^{\log^{1-\epsilon(c)} n}$$

where $\epsilon(c) \stackrel{def}{=} (\log\log n)^{-c}$.

This proposition implies that the soundness parameters indeed provide the desired hardness result for $\mathcal{SSAT}$.

We proceed to prove the soundness theorem.

Proof Structure We begin with $\mathbf{S}_K$, a consistent super-assignment for Ψ_K, of size $\le g$. It induces (by projection) a super-assignment $\mathbf{s}$ for the variables (points). Since $\mathbf{S}_K$ is consistent, $\mathbf{s}$ is well defined. $\mathbf{s}$ is used as the "underlying point super-assignment" for the rest of the proof.

Let us return to the tree view of the construction. We put every cube of the final construction as a leaf in the tree, and the internal nodes are cubes of the intermediate cube-systems (level i in the tree corresponds to the cubes of Ψ_i). We define a 'good' cube (node in the tree) by considering the leaves of its sub-tree:

Definition 8 *Let $\mathcal{C}$ be a cube in Ψ_i. $\mathcal{C}$ is said to be* good *if $avg(\mathcal{C}) \le g^{a^i+1}$; where $avg(\mathcal{C})$ is the average norm of $\mathbf{S}_K$, over the cubes in Ψ_K that are derived from $\mathcal{C}$ (i.e. cubes of Ψ_K that are leaves in $\mathcal{C}$'s subtree). We denote by $Good_i$ the set of cubes of Ψ_i that are good.*

The proof consists of four propositions. We shall first show that in each level most cubes are good (proposition 5), and that most of the children of a good cube are good (proposition 6). We then proceed to show that for every good cube there is a super-polynomial that agrees with $\mathbf{s}$ on most of the cube's points (proposition 7). Finally, we take these super-polynomials, and obtain from them an assignment that satisfies more than half of the $\mathcal{PCP}$ test system Φ (proposition 8).

Most cubes are good

Proposition 5 *Let $0 < i \le K$, at least $1 - g^{-a^i}$ of the cubes in level i of the tree are good.*

Proof: For every cube in level i, consider the average norm of its subtree's leaves. The average over the subtree averages is simply the average norm of the leaves, g. By the Markov inequality, no more than $\frac{1}{l}$ of the subtrees have an average larger than lg. Taking $l \stackrel{def}{=} g^{a^i}$ concludes the argument. ■

Most Subcubes of a good cube are good

For any cube $\mathcal{C}$, denote by $child(\mathcal{C})$ the set of cubes directly beneath $\mathcal{C}$ in the tree. These are the cubes in the embedding extension $ext(\mathcal{C})$.

Proposition 6 *If $\mathcal{C} \in Good_i$, then*

$$\Pr_{\mathcal{C}' \in child(\mathcal{C})}(\mathcal{C}' \in Good_{i+1}) > 1 - g^{-(a-1)\cdot a^i}$$

Proof: $\mathcal{C}$ is good, hence $avg(\mathcal{C}) \leq g^{a^i+1}$. Had there been more than a $g^{-(a-1)\cdot a^i}$ fraction of bad subcubes, then the total average would be

$$> g^{-(a-1)\cdot a^i} \cdot g^{a^{i+1}+1} = g^{a \cdot a^i - (a-1)\cdot a^i + 1} = g^{a^i+1}$$

■

A Super-Polynomial per Good Cube

Proposition 7 *Let $\mathcal{C}_0 \in Good_i$. There exists a super-polynomial of norm $\leq 2^{K-i} avg(\mathcal{C}_0)$ that agrees with* $\mathbf{s}$ *on a $1 - g^{-a^i}$ fraction of $\mathcal{C}_0$'s points.*

Proof: We prove this statement by induction on $K - i$.

The Base of the Induction $(i = K)$**.** We know that $\mathbf{S}_K$ is a consistent super-assignment. Every $\mathcal{C} \in Good_K$ has a super-polynomial ($\mathbf{S}_K(\mathcal{C})$) of norm $\leq g^{a^K+1}$, by definition of $Good_K$. $\mathbf{S}_K(\mathcal{C})$ agrees (non-ambiguously) with $\mathbf{s}$ on at least $1 - amb(r, t, s)$ of the points of $\mathcal{C}$. Since

$$amb(r, t, s) < \frac{rts^2}{|\mathcal{F}|} \leq |\mathcal{F}|^{-\frac{1}{2}}$$

the claim follows, using the consistency lemma (see lemma 2 below).

