Solving Low-Density Multiple Subset Sum Problems with SVP Oracle*

PAN Yanbin · ZHANG Feng

DOI: 10.1007/s11424-015-3324-9
Received: 19 December 2013 / Revised: 29 September 2014

Abstract It is well known that almost all subset sum problems with density less than $0.9408\cdots$ can be solved in polynomial time with an SVP oracle that can find a shortest vector in a special lattice. In this paper, the authors show that a similar result holds for the k-multiple subset sum problem which has k subset sum problems with exactly the same solution. Specially, for the single subset sum problem ($k = 1$), a modified lattice is introduced to make the proposed analysis much simpler and the bound for the success probability tighter than before. Moreover, some extended versions of the multiple subset sum problem are also considered.

Keywords Lattice, low-density, multiple modular subset sum problem, multiple subset sum problem.

1 Introduction

The subset sum problem (or knapsack problem), which is well known to be NP-hard[1], refers to the question of finding variables $(x_1, x_2, \cdots, x_n) \in \{0, 1\}^n$, given positive integers $a_1, a_2, \cdots, a_n$ and s, such that

$$\sum_{i=1}^{n} x_i a_i = s,$$

if there exist. These a_i's are called the weights of the subset sum problem, and the density of the problem is defined as

$$d = \frac{n}{\log_2(\max_i a_i)}.$$

In terms of public-key cryptosystems, the density d is an approximate measure of the information rate at which bits are transmitted, namely

$$d \approx \frac{\text{number of bits in plaintext message}}{\text{average number of bits in ciphertext message}}.$$

PAN Yanbin · ZHANG Feng
Key Laboratory of Mathematics Mechanization, NCMIS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, *China.* Email: panyanbin@amss.ac.cn; zhangfeng@amss.ac.cn.

*This research was supported by the National Natural Science Foundation of China under Grant Nos. 11201458, 11471314, in part by 973 Project under Grant No. 2011CB302401, and in part by the National Center for Mathematics and Interdisciplinary Sciences, Chinese Academy of Sciences.

◇ *This paper was recommended for publication by Editor LI Ziming.*

Lattices are discrete additive subgroups in $\mathbb{R}^n$. Finding the shortest nonzero vector in a given lattice is one of the most famous problem for lattices, which is well known as SVP. The algorithms, such as [2–4], which aims to solve the SVP have been employed widely in cryptography and computational number theory (see [2, 5] for some examples). In respect of the subset sum problem, Lagarias and Odlyzko[6] first considered how to solve it with an SVP oracle, which takes as input a lattice and outputs one of its shortest vectors in unit time. They showed that almost all the subset sum problems with density less than $0.6463\cdots$ would be solved in polynomial time with a single call to an SVP oracle. The bound was improved to $0.9408\cdots$ later by Coster, et al.[7].

There are also some extended versions of the subset sum problem in consideration. For example, the variables x_i's may be in $\{-1, 0, 1\}$ instead of $\{0, 1\}$, or the weights are allowed to be negative, or a modulus is involved. Li and Ma[8] considered an extended version of the subset sum problem where the variables are in $\{-1, 0, 1\}$ and the weights are allowed to be negative. They gave the upper bound $d < 0.488\cdots$ for those problems that can be solved with an SVP oracle with very high probability. Furthermore, Wang, et al.[9] showed the same bound $d < 0.488\cdots$ holds for the extended modular subset sum problems.

In this paper, we consider solving the low-density multiple subset sum problem with an SVP oracle. A multiple subset sum problem refers to the question of finding the solution that satisfies several subset sum problems simultaneously. The number of these subset sum problems involved is called its multiplicity. For simplicity, we denote by k-multiple subset sum problem a multiple subset sum problem of multiplicity k. Similarly, the density of a k-multiple subset sum problem is defined as

$$d = \frac{n}{k \cdot \log_2 A},$$

where A is the maximum of the weights appearing in the subset sum problems. Notice that when $k = 1$, the definition agrees with the one of the original single subset sum problem.

To our best knowledge, Liu, et al.[10] showed how to solve the k-multiple subset sum problem with an SVP oracle by transforming it into a new single subset sum problem, whose density is approximately equal to the density of the multiple subset sum problem we have defined. They obtained the bound $d < 0.9408\cdots$ by employing the known result[7] for the new single subset sum problem. However, just a heuristic explanation, but not a rigorous proof, was given in [10]. Furthermore, it seems hard to apply their method for the extended versions of the multiple subset sum problem.

In this paper, we rigorously prove that almost all the k-multiple subset sum problem with density less than $0.9408\cdots$ would be solved in polynomial time with a single call to an SVP oracle. Moreover, we generalize it to some extended versions of the problem. Specially, when $k = 1$, the results agree with those for single subset sum problems. A modified lattice involved in our proof makes the analysis much simpler than before, and makes the bound for the success probability a little tighter.

The subset sum problem has many applications in cryptography, such as in the construction of the Merkle-Hellman cryptosystem[11]. However, nearly all proposed cryptosystems based on subset sum problem have been broken (see [12]). Some attacks are due to the use of the low-density subset sum problems[6, 7], since in practice an SVP oracle can be stimulated well by some efficient algorithms[2–4] in some cases. However, the subset sum problem with higher density, such as one, have not been solved efficiently by now even with the help of an SVP oracle. Hence, suppose there is a cryptosystem encrypting the message $(x_1, x_2, \cdots, x_n) \in \{0,1\}^n$ to the subset sum $s = \sum_{i=1}^n x_i a_i$, where the public weights a_i's have density one. Then it seems we can not break it even with the help of an SVP oracle. However, a broadcast attack against it can be easily obtained by our results in this paper. The broadcast attack against public key cryptosystems was first proposed by Hästad[13] in 1988, which enables an attacker to recover the single message sent by the single sender to multiple recipients who use the same type of cryptosystem but have different public keys, without requiring any knowledge of the recipients' private keys. Obviously, a broadcast attack against the cryptosystem based on high-density subset sum problem is to solve the corresponding k-multiple subset sum problem, where k is the number of the recipients. According to the definition of the density of k-multiple subset sum problem, we know that although the density of every single subset sum problem may be high, the density of k-multiple subset sum problem decreases as the number of the recipients k increases when A is fixed. By our result, if a message is sent to recipients many enough such that the corresponding density is less than $0.9408\cdots$, the message can be recovered in polynomial time with a single call to an SVP oracle.

