Some Complexity Results and Bit Unpredictable for Short Vector Problem

Cheng Kuan

Abstract

In this paper, we prove that finding the approximate shortest vector with length in $[\lambda_1, \gamma\lambda_1]$ could be reduced to GapSVP. We also prove that shortest vector problem could also be reduced to GapSVP with a small gap. As the complexity of uSVP is not very clear, we improve crurrent complexity results[19] of uSVP, proving uSVP could be reduced from SVP deterministically. What's more, we prove that the search version of uSVP could be reduced to decisional version of uSVP with almost the same gap.

At last, based on the results above, we prove a bit-unpredictable property of SVP.

1 Introduction

Lattice is a wonderful mathematical structure. It is a set of all integer combinations of linearly independent base vector $b_1, b_2, \ldots, b_n$ in $\mathbb{R}^m$. Ajtai discovered that some lattice problems exhibit a wort-case to average-case connection[1]. Cryptosystems could be constructed based on the worst-case hardness of some classic lattice problems. Ajtai-Dwork cryptosystem[2] was the first cryptosystem based on the worst-case hardness of the approximate "unique" Shortest Vector Problem $\text{uSVP}_{O(n^8)}$. Researchers have improved this cryptosystem so that it could be based on worst-case $\text{uSVP}_{O(n^2)}$. Later than that, Regev built a different cryptosystem based on worst-case $\text{uSVP}_{O(n^{1.5})}$[4]. GapSVP is also important in constructing cryptographic primitives. One-way functions could be based on the worst-case hardness of GapSVP, such as the earliest one made by Ajtai[1]. As time went by, cryptosystems based on the hardness of GapSVP came out. Regev[5] built the first cryptosystem based on GapSVP, assuming approximating GapSVP was hard even by quantum algorithms. After that Peikert[10] constructed a cryptosystem based on the hardness of GapSVP under classical reductions. From those works, we can see the complexity of uSVP and GapSVP are the hardness foundation of current lattice cryptosystems and one-way functions. Among all the lattice problems, shortest vector problem is **NP**-hard under random reduction which is proved by Micciancio[24]. It could be randomly reduced from a special version of CVP which is **NP**-hard under deterministic reduction. This work could be regarded as the fundamental complexity result of lattice problems as SVP is the core problem in lattice. Nowadays, algorithms designed to solve SVP still has exponential complexity. Ajtai, Kumar and Sivakumar[6] gave a randomized algorithm running in time (and space) $2^{O(n)}$, typically referred to as the AKS Sieve. Sieve algorithm has been improved recently by Nguyen and Vidick[15], but it still has single exponential time complexity. The deterministic algorithm known with lowest time complexity, $\tilde{O}(2^{2n})$, is discovered by Micciancio[23]. Its complexity is even lower than the best sieve algorithm (not heuristic). But the experimental facts still show that sieve algorithm runs faster practically. High time complexity shows that lattice problem is indeed hard in practice. However Although SVP is **NP**-hard, the complexity of uSVP and GapSVP are still not very explicit. Lyubashevsky and Micciancio[17] proved the reduction from $\text{BDD}_{1/2\gamma}$ to uSVP_γ, uSVP_γ to GapSVP_γ and GapSVP_γ to $\text{BDD}_{\frac{1}{\gamma}\sqrt{n/\log n}}$, which shows the relationship between uSVP and GapSVP. But their relationship with SVP are not very clear.

Kumar and Sivakumar[22] first studied how to make short vector unique using randomized method in polynomial time. Their work proved that short vector could be made unique using randomized method. They constructed a procedure to knock out some of the short vectors so that only one short vector will be left at last probabilistically. Some interesting properties of short vectors are also given in their work. However, the short vector given out by their procedure may not be the shortest one in the original lattice. Indeed, with high probability the short vector left is not the shortest one, because there are so many short vectors. The one left could be any short vector shorter than $\sqrt{2}$ times of the length of the shortest vector. As a result, their procedure do not reveal an reduction from SVP to uSVP. Aggarwal and Dubey[19] found a deterministic method to make the shortest vector unique, a deterministic reduction from SVP to uSVP, using basis transformation which could distinguish vectors with the same length. This is the first deterministic reduction. However, although the shortest vector could be made unique, the gap between λ_1 and λ_2 is very small. Our work is also based on this matrix transformation method, enlarging the gap for a little bit. And it is still an interesting question whether there exists a better method to make the gap even larger.

Bit security is an important topic in cryptology. Hard problems always have hard bits, such as the discrete log problem which is analyzed by Peralta[14]. He proved the last bit unpredictable and next-bit unpredictability property of discrete log. The major method used by Peralta is to recover the bits of discrete log bit by bit using oracle. As lattice problems have been more and more important in cryptology, it is of theoretical and practical interest to identify the hard bits of short vectors. Theoretically, it is just interesting to see that getting those hard bits is just as hard as the whole problem. Practically, those hard bits could be used to generate pseudo random numbers. Hard bits of lattice problem are rarely studied before. In lattice, vector could be represented in at least two forms. In one form, it is just represented as a vector. In the other form, it is represented as a vector of coefficients. For example, if $\mathbf{u}$ is the shortest vector of $\mathcal{L}(B)$, it could be represent as u or c where $\mathbf{u} = B\mathbf{c}$. $\mathbf{c}$ seems to be better than u as it is an integer vector and could reveal the relationship of the shortest vector and its basis. Vectors are stored in computer as strings of binary bits. While it seems difficult to identify which bits in $\mathbf{u}$ are difficult, we find it is easy to identify some hard bits in c. There are n bit of c are difficult where n is the dimension of B. Not only SVP has this kind of property, it seems that uSVP and many other problem also has this kind of property. As the bit unpredictable property is revealed, the next interesting question is whether we could use this kind of property to make pseudo random number generator. It is still an open problem.

In the book "Complexity of Lattice Problems, A Cryptographic Perspective"[24] written by Micciancio, GapSVP$_\gamma$ with $\gamma < \sqrt{2}$ is **NP**-hard under random reduction. The major method is called embedding, finding a special lattice in sphere packing and embedding it to the original lattice basis so that BinCVP is reduced to SVP by constructing a new basis. However, it is still a challenge to find a deterministic reduction. Though it is not easy to find a deterministic method, there are still some interesting work to do. One work is to make it clear the relationship of Approximating SVP and GapSVP. It is a search version to decision version reduction and the focus is on how much the gap is changed. We studied the problem, giving some interesting results.

Our contribution. In section 3 of this paper, we proves that finding the approximate shortest vector with length in $[\lambda_1, \gamma\lambda_1]$ could be reduced to GapSVP$_{\tilde{\gamma}}$ with a small gap.

In section 4, we directly reduce SVP to GapSVP using deterministic method. The gap of GapSVP is still very small.

In section 5, we proved SVP could be reduced to uSVP. Our result is slightly better than [19]. We also proved that the search version of uSVP could be reduced to its decision version, with almost the same gap.