The Inductive Step $(i < K)$**.**

$$avg(\mathcal{C}_0) \geq avg_{\mathcal{C}' \in child(\mathcal{C}_0) \cap Good_{i+1}}(avg(\mathcal{C}'))$$

since taking only the good children can only decrease the average. By the inductive hypothesis, every $\mathcal{C}' \in child(\mathcal{C}_0) \cap Good_{i+1}$ has a super-polynomial with norm $\leq 2^{K-i-1} avg(\mathcal{C}')$. Define a super-assignment to the cubes in $\mathcal{C}' \in Good_{i+1} \cap child(\mathcal{C}_0)$ by setting $\mathbf{S}(\mathcal{C}')$ to be that super-polynomial. The average norm of these super-polynomials is $\leq 2^{K-i-1} avg(\mathcal{C}_0)$. For any cube $\mathcal{C}' \in child(\mathcal{C}_0) \setminus Good_{i+1}$ assign the trivial super-polynomial. It then follows that $||\mathbf{S}|| \leq 2^{K-i-1} avg(\mathcal{C}_0)$.

We now state a consistency lemma that will imply the existence of the desired super-polynomial on $\mathcal{C}_0$.

Lemma 2 (Consistency Lemma) *Let $\mathcal{T}$ be a set of t-tuples. Let $\mathcal{D}$ be a domain, and let $\mathcal{A}_{\mathcal{T}}$ be the set of $(t+3)$-cubes of $\mathcal{D}$ that contain the points of at least one tuple in $\mathcal{T}$. Let $\mathbf{S} : \mathcal{A}_{\mathcal{T}} \to SLDF_{r,t+3}$ be a super-assignment; $||\mathbf{S}|| \leq s < |\mathcal{F}|^{\frac{1}{100}}$. Let* $\mathbf{s}$ *be the underlying super-assignment to the points. Denote by $G \subset \mathcal{A}_{\mathcal{T}}$, the set of all cubes $\mathcal{C}$ for which $\mathbf{S}(\mathcal{C})$ agrees with* $\mathbf{s}$ *on at least $1 - \alpha$ of $\mathcal{C}$'s points, for some $0 < \alpha < \frac{1}{100}$.*

If at least $1 - \alpha$ of the cubes in $\mathcal{A}_{\mathcal{T}}$ are in G then there exists a global super-polynomial $\mathcal{G}$ of degree r on $\mathcal{D}$, with $||\mathcal{G}|| \leq 2s$ and

$$\Pr_{\mathcal{C} \in_R G}(\mathbf{S}(\mathcal{C}) = \pi_{\mathcal{C}}(\mathcal{G})) > 1 - |\mathcal{F}|^{-\frac{1}{2}}(2s + 1)$$

The proof of the lemma is to appear in the full version of the paper. We would like to apply this lemma to the domain $\mathcal{D} = ext(\mathcal{C})$, and the super-assignment $\mathbf{S}$ defined above. Let us see that the super-assignment $\mathbf{S}$ obeys the requirements of the consistency lemma. We take $\alpha = g^{-(a-1)a^i}$, and $s = g^{a^i+1}$. For every cube $\mathcal{C}' \in Good_{i+1} \cap child(\mathcal{C}_0)$, $\mathbf{S}(\mathcal{C}')$ agrees with $\mathbf{s}$ on at least $1 - g^{-a^{i+1}} > 1 - \alpha$ (by the inductive hypothesis). The fraction of good cubes in $child(\mathcal{C}_0)$ is, by proposition 6, $\geq 1 - g^{-(a-1)a^i} = 1 - \alpha$. Hence we can apply the consistency lemma (lemma 2).