Roadmap The remainder of the paper is organized as follows. In Section 2, we give some preliminaries needed. We show how to solve the low-density multiple subset sum problem with an SVP oracle in Section 3, and present some strategies to solve the corresponding SVP in practice in Section 4. Finally, a short conclusion is given in Section 5.

2 Preliminaries

We denote by $\mathbb{Z}$ the integer ring, by $\mathbb{Q}$ the rational number field and by $\mathbb{R}$ the real number field. We use bold letters to denote vectors, in row notation. If $\boldsymbol{v}$ is a vector, then we denote by v_i the i-th component of $\boldsymbol{v}$. Let $\|\cdot\|$ and $\langle\cdot,\cdot\rangle$ be the Euclidean norm and inner product of $\mathbb{R}^n$, respectively.

We also use the big Omega notation and little oh notation. More precisely, by $f(n) = \Omega(g(n))$ we mean there exist $k > 0$ and positive integers n_0, s.t. $f(n) \geq k \cdot g(n)$ for any $n \geq n_0$ and by $f(n) = o(g(n))$ we mean that for any $k > 0$, there exists a positive integer n_0, s.t. $f(n) \leq k \cdot g(n)$ for any $n \geq n_0$.

2.1 Lattice

Let $B = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_n\} \subset \mathbb{R}^m$ be a set of n linearly independent vectors. The lattice $\mathcal{L}$ generated by B is defined as

$$\mathcal{L}(B) = \left\{ \sum_{i=1}^n x_i \boldsymbol{b}_i : x_i \in \mathbb{Z} \right\}.$$

We call B the basis of the lattice and n its dimension. Finding a non-zero shortest vector of a lattice $\mathcal{L}$ is called the shortest vector problem (SVP).

2.2 The Number of Integer Points in $B_n(r)$

We denote by $B_n(r)$ the ball centered at the origin with radius r and by $N(n, r^2)$ the number of integer points in $B_n(r)$, i.e.,

$$N(n, r^2) = \left|\left\{ \boldsymbol{z} \in \mathbb{Z}^n : \sum_{i=1}^{n} z_i^2 \leq r^2 \right\}\right|.$$

By the techniques of Mazo and Odlyzko[14], it can be shown that:

1) $N(n, \frac{n}{2}) \leq 2^{c_0 n}, c_0 = 1.54725\cdots$ (see [6]),

2) $N(n, \frac{n}{4}) \leq 2^{c_1 n}, c_1 = 1.0628\cdots$ (see [7]),

3) $N(n, n) \leq 2^{c_2 n}, c_2 = 2.047\cdots$ (see [8]).

2.3 Solving Low-Density Subset Sum Problems with SVP Oracle

Given positive integer A, consider the random subset sum problem defined by $(a_1, a_2, \cdots, a_n)$ and s, where integer a_i is chosen independently uniformly randomly from $(0, A]$ and $s = \sum_{i=1}^{n} e_i a_i$, with any non-zero vector $\boldsymbol{e} = (e_1, e_2, \cdots, e_n) \in \{0, 1\}^n$. Recall that the density of the subset sum problem is

$$d = \frac{n}{\log_2 A}.$$

Coster, et al. [7] designed a polynomial time algorithm to solve the random subset sum problem above with a single call to an SVP oracle and showed that the failure probability of the algorithm

$$P \leq n(4n\sqrt{n} + 1)\frac{2^{c_1 n}}{A}. \tag{1}$$

Notice that if the density d is less than $\frac{1}{c_1}$ ($\approx 0.9408\cdots$), $P = \frac{1}{2^{\Omega(n)}}$ can be as small as possible when n is large enough. Hence, with probability $1 - P$, which is exponentially close to one, the random subset sum problem can be solved with an SVP oracle.

Li and Ma[8] considered another version of the random subset sum problem that extends the variables range from $\{0, 1\}$ to $\{-1, 0, 1\}$ and allows the weight to be negative. They also estimated the upper bound of the corresponding failure probability P and got

$$P \leq n(n + \frac{1}{2A^t})\frac{2^{c_2 n}}{A^{1-t}} + \frac{2}{A^t}, \tag{2}$$

where $0 < t < 1$. Similarly, if $d < 0.488\cdots$, P would be exponentially small.

The extended modular subset sum problem refers to the question of finding variables $(x_1, x_2, \cdots, x_n) \in \{-1, 0, 1\}^n$, given positive integers $a_1, a_2, \cdots, a_n$, q and s, such that

$$\sum_{i=1}^{n} a_i x_i \equiv s (\text{mod } q),$$

where a_i's are uniformly randomly chosen from $[0, q-1]$. The density of the extended modular subset sum is defined by

$$d = \frac{n}{\log_2 q}.$$

Wang, et al.[9] showed the corresponding failure probability P satisfies

$$P \leq \frac{2^{c_2(n+1)}}{q} n(2n+3)^2, \tag{3}$$

and if $d < 0.488$, then the successful probability is exponentially close to one.

3 Solving Low-Density Multiple SSP with SVP Oracle

In this section, we will consider the multiple subset sum problem (multiple SSP) and the corresponding multiple modular subset sum problem (multiple MSSP).

3.1 The Multiple Subset Sum Problem and Its Extended Versions

Given a positive integer A, the multiple subset sum problem refers to the question of recovering $(x_1, x_2, \cdots, x_n) \in \{0,1\}^n$ from $a_{ji}\,(1 \leq j \leq k, 1 \leq i \leq n)$ and $s_1, s_2, \cdots, s_k$, where a_{ji}'s are uniformly independently randomly chosen from the set of integers between 1 and A, and $s_1, s_2, \cdots, s_k$ satisfy

$$\begin{aligned} \sum_{i=1}^n a_{1,i}x_i &= s_1, \\ \sum_{i=1}^n a_{2,i}x_i &= s_2, \\ &\vdots \\ \sum_{i=1}^n a_{k,i}x_i &= s_k. \end{aligned}$$

The density of the multiple subset sum problem is defined as

$$d = \frac{n}{k \cdot \log_2 A}.$$

Similarly, given a positive integer q, the multiple modular subset sum problem is to find $(x_1, x_2, \cdots, x_n) \in \{0,1\}^n$ from $a_{ji}\,(1 \leq j \leq k, 1 \leq i \leq n)$ and $s_1, s_2, \cdots, s_k$, where a_{ji}'s are uniformly independently randomly chosen from the set of integers between 1 and $q-1$ and $s_1, s_2, \cdots, s_k$ satisfy

$$\begin{aligned} \sum_{i=1}^n a_{1,i}x_i &\equiv s_1 \pmod q, \\ \sum_{i=1}^n a_{2,i}x_i &\equiv s_2 \pmod q, \\ &\vdots \\ \sum_{i=1}^n a_{k,i}x_i &\equiv s_k \pmod q. \end{aligned}$$

The density of the multiple modular subset sum problem is defined as

$$d = \frac{n}{k \cdot \log_2 q}.$$

Notice that when $k = 1$, both of the definitions above are in accordance with those for the single subset sum problems respectively.

