Introduction to Lattice-Based Cryptography

1 Introduction

So far, we have seen that one can find a short vector with approximation factor $\alpha = 2^{\Theta(n)}$ in polynomial time using LLL. However, our best algorithm to find the exact shortest vector ($\alpha = 1$) had complexity $T = 2^{O(n^2)}$. There exists further improvements, which are not covered in these lectures, that result in the fastest known algorithms to have time $T = 2^{O(n)}$. This remains the case even for heuristic algorithms (i.e. algorithms that come *without* proof, but that are backed up by informal and experimental arguments). There are also exist algorithms achieving *trade-offs* between the approximation factor and the running time. That is, the algorithm can be tuned to run in time $T = 2^{\Theta(n^c)}$ and achieve an approximation factor $\alpha = 2^{\Theta(n^{1-c})}$ for any chosen $c \in [0, 1)$. The bottom line is that for polynomial approximation factors $\alpha = n^{O(1)}$, finding short vectors appears exponentially hard. And with hard computational problems, comes the potential for secure cryptography.

To build cryptography, we first need to choose a family of lattices, or even more precisely, a random distribution of lattices. That distribution should be easy to sample from, and convinient to describe and compute with, for efficiency reasons. At the same time, the lattices should not have any weaknesses that would make them suceptible to more efficient attack than 'generic lattices'. This second requirement will be the object of the next Lecture 8.

In this lecture we are simply going to define and study the SIS problem, and study the underlying lattice. We will propose a gentle introduction to security reasoning in cryptography via reductions between problems, and finally show how the SIS problem rather easily give rise to a collision resistant hash function.

2 The Short Integer Solution Problem

Definition 1 *A lattice L is called an* integer lattice *if $L \subset \mathbb{Z}^n$. An integer lattice L is said to be q-ary if $q \cdot \mathbb{Z}^n \subset L$.*

For the rest of this course, q will be prime. The definitions remains valid for non-prime q, but many statements and proofs only hold for prime q. We now define the Short Integer Solution (SIS) problem. This problem is significantly different to computational problems that we have defined previously in this course, in that the instance is specified to come from a certain *distribution* instead of being an arbitrary instance from a set of possible instances: it is an *average-case* problem, not a *worst-case* problem.

Notation We will use $\mathbb{Z}_q$ as a short-hand for the ring $\mathbb{Z}/q\mathbb{Z}$ of integers modulo q, as it is the accepted notation in the cryptographic literature. This should *not* be confused with the ring of q-adic integers. For a finite set S or a measurable compact subset $S \subset \mathbb{R}^n$, we use $\mathcal{U}(S)$ to denote the uniform distribution over S.

Definition 2 (The SIS Problem) *For positive integers n, m, q and a positive real $\beta > 0$, the $\mathrm{SIS}_{n,m,q,\beta}$ problem is defined as follow:*

- *Given a uniformly random matrix* $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n\times m})$
- *Find a non-zero* $\mathbf{z} \in \mathbb{Z}^m \setminus \{\mathbf{0}\}$ *such that* $\mathbf{Az} = 0 \bmod q$ *and* $\|\mathbf{z}\| \leq \beta$.

This is well defined for any norm, though we will be interested mostly in the case of the Euclidean norm, and sometimes in the case of the ℓ_∞ norm; in the latter case we will denote the problem as SIS$^\infty$. The SIS problem can be phrased as a short vector problem, namely, after sampling a random $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n\times m})$, finding non-zero vector of length at most β in the lattice:

$$\Lambda_q^\perp(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m \mid \mathbf{Az} = \mathbf{0} \bmod q\}. \tag{1}$$

The problem is trivial when $\beta \geq q$, as $(q, 0, \ldots, 0)$ is a solution. The problem can also be vacuously hard (i.e. no solution exists) if β is too small, namely if $\beta < \lambda_1(\Lambda_q^\perp(\mathbf{A}))$. More specifically, we are going to establish bounds on the metric properties of $\Lambda_q^\perp(\mathbf{A})$, some of which are probabilistic.