We thus obtain a super-polynomial on $\mathcal{D}$ of norm $\leq 2 \cdot 2^{K-i-1} avg(\mathcal{C}_0) = 2^{K-i} avg(\mathcal{C}_0)$ that agrees with $1 - |\mathcal{F}|^{-\frac{1}{2}}(2^{K-i} avg(\mathcal{C}_0) + 1)$ of the super-polynomials on the cubes $\mathcal{C}' \in child(\mathcal{C}_0) \cap Good_{i+1}$. Recall that on every good cube $\mathbf{S}$ agreed with $\mathbf{s}$ on the tuple-points of the cube. This means that $\mathcal{G}$ agrees with $\mathbf{s}$ on 'almost all' of the tuples (because agreeing with one good cube on a tuple means agreeing with the tuple). Let us examine the meaning of 'almost all':

Since $2^{K-i} avg(\mathcal{C}_0) < |\mathcal{F}|^{\frac{2}{100}}$, we have

$$|\mathcal{F}|^{-\frac{1}{2}}(2^{K-i} avg(\mathcal{C}_0) + 1) \leq |\mathcal{F}|^{-\frac{1}{2}+\frac{2}{100}} < \frac{|\mathcal{F}|^{-\frac{1}{100}}}{2} \leq \frac{g^{-a^i}}{2}$$

hence the fraction of cubes in $child(\mathcal{C}_0)$ that are good, and agree with $\mathcal{G}$ is at least $1 - g^{-a^i}$ (the good cubes make up at least half of the cubes). This implies that $\mathcal{G}$ agrees with at least this fraction of tuples. Since we have exactly one tuple per point in the manifold of $\mathcal{C}_0$ (as defined in the construction), we have that $\mathcal{G}_{\mathcal{C}_0}$ (the restriction of $\mathcal{G}$ to the manifold) agrees with $> 1 - g^{-a^i}$ of the manifold points that correspond to $\mathcal{C}_0$, and with all of the tuple-points of $\mathcal{C}_0$ itself.

■

Constructing an assignment for Φ

Proposition 8 *There is an assignment that satisfies more than half of the $\mathcal{PCP}$ test system Φ.*

Proof: We now have an super-polynomial for every good cube. Define $\mathbf{S}_1$ to assign (as before) the cubes in $Good_1$ this super-polynomial, and the trivial super-polynomial to the rest of the cubes in level 1. By proposition 5 we have that $1 - g^{-(a-1)a}$ of the cubes are good, and their super-polynomial agrees with $1 - g^{-a}$ of their points *and* with their tuple points. We know that the average norm of $\mathbf{S}_1$ is $\leq 2^{K-1} avg(root) = 2^{K-1} g$. Using the consistency lemma one more time with $\alpha = g^{-a}$ yields a global super-polynomial that agrees with almost all of the good cubes in Ψ_1, and in particular with half of the cubes in Ψ_1. This super-polynomial is not trivial, because $\mathbf{s}$ – the underlying point super-assignment – is non-trivial on most of its points (this follows from the low-ambiguity proposition, proposition 2).

Take one LDF P that appears in this super-polynomial. It appears in most of the good cubes (it may be canceled on a negligible fraction of the cubes). For every point $x \in \mathcal{H}^d$, assign $P(x)$. This will satisfy at least half of the tests in the $\mathcal{PCP}$ test-system, Φ. This follows since each $\varphi \in \Phi$ is represented by a tuple that is, in turn, contained by a cube that with probability $> \frac{1}{2}$ has P's restriction appearing in its super-polynomial. Note that the $\mathcal{PCP}$ property that we used was a *constant* error probability.

Using the $\mathcal{PCP}$ property of Φ we deduce that it is satisfiable. This completes the proof of lemma 4 (the soundness of the reduction). ■

4 g-CVP is NP-Hard

We define (again) the Closest Vector Problem (CVP), and its gap version g-CVP. We then define an intermediate problem called Shortest Integer Solution (SIS), and show a reduction from g-SIS to g-CVP. We then show the simple reduction from g-$\mathcal{SSAT}$ to g-SIS and therefore to g-CVP. By theorem 1, we deduce the NP-hardness of g-CVP.

g-CVP

A lattice $L = L(v_1, .., v_n)$, for vectors $v_1, .., v_n \in R^n$ is the set of all integral linear combinations of $v_1, .., v_n$, $L = \{\sum a_i v_i \mid a_i \in Z\}$. The closest-vector problem is defined as follows:

CVP. Given (L, y) where L is a lattice and y a vector in R^n, find the lattice vector closest to y.