3.2 Solving Low-Density Multiple Subset Sum Problems with SVP Oracle

We give the main result for the multiple subset sum problem first.

Theorem 3.1 *Given positive integer A, let a_{ji} $(1 \le j \le k, 1 \le i \le n)$ be independently uniformly random integers between 1 and A, $\boldsymbol{e} = (e_1, e_2, \cdots, e_n)$ be arbitrary non-zero vector in $\{0,1\}^n$ and $s_j = \sum_{i=1}^n a_{ji}e_i$ $(j = 1, 2, \cdots, k)$. If the density $d < 0.9408\cdots$, then with probability $1 - \frac{1}{2^{\Omega(n)}}$, the multiple subset sum problem defined by a_{ji}'s and $s_1, s_2, \cdots, s_k$ can be solved in polynomial time with a single call to an SVP oracle.*

Proof Define vectors $\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_n, \boldsymbol{b}_{n+1}$ as follows:

$$\begin{aligned}
\boldsymbol{b}_1 &= (1, 0, \cdots, 0, 0, Na_{1,1}, Na_{2,1}, \cdots, Na_{k,1}),\\
\boldsymbol{b}_2 &= (0, 1, \cdots, 0, 0, Na_{1,2}, Na_{2,2}, \cdots, Na_{k,2}),\\
&\ \ \vdots\\
\boldsymbol{b}_n &= (0, 0, \cdots, 1, 0, Na_{1,n}, Na_{2,n}, \cdots, Na_{k,n}),\\
\boldsymbol{b}_{n+1} &= \left(\tfrac{1}{2}, \tfrac{1}{2}, \cdots, \tfrac{1}{2}, \tfrac{1}{2}, Ns_1,\ \ Ns_2,\ \ \cdots,\ \ Ns_k\right),
\end{aligned}$$

where N is an integer greater than $\sqrt{\frac{n+1}{4}}$.

Let $\mathcal{L}$ be the lattice generated by $\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_n, \boldsymbol{b}_{n+1}$. Then it can be concluded easily that $\overline{\boldsymbol{e}} = (e_1 - \frac{1}{2}, e_2 - \frac{1}{2}, \cdots, e_n - \frac{1}{2}, -\frac{1}{2}, 0, \cdots, 0)$ is in $\mathcal{L}$. Notice that $\|\overline{\boldsymbol{e}}\| = \sqrt{\frac{n+1}{4}}$.

Let $X = \{\boldsymbol{v} \in \mathcal{L} | 0 < \|\boldsymbol{v}\| \le \|\overline{\boldsymbol{e}}\|, \boldsymbol{v} \notin \{\boldsymbol{0}, \overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}\}\}$. If $X = \emptyset$, then $\overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}$ are the only two non-zero shortest lattice vectors of $\mathcal{L}$. So the SVP oracle will output either $\overline{\boldsymbol{e}}$ or $-\overline{\boldsymbol{e}}$ when queried with $\mathcal{L}$, and we can then recover $\boldsymbol{e}$. Next we will show that the probability of $X = \emptyset$ is exponentially close to one.

We first estimate the value of $\Pr[X \ne \emptyset]$. Since $N > \sqrt{\frac{n+1}{4}}$ and $\|\overline{\boldsymbol{e}}\| = \sqrt{\frac{n+1}{4}}$, we have $v_{n+2} = v_{n+3} = \cdots = v_{n+1+k} = 0$ for any $\boldsymbol{v} \in X$. Suppose that $\boldsymbol{v} = \sum_{i=1}^{n+1} x_i\boldsymbol{b}_i \in X$, then we can express v_i in term of x_i in the following way

$$\begin{aligned}
v_i &= x_i + \frac{1}{2}x_{n+1}, \quad\ \ i = 1, 2, \cdots, n,\\
v_{n+1} &= \frac{1}{2}x_{n+1},\\
v_{n+1+j} &= N \cdot \left(\sum_{i=1}^n a_{ji}x_i + x_{n+1}s_j\right) = 0, \quad\ \ j = 1, 2, \cdots, k,
\end{aligned}$$

which implies that

$$\sum_{i=1}^n a_{ji}(v_i - v_{n+1}) + 2v_{n+1}s_j = 0, \quad j = 1, 2, \cdots, k.$$

Let $\widehat{\boldsymbol{v}} = (v_1, v_2, \cdots, v_{n+1})$ be a truncation of $\boldsymbol{v}$. For simplicity, we denote by $\mathcal{D}$ the following set

$$\left\{\boldsymbol{w}\in\mathbb{Z}^{n+1}\middle|\exists(x_1,x_2,\cdots,x_{n+1})\in\mathbb{Z}^{n+1}\text{ s.t. }w_i=x_i+\frac{1}{2}x_{n+1}(i=1,2,\cdots,n),w_{n+1}=\frac{1}{2}x_{n+1}\right\}.$$

Then

$$\begin{aligned}\Pr[X\neq\emptyset]\leq\Pr\Bigg[&\exists\widehat{\boldsymbol{v}}\in\mathcal{D},\text{ s.t. }0<\|\widehat{\boldsymbol{v}}\|\leq\|\overline{\boldsymbol{e}}\|,\boldsymbol{v}\notin\{\boldsymbol{0},\overline{\boldsymbol{e}},-\overline{\boldsymbol{e}}\},\\&\sum_{i=1}^n a_{ji}(v_i-v_{n+1})+2v_{n+1}s_j=0,j=1,2,\cdots,k\Bigg]\\\leq\Pr\Bigg[&\sum_{i=1}^n a_{ji}(v_i-v_{n+1})+2v_{n+1}s_j=0,j=1,2,\cdots,k,\boldsymbol{v}\notin\{\boldsymbol{0},\overline{\boldsymbol{e}},-\overline{\boldsymbol{e}}\}\Bigg]\\&\cdot\left|\left\{\widehat{\boldsymbol{v}}\in\mathcal{D}\middle|\|\widehat{\boldsymbol{v}}\|\leq\|\overline{\boldsymbol{e}}\|=\sqrt{\frac{n+1}{4}}\right\}\right|.\end{aligned}$$

