Lattice Reduction with Applications to Cryptography

Boxiang Fu with supervisor **Peter Forrester**

Vacation Research Project

Introduction

Lattice reduction is to find a nearly orthogonal short basis for a given input lattice. Such a problem is usually very hard to solve exactly on both classical and quantum computers. We can use this fact to build a (presumably) secure post-quantum cryptosystem.

Lattices

Given a set of *n* linearly independent vectors $B = \{\mathbf{b_1}, ..., \mathbf{b_n}\}$ in $\mathbb{R}^n$, a lattice is generated by the integral linear combination of B:

$$L = \{\sum_{i=1}^{n} \lambda_i \mathbf{b_i} | \lambda_i \in \mathbb{Z}\}$$

We note that the basis B is not unique. For $n \geq 2$, every lattice has infinitely many bases. For a matrix $\boldsymbol{X}$ whose rows contain the basis of the lattice L, any of the below unimodular row operations can be applied to get another basis of the same lattice:

- Multiply any row by -1
- Interchange any two rows
- Add an integral multiple of any row to another other row

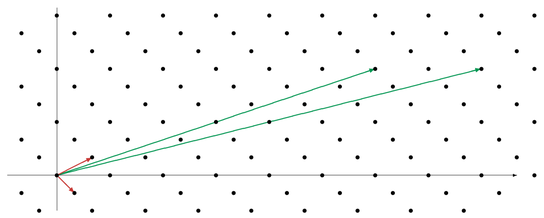

Fig. 1: A 2 Dimensional Lattice with a "Good" (Red) and "Bad" (Green) Bases [2]

In general, it is hard to take a basis of long "bad" vectors and reduce it to a basis of short "good" vectors. This fundamental problem is called **Lattice Basis Reduction**. Two hard problems arising from this is to find a non-zero vector of minimal Euclidean length (**Shortest Vector Problem**) and given a vector $\mathbf{t} \in \mathbb{R}^n$ not in L, find a vector in L closest to $\mathbf{t}$ (**Closest Vector Problem**).

Lattice Reduction Algorithms

In 2-dimensional lattices $L = \{a\mathbf{x} + b\mathbf{y} | a, b \in \mathbb{Z}\}$ the Lagrange-Gauss algorithm terminates with a minimal basis for the lattice. Its pseudo-code is presented next [1].

Lattice Reduction Algorithms

Algorithm 1 Lagrange-Gauss Algorithm

Require: A basis $\mathbf{x}$, $\mathbf{y}$ of a lattice in $\mathbb{R}^2$ such that $|\mathbf{x}| \leq |\mathbf{y}|$

Set $\mathbf{v_1} \leftarrow \mathbf{x}$ and $\mathbf{v_2} \leftarrow \mathbf{y}$. Set finished $\leftarrow$ **false**
while not finished **do**
  Set $m \leftarrow \lceil \frac{\mathbf{v_2} \cdot \mathbf{v_1}}{\mathbf{v_1} \cdot \mathbf{v_1}} \rfloor$
  Set $\mathbf{v_2} \leftarrow \mathbf{v_2} - m\mathbf{v_1}$
  if $|\mathbf{v_1}| \leq |\mathbf{v_2}|$ **then**
    Set finished $\leftarrow$ **true**
  else
    Interchange $\mathbf{v_1}$ and $\mathbf{v_2}$
  end if
end while
return $\mathbf{v_1}$ and $\mathbf{v_2}$ as a minimal basis of the input lattice

However, in higher dimensions, there generally does not exist an exact algorithm for lattice reduction. The best currently known method is the LLL (Lenstra–Lenstra–Lovász) algorithm which finds a moderately short bases in polynomial time. An implementation can be found in [3].

