Lattice-based Cryptography

Daniele Micciancio[1]* and Oded Regev[2]†

[1] CSE Department, University of California, San Diego.
[2] School of Computer Science, Tel-Aviv University.

1 Introduction

In this chapter we describe some of the recent progress in *lattice-based cryptography.* Lattice-based cryptographic constructions hold a great promise for post-quantum cryptography, as they enjoy very strong security proofs based on worst-case hardness, relatively efficient implementations, as well as great simplicity. In addition, lattice-based cryptography is believed to be secure against quantum computers. Our focus here will be mainly on the practical aspects of lattice-based cryptography and less on the methods used to establish their security. For other surveys on the topic of lattice-based cryptography, see, e.g., [36, 52, 60, 71] and the lecture notes [51, 67]. The survey by Nguyen and Stern [60] also describes some applications of lattices in cryptanalysis, an important topic that we do not discuss here. Another useful resource is the book by Micciancio and Goldwasser [49], which also contains a wealth of information on the computational complexity aspects of lattice problems.

So what is a lattice? A lattice is a set of points in n-dimensional space with a periodic structure, such as the one illustrated in Figure 1. More formally, given n-linearly independent vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n \in \mathbb{R}^n$, the lattice generated by them is the set of vectors

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

The vectors $\mathbf{b}_1, \ldots, \mathbf{b}_n$ are known as a *basis* of the lattice.

The way lattices can be used in cryptography is by no means obvious, and was discovered in a breakthrough paper by Ajtai [7]. His result has by now

* Supported in part by NSF Grant CCF 0634909.

† Supported by the Binational Science Foundation, by the Israel Science Foundation, by the European Commission under the Integrated Project QAP funded by the IST directorate as Contract Number 015848, and by a European Research Council (ERC) Starting Grant.

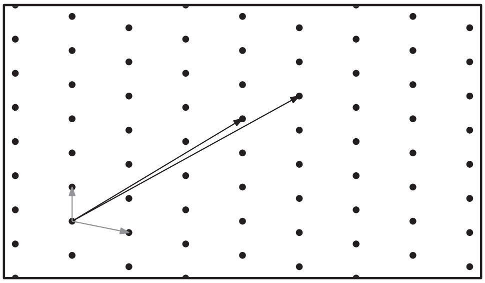

Fig. 1. A two-dimensional lattice and two possible bases.

developed into a whole area of research whose main focus is on expanding the scope of lattice-based cryptography and on creating more practical lattice-based cryptosystems. Before discussing this area of research in more detail, let us first describe the computational problems involving lattices, whose presumed hardness lies at the heart of lattice-based cryptography.

1.1 Lattice problems and algorithms

Lattice-based cryptographic constructions are based on the presumed hardness of lattice problems, the most basic of which is the *shortest vector problem* (SVP). Here, we are given as input a lattice represented by an arbitrary basis, and our goal is to output the shortest nonzero vector in it. In fact, one typically considers the approximation variant of SVP where the goal is to output a lattice vector whose length is at most some approximation factor $\gamma(n)$ times the length of the shortest nonzero vector, where n is the dimension of the lattice. A more precise definition of SVP and several other lattice problems appears in Section 2.

The most well known and widely studied algorithm for lattice problems is the *LLL algorithm*, developed in 1982 by Lenstra, Lenstra, and Lovász [39]. This is a polynomial time algorithm for SVP (and for most other basic lattice problems) that achieves an approximation factor of $2^{O(n)}$ where n is the dimension of the lattice. As bad as this might seem, the LLL algorithm is surprisingly useful, with applications ranging from factoring polynomials over the rational numbers [39], to integer programming [31], as well as many applications in cryptanalysis (such as attacks on knapsack-based cryptosystems and special cases of RSA).

In 1987, Schnorr presented an extension of the LLL algorithm leading to somewhat better approximation factors [73]. The main idea in Schnorr's algorithm is to replace the core of the LLL algorithm, which involves 2×2 blocks, with blocks of larger size. Increasing the block size improves the approximation factor (i.e., leads to shorter vectors) at the price of an increased running time. Schnorr's algorithm (e.g., as implemented in Shoup's NTL package [75])