Lemma 3 *For any* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $L := \Lambda_q^\perp(\mathbf{A})$ *is a q-ary lattice of rank m, and* $\det(L) \leq q^n$.

Proof: The inclusion $q\mathbb{Z}^m \subset L \subset \mathbb{Z}^m$ follows from definition of $\Lambda_q^\perp(\mathbf{A})$, and hence L is q-ary of rank m. For the determinant, note that L contains $q\mathbb{Z}^m$, and that $L = \ker(\mathbf{A} : \mathbb{Z}^m \to \mathbb{Z}_q^n)$, when $\mathbf{A}$ is seen as a group homomorphism. The isomorphism theorem tells us that $|\mathbb{Z}^m/L| = |\mathrm{im}(\mathbf{A})| \leq |\mathbb{Z}_q^n| = q^n$. Now use that $|\mathbb{Z}^m/L| = \det(L)/\det(\mathbb{Z}^m) = \det(L)$. □

We can then get an upper-bound on λ_1 by Minkowski's bound.

Corollary 4 *For any* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $L := \Lambda_q^\perp(\mathbf{A})$, *it holds that:*

$$\lambda_1^\infty(L) \leq q^{n/m} \quad \textit{and} \quad \lambda_1^2(L) \leq q^{n/m}\sqrt{2n/\pi e} + o(\sqrt{n}).$$

We now prove that, with overwhelming probability, this upper bound is in fact tight up to a factor close to 2.

Lemma 5 *For any* $f > 1$, *and a random* $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{n\times m})$, $L := \Lambda_q^\perp(\mathbf{A})$, *it holds except with probability at most* f^{-n} *that:*

$$\lambda_1^\infty(L) > \frac{(q/f)^{n/m} - 1}{2}.$$

Proof: We first claim that for any $\mathbf{x} \in \mathbb{Z}_q^m \setminus \{\mathbf{0}\}$,

$$\mathbb{P}_{\mathbf{A}\leftarrow\mathcal{U}(\mathbb{Z}_q^{n\times m})}[\mathbf{Ax} = \mathbf{0} \bmod q] = q^{-n}.$$

Indeed, because $\mathbb{Z}_q$ is also a finite field, one can verify that $\mathbf{Ax}$ is uniform in $\mathbb{Z}_q^n$ over the uniform choice of $\mathbf{A}$. Let us denote $\beta = \frac{(q/f)^{n/m}-1}{2}$, and count the number of vectors of ℓ_∞ norm less or equal to β:

$$|\{\mathbf{x} \in \mathbb{Z}^m \mid \|\mathbf{x}\|_\infty \leq \beta\}| = (2\beta+1)^m.$$

Using the union bound, we have

$$\mathbb{P}_{\mathbf{A}}[\exists \mathbf{x} \text{ nonzero with } \|\mathbf{x}\|_\infty \leq \beta \text{ and } \mathbf{Ax} = 0 \text{ modulo } q] \leq (2\beta+1)^m q^{-n} = (q/f)^n q^{-n} = f^{-n}.$$

So, the probability that $\lambda_1^\infty(L) \leq \beta$ is less than or equal to f^{-n}. □

One can also derive an lower bound for the Euclidean norm generically as $\lambda_1^2(L) \geq \lambda_1^\infty(L)$, but this can be shown to be loose, and the bound can be improved by a factor $\Theta(\sqrt{n})$ for the Euclidean norm. Alternatively, one may reproduce the above proof strategy directly for the Euclidean norm, to obtain a tight bound. This requires counting the number of points of $\mathbb{Z}^n$ in a Euclidean ball of a given radius, and is left to exercise in Sheet 7.

2.1 Geometric Digression

Before going back to considerations of algorithmic hardness and our path towards cryptography, let us make a few comments on purely geometric matters related to these random lattices.