For the second factor of the above expression, notice that for every $\widehat{\boldsymbol{v}}\in\mathcal{D}$,

1) if the corresponding x_{n+1} is odd, then $\|\widehat{\boldsymbol{v}}\|\geq\sqrt{\frac{n+1}{4}}$, which implies $\|\widehat{\boldsymbol{v}}\|=\sqrt{\frac{n+1}{4}}$. Moreover, $|v_i|=\frac{1}{2}$ $(i=1,2,\cdots,n+1)$ and then $x_{n+1}=\pm1$. It can be easily concluded that for either $x_{n+1}=1$ or $x_{n+1}=-1$, there are exactly 2^n vectors $(x_1,x_2,\cdots,x_n)\in\mathbb{Z}^n$ s.t. $|v_i|=\frac{1}{2}$ $(i=1,2,\cdots,n)$;

2) if the corresponding x_{n+1} is even, the second factor is exactly the number of the integer points in $B_{n+1}(\sqrt{\frac{n+1}{4}})$.

Finally, we have

$$\left|\left\{\widehat{\boldsymbol{v}}\in\mathcal{D}\middle|\|\widehat{\boldsymbol{v}}\|\leq\|\overline{\boldsymbol{e}}\|=\sqrt{\frac{n+1}{4}}\right\}\right|=N\left(n+1,\frac{n+1}{4}\right)+2^{n+1}\leq2^{c_1(n+1)+1}.$$

Next we consider the first factor. For $j=1,2,\cdots,k$, since $s_j=\sum_{i=1}^n a_{ji}e_i$, we rewrite $\sum_{i=1}^n a_{ji}(v_i-v_{n+1})+2v_{n+1}s_j=0$ as

$$\sum_{i=1}^n a_{ji}z_i=0,\text{ where }z_i=v_i-v_{n+1}+2v_{n+1}e_i=x_i+x_{n+1}e_i.$$

We claim that there must exist $t,1\leq t\leq n$, such that $z_t\neq0$. Otherwise, $v_i=v_{n+1}(1-2e_i)=\pm v_{n+1}$, since $e_i\in\{0,1\}$ for $i=1,2,\cdots,n$. Thus, $\|\widehat{\boldsymbol{v}}\|=\sqrt{n+1}|v_{n+1}|=\frac{\sqrt{n+1}}{2}|x_{n+1}|$ since $v_{n+1}=\frac{1}{2}x_{n+1}$. By the fact $\widehat{\boldsymbol{v}}\in X$, we know that

$$\|\widehat{\boldsymbol{v}}\|=\frac{\sqrt{n+1}}{2}|x_{n+1}|\leq\frac{\sqrt{n+1}}{2},$$

which implies that $|x_{n+1}|\leq1$. Since $x_{n+1}\in\mathbb{Z}$, x_{n+1} takes value only $-1,0$, or 1, corresponding to $\widehat{\boldsymbol{v}} = \overline{\boldsymbol{e}}$, $\mathbf{0}$, or $-\overline{\boldsymbol{e}}$ respectively, which contradicts to the definition of X. Thus,

$$\begin{aligned}
&\Pr\Bigg[\sum_{i=1}^{n} a_{ji}(v_i - v_{n+1}) + 2v_{n+1}s_j = 0, j = 1, \cdots, k, \boldsymbol{v} \notin \{\mathbf{0}, \overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}\}\Bigg] \\
&\leq \Pr\Bigg[\sum_{i=1}^{n} a_{ji}z_i = 0, j = 1, 2, \cdots, k\Bigg] \\
&= \Pr\Bigg[a_{jt} = -\frac{\sum_{i=1, i\neq t}^{n} a_{ji}z_i}{z_t}, j = 1, 2, \cdots, k\Bigg] \\
&= \prod_{j=1}^{k} \Pr\Bigg[a_{jt} = -\frac{\sum_{i=1, i\neq t}^{n} a_{ji}z_i}{z_t}\Bigg] \\
&\leq \frac{1}{A^k}.
\end{aligned}$$

Thus,

$$\Pr[X \neq \emptyset] \leq \frac{2^{c_1 n}}{A^k} 2^{c_1+1}. \tag{4}$$

If $d < \frac{1}{c_1} = 0.9408\cdots$, it can be easily concluded that the probability of the event X is not empty is $\frac{1}{2^{\Omega(n)}}$. Hence the theorem follows. ▌

Remark 3.2 Notice that for $k = 1$, it turns out to be the single subset sum problem. The analysis is much simpler than before and the upper bound (4) is also a little tighter than the previous result (1).

If we extend the variables range from $\{0, 1\}$ to $\{-1, 0, 1\}$, and allow the weight to be negative, we can investigate the lattice $\mathcal{L}$ generated by $\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_n, \widetilde{\boldsymbol{b}}_{n+1}$ where $\boldsymbol{b}_i$'s $(i = 1, 2, \cdots, n)$ are exactly the same to those in the proof of Theorem 3.1, and

$$\widetilde{\boldsymbol{b}}_{n+1} = (0, 0, \cdots, 0, 1, \quad Ns_1, \quad Ns_2, \quad \cdots, \quad Ns_k),$$

where N is an integer greater than $\sqrt{n+1}$.

Notice that there is a vector $\overline{\boldsymbol{e}} = (e_1, e_2, \cdots, e_n, 1, 0, \cdots, 0) \in \mathcal{L}$ with length $\sqrt{n+1}$. Similarly, we can define $X = \{\boldsymbol{v} \in \mathcal{L} | 0 < \|\boldsymbol{v}\| \leq \|\overline{\boldsymbol{e}}\|, \boldsymbol{v} \notin \{\mathbf{0}, \overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}\}\}$ and get

$$\Pr[X \neq \emptyset] \leq \frac{2^{c_2 n}}{A^k} 2^{c_2}. \tag{5}$$

Thus, a similar result holds for $d < \frac{1}{c_2} = 0.488\cdots$. Notice that for $k = 1$, the analysis is much simpler than before and the upper bound (5) is better than the previous result (2).