In section 6, we proved a bit-unpredictable property of SVP. Some bits of coefficients of the shortest vector are hard to obtain.

2 Preliminaries

.

2.1 Basic Notations

The sets of real numbers and integers are denoted by $\mathbb{R}$ and $\mathbb{Z}$ respectively. Vectors are represented as lower-case letters, e.g. $\mathbf{x}$. Matrix and basis are denoted by upper-case letters, e.g. B. $\mathcal{L}(B)$ denote the lattice generated by basis B. For a vector $\mathbf{x}$, the ith coordinate is denoted by x_i. The inner product between $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is $\mathbf{x} \cdot \mathbf{y} = \langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i y_i$.

The l_p norm of $\mathbf{x}$ is $\|\mathbf{x}\|_p = (\sum_{i=1}^{n} |x_i|^p)^{1/p}$ for any $p \in [1, \infty)$ and the l_∞ norm is $\|\mathbf{x}\|_\infty = \max_{1 \leq i \leq n} |x_i|$. We omit the subscript when $p = 2$.

2.2 Definitions and Useful Lemmas

Given a base of n-dimensional space, a Lattice could be formulated by linear combination of base vectors using integer coefficients.

Definition 1 (Lattice)**.** *Given n linearly independent vectors $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n \in \mathbb{R}^m$, the lattice generated by them is defined as*

$$\mathcal{L}(\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n) = \{\sum_{i=1}^{n} x_i \mathbf{b}_i | x_i \in \mathbb{Z}\}.$$

We refer to $\mathbf{b}_1, \ldots, \mathbf{b}_n$ as a basis of the lattice. Equivalently, if we define B as the $m \times n$ matrix whose columns are $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n$, then the lattice generated by B is

$$\mathcal{L}(B) = \mathcal{L}(\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n) = \{B\mathbf{x} | \mathbf{x} \in \mathbb{Z}^n\}.$$

Definition 2 (Span)**.** *The span of a lattice $\mathcal{L}(B)$ is the linear space spanned by its vectors,*

$$span(\mathcal{L}(B)) = span(B) = \{B\mathbf{y} | \mathbf{y} \in \mathbb{R}^n\}.$$

Definition 3 (Shortest Vector Problem (SVP))**.** *Given a basis $B \in \mathbb{R}^{m \times n}$, find a nonzero lattice vector $B\mathbf{x}$(with $\mathbf{x} \in \mathbb{Z}^n \setminus \{0\}$) such that*

$$\|B\mathbf{x}\| \leq \|B\mathbf{y}\|$$

for any other $\mathbf{y} \in \mathbb{Z}^n \setminus \{0\}$.

Always, researchers use λ_i (also $\lambda_i(B)$ with respect to basis B) to denote the ith shortest vector in a lattice. Pay attention that λ_0 is 0, and λ_1 is the shortest vector. I also use Λ to denote a lattice.

GapSVP is defined as the following.

Definition 4 (GapSVP$_\gamma$)**.** *The input consists of $B \in \mathbb{Z}^{m \times n}$ and $r \in \mathbb{Q}$.*

- *In YES instances, $\lambda_1(\mathcal{L}(B)) \leq r$.*
- *In NO instances, $\lambda_1(\mathcal{L}(B)) > \gamma \cdot r$.*

Now I give the definition of the problem finding the approximate shortest vector problem of a lattice.

Definition 5 (SVP$_\gamma$)**.** *Given lattice $\mathcal{L}(B)$ where basis $B \in \mathbb{R}^{m \times n}$, find the vector v such that $\|v\| \in [\lambda_1(B), \gamma\lambda_1(B))$.*

Definition 6 (Unique Shortest Vector Problem(uSVP$_\gamma$)). *Given a lattice B such that $\lambda_2(B) > \gamma\lambda_1(B)$, find a nonzero vector $v \in \mathcal{L}(B)$ of length $\lambda_1(B)$.*

In the following passage, without loss of generality, assume that the input basis B is full rank and is an integer matrix.

The LLL basis reduction algorithm[16], on input a lattice basis, outputs a basis for the same lattice such that $\|\mathbf{b}_{i+1}^*\|^2 \geq \frac{\sqrt{3}}{2}\|\mathbf{b}_i^*\|^2$ for all i. It runs in polynomial time. A useful facts implied by LLL algorithm is the following lemma, proposed in [19].

Lemma 1. *For a LLL reduced basis $B = [\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_n]$, if $\mathbf{u} = \sum_i \alpha_i \mathbf{b}_i$ is a shortest vector, then $|\alpha_i| < 2^{3n/2}$ for all $i \in [n]$.*

In the following passage, we will improve this lemma, to make the upper bound smaller.

3 The Reduction from SVP$_\gamma$ to GapSVP$_{\tilde{\gamma}}$

In this section, the first result of this paper is given.

3.1 The Capability of GapSVP Oracle

Lemma 2. *Given Lattice Basis $B \in \mathbb{Z}^{n\times n}$, using GapSVP$_\gamma$ oracle, a range $(\alpha, \gamma\alpha]$, $\alpha \in \mathbb{R}$, could be found such that $\lambda_1(B) \in (\alpha, \gamma\alpha]$.*

Proof. We propose an algorithm just like binary search to prove this lemma.

First, pick a real number α_0 such that $\alpha_0 \geq \sqrt{n}\det(B)^{1/n} \geq \lambda_1$. Run GapSVP$_\gamma$ oracle on input instance $\langle B, \alpha_0\rangle$. If it returns "yes", then set $\alpha_1 = \alpha_0/2$. And the range of λ_1 reduces to $(0, \gamma\alpha_0]$. It's impossible for the oracle to return "no" at this step.

Each time we got $\alpha_i, i \geq 0$, we do the following operations. Check the result of GapSVP$_\gamma$ oracle on input instance $\langle B, \alpha_i\rangle$. If it returns "yes", then set $\alpha_{i+1} = \alpha_i/2$. The range of λ_1 reduces to $(0, \gamma\alpha_i]$. We redo this step until the oracle returns "no".

Once the oracle returns "no", the situation needs to be discussed. We enter the second part of our algorithm. Here, we just consider situation of the most difficult input instance, because other input instance will lead us to get a even smaller range of λ_1.

To be more precisely, assume the oracle returns "no" on input $\langle B, \alpha_{k+1}\rangle$. The range of λ_1 is $(\alpha_{k+1}, \gamma\alpha_k]$. Two situations are here.

- λ_1 is in $(\alpha_k, \gamma\alpha_k]$.
- λ_1 is in $(\alpha_{k+1}, \alpha_k]$.

No matter in which situation, next step, we will set $\alpha_{k+2} = (\alpha_{k+1} + \alpha_k)/2$. On input $\langle B, \alpha_{k+2}\rangle$, if the oracle returns "no", then the range reduces to $(\alpha_{k+2}, \gamma\alpha_k]$. If the oracle returns "Yes", then the range reduces to $(\alpha_{k+1}, \gamma\alpha_{k+2}]$.