We measure distance by l_1 norm, although any l_p norm $p > 1$ can be used. Approximating CVP to within factor $g = g(n)$ means finding a vector whose distance from y is no more than g times the minimal distance. The gap version of CVP is a decision problem as follows,

g-CVP. Given (L, y) for a lattice L and a vector $y \in R^n$, distinguish between the following two cases:

[Yes] The closest lattice vector to y is of distance d or less.

[No] All lattice vectors are of distance at least $g \cdot d$ from y.

Proving that g-CVP is NP-hard implies that unless $P = NP$ there is no polynomial-time algorithm that approximates CVP to within a factor g.

4.1 Shortest Integer Solution - SIS

We define a variant of CVP named Shortest Integer Solution (SIS), its gap version referred to as g-SIS. We then show a simple reduction from g-SIS to g-CVP.

SIS: Given (B, t) for an integer matrix B with columns $b_1, .., b_n$ and a target vector $t \in L(b_1, ..., b_n)$, find integer coefficients a_i such that $\sum a_i b_i = t$ (we assume such a_i exist), and such that $\sum |a_i|$ is minimal. In other words, find the shortest integer solution for the linear system $B \cdot x = t$.

The gap version of SIS is as follows,

g-SIS: Given (B, t) as before, distinguish between the following two cases:

Yes: The shortest integer solution is of length d or less.

No: The shortest integer solution is of length at least $g \cdot d$.

Reducing g-SIS to g-CVP

Given an instance of g-SIS, (B, t), we can efficiently construct a lattice L and a target vector y such that 'yes' instances of g-SIS are translated into 'yes' instances of g-CVP and 'no' instances are translated into 'no' instances. The lattice L is constructed by multiplying the matrix B by a very large number w, and adding a distinct 1-coordinate to each column. The vector y (that we are to approximate from within the lattice), is constructed by taking t multiplied by w, and putting zeros in the n additional coordinates.

$$L = \begin{pmatrix} & wB & \\ 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{pmatrix} \qquad y = \begin{pmatrix} \vdots \\ wt \\ \vdots \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

The simple correctness proof of the reduction is omitted.

From $\mathcal{SSAT}$ to g-SIS

We shall prove that g-SIS is NP hard for $g = 2^{(\log n)^{1-\epsilon}}$ for $\epsilon = (\log\log n)^{-c}$ for any $c < \frac{1}{2}$ by reducing it to $\mathcal{SSAT}$.

We take this $\mathcal{SSAT}$ test system Ψ and (efficiently) construct from it an instance of g-SIS, (B, t). We then show that the 'yes' instances of $\mathcal{SSAT}$ are mapped to 'yes' instances of g-SIS and 'no' instances to 'no' instances.

We show that a natural consistent assignment to Ψ translates to a short solution for (B, t). On the other hand we show that any solution that is shorter than g, translates to a consistent super-assignment of size $\le g$ for Ψ.

The General Construction

The matrix B will have a column for every pair of test $\psi \in \Psi$ and an assignment $r \in \mathcal{R}_\psi$ for it. We will be able to translate the shortest integer solution into a consistent super-assignment for Ψ. The upper rows of B will take care of consistency, and the lower rows will take care of non-triviality.

Non-Triviality Rows. There will be a row designated to each test. In the row of ψ all of ψ's columns will have a 1, and all other columns will have zero.