3.3 Solving the Multiple Modular Subset Sum Problems with SVP Oracle

For the multiple modular subset sum problems, we have

Theorem 3.3 *Given a positive integer q which is greater than $\sqrt{\frac{n+1}{4}}$, let $a_{ji}\,(1 \leq j \leq k, 1 \leq i \leq n)$ be independently uniformly random integers between 1 and $q-1$, $\boldsymbol{e} = (e_1, e_2, \cdots e_n)$ be arbitrary non-zero vector in $\{0, 1\}^n$, and $s_j \equiv \sum_{i=1}^{n} a_{ji}e_i (\mathrm{mod}\ q)$, $(j = 1, 2, \cdots, k)$, then with probability greater than $1 - \frac{2^{c_1 n}}{q^k} 2^{c_1}((n+1)\sqrt{n}+1)^k$, the multiple modular subset sum problem defined by a_{ji}'s and $s_1, s_2, \cdots, s_k$ can be solved in polynomial time with a single call to an SVP oracle.*

Proof The proof is similar to the one of Theorem 3.1. Define the vectors as follows:

$$\begin{aligned}
\boldsymbol{b}'_1 &= (1,0,\cdots,0,0,Na_{1,1},Na_{2,1},\cdots,Na_{k,1}),\\
\boldsymbol{b}'_2 &= (0,1,\cdots,0,0,Na_{1,2},Na_{2,2},\cdots,Na_{k,2}),\\
&\ \vdots\\
\boldsymbol{b}'_n &= (0,0,\cdots,1,0,Na_{1,n},Na_{2,n},\cdots,Na_{k,n}),\\
\boldsymbol{b}'_{n+1} &= (0,0,\cdots,0,0,\ \ Nq,\quad 0,\quad \cdots,\quad 0\ \ ),\\
\boldsymbol{b}'_{n+2} &= (0,0,\cdots,0,0,\quad 0,\quad Nq,\quad \cdots,\quad 0\ \ ),\\
&\ \vdots\\
\boldsymbol{b}'_{n+k} &= (0,0,\cdots,0,0,\quad 0,\quad 0,\quad \cdots,\quad Nq\ ),\\
\boldsymbol{b}'_{n+k+1} &= \left(\tfrac{1}{2},\tfrac{1}{2},\cdots,\tfrac{1}{2},\tfrac{1}{2},Ns_1,\ \ Ns_2,\ \ \cdots,\ \ Ns_k\right),
\end{aligned}$$

where $N > \sqrt{\frac{n+1}{4}}$, and consider the lattice $\mathcal{L}'$ generated by $\boldsymbol{b}'_1, \boldsymbol{b}'_2, \cdots, \boldsymbol{b}'_{n+k+1}$.

Then we can easily know that $\overline{\boldsymbol{e}} = (e_1 - \frac{1}{2}, e_2 - \frac{1}{2}, \cdots, e_n - \frac{1}{2}, -\frac{1}{2}, 0, \cdots, 0)$ is in $\mathcal{L}'$. Notice that $\|\overline{\boldsymbol{e}}\| = \sqrt{\frac{n+1}{4}}$.

Similarly, let $X = \{\boldsymbol{v} \in \mathcal{L}' | 0 < \|\boldsymbol{v}\| \le \|\overline{\boldsymbol{e}}\|, \boldsymbol{v} \notin \{\boldsymbol{0}, \overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}\}\}$, and we also have $v_{n+2} = v_{n+3} = \cdots = v_{n+k+1} = 0$ for any $\boldsymbol{v} \in X$. Suppose that $\boldsymbol{v} = \sum_{i=1}^{n} x_i \boldsymbol{b'}_i + \sum_{i=1}^{k} y_i \boldsymbol{b'}_{n+i} + x_{n+1}\boldsymbol{b'}_{n+k+1} \in X$, then we have

$$\begin{aligned}
v_i &= x_i + \frac{1}{2}x_{n+1}, \quad i = 1,2,\cdots,n,\\
v_{n+1} &= \frac{1}{2}x_{n+1},\\
v_{n+1+j} &= N \cdot \left(\sum_{i=1}^{n} a_{ji}x_i + x_{n+1}s_j + qy_j\right) = 0, \quad j = 1,2,\cdots,k.
\end{aligned}$$

This implies that

$$\sum_{i=1}^{n} a_{ji}(v_i - v_{n+1}) + 2v_{n+1}s_j + qy_j = 0, \quad j = 1,2,\cdots,k.$$

Let $\widehat{\boldsymbol{v}} = (v_1, v_2, \cdots, v_{n+1})$ and $\mathcal{D}$ be the set as in the proof of Theorem 3.1. Then

$$\begin{aligned}
\Pr[X \ne \emptyset] &\le \Pr\Bigg[\exists \widehat{\boldsymbol{v}} \in \mathcal{D}, y_1, y_2, \cdots, y_k \in \mathbb{Z}, \text{ s.t. } 0 < \|\widehat{\boldsymbol{v}}\| \le \|\overline{\boldsymbol{e}}\|, \boldsymbol{v} \notin \{\boldsymbol{0}, \overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}\},\\
&\qquad \sum_{i=1}^{n} a_{ji}(x_i + e_i x_{n+1}) + qy_j = 0, j = 1,\cdots,k\Bigg]\\
&\le \left|\left\{\widehat{\boldsymbol{v}} \in \mathcal{D} | \|\widehat{\boldsymbol{v}}\| \le \sqrt{\frac{n+1}{4}}\right\}\right| \cdot \prod_{j=1}^{k} \left|\left\{y_j | \sum_{i=1}^{n} a_{ji}z_i = -qy_j\right\}\right|\\
&\quad \cdot \prod_{j=1}^{k} \Pr\left[\sum_{i=1}^{n} a_{ji}z_i = -qy_j\right],
\end{aligned}$$

where $z_i = x_i + e_i x_{n+1} = v_i - v_{n+1} + 2v_{n+1}e_i$.

For the first factor, it is clear that $\left|\{\widehat{\boldsymbol{v}}|\|\widehat{\boldsymbol{v}}\| \leq \|\overline{\boldsymbol{e}}\| = \sqrt{\frac{n+1}{4}}\}\right| \leq 2^{c_1(n+1)+1}$ as in the proof of Theorem 3.1.

