CS 4501-6501 Topics in Cryptography January 28, 2015

Lecture 5

Lecturer: Mohammad Mahmoody *Scribe: Ameer Mohammed*

1 Learning with Errors: Motivation

An important goal in cryptography is to find problems that are conjectured to be computationally hard to solve as well as structurally well-suited for cryptographic constructions to be based on. Examples of such problems include number-theoretic assumptions (e.g. integer factorization is hard). Here we will discuss another such problem called the *learning with errors* (LWE) problem that was introduced by Regev in [Reg05] where it was shown that the hardness of LWE can be reduced to the hardness of the approximate shortest vector problem (SVP), a well-studied lattice problem that is believed to be computationally hard. Interestingly, unlike classical number-theoretical assumptions, LWE's hardness extends to quantum algorithms as well, making it a suitable candidate for use in post-quantum cryptography.

Besides its proven hardness, LWE is also a versatile problem that was used as the basis for several diverse constructions including public-key encryption schemes [Reg05, PW08], oblivious transfer protocols [PVW08] and, more pertinently, fully homomorphic encryption schemes [BV11, BGV12, GSW13]. To get a better understanding of the LWE problem, we will first look at some related concepts in coding theory before transitioning to their lattice-based variants.

2 Coding Theory

One of the more widely studied topics in coding theory is *error correcting codes* where we are interested in finding a (usually linear) message-encoding function f that transforms a message $\vec{x} \in \{0,1\}^n$ into a codeword $f(\vec{x}) \in \{0,1\}^m$ where $m \geq n$ such that, even after flipping a "large" fraction of the bits in $f(\vec{x})$, we are still able to (efficiently) decode it and get back $\vec{x}$.

2.1 Linear Codes Background

Throughout this section, we focus on working in GF(2). We first recall some general definitions from coding theory:

Definition 1 (Hamming distance)**.** *For any two vectors $\vec{x}, \vec{y} \in \{0,1\}^m$, their Hamming distance $hd(\vec{x}, \vec{y})$ is the number of positions i where $\vec{x}_i \neq \vec{y}_i$*

Definition 2 (Hamming weight)**.** *For any vector $\vec{x} \in \{0,1\}^m$, its Hamming weight $hw(\vec{x})$ is the number of positions i where $\vec{x}_i = 1$*

Define $\mathbf{A} \in \mathbb{F}_2^{m \times n}$ to be some binary matrix. We represent the linear encoding operation as a function $f_{\mathbf{A}} : \mathbb{F}_2^n \to \mathbb{F}_2^m$ that accepts a message $\vec{x} \in \{0,1\}^n$ and outputs the corresponding codeword $f_{\mathbf{A}}(\vec{x}) = \mathbf{A}\vec{x} = \vec{y} \in \{0,1\}^m$. A *linear code* C is simply the set of codewords generated by matrix $\mathbf{A}$ which form an n-dimensional subspace in $\mathbb{F}_2^m$.

If the codeword $\vec{y}$ is error-free (no bits were flipped) and the matrix $\mathbf{A}$ is full column rank (or, if $n = m$, it is invertible/non-singular) then we can uniquely and efficiently recover $\vec{x}$ using Gaussian elimination. However, this is not always the case when we add noise to $\vec{y}$. Consider, for example, the effect of flipping half the bits in $\vec{y}$. This would result in a string $\vec{z}$ that we cannot decode because if we have another codeword $\vec{y}'$ such that $hd(\vec{y}, \vec{y}') = m$, then we cannot determine whether $\vec{z}$ is actually $\vec{y}$ or $\vec{y}'$ without the noise.

Thus, it is important to determine the *error tolerance* of a code, which is given by the maximum number of bits that we can flip in $\vec{y}$ while preserving the ability to decode it and get back the correct $\vec{x}$. In addition, we should also take into consideration the length of the codewords m, as we generally prefer to have m to be on the same order of n (no excessive redundancy) while still providing comparative error tolerance. Fortunately, the following theorem whose proof we omit shows that this is possible.