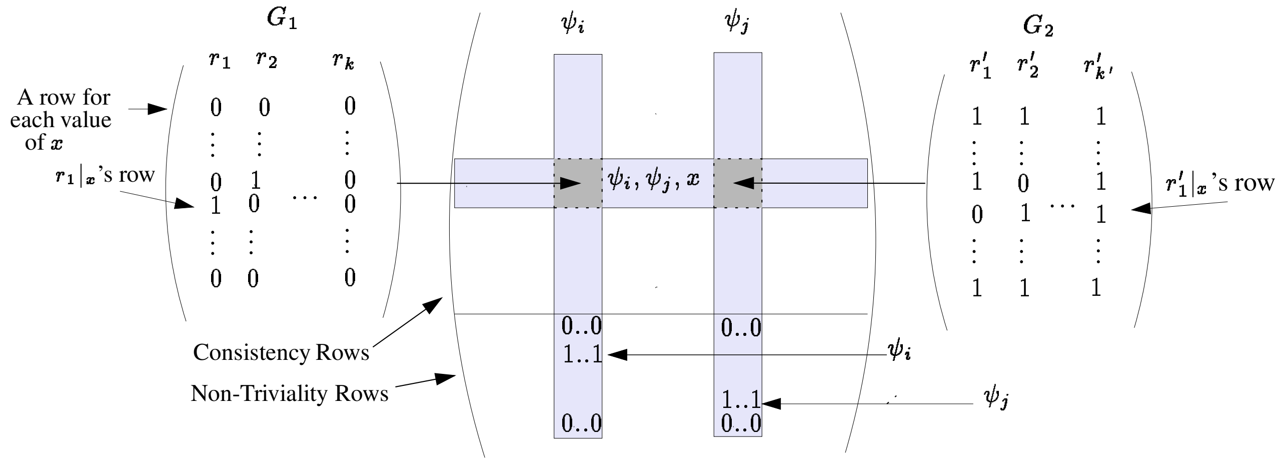

Figure 1: The SIS matrix B

Consistency Rows. We shall have $|\mathcal{F}|$ rows for each pair of tests ψ_i and ψ_j that depend on a mutual variable x. The columns of tests other than ψ_i and ψ_j will have zeros in all of these rows. These rows serve as a consistency-ensuring gadget and only the vectors of ψ_i and ψ_j will have non-zero values in these rows. The gadget will ensure that the super-assignments to ψ_i and ψ_j *are consistent* on their mutual variable x.
The **target vector** will be an all-1 vector.
We now turn to describe the structure of the gadget itself. This will complete the description of the g-SIS instance.

The Gadget. Let's concentrate on the gadget for the pair of tests ψ_i and ψ_j with mutual variable x. This is a pair of matrices G_1 of dimension $\left(|\mathcal{F}| \times |\mathcal{R}_{\psi_i}|\right)$ and G_2 of dimension $\left(|\mathcal{F}| \times \left|\mathcal{R}_{\psi_j}\right|\right)$. Let $r \in \mathcal{R}_{\psi_i}$ be a satisfying assignment for ψ_i and $r' \in \mathcal{R}_{\psi_j}$ be a satisfying assignment for ψ_j. The column in G_1 corresponding to r is a unit vector with a 1 in the $r|_x$-th coordinate. The column in G_2 corresponding to r' is the negation of a unit vector (all ones except for one 0) with a zero in the $r'|_x$-th coordinate (see figure 1).

Proving Correctness - a Sketch

The full correctness proof appears in the full version of the paper. We now sketch why 'yes' instances of the $\mathcal{SSAT}$ map to 'yes' instances of the g-SIS.

Lemma 3 *If there is a consistent natural assignment, then there is a solution of length $|\Psi|$ to the above g-SIS instance.*

Proof: Let $\mathbf{S}$ be the consistent natural assignment. The solution to the g-SIS instance is constructed by taking the vectors $\mathbf{S}(\psi_1)\mathbf{S}(\psi_2)$..., each with coefficient "+1". It is easy to show that this is indeed the desired solution.

■

We will now show that 'no' instances of the $\mathcal{SSAT}$ map to 'no' instances of the g-SIS, by showing that if we did not end up with an 'no' instance of g-SIS, then we must have started with a non-'no' instance of $\mathcal{SSAT}$.

Lemma 4 *Let s be a solution to the above g-SIS instance, $||s|| \leq g\,|\Psi|$ then there exists a consistent super-assignment $\mathbf{S}$ of size $\leq g$ for the $\mathcal{SSAT}$ instance.*

Proof: We show how to construct $\mathbf{S}$ from s: we 'break' s into $|\Psi|$ pieces, one for each test $\psi \in \Psi$.

For any arbitrary $\psi \in \Psi$, the target vector is reached in the ψ-th row of the non-triviality rows. This implies that

$$\sum_{r \in \mathcal{R}_\psi} \mathbf{S}(\psi)[r] = 1 \tag{1}$$

and in particular $\mathbf{S}$ is non-trivial.