Now we consider the second factor. Let $g_j = \frac{a_{ji}}{q}, j = 1, 2, \cdots, k$, then $|g_j| \leq 1, j = 1, 2, \cdots, k$. Since $\sum_{i=1}^n a_{ji}z_i = -qy_j$, we have

$$\begin{aligned}
|y_j| &= \left|\sum_{i=1}^n g_i(v_i - v_{n+1} + 2v_{n+1}e_i)\right| \\
&= \left|\sum_{i=1}^n g_i v_i + \sum_{i=1}^n (2e_i - 1)v_{n+1}g_i\right| \\
&\leq \sum_{i=1}^n |g_i||v_i| + \sum_{i=1}^n |g_i||v_{n+1}| \\
&= (|v_1|, |v_2|, \cdots, |v_{n+1}|) \begin{pmatrix} |g_1| \\ \vdots \\ |g_n| \\ \sum\limits_{i=1}^n |g_i| \end{pmatrix} \\
&\leq \|v\| \cdot \sqrt{n + n^2} \\
&\leq \sqrt{\frac{n+1}{4}}\sqrt{n(n+1)} \\
&= \frac{n+1}{2}\sqrt{n}.
\end{aligned}$$

Thus, we have $\prod_{j=1}^k |\{y_j| \sum_{i=1}^n a_{ji}z_i = -qy_j\}| \leq ((n+1)\sqrt{n}+1)^k$.

Finally, for the last factor, we can similarly conclude that for $\sum_{i=1}^n a_{ji}z_i = -qy_j, 1 \leq j \leq k$, there exists t, $1 \leq t \leq k$, such that $z_t \neq 0$, and

$$\prod_{j=1}^k \Pr\Big[\sum_{i=1}^n a_{ji}z_i = qy_j\Big] \leq \frac{1}{q^k}.$$

Thus,

$$\Pr[X \neq \emptyset] \leq \frac{2^{c_1 n}}{q^k} 2^{c_1+1}((n+1)\sqrt{n}+1)^k$$

and the theorem follows. ∎

An corollary can be easily obtained by Theorem 3.3.

Corollary 3.4 *If the density $d < 0.9408\cdots$ and $k = o(\frac{n}{\log_2((n+1)\sqrt{n}+1)})$, with probability $1 - \frac{1}{2^{\Omega(n)}}$, the multiple modular subset sum problem defined in Theorem* 3.3 *can be solved in polynomial time with a single call to an SVP oracle.*

Notice that $d < 0.9408\cdots$ and $k = o(\frac{n}{\log_2((n+1)\sqrt{n}+1)})$ yield that q must be large enough implicitly. The restriction for k comes from the factor $((n+1)\sqrt{n}+1)^k$ in Theorem 3.3. It is still a problem whether there exists polynomial poly(n), independent of k, such that

$$\Pr[X \neq \emptyset] \leq \frac{2^{cn}}{q^k} \cdot \text{poly}(n).$$

If such poly(n) existed, then only the condition $d < 0.9408\cdots$ was needed to make Corollary 3.4 hold.

Interestingly, if q is a prime larger than $\sqrt{\frac{n+1}{4}}$, we can get a better result.

Theorem 3.5 *Given a positive prime q greater than $\sqrt{\frac{n+1}{4}}$, let $a_{ji}\,(1 \le j \le k, 1 \le i \le n)$ be independently uniformly random integers between 1 and $q-1$, $\boldsymbol{e} = (e_1, e_2, \cdots, e_n)$ be arbitrary non-zero vector in $\{0,1\}^n$, and $s_j \equiv \sum_{i=1}^n a_{ji}e_i (\bmod\ q)$ $(j = 1, 2, \cdots, k)$. If the density $d < 0.9408\cdots$, then with probability $1 - \frac{1}{2^{\Omega(n)}}$, the multiple modular subset sum problem defined by a_{ji}'s and $s_1, s_2, \cdots, s_k$ can be solved in polynomial time with a single call to an SVP oracle.*

Proof The first part of the proof is exactly same to the one for Theorem 3.3 until we have

$$\begin{aligned}
&\Pr[X \neq \emptyset]\\
&\le \Pr\Big[\exists \widehat{\boldsymbol{v}} \in \mathcal{D}, y_1, y_2, \cdots, y_k \in \mathbb{Z}, \text{ s.t. } 0 < \|\overline{\boldsymbol{v}}\| \le \|\overline{\boldsymbol{e}}\|, \boldsymbol{v} \notin \{\boldsymbol{0}, \overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}\},\\
&\qquad \sum_{i=1}^n a_{ji}(x_i + e_i x_{n+1}) + q y_j = 0, j = 1, 2, \cdots, k\Big]\\
&= \Pr\Big[\exists \widehat{\boldsymbol{v}}, \text{ s.t. } 0 < \|\overline{\boldsymbol{v}}\| \le \|\overline{\boldsymbol{e}}\|, \boldsymbol{v} \notin \{\boldsymbol{0}, \overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}\}, \sum_{i=1}^n a_{ji} z_i \equiv 0 (\bmod\ q), j = 1, 2, \cdots, k\Big]\\
&\le \left|\left\{\widehat{\boldsymbol{v}} \in \mathcal{D} \middle| \|\overline{\boldsymbol{v}}\| \le \|\overline{\boldsymbol{e}}\| = \sqrt{\frac{n+1}{4}}\right\}\right| \cdot \prod_{j=1}^k \Pr\Big[\sum_{i=1}^n a_{ji} z_i \equiv 0 (\bmod\ q)\Big].
\end{aligned}$$

The first factor can be bounded by $2^{(c_1+1)n+1}$.

To bound the second factor, we first show that there must exist $t, 1 \le t \le k$ such that $(z_t, q) = 1$ when q is a prime. Otherwise, suppose $z_i \equiv 0 (\bmod\ q)$ for $i = 1, 2, \cdots, n$. Then we have

$$v_i \equiv v_{n+1}(1 - 2e_i) = \pm v_{n+1} (\bmod\ q), \tag{6}$$

since $e_i \in \{0, 1\}$, for $i = 1, 2, \cdots, n$. Notice that here, if v_i or v_{n+1} is a fraction, then $v_i \equiv \pm v_{n+1} (\bmod\ q)$ means $q | (v_i \mp v_{n+1})$.

We claim that $v_{n+1} \neq 0$. Otherwise there must exist some v_j such that $v_j = \alpha q$, where $\alpha \in \mathbb{Z}$ and $\alpha \neq 0$, since $\boldsymbol{v} \neq \boldsymbol{0}$. So $\frac{\sqrt{n+1}}{2} \ge \|\widehat{\boldsymbol{v}}\| \ge |\alpha| q > \frac{\sqrt{n+1}}{2}$, which is a contradiction. Similarly, there does not exist nonzero integer α such that $v_{n+1} = \alpha q$. So we have $v_{n+1} \not\equiv 0 (\bmod\ q)$.