Then at each of the following step (considering the $\tilde{k}$th step), we set $\alpha_{\tilde{k}} = (a + b/\gamma)/2$, where $(a, b]$ denotes the range we got of λ_1 at hand. Run GapSVP oracle on $\langle B, \alpha_{\tilde{k}}\rangle$. No matter what the result is, the range reduces. Renew the value of a, b. Do the $\tilde{k} + 1$th step the same way as we do the $\tilde{k}$th until, $b < \gamma a$. Let $\alpha = a$. Finally, we got that $\lambda_1(B) \in [\alpha, \gamma\alpha]$.

$\lambda_1 \leq 2^{p(m)}$, where m is the length of input. As a result, the proposed algorithm could be done in polynomial time of input length. This algorithm directly proves the lemma. □

3.2 The Reduction

We give the following theorem to reduce the approximation version of SVP to GapSVP. The method is just adapted from the proof reducing uSVP to GapSVP in [17].

Theorem 1. *For any given $\gamma \geq 1$, Approximate $SVP_\gamma \leq_p GapSVP_{\tilde{\gamma}}, \tilde{\gamma} = \gamma^{\frac{1}{n(n+\log_2(\gamma n))(n-1)}}$.*

Proof. Given the input instance $B = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ which is the basis of a lattice. Now I will show the algorithm to compute a vector $\mathbf{v}$ such that $\|\mathbf{v}\| \leq \gamma\lambda_1(B)$.

The main idea is to obtain lower rank sublattice of $\mathcal{L}(B)$ such that the approximate shortest vector are still in the sublattice.

As $\mathcal{L}(B)$ is n dimensional lattice, we only need to lower the rank by $n-1$ times.

To be more precisely, for the given lattice basis $\mathcal{L}(B)$, we find a serial of sublattices with rank decreased gradually. Assume that the serial of sublattices are denoted by $B_1, \ldots, B_n$, where $B_1 = B$ and $\text{rank}(B_i) - 1 = \text{rank}(B_{i+1})$.

We use the following method to lower the rank by 1.

Now I describe how to obtain B_{i+1} from B_i.

Given basis B_i, applying the method proposed in lemma 2, we could found the range $r = [\alpha, \tilde{\gamma}\alpha]$ where $\lambda_1(B_i)$ is in.

Generate three sublattices of $\tilde{B}_0 = B_i$. They are $\hat{B}_0 = [2\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k]$, $\hat{B}_c = [\mathbf{b}_1 + c\mathbf{b}_2, 2\mathbf{b}_2, \mathbf{b}_3, \ldots, \mathbf{b}_k], c = 1, 2$. It could be found that the shortest vector of $\tilde{B}_0$ is in one of the three generated sublattices. Apply the method proposed in lemma 2, we could find the range containing the length of shortest vector for each of the three sublattice. Assume the three ranges we got are $r_j, j = 0, 1, 2$. At least one sublattices has r_j intersecting r. Set $\tilde{B}_1$ to be $\hat{B}_j$ if r_j intersects r. If more than one sublattice has r_j intersecting r, set $\tilde{B}_1$ to be arbitrarily one of them. We do this for t times to get a serial of sublattices, where

$$\mathcal{L}(\tilde{B}_0) \supset \mathcal{L}(\tilde{B}_1) \supset \cdots \supset \mathcal{L}(\tilde{B}_t)$$

.

Here, $t > n(n + \log_2(\gamma n))$.

It could be concluded that $\lambda_1(\tilde{B}_t) \leq \tilde{\gamma}^t \lambda_1(\tilde{B}_0)$. Also we know that $\det(\tilde{B}_t) \geq 2^t \det(\tilde{B}_0)$, because each time we select a sublattice the value of the determinant at least doubles. Assume D to be the dual of $\tilde{B}_t$. $\det(D) \leq 1/(2^t \det(\tilde{B}_0))$. Applying the LLL algorithm we can find a vector $\mathbf{u} \in \mathcal{L}(D)$ such that

$$\|\mathbf{u}\| \leq 2^n \sqrt{n} \det(D)^{1/n} \leq \frac{\sqrt{n}2^n}{2^{t/n} \det(\tilde{B}_0)^{1/n}}.$$

Suppose the shortest vector of $\mathcal{L}(\tilde{B}_0)$ is $\tilde{u}_0$. According to Minkowski's bound, we have $\|\tilde{\mathbf{u}}_0\| \leq \sqrt{n} \det(\tilde{B}_0)^{1/n}$. Consequently, the shortest vector of $\tilde{B}_t$ (suppose to be $\tilde{\mathbf{u}}_t$) meet the following bound.

$$\|\tilde{\mathbf{u}}_t\| \leq \tilde{\gamma}^t \sqrt{n} \det(\tilde{B}_0)^{1/n}$$

Using Cauchy-Schwarz inequality,

$$|\langle \tilde{\mathbf{u}}_t, u \rangle| \leq \|\tilde{\mathbf{u}}_t\| \cdot \|\mathbf{u}\| \leq \tilde{\gamma}^t n 2^{n-t/n} < 1.$$

We know $\tilde{\mathbf{u}}_t \in \mathcal{L}(\tilde{B}_t)$, $\mathbf{u} \in \mathcal{L}(D)$. It means that $\langle \tilde{\mathbf{u}}_t, \mathbf{u} \rangle$ is an integer. This concludes that $|\langle \tilde{\mathbf{u}}_t, \mathbf{u} \rangle| = 0$. Taking the sublattice of $\tilde{B}_t$ orthogonal to $\mathbf{u}$, we get a lower rank sublattice $\mathcal{L}(B_{i+1}) \subset \mathcal{L}(B_i)$ such that $\lambda_1(B_{i+1}) \leq \tilde{\gamma}^t \lambda_1(B_i)$.

Finally, after lowering rank for $n-1$ times, we could finally got $\mathcal{L}(B_n)$ such that its rank is 1 which mean its shortest vector could be found trivially. Also we have

$$\lambda_1(B_n) \leq \tilde{\gamma}^{(n-1)t} \lambda_1(B).$$

We know $\tilde{\gamma} = \gamma^{\frac{1}{n(n+\log_2(\gamma n))(n-1)}}$. As a result, the final conclusion is

$$\lambda_1(B_n) \leq \gamma \lambda_1(B).$$

This completes our proof. □

4 The Reduction from SVP to GapSVP

This proof also uses the method adapted from the method in [17]. We just make a little adjustment.

Theorem 2. *SVP could be cook-reduced to GapSVP$_\gamma$, where $\gamma = \sqrt{1 + \frac{1}{\lambda_1(\mathcal{L}(B))^2}}$.*

Proof. Given the input instance $B = [\mathbf{b}_1, \ldots, \mathbf{b}_n]$ which is the basis of a lattice. The following algorithm computes the shortest vector of $\mathcal{L}(B)$, saying $\mathbf{v}$, using GapSVP oracle.

The main idea is to obtain a lower rank sublattice of $\mathcal{L}(B)$ such that the shortest vector are still in the sublattice.