Firstly, random lattices provide a lower bound on the Hermite constant, closely matching Minkowski's upper bound $\gamma_n \leq 2n/\pi e + o(n)$. This is (more or less) the Theorem of Minkowski-Hlawka.

THEOREM 6 (MINKOWSKI-HLAWKA) $\gamma_n \geq n/2\pi e - o(n)$.

What is remarkable is that apart from random lattices, we know very few explicit families of lattices that get close to Minkowski's bound up to a large constant factor, and their definitions are rather technical.[1]

Secondly, one can also deduce an upper-bound on $\lambda_m(L)$ by resorting to *transference*: the dual lattice can also be shown to be a scaling of a random q-ary lattice, but we do not have all the tools necessary to get there in this course. The conclusion is that for such random lattices, with overwhelming probability the successive minimal distances $\lambda_i(L)$ and the covering radius $\mu(L)$ are all within a constant factor of $q^{n/m}/2$ for the ℓ_∞ norm, and to $q^{n/m}\sqrt{n/2\pi e}$ for the ℓ_2 norm.

2.2 Hardness regimes of SIS

We return to our hardness considerations. Having determined that $\lambda_1(L) \approx q^{n/m}$ with overwhelming probability, we have three clear regimes:

1. If $\beta \geq q$ then the problem admits a trivial solution $(q, 0, \ldots, 0)$
2. If $\beta \geq (\gamma_2 + \varepsilon)^m \cdot q^{n/m}$, the problem can be solved by running LLL on the lattice L. In fact, one may notice that one can always drop columns of the matrix $\mathbf{A}$ to decrease the dimension n. So LLL can in fact solve the problem when $\beta \geq (\gamma_2 + \varepsilon)^{m'} \cdot q^{n/m'}$ for any $m' \leq m$.
3. If $\beta < q^{n/m}$, then the problem is vacuously hard with overwhelming probability.

Between Regime 2 and Regime 3 lies a potentially computationally hard problem that we can harness to build cryptography. More specifically, when $\beta = q^{n/m} \cdot n^{O(1)}$ (that is when β is polynomially larger than the expected minimal distance) all the lattice algorithm we know solve the problem in exponential time.

3 Collision resistance

An important notion in cryptography is *collision resistance*. Informally, a function f is collision resistant whenever it is computationally hard to find two different inputs x_1, x_2 such that $f(x_1) = f(x_2)$. To make this notion formal, we need some explanation and notation.

Of course, the identity function is a trivial answer: finding collisions is vacuously hard for it. The cryptographic challenge for such function is to achieve collision resistance while being *compressing*: the input space should be quite larger than the output space. In that case, collision

[1] To the best of my knowledge, there is only the class field tower construction of Martinet, and the so-called Mordell-Weil lattices based on pairing of elliptic curves from Shioda and Elkies.

will exists, but may still be hard to find. However, the output space needs to be large enough so that a collision can not be found by brute force.

Application. Collision resistant hash functions are one of the most basic building blocks for cryptography. But they have a simple direct application, namely *fingerprinting* large files F. Suppose that Alice wants to transmit a very large file to Bob, and the only practical solution is to use a third party (google drive, Wetransfer, ...). They are concerned that the file F could be tampered with, and want to be sure what Bob received F' is exactly what Alice sent: $F = F'$. To ensure this, Alice would compute a hash $h = H(F)$, and Bob would compute $h' = H(F')$. This hash can be quite short (64 bytes), and therefore can be transmitted more directly, say via a secure texting app (Signal, Whatsapp, ...). If $h = h'$, and if H is collision resistant, then they can be confident that $F = F'$.

Formalizing Security notions. To make this notion formal, we need some explanation and notation.