To see that $\mathbf{S}$ is consistent, let $\psi_i, \psi_j \in \Psi$ be arbitrary tests with a mutual variable x. We shall show that $\pi_x(\mathbf{S}(\psi_i)) = \pi_x(\mathbf{S}(\psi_j))$. Consider the $|\mathcal{F}|$ rows that correspond to ψ_i, ψ_j, x. In each of these rows the sum of the vectors is 1, in other words, for any $f \in \mathcal{F}$,

$$\sum_{r\,:\,r|_x = f} \mathbf{S}(\psi_i)[r] + \sum_{r\,:\,r|_x \neq f} \mathbf{S}(\psi_j)[r] = 1 \tag{2}$$

Subtracting (1) for ψ_j from (2) gives,

$$\sum_{r\,:\,r|_x = f} \mathbf{S}(\psi_i)[r] = \sum_{r\,:\,r|_x = f} \mathbf{S}(\psi_j)[r]$$

which, by definition of the projection means $\pi_x(\mathbf{S}(\psi_i)) = \pi_x(\mathbf{S}(\psi_j))$. We hence have a consistent super-assignment of size $\frac{1}{|\Psi|}||s|| \leq g$.

■

The two above lemmas complete the reduction of $\mathcal{SSAT}$ to g-SIS.

5 Discussion

- **CVP over Z_p.** We can actually show that g-CVP over a finite field Z_p (for any prime p) is NP-hard. This problem (with $p = 2$) was referred to as 'Nearest-Codeword' in [4] and shown to be quasi-NP-hard to approximate to within a factor of $2^{(\log n)^{1-\epsilon}}$ for any constant $\epsilon > 0$.

  We first need to define a variant of $\mathcal{SSAT}$, $\mathcal{SSAT}_{mod\ p}$ – where consistency is defined as equality of projections *modulo* p – and show that it too is NP-hard. The NP-hardness proof is carried out almost word for word if we notice that a variable was considered ambiguous if any two of its assigned values collided, disregarding the value of the coefficient.

  The result for Nearest Codeword easily follows, using the same reduction from $\mathcal{SSAT}_{mod\ p}$ to $CVP_{mod\ p}$.

- Using the general structure of the proof described herein, however improving the construction, Ran Raz has obtained a better hardness result than claimed here. Namely, that the range of ϵ for which the claim of theorem 1 holds, can be extended to $\epsilon = \frac{\log\log\log n}{\log\log n}$ and therefore CVP is NP-hard to approximate to within a factor of

$$2^{\frac{\log n}{\log\log n}}$$

6 Acknowledgments

We would like to thank Ran Raz for many helpful discussions.

References

[1] M. Ajtai. Generating hard instances of lattice problems. In *Proc. 28th ACM Symp. on Theory of Computing*, pages 99–108, 1996.

[2] M. Ajtai. The shortest vector problem in l_2 is NP-hard for randomized reductions. manuscript, May 1997.

[3] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 284–293, El Paso, Texas, 4–6 May 1997.

[4] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes and linear equations. In *Proc. 34th IEEE Symp. on Foundations of Computer Science*, pages 724–733, 1993.

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

[6] J. Cai and A. Nerurkar. Approximating the SVP to within a factor $(1 + 1/\dim^{\epsilon})$ is NP-hard under randomized reductions. In *Proc. of the 13th Annual IEEE Conference on Computational Complexity*, pages 46–55. 1998.

[7] S. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd ACM Symp. on Theory of Computing*, pages 151–158, 1971.

[8] I. Dinur, E. Fischer, G. Kindler, R. Raz, and S. Safra. PCP characterizations of NP: Towards a polynomially-small error-probability. Manuscript, 1998.

[9] O. Goldreich and S. Goldwasser. On the limits of non-approximability of lattice problems. In *Proc. 30th ACM Symp. on Theory of Computing*, pages 1–9, 1998.

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

[11] A. Lenstra, H. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:513–534, 1982.

[12] D. Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. In *Proc. 39th IEEE Symp. on Foundations of Computer Science*, 1998.

[13] C. Schnorr. A hierarchy of polynomial-time basis reduction algorithms. In *Proceedings of Conference on Algorithms, Pécs (Hungary)*, pages 375–386. North-Holland, 1985.

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