As $\mathcal{L}(B)$ is n dimensional lattice, we need to lower the rank by $n-1$ times.

To be more precisely, for the given lattice basis $\mathcal{L}(B)$, we find a serial of sublattices with rank decreased gradually. Assume that the serial of sublattices are denoted by $B_1, \ldots, B_n$, where $B_1 = B$ and $\operatorname{rank}(B_i) - 1 = \operatorname{rank}(B_{i+1})$.

We use the following method to lower the rank by 1. The method describes how to obtain B_{i+1} from B_i.

Given basis B_i, applying the method proposed in lemma 2, we could found the range $r = (\alpha, \gamma\alpha]$ which $\lambda_1(B_i)$ is in.

Generate three sublattices of $\tilde{B}_0 = B_i$. They are $\hat{B}_0 = [2\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_k]$, $\hat{B}_c = [\mathbf{b}_1 + c\mathbf{b}_2, 2\mathbf{b}_2, \mathbf{b}_3, \ldots, \mathbf{b}_k], c = 1, 2$. It could be found that the shortest vector of $\tilde{B}_0$ is in at least one of the three generated sublattices. Apply the method proposed in lemma 2, we could find the range containing the length of shortest vector for each of the three sublattice. Assume the three ranges we got are $r_j, j = 0, 1, 2$. At least one of them contain $\lambda_1(B)$. Assume $\lambda_1(B)$ is in r_i, corresponding to $\mathcal{L}(\hat{B}_i)$. If $\mathcal{L}(\hat{B}_j)$ do not have the shortest vector of $\mathcal{L}(B)$, then $\lambda_1 \notin r_j$. This is because, according to lemma 2, if $\lambda_1 \in r_j$, r_j will not contain $\lambda_1(\hat{B}_j)$, as the gap of the oracle is $\sqrt{1 + \frac{1}{\lambda_1(\mathcal{L}(B))^2}}$ and $\lambda_1(\hat{B}_j)^2 \geq \lambda_1(B)^2 + 1$.

$$\sup\{x | x \in r_j\} < \gamma \lambda_1(B) \leq \lambda_1(\hat{B}_j)$$

As a result, we could find the sublattice $\mathcal{L}(\hat{B}_i)$ which contains the shortest vector. Set $\tilde{B}_1 = \hat{B}_i$. We do this for t times to get a serial of sublattices, where

$$\mathcal{L}(\tilde{B}_0) \supset \mathcal{L}(\tilde{B}_1) \supset \cdots \supset \mathcal{L}(\tilde{B}_t)$$

.

Here, $t > n(n + \log_2 n)$.

It could be concluded that $\lambda_1(\tilde{B}_t) = \lambda_1(B_0)$. Also we know that $\det(\tilde{B}_t) \geq 2^t \det(\tilde{B}_0)$, because each time we select a sublattice the value of the determinant at least doubles. Assume D to be the dual of $\tilde{B}_t$. $\det(D) \leq 1/(2^t \det(\tilde{B}_0))$. Applying the LLL algorithm we can find a vector $\mathbf{u} \in \mathcal{L}(D)$ such that

$$\|\mathbf{u}\| \leq 2^n \sqrt{n} \det(D)^{1/n} \leq \frac{\sqrt{n} 2^n}{2^{t/n} \det(\tilde{B}_0)^{1/n}}.$$

According to Minkowski's bound,

$$\|\mathbf{v}\| \leq \sqrt{n} \det(\tilde{B}_0)^{1/n}$$

Using Cauchy-Schwarz inequality,

$$|\langle \mathbf{v}, \mathbf{u} \rangle| \leq \|\mathbf{v}\| \cdot \|\mathbf{u}\| < 1.$$

As $|\langle \mathbf{v}, \mathbf{u} \rangle|$ is an integer, $|\langle \mathbf{v}, \mathbf{u} \rangle| = 0$. Taking the sublattice of $\tilde{B}_t$ orthogonal to $\mathbf{u}$, we get a lower rank sublattice $\mathcal{L}(B_{i+1}) \subset \mathcal{L}(B_i)$ such that $\lambda_1(B_{i+1}) = \lambda_1(B)$.

Finally, after lowering rank for $n-1$ times, we could finally got $\mathcal{L}(B_n)$ such that its rank is 1 which mean the shortest vector could be found trivially.

□

5 Complexity Results of Unique Shortest Vector Problem

5.1 Reduction from SVP to uSVP

In [19], it is proved that SVP $\leq_p$ uSVP$_\gamma, \gamma = \sqrt{1 + \frac{1}{c \cdot 2^{4n^2} \lambda_1^2}}$. This could be improved.

Lemma 3. *For basis $B = [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n]$, it could be reduced to B' using slide reduction[21] such that, if $\mathbf{u} = \sum_i \alpha_i \mathbf{b}'_i$ is a shortest vector, then $|\alpha_i| < cn(\frac{3}{2})^{n-i} \cdot (r_k(1+\epsilon))^{\frac{pk}{k-1}}$,where r_k denotes the hermite constant, k is the block size $(O(\log n/(\log\log n)))$ and $p = n/k$.*

If using LLL reduction the corresponding conclusion is $\forall i \in [n], |\alpha_i| < 2^{n/2}(\frac{3}{2})^{n-i}$.

Proof. This could be done according to the slide reduction method proposed in [21]. Here I will explain why this basis reduction will meet the proposed property. The reduction method proposed in [21] require the dimension to be $n = pk$, here I made a little adjustment so that n could be any positive integer.

A basis B of an n-dimensional lattice L where $n = kp + q$ is slide reduced with a factor $\varepsilon \geq 0$ if it is size-reduced and satisfies the following two conditions.

- $\forall i \in [0, p-1]$, the block $B_{[ik+1, ik+k]}$ is HKZ-reduced.
- $\forall i \in [0, p-1]$, the block $B_{[ik+2, ik+k+1]}$ is $(1+\epsilon)$-DSVP-reduced. (If $q = 0$, $i \in [0, p-2]$.)

As a result, we have $\|\mathbf{b}^*_{ik+1}\| \leq (r_k(1+\epsilon))^{\frac{k}{k-1}} \|\mathbf{b}^*_{ik+k+1}\|$.

This induces $\|\mathbf{b}^*_1\| \leq (r_k(1+\epsilon))^{\frac{ik}{k-1}} \|\mathbf{b}^*_{ik+1}\| \Rightarrow \|\mathbf{b}^*_1\| \leq (r_k(1+\epsilon))^{\frac{pk}{k-1}} \|\mathbf{b}^*_{pk+1}\|$.