Theorem 3. *There exists a linear coding function $f_{\mathbf{A}} : \mathbb{F}^n \to \mathbb{F}^m$ with $m = O(n)$ and error tolerance $\Omega(m)$*

For now, we can set $m = 10n$ and the error tolerance to be $m/10$. We say that a linear coding function $f_{\mathbf{A}}$ is *decodable* if for every pair of messages $\vec{x} \neq \vec{x}'$, their Hamming distance is $hd(\mathbf{A}\vec{x}, \mathbf{A}\vec{x}') \geq m/5$. This guarantees that any codeword $\mathbf{A}\vec{x}$ can be decoded into some unique $\vec{x}$. Note that while the encoding procedure is efficient (simple matrix multiplication), the decoding procedure might not necessarily be so, even though, information-theoretically, there exists a way to correctly decode a codeword. We can efficiently generate a decodable coding function using the following theorem.

Theorem 4. *A randomly chosen $\mathbf{A} \in \mathbb{F}_2^{m \times n}$ can be used to get a decodable linear coding function $f_{\mathbf{A}}$ with high probability.*

Lastly, we define the *code distance* d of a linear code $f_{\mathbf{A}}$ which is the minimum Hamming distance between any two codewords. It can be easily shown that d is also equal to the codeword with the minimum Hamming weight. That is:

$$d = \min_{\vec{x} \neq \vec{x}'} hd(\mathbf{A}\vec{x}, \mathbf{A}\vec{x}') = \min_{\vec{x} \neq 0} hw(\mathbf{A}\vec{x}) \tag{1}$$

Proof. A good simple exercise! □

2.2 Problem Definitions

Here we discuss the problem definitions related to coding theory that are conjectured to be computationally hard. While it is easy to find a decodable linear code (see Theorem 4), we do not yet know of any efficient procedure to test that it is decodable. That is, finding the code distance d is computationally hard. By Equation (1), finding the minimum weight among all (non-zero) codewords is also equivalently hard.

Definition 5 (Shortest vector problem)**.** *Given matrix* $\mathbf{A}$ *for some linear code* $f_{\mathbf{A}}$ *with code distance* d*, find* $\vec{y} = \mathbf{A}\vec{x}$ *such that* $hw(\vec{y}) = d$*. That is, find the shortest non-zero vector in* $\{\mathbf{A}\vec{x} \mid \vec{x} \in \{0,1\}^n\} \subseteq \{0,1\}^m$

In contrast to the above problem of testing whether a linear code is decodable or not, the following problem is based on the hardness of decoding a noisy codeword from a random linear code.

Definition 6 (Decoding random linear codes)**.** *Given some random matrix* $\mathbf{A} \in \mathbb{F}_2^{m \times n}$ *for some linear code* $f_{\mathbf{A}}$ *and a perturbed codeword* $\vec{y} \leftarrow \mathbf{A}\vec{x} + \vec{e}$ *where* $\vec{x} \in \{0,1\}^n$ *is chosen uniformly at random and* $\vec{e} \in \{0,1\}^m$ *is a random error vector with weight at most* $m/10$*, find* $\vec{x}$*.*