First, we would like to settle what it means to be 'computationally hard'. In computer science, this is often formalized using asymptotics and (probabilistic) polynomial-time Turing machines. In order to be able to properly talk about asymptotics, it makes no sense to talk about computational hardness of one function alone. Instead, one observes a function *family*, which is parametrized by a variable λ. Mathematically, one would denote that as follows:

$$\mathcal{F} = \{f_\lambda : \mathcal{K}_\lambda \times \mathcal{M}_\lambda \to \mathcal{H}_\lambda \mid \lambda \in \mathbb{N}\}.$$

The parametrization variable λ is often called the *security parameter*, since it indicates how 'hard' the collision problem is (which will be defined properly very soon). The letters for the domain and codomain of the functions f_λ come from their cryptographic interpretation. Namely, $\mathcal{K}$ stands for *key space*, $\mathcal{M}$ for *message space* and $\mathcal{H}$ for *hashing space*. In cryptography, collision resistant functions are often called hash functions, and inputs to such functions are viewed as messages. For fixed λ, the key space allows to use many different hash functions from $\mathcal{M}$ to $\mathcal{H}$, instead of just one.

Now, computational hardness of finding a collision can be defined as follows.

Definition 7 (Collision resistance) *A function family $\mathcal{F} = \{f_\lambda : \mathcal{K}_\lambda \times \mathcal{M}_\lambda \to \mathcal{H}_\lambda \mid \lambda \in \mathbb{N}\}$ is said to be collision resistant if, for all λ, and all probabilistic-polynomial time algorithms $\mathcal{A}$, holds that*

$$\mathop{\mathbb{P}}_{\substack{k \in \mathcal{K} \\ \mathcal{A}}}[f_\lambda(k, x_0) = f_\lambda(k, x_1) \text{ and } x_0 \neq x_1 \mid (x_0, x_1) \leftarrow \mathcal{A}(k)] \leq \mathit{negl}(\lambda).$$

Here, $\mathrm{negl}(\lambda)$ is any function in $\lambda^{-\omega(1)}$. Note that randomness is taken uniformly over the key space $\mathcal{K}$, as well as over the randomness of the algorithm $\mathcal{A}$.

In human language, the above definition says that a function family $\mathcal{F}$ is collision resistant whenever no algorithm (that runs in reasonable time) is able to find a collision in f_λ with some significant probability. Note that the notion of collision resistance is only interesting whenever the functions $f_\lambda(k, \cdot)$ are not injective. In real life, one therefore almost only looks at the case when the hashing space $\mathcal{H}$ is (much) smaller than $\mathcal{M}$.

4 Cryptography and reductions

In cryptography, one often relies on certain 'hardness assumptions'. For example, people tend to say that the security of the famous cryptosystem RSA relies on the hardness of factoring[2]. In lattice based cryptosystems, something similar is happening; for example, we would like to show that breaking a specific cryptographic protocol is at least as hard as some lattice problem (that is believed to be hard).

For example, suppose we wanted to prove that some prolem Y is at least as hard as problem X. This can be achieved by proving the following statement:

$$X \text{ is hard } \Rightarrow Y \text{ is hard.}$$

In computer science, this is proved as follows. Assume that there exists an *oracle* for Y. That is a miraculous (usually imaginary) algorithm that solves Y instantaneously. Now construct an algorithm $\mathcal{A}^Y$, that has access to the 'oracle' of Y and that solves X. If this algorithm is poly-time, one has proven that Y is at least as hard as X.

4.1 Collision-Resistance of Merkle-Damgård Domain Extension

An hopefully clarifying example is the following so-called Merkle-Damgård domain extension of hash functions. Let $f : \mathcal{K} \times \{0,1\}^m \to \{0,1\}^n$ with $m > n$ be any function. Let $b = m - n > 0$. Set

$$f_{MD}^{(\ell)} : \mathcal{K} \times \{0,1\}^{\ell \cdot b} \to \{0,1\}^n$$
$$f_{MD}^{(\ell)}(x_1 \cdots x_\ell) = f_k(\cdots f_k(f_k(f_k(0^n|x_1)|x_2)|x_3)\cdots)|x_\ell),$$

where $x_i \in \{0,1\}^b$.