According to [21] B' is LLL-reduced. We have

$$\|\mathbf{b}^*_1\| \leq (r_k(1+\epsilon))^{\frac{pk}{k-1}} \|\mathbf{b}^*_{pk+1}\| \leq (\sqrt{\frac{4}{3}}(1+\epsilon))^{q-1} (r_k(1+\epsilon))^{\frac{pk}{k-1}} \|\mathbf{b}^*_n\|$$

$$\|\mathbf{b}_1\| = \|\mathbf{b}^*_1\| \geq \|\mathbf{u}\| \geq |\alpha_n| \|\mathbf{b}^*_n\| \geq ((\sqrt{\frac{4}{3}}(1+\epsilon))^{q-1} (r_k(1+\epsilon))^{\frac{pk}{k-1}})^{-1} |\alpha_n| \|\mathbf{b}^*_1\|$$

This implies $|\alpha_n| \leq (\sqrt{\frac{4}{3}}(1+\epsilon))^{q-1} (r_k(1+\epsilon))^{\frac{pk}{k-1}}$. $q < k = O(\log n/\log\log n)$, so $(\sqrt{\frac{4}{3}}(1+\epsilon))^{q-1}$ is a linear polynomial of n.

Suppose the lemma holds for $\alpha_i, \forall i > n-l$. According to Gram-Schmidt orthogonal method, we have the following.

$$\begin{aligned} \|\mathbf{b}^*_1\| \geq \|\mathbf{u}\| &\geq |\alpha_{n-l} + (\sum_{j=n-l+1}^{n} \mu_{j,n-l}\alpha_j)| \|\mathbf{b}^*_{n-l}\| \\ &\geq \{(r_k(1+\epsilon))^{\frac{\lfloor (n-l-1)/k \rfloor}{k-1}} (\sqrt{\frac{4}{3}}(1+\epsilon))^{(n-l-1) \mod k}\}^{-1} \\ &\quad \cdot |\alpha_{n-l} + (\sum_{j=n-l+1}^{n} \mu_{j,n-l}\alpha_j)| \|\mathbf{b}^*_1\| \end{aligned} \tag{1}$$

$\sqrt{\frac{4}{3}}(1+\epsilon))^{(n-l-1) \mod k}$ is also a linear polynomial of n. Suppose it is less than $c_2 n$. We also knows that $\forall 1 \leq j < i \leq n, |\mu_{i,j}| \leq 1/2$.

$$\begin{aligned}|\alpha_{n-l}| &\leq c_2 n(r_k(1+\epsilon))^{\frac{\lfloor (n-l-1)/k \rfloor}{k-1}} + (\sum_{j=n-l+1}^{n} |\mu_{j,n-l}\alpha_j|) \\ &\leq c_2 n(r_k(1+\epsilon))^{\frac{\lfloor (n-l-1)/k \rfloor}{k-1}} + 1/2 \sum_{j=n-l+1}^{n} |\alpha_j|\end{aligned} \tag{2}$$

$$c_2 n(r_k(1+\epsilon))^{\frac{\lfloor (n-l-1)/k \rfloor}{k-1}} + 1/2 \sum_{j=n-l+1}^{n} |\alpha_j| \leq c_3 n(r_k(1+\epsilon))^{\frac{pk}{k-1}}(\frac{3}{2})^{n-l}$$

As a result, we could conclude that $\forall i, |\alpha_i| < cn(\frac{3}{2})^{n-i} \cdot (r_k(1+\epsilon))^{\frac{pk}{k-1}}$.

Using similar methods, we could get the corresponding conclusion for LLL, $\forall i \in [n], |\alpha_i| < 2^{n/2}(\frac{3}{2})^{n-i}$.

□

Theorem 3. *If the shortest vector* $\mathbf{u}$ *of the input lattice* $\mathcal{L}(B)$ *could be denoted as* $\mathbf{u} = \sum_i \alpha_i \mathbf{b}_i$, *knowing* $|\alpha_i| < t_i$, *then* $SVP \leq_p uSVP_\gamma$, $\gamma = \sqrt{1 + \frac{1}{c(\prod_{i=1}^{n} t_i)^2 \lambda_1(\mathcal{L}(B))^2}}$.

Proof. Suppose $m_j = \prod_{i=1}^{j} t_i$. Denote $m_0 = 1$.

Consider the following matrix.

$$B' = \begin{pmatrix} m_n\mathbf{b}_1 & m_n\mathbf{b}_2 & \dots & m_n\mathbf{b}_{n-1} & m_n\mathbf{b}_n \\ 1 & & & & \\ & m_1 & & & \\ & & \dots & & \\ & & & m_{n-2} & \\ & & & & m_{n-1} \end{pmatrix}$$

I will prove that, $\mathcal{L}(B')$ has a unique shortest vector corresponding to one of the shortest vector of $\mathcal{L}(B)$.

$$\forall \mathbf{x} \in \mathbb{Z}^n, \|B'\mathbf{x}\|^2 = m_n^2\|B\mathbf{x}\|^2 + \sum_{i=1}^{n}(m_{i-1}x_i)^2 < m_n^2(\|B\mathbf{x}\|^2 + 1)$$

If $\|B\mathbf{x}\| = \|B\mathbf{y}\| = \lambda_1(B)$, then $\exists k, \forall i, k < i \leq n, x_i = y_i, |x_k| > |y_k|$. We could see $\|B'\mathbf{x}\| > \|B'\mathbf{y}\|$. The reason follows.

It is easy to see that $(x_k m_{k-1})^2 - (y_k m_{k-1})^2 \geq m_{k-1}^2 = (t_{k-1}m_{k-2})^2 \geq (|y_{k-1}|m_{k-2})^2 + m_{k-2}^2 \geq \sum_{j=1}^{k-1}(|y_j|m_{j-1})^2 + m_0^2 > \sum_{j=1}^{k-1}(|y_j|m_{j-1})^2$. So we have that $|x_i| = |y_i|$.

According to [**?**], when two shortest vectors, say $B\mathbf{x}$ and $B\mathbf{y}$, have the same parity vector (for $B\mathbf{x}$ the parity vector is $p(\mathbf{x}) = [\mathbf{x}_1 \mod 2, \dots, \mathbf{x}_n \mod 2]$), then $B\mathbf{x} = B\mathbf{y}$. This implies if $\|B\mathbf{x}\| = \|B\mathbf{y}\| = \lambda_1, \forall i \in [n], |x_i| = |y_i|$, then $\forall i \in [n], x_i = y_i$.

As a result, we could see that there is only one unique shortest vector for $\mathcal{L}(B')$.

$\lambda_2^2(B') - \lambda_1^2(B') \geq 1$. So the gap is $\sqrt{1 + \frac{1}{c(\prod_{i=1}^{n} t_i)^2 \lambda_1(\mathcal{L}(B))^2}}$ □

Theorem 4. *SVP could be reduced to* $uSVP_\gamma$ *with*

$$\gamma = \sqrt{1 + \frac{1}{c_1(c_2 n)^{2n}(3/2)^{(n-1)n}(r_k(1+\epsilon))^{2n\frac{pk}{k-1}}\lambda_1(\mathcal{L}(B))^2}},$$

for some constant c_1, c_2. r_k *is the kth hermite constant.* k *is the block size* $(O(\log n/(\log \log n)))$ *and* $p = n/k$.