Applications (Cryptosystems)

Shor's algorithm meant that current public-key cryptosystems such as RSA and discrete logarithm problems are rendered insecure once large-scale quantum computers are readily available. The (postulated) hardness of lattice problems make lattice-based post-quantum cryptosystems look promising. We look at one such cryptosystem below:

The GGH (Goldreich-Goldwasser-Halevi) Public Key Cryptosystem [2]
Main idea: Public key is a "bad" (e.g. Hermite normal form) basis of some lattice, private key is its "good" basis. Sender uses public key to map to a lattice point and adds a small error term in the neighbourhood of the lattice point. The receiver then solves the Closest Vector Problem (CVP) using the "good" basis trapdoor, which is hard to solve using the "bad" basis.
Key Creation:
Private Key $= \{\mathbf{v_1}, ..., \mathbf{v_n}\}$ a "good" basis of the lattice L
Public Key $= \{\mathbf{w_1}, ..., \mathbf{w_n}\}$ a "bad" basis of the lattice L
Encryption: Write out plaintext $\mathbf{m}$ as a binary vector. Ciphertext is $\mathbf{e} = m_1\mathbf{w_1} + m_2\mathbf{w_2} + ... + m_n\mathbf{w_n} + \mathbf{r}$. Where $\mathbf{r}$ is a small error term.
Decryption: Find lattice point $\mathbf{u}$ closest to $\mathbf{e}$ by solving CVP. This can be done using the "good" basis trapdoor by writing $\mathbf{e} = \mu_1\mathbf{v_1} + \mu_2\mathbf{v_2} + ... + \mu_n\mathbf{v_n}$, $\mu_1, ..., \mu_n \in \mathbb{R}$. We then round $\mu_1, ..., \mu_n$ to the nearest integer to give $\mathbf{u} = \lfloor \mu_1 \rceil \mathbf{v_1} + \lfloor \mu_2 \rceil \mathbf{v_2} + ... + \lfloor \mu_n \rceil \mathbf{v_n}$.
For appropriate (small) $\mathbf{r}$, $\mathbf{u} = m_1\mathbf{w_1} + m_2\mathbf{w_2} + ... + m_n\mathbf{w_n}$ so we recover $\mathbf{m}$.

Applications (Attacks)

Here we look at an attack on knapsack cryptosystems using lattice reduction techniques:

The Merkle-Hellman Knapsack Cryptosystem [2]
Main idea: The ciphertext is $t = x_1a_1 + ... + x_na_n$. This can be mapped to a very short vector in the lattice L with length at most $\sqrt{n}$. We can use LLL to find lattice vectors that meet this criteria.
Key Creation:
Private Key: Superincreasing sequence of $b_1, b_2, ..., b_n$ with $b_1 \approx 2^n$ and $b_n \approx 2^{2n}$. Positive integers m and w satisfying $m > \sum_{i=1}^{n} b_i$ and $gcd(m, w) = 1$. And a permutation π of $\{1, ..., n\}$.
Public Key: Set $\{a_1, ..., a_n\}$ with $a_i \equiv wb_{\pi(i)} \pmod{m}$
Encryption: Write out plaintext $\mathbf{x} = (x_1, ..., x_n) \in \{0, 1\}^n$ as a binary vector. Ciphertext is $t = x_1a_1 + ... + x_na_n$.
Attack: If $\mathbf{x} = (x_1, ..., x_n) \in \{0, 1\}^n$ solves $t = x_1a_1 + ... + x_na_n$, then $\mathbf{v} = (x_1, ..., x_n, 0) \in L$, where

$$L = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 & a_1 \\ 0 & 1 & 0 & \cdots & 0 & a_2 \\ & & & \ddots & & \\ 0 & 0 & 0 & \cdots & 1 & a_n \\ 0 & 0 & 0 & \cdots & 0 & -t \end{bmatrix}$$

Since $\mathbf{v}$ has a short length of at most $\sqrt{n}$, applying LLL on L to find short basis vectors in $\{0, 1\}^n$ finds $\mathbf{v}$ with high probability. Note that the matrix L consists of only public information.

Acknowledgements

I would like to sincerely thank my supervisor Peter Forrester for his guidance and encouragement over the Vacation Scholar period. I would also like to express my gratitude to the School of Mathematics and Statistics for providing me with this opportunity.

References

[1] M. Bremner. *Lattice Basis Reduction: An Introduction to the LLL Algorithm and Its Applications*. Taylor Francis Group, 2011.

[2] J. Silverman. "An Introduction to the Theory of Lattices and Applications to Cryptography". In: *Computational Number Theory and Applications to Cryptography*. 2006.

[3] The FPLLL development team. "fplll, a lattice reduction library, Version: 5.4.1". Available at `https://github.com/fplll/fplll`. 2021. URL: `https://github.com/fplll/fplll`.