Fast Algorithm to solve a family of SIS problem with l_∞ norm

Jintai Ding[1,2]

Southern Chinese University of Technology, 1
University of Cincinnati, 2
ding@math.uc.edu

Abstract. In this paper, we present a new algorithm, such that, for the small integer solution (SIS) problem, if the solution is bounded (by an integer β in l_∞ norm, which we call a bounded SIS (BSIS) problem, *and if the difference between the row dimension n and the column dimension m of the corresponding matrix is relatively small with respect the row dimension m*, we can solve it easily with a complexity of polynomial in m.

Keywords: SIS, Lattice, l_∞ norm bounded, multivariate polynomials, linerization

1 Introduction

The Learning with Errors (LWE) problem, introduced by Regev in 2005 [5], has attracted a lot of attentions in theory and applications due to its usage in cryptographic constructions with some good provable secure properties. The main claim is that it is hard as worst-case lattice problems and hence the related cryptographic constructions.

In the first paper of this series[4], we present a new algorithm to solve a subclass of the LWE problems, which, we call, the learning with bounded errors (LWBE) problems, namely the errors from the queries do not span the whole finite field but a fixed known subset of size D ($D < q$). We show that we can solved this problem with a polynomial complexity.

In this paper, we will deal with a closely related problem, which is the SIS problem, which was formulated by Ajtai.

SIS problem can be described as follows.

Let q be a prime number and A $\in \mathbb{Z}_q^{n\times m}$, where A is chosen from a distribution negligibly close to uniform over $\mathbb{Z}_q^{n\times m}$.

$\Lambda_q^\perp(A) = \{\boldsymbol{x} \in \mathbb{Z}^m : A\boldsymbol{x} \equiv \mathbf{0} \in \mathbb{Z}^n \pmod q\}$ is an m-dimensional lattice. The SIS problem is to find a vector $\boldsymbol{v} \in \Lambda_q^\perp(A)$ with $\|\boldsymbol{v}\|_p \le \beta$.

Here, there are 5 key parameters n, m and q, p and β

We would remark here that our algorithm also works in the case the solution are not small integers but integers in a fixed small subset of size like β.

Clearly here we exclude the case q = 2, just like our previous paper for the case of LWE problem[4].

On the theory side, there are many arguments that the SIS problem is very hard due to the connection of SIS problems with the worst-case lattice problems such as SIVP (the shortest independent vectors problem).

In this paper, we present a new algorithm to solve a subclass of the SIS problems, which, we call, the bounded SIS (BSIS) problems, namely

$$\beta = \infty,$$

where the small integer component of (x) are all bounded by the integer β, and if

$$m > \text{or} \approx Q(m-n, D),$$

where

$$D = 2\beta + 1,$$

$$Q(y) = \binom{D}{y+D} = \frac{(y+D)!}{D!(y-1)!}.$$

Here $Q(y)$ is the number of monomial (including 1) in the polynomial ring $F_q[x_1, ..., x_y]$ when D is less than q. Therefore the number of monomials (excluding 1) is exactly $Q(y) - 1$.

We show that we can solved this problem with polynomial complexity in $O(m^3)$.

The main motivation to consider this problem comes from the consideration that often in the lattice related problems, the short vector (or the 'error') are often select mainly from the small set $\{1, -1\}$ like in NTRU, SWIFT and other type of systems.

This paper is organized as follows. Section 2 presents the algorithm and explains how it works with an example. Section 3 presents the basic analysis of the algorithm and the complexity. The last section is devoted to conclusion and discussions.

2 The new algorithm

Let us first define BSIS problem.

2.1 The BSIS

BSIS problem is given as follows.

There are 3 parameter n, m, a prime modulus q, and a fixed positive integer β.

The BSIS problem is to find a vector $\boldsymbol{v} \in \Lambda_q^{\perp}(A)$ with $\|\boldsymbol{v}\|_\infty \leq \beta$, where A $\in \mathbb{Z}_q^{n\times m}$, is chosen from a distribution negligibly close to uniform over $\mathbb{Z}_q^{n\times m}$, and

$$\Lambda_q^{\perp}(A) = \{\boldsymbol{x} \in \mathbb{Z}^m : A\boldsymbol{x} \equiv \mathbf{0} \in \mathbb{Z}^n \pmod q\}$$

, which can be view as an (m-n)-dimensional lattice.

Here, let us add our first assumption, that the matrix A is row independent, if not, we can always to a Gaussian to make the rows linearly independent.

From now on, let us assume that $n < m$.