If using LLL reduction, we could get that SVP $\leq_p$ uSVP$_\gamma, \gamma = \sqrt{1 + \frac{1}{c2^{n^2}(\frac{3}{2})^{(n-1)n}\lambda_1^2}}$.

Proof. The theorem follows immediately from lemma 4 and theorem 5. □

5.2 Search versus Decision

We will show that the search version uSVP could be reduced to decision version uSVP maintaining almost the same gap.

In order to do this reduction, we adapted the methods of Kannan[20] and the methods of Hu and Pan[18].

Both of the two methods aimed to reduce SVP to decisional SVP. However, the parameters in their methods are very large. If just apply their methods to do the reduction we cannot get results better than [19]. we come up with the following method.

Lemma 4. *Given the value of an integer r, knowing $r = mp_{n+1} + \sum_{i=1}^{n} \alpha_i p_i$, where $p_i | p_{i+1}$, $\alpha_i < \lfloor \frac{p_{i+1}}{p_i} \rfloor$, α_i $(i \in [n])$ and m could be computed. Here, $m \geq 0, \forall i, p_i > 0, \alpha_i \geq 0$.*

Proof. First, we compute $r_n = r \mod p_{n+1}, m = r/p_{n+1}$. Once we have r_i, we compute $r_{i-1} = r_i \mod p_i$, $\alpha_i = r_i/p_i$. In this way, we could compute $\alpha_i, (i \in [n])$ one by one. □

Lemma 5. *Using duSVP oracle, the exact length of the shortest vector of the given input lattice could be found.*

Proof. This could be done using binary search.

According to Minkowski's bound, we have the following bound for shortest vector $\mathbf{u}$ of input lattice $\mathcal{L}(B)$.

$$\|\mathbf{u}\| \leq \sqrt{n} \det(B)^{1/n}$$

First we just take $\langle B, d \rangle$, where $d = \sqrt{n} \det(B)^{1/n}$, as the input for duSVP oracle. Set the original range of $\lambda_1(B)$ to be $[a, b] = [0, d]$ (means $a = 0, b = d = \sqrt{n} \det(B)^{1/n}$). We do the following iteration.

For each time run the duSVP oracle on $\langle B, d \rangle, d = (a + b)/2$. If it returns "Yes", then the range of $\lambda_1(B)$ is set to be $[a, b] = [a, d]$, else set the range to be $[a, b] = [d, b]$. Finally, the length of $\lambda_1(B)$ could be settled in polynomial time of the input length. □

Theorem 5.

$$\textit{search-uSVP}_\gamma \leq_p \textit{decision-uSVP}_{\gamma\sqrt{1-\epsilon}}$$

Proof. According to lemma 4, we could assume that we have the oracle O which could output the length of the unique shortest vector given any input lattice basis with gap γ'.

Now, given the input lattice basis B, we construct the following new lattice.

$$B' = LLL(B)$$

$$B'' = \begin{pmatrix} m_n\mathbf{b}'_1 & m_n\mathbf{b}'_2 & \dots & m_n\mathbf{b}'_{n-1} & m_n\mathbf{b}'_n \\ 1 & & & & \\ & m_1 & & & \\ & & \dots & & \\ & & & m_{n-2} & \\ & & & & m_{n-1} \end{pmatrix}$$

$t_i = 2^{n/2}(\frac{3}{2})^{n-i}$, $m_i = \prod_{j=1}^{i} t_j$. We already know that if $\mathbf{u} = \sum_{i=1}^{n} \alpha_i \mathbf{b}'_i$ then $\alpha_i < t_i$. Assume $m_0 = 1$.

If $B''\mathbf{x}$ is the shortest vector of $\mathcal{L}(B'')$ then, $B'\mathbf{x}$ is the shortest vector of $\mathcal{L}(B')$. If not, assume $B'\mathbf{y}$ is the shortest vector of $\mathcal{L}(B')$. That is $\|B'\mathbf{y}\| < \|B'\mathbf{x}\|$. It means $\|B'\mathbf{x}\|^2 - \|B'\mathbf{y}\|^2 \geq 1$. Consider the vector $B''\mathbf{y}$ in $\mathcal{L}(B'')$. $\|B''\mathbf{y}\|^2 = m_n^2\|B'\mathbf{y}\|^2 + \sum_{i=1}^n(\mathbf{y}_i m_{i-1})^2 < m_n^2\|B'\mathbf{x}\|^2 < \|B''\mathbf{x}\|^2$. This contradicts $B''\mathbf{x}$ is the shortest vector of $\mathcal{L}(B'')$.

We could also know that $\mathcal{L}(B'')$ has a unique shortest vector. If not, assume that $B''\mathbf{x}$,$B''\mathbf{y}$ are two shortest vector $B''\mathbf{x} \neq \pm B''\mathbf{y}$. $\|B''\mathbf{x}\|^2 = m_n^2\|B'\mathbf{x}\| + \sum_{i=1}^n(x_i m_{i-1})^2$. It should be $\|B''\mathbf{x}\| = \|B''\mathbf{y}\|$. So we have $\|B'\mathbf{x}\| = \|B'\mathbf{y}\|$. It means both $B'\mathbf{x}$ and $B'\mathbf{y}$ are the shortest vector of $\mathcal{L}(B')$. This is impossible, as $\mathcal{L}(B')$ a unique shortest vector.

Suppose $B''\mathbf{x}$ is the shortest vector of $\mathcal{L}(B'')$. Using our oracle, we could get $\lambda_1(B'')$. According the above lemma, we could get $|x_i|, i = 1, \ldots, n$.

Now we compute the sign for each x_i.

Construct the following basis.

$$\tilde{B} = \begin{pmatrix} m_1x_1\mathbf{b}'_1 & m_1x_2\mathbf{b}'_2 & \ldots & m_1x_{n-1}\mathbf{b}'_{n-1} & m_1x_n\mathbf{b}'_n \\ 1 & -1 & & & \end{pmatrix}$$

Assume that $x_1 > 0$, $x_2 \neq 0$. Now we compute the sign of x_2. It is easy to see that $\tilde{B}$ has unique shortest vector.

Run O on $\tilde{B}$. We get $\lambda_1(\tilde{B})$. According to lemma 3 and 4, we could get $\lambda_1(\tilde{B}) \mod m_1$. If it is 0, we know x_2 is positive, else it is negative. In this way, all the sign of x_i could be got. So we could get the shortest vector of B.

Next we analysis the gap that O need.

Denote the gap between λ_1 and λ_2 of $\mathcal{L}(B'')$ to be γ''.

If $\gamma' < \gamma''$, we could run O on B''.

$$\gamma'' = \sqrt{\frac{(\lambda_2'')^2}{(\lambda_1'')^2}} > \sqrt{\frac{\lambda_2^2}{\lambda_1^2 + 1}} = \gamma\sqrt{\frac{\lambda_1^2}{\lambda_1^2 + 1}}$$

Set $\gamma' = \gamma\sqrt{\frac{\lambda_1^2}{\lambda_1^2+1}} = \gamma\sqrt{1 - \frac{1}{\lambda_1^2+1}} = \gamma\sqrt{1-\epsilon}$.

