Algorithm for SIS and MultiSIS problems

Igor Semaev
Department of Informatics, University of Bergen, igor@ii.uib.no

August 18, 2020

Abstract

SIS problem has numerous applications in cryptography. Known algorithms for solving that problem are exponential in complexity. A new algorithm is suggested in this note, its complexity is sub-exponential for a range of parameters.

1 Introduction

Let $A$ be any integer $m \times n$ matrix, where $m > n$ and $q$ be a prime. Assume $A$ is of rank $n$ modulo $q$. Let $c = (c_1, \ldots, c_m)$ be an integer vector of length $m$ and $|c| = (c_1^2 + \ldots + c_m^2)^{1/2}$ denote its norm (Euclidean length) and $\nu$ be a positive real. The SIS (Short Integer Solution) problem is to construct a non-zero integer row vector $c$ of length $m$ and norm at most $\nu$ such that $cA \equiv 0 \mod q$. The problem of constructing several such short vectors is called MultiSIS problem.

The inhomogeneous SIS problem asks for a short vector $c$ such that $cA \equiv a \mod q$ for a non-zero row vector $a$ of length $n$. The inhomogeneous SIS problem may be reduced to a homogeneous SIS problem. Let $A_1 = \begin{pmatrix} A \\ a \end{pmatrix}$ be a concatenation of the matrix $A$ and the vector $a$. Assume one constructs a number of short solutions $c_1$ to $c_1A_1 \equiv 0 \mod q$ with non-zero last entry. One of them may likely be $c_1 = (c, 1)$ and that gives a solution to $cA \equiv a \mod q$, or such a vector may be found as a combination of the solutions to the SIS problem.

Typical SIS problem parameters are $\nu \geq \sqrt{\frac{n \log_2 q}{2}}$ and $m > n \log_2 q$, where $q$ is bounded by a polynomial in $n$. The problem may be reduced to constructing short vectors in general lattices, which is considered hard, see [1]. The SIS problem has a number of applications in cryptography, see [6]. For instance, the hash function $x \rightarrow xA$ was suggested in [1].

Integer vectors $c$ such that $cA \equiv 0 \mod q$ is a lattice of dimension $m$ and volume $q^m$. So all vectors of norm $\leq \nu$ may be computed with a lattice enumeration in time $m^{O(m)}$, see [3]. Alternatively, one may apply a lattice reduction algorithm. The reduction cost is $2^{O(m)}$ operations according to [3]. The so-called combinatorial algorithms to solve the
SIS problem and its inhomogeneous variant, where the entries of \( c \) are 0 or 1, are surveyed in [2]. They have complexity \( 2^{O(m)} \) operations. All above methods are thus exponential in complexity. In this note a new algorithm for solving SIS and MultiSIS problems is introduced. The complexity of the algorithm is sub-exponential for a range of parameters.

2 MultiSIS Problem

How to construct \( N \) different non-zero vectors \( c \) of norm at most \( \nu \) such that \( cA \equiv 0 \mod q \)? The vectors generated by the rows of the matrix \( qI_m \), where \( I_m \) denotes a unity matrix of size \( m \times m \), are trivial solutions and not counted. We call this MultiSIS problem. Obviously, a solution to the MultiSIS problem implies a solution to the homogeneous SIS problem. That may also imply a solution to a relevant inhomogeneous problem as it is explained earlier.

The MultiSIS problem may be solved by lattice enumeration. Alternatively, one perturbs the initial basis of the lattice \( N \) times and apply a lattice reduction algorithm after each perturbation. So the overall complexity is \( N2^{O(m)} \), though we do not know if that really solves the problem as the vectors in the reduced bases may repeat.