Also for the simplicity, the problem we will look into is a subclass of the BSIS problems, namely we require that $n - m$ is relatively small which is defined as:

$$m > \text{or} \approx Q(m-n, D).$$

Tn the case $D = 2$, this means that

$$m - n \approx 2\sqrt{m};$$

and in the case of $D = 3$,

$$m - n \approx 6\sqrt[3]{m}.$$

2.2 The new algorithm

Clearly the vector $\boldsymbol{v}$ is a short solution to the equation:

$$A(x) = 0.$$

where

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{pmatrix}.$$

Since we know that $\boldsymbol{v}$ is bounded in the l_∞ norm β. Then we immediately have that

$$\prod_{j=-\beta}^{j=\beta} (x_i - j) = 0.$$

This is a set of m degree D equations with m

Then we also have a set of linear equations

$$AX = 0,$$

which is a set of linear equations.

For the set of linear equations, we can perform Gaussian elimination, and then we substitute these equations into the set of m degree D equations.

Then we have a set of m degree D equations with only now $m - n$ variables.

1. If $m \approx Q(m-n, D)$, then we can perform a partial enlargement as in [3], to produce a large set of more that $Q(m-n, D+1)$ degree less or equal to $D+1$ equations, then we can solve it the complexity is roughly $O(m^{3+3/d})$.
2. If $m > Q(m-n, D)$, then we can perform a linearization as in the case of [4], namely we treat each monomial as an independent variable and by solving a set of linear equations, we should be able to find the solution we are looking for, and the complexity is roughly $O(m^3)$. If it does not work , we will do as in the case above.

3 Analysis of the Algorithm and Complexity

One can see easily that the success of the algorithm depends on if we can solve the linear equation, which comes from the linearization of the set of m polynomial equations of degree D with $m-n$ variables.

Since the entries of follows certain almost uniformly distribution, it is not unreasonable to assume that coefficients of the matrix that those m nonlinear equations are quite generic and whose coefficients are somewhat randomly and uniformly in this case, it is not at all difficult to deduce that we have a very good probability to succeed at degree D or for sure we will succeed in degree $D+1$ in the linearization solving step.

In all the extensive experiments (with relative large q and $D = 3$), we have never failed. Therefore the conclusion is that algorithm works nearly 100 percent.

Now let us look at the complexity. It is clear that the matrix size of the linearization step is either m, if we solve at degree D or the size of roughly $m \times \sqrt[D]{m}$, if we solve at degree $D+1$. Then the complexity should be either $O(m^3)$ or $O(m^{3+3/d})$.

therefore the complexity is for sure polynomial in m.

On the other hand, surely the biggest memory requirement is to store the matrix associate with linearization, which is of the size roughly

$$O(m^2)).$$

Then one may ask about the case when $m-n$ is large, for example, in the case of NTRU, where the associated SIS problem is the case $m = 2n$. In such a case, we can use some of the polynomial solving algorithms such as Groebner basis or MXL algorithms [3], the complexity is surely expected to be much higher and we do not expect to solve such a problem easily using directly our algorithm. The details will be discussed further a subsequent paper.

Another remark we have is that our algorithm is designed to solve a subclass of the SIS problem, but we can easily transform it into an algorithm a short vector problem, where the short vector is bounded by a l_∞ norm of size β, the conclusion, we can draw here that does not really depend on the distribution of the errors on the error set. Similarly we can draw here is that if the dimension of the lattice is relative small compared with the total dimension of the space and it is l_∞ bounded, then this is an easy problem just like the problem we solve in this paper, since they are equivalent.

4 Conclusion

We present a new algorithm to solve a subclass of the small integer solution (SIS) problem, if the solution is bounded (by an integer β in l_∞ norm, which we call a bounded SIS (BSIS) problem, and *and if the difference between the row dimension and the column dimension of the corresponding matrix is relatively small with respect the row dimension*, the complexity is polynomial. in m, th dimension of the solution vector.

This algorithm, we believe, present a new direction to look at the security of the cryptographic algorithms that are related to the SIS problem, in particular the NTRU cryptosystems and the SWIFT family of algorithms. We hope that we can develop new attack tools along this line.

5 Acknowledgment

I would like to thank R. Linderner and D. Cabarcas for useful discussions.

This work is partially supported by NSF China and the Taft research center at the University of Cincinnati.

References

1. Sanjeev Arora, Rong Ge, Learning Parities with Structured Noise, TR10-066, April 2010
2. A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. Journal of the ACM, 50(4):506519, 2003.
3. Johannes Buchmann, Daniel Cabarcas, Jintai Ding, Mohamed Saied Emam Mohamed: Flexible Partial Enlargement to Accelerate Grbner Basis Computation over F2. AFRICACRYPT 2010: 69-81
4. Jintai Ding, Solving LWE problem with bounded errors in polynomial time, Cryptology ePrint Archive, Report 2010/558, 2010, http://eprint.iacr.org/.
5. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM, 56(6):34, 2009. Preliminary version in STOC05.
6. Oded Regev, The Learning with Errors Problem (Invited Survey), CCC, pp.191-204, 2010 25th Annual IEEE Conference on Computational Complexity, 2010