We also use oracle O in the step computing the sign of x_i. In the same way it could be proved that $\gamma' = \gamma\sqrt{1-\epsilon}$ is a suitable gap in this step.

As a result,

$$\text{search-uSVP}_\gamma \leq_p \text{decision-uSVP}_{\gamma\sqrt{1-\epsilon}}.$$

The proof is complete.

□

6 Bit Unpredictable Property of SVP

In this section, we propose an interesting bit unpredictable property of SVP.

Theorem 6. *For lattice $\mathcal{L}(B)$, an oracle O could compute the last significant bit of c_1, where $\mathbf{u} = \sum_{i=1}^n c_i\mathbf{b}_i$ is the unique shortest vector, $\lambda_2 > \gamma\lambda_1$. $\mathbf{u}$ could be found in polynomial time.*

Proof. The proof includes two major parts. In the first part, we will compute $|c_i|, i = 1, \ldots, n$. In the second part, we will compute the sign for each coefficient.

For the first part, compute the LLL reduced basis B' of B. According to lemma 3, we know that if $\mathbf{u} = \sum_{i=1}^n c_i\mathbf{b}'_i$ is the shortest vector then $|c_i| < \alpha_i = 2^{\frac{n}{2}}(\frac{3}{2})^{n-i} < 2^{\frac{n}{2}}(\frac{3}{2})^n = \alpha$. Construct the following basis as $\hat{B}_0$.

$$\hat{B}_0 = \begin{pmatrix} \tilde{\mathbf{b}}_1 & \tilde{\mathbf{b}}_2 & \ldots & \tilde{\mathbf{b}}_{n-1} & \tilde{\mathbf{b}}_n \end{pmatrix} = 2\alpha^2 \begin{pmatrix} \mathbf{b}'_1 & \mathbf{b}'_2 & \ldots & \mathbf{b}'_{n-1} & \mathbf{b}'_n \end{pmatrix}$$

The shortest vector of $\mathcal{L}(\hat{B}_0)$ is $\tilde{\mathbf{u}} = \sum_{i=1}^n c_i\tilde{\mathbf{b}}_i$.

Now we compute $|c_1|$.

Run our oracle O on $\hat{B}_0$. The output is o. Construct $\hat{B}_1$ to be

$$\begin{pmatrix} 2\tilde{\mathbf{b}}_1 & \mathbf{e} & \tilde{\mathbf{b}}_2 & \dots & \tilde{\mathbf{b}}_{n-1} & \tilde{\mathbf{b}}_n \\ 1 & & & & & \\ & \alpha & & & & \end{pmatrix}.$$

$$e = o\tilde{\mathbf{b}}_1$$

Run O on this basis. We could get the second last bit of c_1. Denote the jth last bit of c_i to be $c_{i,j}$. After we get $c_{1,j}$, we turn $\hat{B}_j$ to be the following.

$$\begin{pmatrix} 2^j\tilde{\mathbf{b}}_1 & \mathbf{e} & \tilde{\mathbf{b}}_2 & \dots & \tilde{\mathbf{b}}_{n-1} & \tilde{\mathbf{b}}_n \\ 1 & & & & & \\ & \alpha & & & & \end{pmatrix}.$$

Here,

$$\mathbf{e} = \mathbf{e} + c_{1,j} \cdot 2^{j-1}\tilde{b}_1.$$

Suppose $\mathbf{e} = k_j\tilde{b}, k > 0$ at this step.

Run O on this basis. We could get $c_{1,j+1}$. In this way, we could at last get $|c_1|$ as $|c_i| < \alpha_i = 2^{\frac{n}{2}}(\frac{3}{2})^{n-i}$.

Here we have to explain why the above steps are all right.

If $\mathbf{e} = \mathbf{0}$, then it is trivial. We will show the situation that $\mathbf{e} \neq \mathbf{0}$.

After we have got $c_{1,j}$,

$$\|\hat{B}_j\mathbf{x}\|^2 = \|2^j\tilde{\mathbf{b}}_1x_1 + \mathbf{e}x_2 + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ix_{i+1}\|^2 + x_1^2 + (\alpha x_2)^2.$$

Assume $\hat{B}_j\mathbf{x}$ is the shortest vector, shorter than $\hat{\mathbf{u}} = \hat{B}_j\mathbf{c'}$ which is a vector in $\mathcal{L}(\hat{B}_j)$.

$$\|\hat{\mathbf{u}}\|^2 = \|2^j\tilde{\mathbf{b}}_1c_1' + \mathbf{e}c_2' + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ic_i\|^2 + (c_1')^2 + (\alpha c_2')^2 = \tilde{\mathbf{u}}^2 + (c_1')^2 + (\alpha c_2')^2$$

Here,

$$2^j\tilde{\mathbf{b}}_1c_1' + e_2c_2' = c_1\tilde{\mathbf{b}}_1, c_1'c_2' >= 0$$

If $c_1 \geq 0$ then $c_2' = 1$, else $c_2' = -1$.

$$\sum_{i=1}^{n}\tilde{\mathbf{b}}_ic_i = 2^j\tilde{\mathbf{b}}_1c_1' + \mathbf{e}c_2' + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ic_i.$$

As a result, $|c_1'| \leq |c_1|$, $c_{1,j+1}$ is the last bit of c_1'.

There should be

$$\|\sum_{i=1}^{n}\tilde{\mathbf{b}}_ic_i\| = \|2^j\tilde{\mathbf{b}}_1c_1' + ec_2' + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ic_i\| \geq \|2^j\tilde{\mathbf{b}}_1x_1 + \mathbf{e}x_2 + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ix_{i+1}\|.$$

If not,

$$\|2^j\tilde{\mathbf{b}}_1x_1+\mathbf{e}x_2+\sum_{i=2}^{n}\tilde{\mathbf{b}}_ix_{i+1}\|^2 > \|2^j\tilde{\mathbf{b}}_1c_1'+\mathbf{e}c_2'+\sum_{i=2}^{n}\tilde{\mathbf{b}}_ic_i\|^2+4\alpha^4 > \|\sum_{i=1}^{n}\tilde{\mathbf{b}}_ic_i\|^2+(c_1')^2+(\alpha c_2')^2 = \|\hat{\mathbf{u}}\|^2.$$

Then $\hat{B}_jx$ is not the shortest. As $\tilde{\mathbf{u}}$ is the shortest vector of $\hat{B}_0$, $(2^j\tilde{\mathbf{b}}_1x_1 + \mathbf{e}x_2 + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ix_{i+1}) = \pm\sum_{i=1}^{n}\tilde{\mathbf{b}}_ic_i$. Without loss of generality, assume $(2^j\tilde{\mathbf{b}}_1x_1 + \mathbf{e}x_2 + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ix_{i+1}) = \sum_{i=1}^{n}\tilde{\mathbf{b}}_ic_i$. This implies $x_{i+1} = c_i, i = 2, \dots, n$ and $2^jx_1 + k_jx_2 = c_1$.