If \( m = o(\nu^2) \), then the number of integer vectors \( c \) of norm at most \( \nu \) is approximately the volume of a ball of radius \( \nu \) centred at the origin. With probability \( 1/q^n \) the vector \( c \) satisfies \( cA \equiv 0 \). Therefore the number of such relations is around

\[
\frac{\pi^{m/2} \nu^m}{\Gamma(m/2 + 1)} q^n \approx \frac{(2\pi e)^{m/2}}{\sqrt{\pi m}} \left( \frac{\nu}{\sqrt{m}} \right)^m \frac{1}{q^n}
\]

and should be at least \( N \) to make the problem solvable. That fits the so-called Gaussian heuristic, see [4].

According to [5], if \( \nu = O(\sqrt{m}) \) the Gaussian heuristic does not generally hold. We will use a different argument still heuristic. Let \( \nu < \sqrt{m} \) and \( d = \lfloor \nu^2 \rfloor \). For each subset \( A_{i_1}, \ldots, A_{i_r} \) of \( r \leq d \) rows of \( A \) there are \( 2^r \) linear combinations \( c_1A_{i_1} + \ldots + c_rA_{i_r} \), where \( c_i = \pm 1 \) and so \( c = (c_1, \ldots, c_r) \) is of norm \( \leq \nu \). We do not distinguish between \( c \) and \( -c \). So the expected number of such zero combinations is \( 2^{r-1}/q^n \). For the whole matrix the expected number of different \( c \) of norm at most \( \nu \) such that \( cA \equiv 0 \) is at least

\[
\sum_{r=1}^{d} \binom{m}{r} 2^{r-1}/q^n.
\]

Therefore, \( N \) such relations do exist if

\[
\sum_{r=1}^{d} \binom{m}{r} 2^{r-1}/q^n \geq N,
\]

minding that the inequality is approximate.

2.1 MultiSIS Algorithm

Let \( \delta = m/n \ln q \) and \( \eta = \nu^2/n \ln q \). In this section we present the algorithm to construct vectors \( c \) of norm at most \( \nu \) such that \( cA \equiv 0 \mod q \). In Section 2.2 we will show that if at least one of \( \delta \) or \( \eta \) tends to infinity, then one may construct \( q^{\frac{\nu^2}{(1+o(1))}} \) such vectors with the complexity \( q^{t(1+o(1))} \) operations, where \( t = \lfloor \log_2 \sqrt{\eta \ln \delta} \rfloor (1 + o(1)) \). The latter tends to infinity, so the complexity is sub-exponential. If both \( \delta \) and \( \eta \) are bounded, then
the complexity is represented by the same expression for some bounded $t$ and therefore exponential. The analysis is heuristic.

Let $d \geq 2, k < m, N$ be integer parameters such that $\nu = d\sqrt{k}$. We may assume that $d = 2^t$ for an integer $t = \log_2 d$ and $n = st$ for an integer $s$. Otherwise, the algorithm below is easy to adjust. Let $m(k)$ be the number of integer vectors of length $m$ and of norm $\leq \sqrt{k}$ up to a multiplier $-1$. It is easy to see that $m(k) \geq \sum_{i=1}^{k} \binom{m}{i}2^{t-1}$.

1. Put $\mathfrak{A}_0 = C_0A$, where $C_0$ be a matrix of size $m(k) \times m$ and each row of $C_0$ is an integer vector of norm at most $\sqrt{k}$.

2. Let $N_i$ for $i$ in $0, \ldots, t - 1$ be integers such that $N_i = q^{s(1+o(1))}$, where $N_0 \leq m$ and $N_t = N$.

3. For $i = 0, \ldots, t - 1$ do the following. Represent $\mathfrak{A}_i = \mathfrak{A}_{i1}|\mathfrak{A}_{i2}$ as a concatenation of two matrices, where $\mathfrak{A}_{i1}$ is of size $N_i \times s$ and $\mathfrak{A}_{i2}$ is of size $N_i \times s(t - i - 1)$. As $N_i = q^{s(1+o(1))}$ there are $N_{i+1} = q^{s(1+o(1))}$ relations $c\mathfrak{A}_{i1} \equiv 0$, where $c$ is a vector of length $N_i$ and it has at most two non-zero entries which are $\pm 1$. Let $C_{i+1}$ be a matrix of size $N_{i+1} \times N_i$ with such rows. Equivalently, there are $q^{t(1+o(1))}$ pairs of rows in $\mathfrak{A}_{i1}$, where one row differs from another by a multiplier $\pm 1$, and zero rows in $\mathfrak{A}_{i1}$. Such pairs of rows and zero rows in $\mathfrak{A}_{i1}$ may be computed in $N_i^{1+o(1)}$ operations by sorting. Put $\mathfrak{A}_{i+1} = C_{i+1}\mathfrak{A}_{i2}$ and repeat the step.