From (6), we know that there exist integers $\beta_1, \beta_2, \cdots, \beta_n$ such that $v_i = \pm v_{n+1} + q\beta_i$, $i = 1, 2, \cdots, n$. Since $v_{n+1} \not\equiv 0 (\bmod\ q)$ and $v_{n+1} = \frac{1}{2} x_{n+1}$, we have $|v_i| \ge \frac{1}{2}$. Together with $|v_{n+1}| \ge \frac{1}{2}$, we have

$$\|\widehat{\boldsymbol{v}}\| \ge \frac{\sqrt{n+1}}{2}.$$

Since $\|\widehat{\boldsymbol{v}}\| \le \frac{\sqrt{n+1}}{2}$, we know that $\|\widehat{\boldsymbol{v}}\| = \frac{\sqrt{n+1}}{2}$, which implies that $|v_i| = \frac{1}{2}$ for $i = 1, 2, \cdots, n+1$. Hence $x_{n+1} = \pm 1$, $\beta_i = 0$, which corresponds to $\boldsymbol{v} = \mp \boldsymbol{e}$. This is also a contradiction.

Thus, there must exist $t, 1 \le t \le k$ such that $(z_t, q) = 1$. We have

$$\Pr\Big[\sum_{i=1}^n a_{ji} z_i \equiv 0 (\bmod\ q)\Big] = \Pr\Big[a_{jt} \equiv -\frac{\sum_{i=1, i\neq t}^n a_{ji} z_i}{z_t} (\bmod\ q)\Big] \le \frac{1}{q}.$$

Hence,

$$\Pr[X \neq \emptyset] \leq \frac{2^{c_1 n}}{q^k} 2^{c_1+1}.$$

The proof is finished. ∎

Remark 3.6 Similarly, if we allow the variables $x_i \in \{-1, 0, 1\}$, we can consider the lattice $\mathcal{L}$ spanned by vectors $\boldsymbol{b}'_1, \boldsymbol{b}'_2, \cdots, \boldsymbol{b}'_{n+k}, \widetilde{\boldsymbol{b}}'_{n+k+1}$ where $\boldsymbol{b}'_i$'s $(i = 1, 2, \cdots, n+k)$ are exactly the same to those in the proof of Theorem 3.3, and

$$\widetilde{\boldsymbol{b}}'_{n+k+1} = (0, 0, \cdots, 0, 1, Ns_1, \quad Ns_2, \quad \cdots, \quad Ns_k),$$

where N is an integer greater than $\sqrt{n+1}$.

Notice that there is a vector $\overline{\boldsymbol{e}} = (e_1, e_2, \cdots, e_n, 1, 0, \cdots, 0) \in \mathcal{L}$ with length $\sqrt{n+1}$. Similarly, we can define $X = \{\boldsymbol{v} \in \mathcal{L} | 0 < \|\boldsymbol{v}\| \leq \|\overline{\boldsymbol{e}}\|, \boldsymbol{v} \notin \{\boldsymbol{0}, \overline{\boldsymbol{e}}, -\overline{\boldsymbol{e}}\}\}$ and get

$$\Pr[X \neq \emptyset] \leq \frac{2^{c_2 n}}{q^k} 2^{c_2}((n+1)\sqrt{n}+1)^k. \tag{7}$$

Furthermore, when q is a prime larger than $\sqrt{n+1}$, the probability turns out to be

$$\Pr[X \neq \emptyset] \leq \frac{2^{c_2 n}}{q^k} 2^{c_2}. \tag{8}$$

Notice that for $k = 1$, the analysis is much simpler than before and the upper bounds (7), (8) are better and simpler than the previous result (3).

4 Some Strategies to Solve the Corresponding SVP in Practice

In this section, we give some strategies to solve the SVP in the corresponding lattices of the multiple SSP and multiple MSSP in practice.

4.1 By Computing the Intersection of Lattices

Take the multiple SSP as an example. We should determine the non-zero shortest vector of the lattice $\mathcal{L}$, whose basis is

$$\begin{bmatrix} 1, \ 0, \ \cdots, \ 0, \ 0, \ Na_{1,1}, \ Na_{2,1}, \ \cdots, \ Na_{k,1} \\ 0, \ 1, \ \cdots, \ 0, \ 0, \ Na_{1,2}, \ Na_{2,2}, \ \cdots, \ Na_{k,2} \\ \vdots \qquad\qquad \vdots \\ 0, \ 0, \ \cdots, \ 1, \ 0, \ Na_{1,n}, \ Na_{2,n}, \ \cdots, \ Na_{k,n} \\ \frac{1}{2}, \ \frac{1}{2}, \ \cdots, \ \frac{1}{2}, \ \frac{1}{2}, \ Ns_1, \quad Ns_2, \ \cdots, \ Ns_k \end{bmatrix}.$$

Notice that solving the shortest non-zero vector of $\mathcal{L}$ is equivalent to solving the shortest non-zero vector of the lattice

$$\mathcal{L}_1 \cap \mathcal{L}_2 \cap \cdots \cap \mathcal{L}_k,$$

where $\mathcal{L}_i$ is the lattice generated by

$$\begin{bmatrix} 1, \ 0, \ \cdots, \ 0, \ 0, \ Na_{i,1} \\ 0, \ 1, \ \cdots, \ 0, \ 0, \ Na_{i,2} \\ \vdots \\ 0, \ 0, \ \cdots, \ 1, \ 0, \ Na_{i,n} \\ \frac{1}{2}, \frac{1}{2}, \cdots, \frac{1}{2}, \frac{1}{2}, \ Ns_i \end{bmatrix},$$

$i = 1, 2, \cdots, k$.

As we know, the intersection of lattices is always another lattice, which might live in a lower dimension. Thus, we can obtain the shortest vector by computing the shortest vector in $\mathcal{L}_1 \cap \mathcal{L}_2 \cap \cdots \cap \mathcal{L}_k$, which may be easier since it has lower dimension than $\mathcal{L}$.