THEOREM 8 *If f is collision resistant, then so is $f_{MD}^{(\ell)}$.*

PROOF: Suppose the latter is not collision resistant, meaning that there is an algorithm $\mathcal{A}$ that outputs a valid collision $(x = x_1 \cdots x_\ell, x' = x'_1 \cdots x'_\ell)$ with $f_{MD}^{(\ell)}(k, x_1 \cdots x_\ell) = f_{MD}^{(\ell)}(k, x'_1 \cdots x'_\ell)$ with probability $\frac{1}{p(\lambda)}$, where $p(\lambda)$ is a polynomial function of the security parameter.

Take the largest $i < \ell$ such that $f_{MD}^{(i)}(x_1 \cdots x_i) \neq f_{MD}^{(i)}(x'_1 \cdots x'_i)$. Then, taking $y = f_{MD}^{(i)}(x_1 \cdots x_i)|x_{i+1}$ and $y' = f_{MD}^{(i)}(x'_1 \cdots x'_i)|x'_{i+1}$ yields a collision for f.

If no such i exists, then let j be the smallest integer such that $x_j \neq x'_j$. If $j = 1$ then $y = 0^n|x_1$ and $y' = 0^n|x'_1$ yields a collision. Otherwise $y = f_{MD}^{(j-1)}(x_1 \cdots x_{j-1})|x_j$ and $y' = f_{MD}^{(j-1)}(x'_1 \cdots x'_{j-1})|x'_j$ yields a collision for f.

Then we also have an algorithm that finds a collision for f with polynomial probability, and f is therefore also not collision resistant. By contraposition, we obtain the claim. □

[2]In reality, this hardness assumption is slightly more complicated.

4.2 Collision resistant functions from the SIS problem

The key space $\mathcal{K} = \mathbb{Z}_q^{n\times m}$, is the set of all $n \times m$ matrices with coefficients in $\mathbb{Z}_q$. Set $\mathcal{M} = \{0,1,\ldots,b\}^m \approx \{0,1\}^{m\log_2 b}$ and $\mathcal{H} = \mathbb{Z}_q^n \approx \{0,1\}^{n\log_2 q}$. For key $\mathbf{A}$ and input message $x \in \mathcal{M}$, set

$$f_{\mathbf{A}}(x) := \mathbf{A}\mathbf{x} \bmod q,$$

where $\mathbf{x} \in \{0,1,\ldots,b\}^m$ denotes the adequate parsing of $x \in \{0,1\}^{m\log_2 b}$.

LEMMA 9 *If $\mathrm{SIS}^{\infty}_{m,n,q,b}$ is hard, then $f_\bullet : \mathcal{K}\times\mathcal{M} \to \mathcal{H}$ is collision resistant.*

PROOF: Let $\mathbf{A} \in \mathcal{K}$. Suppose we are able to find two distinct $x_1 \neq x_2 \in \mathcal{M}$ such that $f_{\mathbf{A}}(x_1) = f_{\mathbf{A}}(x_2)$. Let $\mathbf{x} = \mathbf{x}_1 - \mathbf{x}_2$. Then $\mathbf{A}\mathbf{x} = 0$, with $\|\mathbf{x}\|_\infty \leq b$ and $\mathbf{x} \neq \mathbf{0}$. So any algorithm that finds a collision of $f_\bullet$, can be used to solve SIS. More formally, one would define the reduction $\mathcal{R}^{\mathcal{A}}$ using an SIS oracle $\mathcal{A}$ as $\mathcal{R}^{\mathcal{A}}(\mathbf{A}) = \mathbf{x}_0 - \mathbf{x}_1$ where $(x_0, x_1) \leftarrow \mathcal{A}(\mathbf{A})$. □