3 Lattice-based Cryptography

3.1 Lattices Background

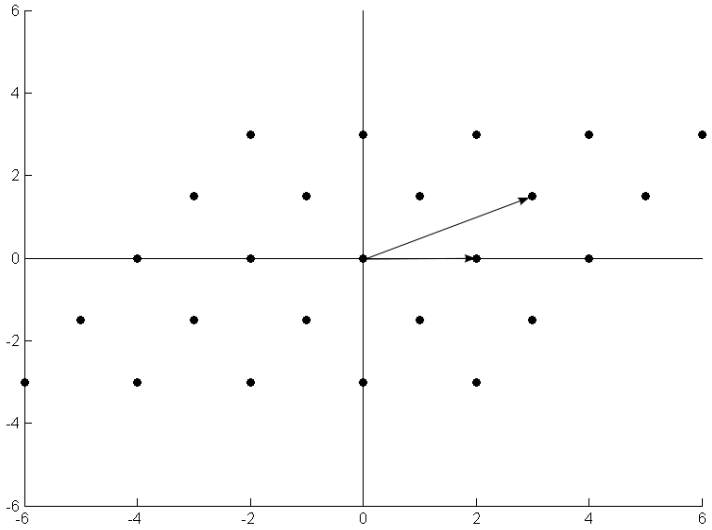

Figure 1: An example of a lattice in $\mathbb{R}^2$

An m-dimensional *lattice* $\mathcal{L}$ is a discrete additive subgroup of $\mathbb{R}^m$ and is represented as the set of all possible integer linear combinations of n linearly independent m-dimensional (column) vectors $\vec{a}_1, ..., \vec{a}_n$. We call the matrix $\mathbf{A} = [\vec{a}_1, ..., \vec{a}_n]$ the *basis* of lattice $\mathcal{L}$ and we can define it more formally as follows:

$$\mathcal{L}(\mathbf{A}) = \mathcal{L}(\vec{a}_1, ..., \vec{a}_n) = \left\{ \sum_{i=1}^{n} x_i \vec{a}_i : x_i \in \mathbb{Z} \right\} = \{\mathbf{A}\vec{x} : \vec{x} \in \mathbb{Z}^n\}$$

An example of 2-dimensional lattice is shown in Figure 1 where the explicitly drawn arrows represent a basis for the lattice from which we can generate all other points. For any vector $\vec{y} \in \mathcal{L}$, we usually refer to the length of a vector $||\vec{y}||$ using the Euclidean norm however other norms are also applicable.

3.2 Problem Definitions

We list here the definitions of the relevant lattice-based problem that are conjectured to be computationally hard. These problems are the lattice analogue of the ones described in Section 2.2.

Definition 7 (Shortest vector problem in lattices)**.** *Given a basis* $\mathbf{A}$ *for lattice* $\mathcal{L}$*, find the shortest non-zero vector in* $\mathcal{L}(\mathbf{A})$*.*

The shortest vector problem (SVP) was one of the first lattice problems to have been used as the basis for a public-key cryptosystem [AD97]. The exact version of SVP was proven to be NP-hard [Ajt98] and the approximate version was shown to be computationally hard even for constant approximation factors [Kho05].

Definition 8 (Learning with Errors)**.** *For uniformly random* $\vec{x} \in \mathbb{F}^n$ *define* $O_{\vec{x}}$ *to be an LWE oracle that upon request returns a tuple* $(\vec{a}, \langle \vec{a}, \vec{x} \rangle + \vec{e})$ *where* $\vec{a} \in \mathbb{F}^n$ *is a vector chosen uniformly at random and* $e \in \mathbb{F}$ *is an error term chosen from some suitable distribution. Given* m *samples from* $O_{\vec{x}}$*, where* m *can be arbitrary, the LWE problem is to find* $\vec{x}$*.*

Unlike the two problems in Section 2.2, the two problems showcased here are known to be related. Specifically, Regev [Reg05] shows a reduction from approximate-SVP to LWE. Usually, the field we use on which to perform these computations is $\mathbb{Z}_p^*$ and the error vector is drawn from some suitable Gaussian distribution.

References

[AD97] Miklós Ajtai and Cynthia Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC)*, pages 284–293. ACM Press, 1997. See also ECCC TR96-065.

[Ajt98] Miklós Ajtai. The shortest vector problem in l_2 is np-hard for randomized reductions (extended abstract). In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98, pages 10–19, New York, NY, USA, 1998. ACM.

[BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 309–325, New York, NY, USA, 2012. ACM.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. In Rafail Ostrovsky, editor, *FOCS*, pages 97–106. IEEE, 2011.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO*, pages 75–92. Springer, 2013.

[Kho05] Subhash Khot. Hardness of approximating the shortest vector problem in lattices. *J. ACM*, 52(5):789–808, 2005.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *Advances in Cryptology CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571. Springer Berlin Heidelberg, 2008.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 187–196. ACM, 2008.

[Reg05] Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 2005.