4. The matrix $C = C_t \ldots C_1C_0$ is of size $N \times m$ and it satisfies $CA \equiv 0$. Each row of $C$ has norm $\leq \nu = d\sqrt{k}$.

The rows of $C_0$ are different and non-zero. At each step of the algorithm one may choose $C_i$ such that the rows of $C_i \ldots C_1C_0$ are different. As the rows of $C_{i+1}$ have at most two non-zero entries which are $\pm 1$, the rows of $C_{i+1}C_i \ldots C_0$ are all non-zero. Though we can not guarantee theoretically that all constructed vectors are different, the algorithm works well in practice.

2.2 Analysis of the Algorithm

The algorithm constructs $q^n^{k(1+o(1))}$ integer vectors $c$ of norm at most $\nu$ such that $cA \equiv 0 \pmod{q}$ and its complexity is $q^{t(1+o(1))}$ operations. We will define an optimal $t = \log_2 d$. For any input parameters $n, q, m, \nu$ one may find $t$ by solving numerically the system $m(k) \geq q^t$ and $\nu = 2^t \sqrt{k}$.

Let $\delta = m/n \ln q$ and $\eta = \nu^2/n \ln q$ and at least one of them is an increasing function in $n$. We will represent $t$ as a function of $\delta, \eta$. First, we find $k$ such that $m(k) \geq q^\tilde{\eta}$ for large $n$. One may solve a stronger inequality $\binom{m}{k}2^{k-1} \geq q^\tilde{\eta}$ instead. With the Stirling approximation to the factorial function, it is easy to see that one may take $k = \frac{\alpha n}{\delta} (1 + o(1))$, where

$$\alpha = \frac{\ln q}{\ln m - \ln \ln q^\tilde{\eta}} = \frac{\ln q}{\ln(\delta t)}.$$
So $k = \frac{n \ln q}{t \ln(\delta t)} (1 + o(1))$ and the equation $\nu = d \sqrt{k}$ is equivalent to

$$\eta = \frac{4t}{t \ln(\delta t)} (1 + o(1)).$$

(1)

The solution to (1) is

$$t = \log_2 \sqrt{\eta \ln \delta} \cdot (1 + o(1)).$$

Experimentally, $t > \log_2 \sqrt{\eta \ln \delta}$ and they converges for very large parameters. The complexity of the algorithm is $q^{\frac{n \ln q}{t \ln(\delta t)} (1 + o(1))}$.

References

[1] M. Ajtai, *Generating hard instances of lattice problems*. Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. ACM, 1996.

[2] S. Bai, S. Galbraith, L. Li, D. Sheffield, *Improved Combinatorial Algorithms for the Inhomogeneous Short Integer Solution Problem*, J Cryptol 32, pp. 35–83 (2019)

[3] G. Hanrot, X. Pujol and D. Stehlé, *Algorithms for the Shortest and Closest Lattice Vector Problems*, in IWCC 2011. LNCS 6639, Springer, Berlin, Heidelberg, pp. 159–190.

[4] P.Q. Nguyen and B. Vallée(eds), *The LLL Algorithm. Survey and Applications*, Springer-Verlag, Berlin, Heidelberg, 2010.

[5] Mazo, J.E., Odlyzko, A.M. *Lattice points in high-dimensional spheres*. Monatshefte für Mathematik 110 (1990), pp. 47–61.

[6] C. Peikert, *A decade of lattice cryptography*. Cryptology ePrint Archive, Report 2015/939, 2015.