There also should be $(\alpha x_2)^2 \leq (\alpha c_2')^2$. If not, $(\alpha x_2)^2 \geq (\alpha c_2')^2 + \alpha^2 > (\alpha c_2')^2 + (c_1')^2$. This implies $\hat{B}_j\mathbf{x}$ is not the shortest. x_2 could not be 0, because if it is 0,

$$2^j\tilde{\mathbf{b}}_1c_1' + \mathbf{e}c_2' + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ic_i \neq 2^j\tilde{\mathbf{b}}_1x_1 + \mathbf{e}x_2 + \sum_{i=2}^{n}\tilde{\mathbf{b}}_ix_{i+1}$$

so $\hat{B}_j\mathbf{x}$ could not be the shortest. This proves $\|(\alpha x_2)\| = \|(\alpha c_2')\|$, $x_2 = \pm 1$.

At last, there should be $|c_1'| = |x_1|$. This is because $2^j\tilde{\mathbf{b}}_1c_1' + e = \tilde{\mathbf{b}}_1c_1 = 2^j\tilde{\mathbf{b}}_1x_1 + \mathbf{e}x_2$. $2^jc_1' + k_j = c_1 = 2^jx_1 + k_jx_2$. Suppose $x_2 = 1$. $x_1 = c_1'$. If $x_2 = -1$, we could get that x_1 should be $-c_1' - 2k_j$. $|c_1'| < |-c_1' - 2k_j|$, so $x_2 = 1$, $x_1 = c_1'$.

Based on the above proof, we know $\|\hat{\mathbf{u}}\| = \|\hat{B}_j\mathbf{x}\|$, $\hat{\mathbf{u}} = \pm\hat{B}_j\mathbf{x}$ which is the shortest vector. This proves our operations are all right.

Using the same method we could get each $|c_i|, i = 1, \ldots, n$.

Now we do the second part of our proof. We need to know the sign for each c_i. Assume c_1 is positive and $c_2 \neq 0$. Set B to be the following.

$$B = \begin{pmatrix} 2|c_2|\tilde{\mathbf{b}}_2 & |c_1|\tilde{\mathbf{b}}_1 - |c_2|\tilde{\mathbf{b}}_2 & |c_3|\tilde{\mathbf{b}}_3 & \ldots & |c_n|\tilde{\mathbf{b}}_n \end{pmatrix}.$$

Run O on B.

As $\mathcal{L}(B)$ has a unique shortest vector $\tilde{\mathbf{u}} = c_1\tilde{\mathbf{b}}_1 + \sum_{i=2}^{n} p_i|c_i|\tilde{\mathbf{b}}_i$, $p_i \in \{1, -1\}$. If $p_2 = 1$, the only possible situation for $B\mathbf{x} = \tilde{\mathbf{u}}$ is that $x_2 = 1, x_1 = 1$. If $p_2 = -1$, the only possible situation for $B\mathbf{x} = \tilde{\mathbf{u}}$ is that $x_2 = 1, x_1 = 0$. Finally, it is easy to see that, if it returns 0, then we know c_2 is negative. If it returns 1, we know c_2 is positive.

In the same way, we could compute the sign of each c_i. At last, we could get all c_i. As a result, we get the shortest vector of the input lattice.

□

According to last section and paper 6, we have the following lemma.

Lemma 6. *Given lattice $\mathcal{L}(B)$, there is a polynomial time algorithm which could turn B to B' which has the following property.*

- *$\mathcal{L}(B')$ has a unique shortest vector $\mathbf{u} = B'\mathbf{x}$.*
- *$B\mathbf{x}$ is a shortest vector of $\mathcal{L}(B)$.*

Theorem 7. *For a given lattice $\mathcal{L}(B)$, an oracle O could compute the last significant bit of c_1, where $\mathbf{u} = \sum_{i=1}^{n} c_i\mathbf{b}_i$ is the shortest vector. $\mathbf{u}$ could be found in polynomial time.*

Proof. According to the lemma 5, $\mathcal{L}(B)$ could be turned to $\mathcal{L}(B')$ which has a unique shortest vector. Applying the algorithm in the theorem 8, the shortest vector $B'\mathbf{x}$ of $\mathcal{L}(B')$ could be found. As a result, the shortest vector $B\mathbf{x}$ of $\mathcal{L}(B)$ could be found.

□

References

[1] M. Ajtai. Generating hard instances of lattice problems. STOC, 1996, 99-108.

[2] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. STOC, 1997.

[3] M. Ajtai. The Shortest vector problem in l_2 is **NP**-hard for randomized reductions. STOC, 1998, 10-19.

[4] O. Regev. New lattice-based cryptographic constructions. J. ACM 51(2004).

[5] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. J. ACM, 2009.

[6] M. Ajtai, R. Kumar and D. Sivakumar. A sieve algorithm for the shortest vector lattice vector problem. STOC, 1998, 266-275.

[7] D. Micciancio. Efficient reductions among lattice problems. SODA, 2008, 84-93.

[8] D. Micciancio. The shortest vector problem is **NP**-hard to approximate within some constant. SIAM journal on Computing, 2001, 30(6), 2008-2035.

[9] D. Micciancio and O. Regev. Lattice-based cryptography. In *Post Quantum Cryptography*, pages 147-191. Springer, February 2009.

[10] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. STOC, 2009.

[11] L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. Combinatorica, 1986.

[12] W.Banaszczyk. New bounds in some transference theorems in the geometry of numbers. Mathematische Annalen, 1993.

[13] D. Micciancio, Chris Peikert. Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. 2011.

[14] René Peralta. Simultaneous Security of Bits in the Discrete Log. Springer-verlag, 1998.

[15] P. Nguyen and T. Vidick. Sieve algorithms for the shortest vector problem are practical. J. of Mathe-matical Cryptology, 2(2):181-207, jul 2008.

[16] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. Mathematische Annalen, 261:513-534, 1982.

[17] Vadim Lyubashevsky, Daniele Micciancio. On Bounded Distance Decoding, Unique Shortest Vectors, and the Minimum Distance Problem. 2009.

[18] Gengran Hu, Yanbin Pan. A New Reduction from Search SVP to Optimization SVP. 2012.

[19] Divesh Aggarwal, Chandan Dubey. Improved Hardness Results for Unique Shortest Vector Problem. 2011.

[20] Ravi Kannan. Minkowski's Convex Body Theorem and Integer Programming. 1987.

[21] Nicolas Gama, Phong Q. Nguyen. Finding Short Lattice Vectors within Mordell's inequality. 2008.

[22] R Kumar, D Sivakumar. A note on the shortest lattice vector problem. 1999.

[23] Daniele Micciancio, Panagiotis Voulgaris. A Deterministic Single Exponential Time Algorithm for Most Lattice Problems based on Voronoi Cell Computations. STOC, 2010.

[24] S. Goldwasser and D. Micciancio. Complexity of lattice problems. Springer, 2002.