Obviously, we can compute the intersection of lattices if we can determine a lattice basis for the intersection of two n-dimensional lattices. For simplicity, we show how to compute a lattice basis for the intersection of two n-dimensional lattices, say $\mathcal{L}_1$ and $\mathcal{L}_2$. Denote by B_1, B_2 the basis matrix of $\mathcal{L}_1$, $\mathcal{L}_2$ respectively. Then $\mathcal{L}_1 \cap \mathcal{L}_2$ is the set of all the vectors which can be represented as $\boldsymbol{x} \cdot B_1 = \boldsymbol{y} \cdot B_2$, where $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{Z}^n$. Thus we have,

$$(\boldsymbol{x}, \boldsymbol{y}) \begin{pmatrix} B_1 \\ -B_2 \end{pmatrix} = \mathbf{0}.$$

The set of all the integer solutions of the equation above is also a lattice $\mathcal{L}'$. By the algorithm in [15] (page 74), we can find an integral basis $(x_1, y_1), (x_2, y_2), \cdots, (x_k, y_k)$ of $\mathcal{L}'$. Then, it can be concluded that $x_1 B_1, x_2 B_1, \cdots, x_k B_1$ will be a basis of the lattice $\mathcal{L}_1 \cap \mathcal{L}_2$.

4.2 By Computing the Kernel

Furthermore, solving the multiple SSP can also be seen as finding a small solution of a system of some inhomogeneous linear equations, which can be viewed as a closest vector problem by considering the corresponding homogeneous linear equations and an arbitrary solution of the inhomogeneous equations.

Assume the inhomogeneous linear equations defined by the multiple subset sum problem

$$\begin{aligned} \sum_{i=1}^{n} a_{1,i} x_i &= s_1, \\ \sum_{i=1}^{n} a_{2,i} x_i &= s_2, \\ &\vdots \\ \sum_{i=1}^{n} a_{k,i} x_i &= s_k, \end{aligned}$$

has a solution $\boldsymbol{e} = (e_1, e_2, \cdots, e_n) \in \{0,1\}^n$. Let L be the set of all integer solutions to the corresponding homogeneous equations, that is,

$$L = \left\{ (x_1, x_2, \cdots, x_n) \in \mathbb{Z}^n \,\middle|\, \sum_{i=1}^{n} a_{j,i} x_i = 0 \text{ for } j = 1, 2, \cdots, k \right\}.$$

L is in fact a lattice which has smaller dimension. Let $(y_1, y_2, \cdots, y_n)$ be an arbitrary integer solution of the inhomogeneous equations. Then the vector $\boldsymbol{v} = (y_1 - e_1, y_2 - e_2, \cdots, y_n - e_n) \in L$. Notice that $\boldsymbol{v}$ is very close to the vector $\boldsymbol{t} = (y_1 - \frac{1}{2}, y_2 - \frac{1}{2}, \cdots, y_n - \frac{1}{2})$. Thus, by finding the closet vector to $\boldsymbol{t}$ in the lattice L, we may recover $\boldsymbol{v}$ and hence $\boldsymbol{e}$.

Remark 4.1 To find a lattice vector which is closest to a given target vector is called the closest vector problem (CVP), which is also an NP-hard problem. With a CVP oracle, we can solve all the (multiple) subset sum problems regardless of their density (for example, see [16]). However, if we only have an SVP oracle, we can also replace the CVP oracle with the SVP oracle by Kannan's well-known embedding technique[17]. Nguyen and Stern[18] considered such a reduction for the single subset sum problem. We can also generalize it to the multiple SSP and multiple MSSP.

5 Conclusions

In this paper, we show how to solve low-density multiple subset sum problem with an SVP oracle. Some extended versions of the multiple subset sum problem are also considered.

References

[1] Garey M R and Johnson D S, *Computer and Intractability: A Guide to the Theory of NP-Completeness*, Freeman W H and San Francisco, 1979.

[2] Lenstra A K, Lenstra Jr H W, and Lovász L, Factoring polynomials with rational coefficients, *Mathematische Ann.*, 1982, **261**: 513–534.

[3] Schnorr C P, A hierarchy of polynomial lattice basis reduction algorithms, *Theoretical Computer Science*, 1987, **53**: 201–224.

[4] Schnorr C P and Euchner M, Lattice basis reduction: Improved practical algorithms and solving subset sum problems, *Mathematics of Programming*, 1994, **66**: 181–199.

[5] Feng Y, Wu W Y, Zhang J Z, and Chen J W, Exact bivariate polynomial factorization over $\mathbb{Z}$ by approximation of roots, *Journal of Systems Science and Complexity*, 2015, **28**(1): 243–260.

[6] Lagarias J C and Odlyzko A M, Solving low-density subset sum problems, *J. Assoc. Comp. Mach.*, 1985, **32**(1): 229–246.

[7] Coster M J, Joux A, LaMacchia B A, Odlyzko A M, Schnorr C P, and Stern J, Improved low-density subset sum algorithms, *Comput. Complexity*, 1992, **2**: 111–128.

[8] Li D and Ma S, Two notes on low-density subset sum algorithm, *Proceedings of ISAAC'94*, Ed. by Du D and Zhang X, *Lecture Notes in Computer Science*, Springer-Verlag, 1994, **834**: 164–171.

[9] Wang H, Xiao H, and Xiao G, EMSSP and its lattice reduction analysis, *J. of Xidian University*, 2000, **27**(5): 616–618 (in Chinese).

[10] Liu M J, Wang X Y, Bi J G, and Zheng X, Finding shortest lattice vector for lattice with gaps, previous version, available at http://eprint.iacr.org/2011/139.

[11] Merkle R and Hellman M, Hiding information and signatures in trapdoor knapsacks, *IEEE Transactions on Information Theory*, 1978, **24**(5): 525–530.

[12] Odlyzko A M, The rise and fall of knapsack cryptosystems, *Proceedings of Symposia in Applied Mathematics*, Ed. by Pomerance C, *Cryptology and Computational Number Theory, Ameri. Math. Soc.*, 1990, **42**: 75–88.

[13] Hästad J, Solving simultaneous modular equations of low degree, *SIAM J. Comput.*, 1988, **17**: 336–341.

[14] Mazo J E and Odlyzko A M, Lattice points in high-dimensional spheres, *Monatsh. Math*, 1990, **110**: 47–61.

[15] Cohen H, *A Course in Computational Algebraic Number Theory*, Springer-Verlag, Germany, 1996.

[16] Micciancio D, The hardness of the closest vector problem with preprocessing, *IEEE Transactions on Information Theory*, 2001, **47**(3): 1212–1215.

[17] Kannan R, Minkowski's convex body theorem and integer programming, *Math. Oper. Res.*, 1987, **12**(3): 415–440.

[18] Nguyen P Q and Stern J, Adapting density attacks to low-weight knapsacks, *Proceedings of Asiacrypt* 2005, Ed. by Lee P, *Lecture Notes in Computer Science*, Springer, 2005, **3788**: 41–58.