Complementing LLL Lattice Reduction Algorithm with BIROT to find Shortest Vectors in NTRU Cryptosystem

Tran Nguyen Bao Long, Liu Renhang

Supervisor: Wong Wei Pin

The NTRUEncrypt

- A form of **Public Key Cryptography** *(Figure 1)*:
  - **Key generator**: outputs a public key and a private key with given security parameters (*N, p, q, d*)
  - **Encryption** algorithm: takes a public key and a message and output a ciphertext
  - **Decryption** algorithm: takes a private key and a ciphertext, and either output the same message (if successful)
- First cryptographic construction using **Quotients of Polynomial Rings** which is most usefully interpreted in terms of **algebraically structured lattices**
  - Integer ***N*** and two moduli ***p*** and ***q*** gives **convolution polynomial rings**:

$$R = \frac{\mathbb{Z}[x]}{(x^N - 1)}, \qquad R_p = \frac{(\mathbb{Z}/p\mathbb{Z})[x]}{(x^N - 1)}, \qquad R_q = \frac{(\mathbb{Z}/q\mathbb{Z})[x]}{(x^N - 1)},$$

  - A polynomial *a(x)* ∈ *R* can be naturally mapped to R_p and R_q by reducing its coefficients modulo *p* or *q*. In other directions, we use center-lifts to move elements from R_p or R_q to *R*.
  - Polynomials in $\mathcal{T}(d_1, d_2)$ are called **ternary polynomials**:

$$\mathcal{T}(d_1, d_2) = \left\{ a(x) \in R : \begin{array}{l} a(x) \text{ has } d_1 \text{ coefficients equal to } 1, \\ a(x) \text{ has } d_2 \text{ coefficients equal to } -1, \\ a(x) \text{ has all other coefficients equal to } 0 \end{array} \right\}$$

Public Parameter Creation	
Choose *(N, p, q, d)* s.t *N* and *p* are primes, and *gcd(N, q)* = *gcd(p, q)* = *1*, and *q* > *(6d + 1)p*.	
Alice	**Bob**
Key creation	
Choose private $f \in \mathcal{T}(d+1, d)$ that is invertible in both R_p and R_q Compute inverse of f in R_p : F_p Compute inverse of f in R_q : F_q Publish the public key: $h = F_q \star g$	
Encryption	
	Choose plaintext $m \in R_p$ Choose a random $r \in \mathcal{T}(d, d)$ Use Alice's public key h to compute cipher text $e \equiv pr \star h + m$ (mod q)
Decryption	
Compute $f \star e \equiv pg \star r + f \star m$ (mod q) Center-lift to $a \in R$ and compute the original message $m \equiv F_p \star a$ (mod p)	

Figure 1: NTRU Public Key Cryptosystem summary

SVPs and NTRU Lattices

- Basic Lattices Definitions *(Figure 2)*:
  - The **lattice *L*** generated by *n* **linearly independent vectors** $v_1, \ldots, v_n \in \mathbb{R}^m$ is the set of linear combinations of these vectors with integer coefficients:

$$L = \{a_1 v_1 + a_2 v_2 + \cdots + a_n v_n : a_1, a_2, \ldots, a_n \in \mathbb{Z}\}.$$

  - **Integral lattice** is one whose vectors have integer coordinates
  - The **basis** for L is not **unique** and basis whose vectors are more **orthogonal** will have a **Hadamard ratio** closer to 1:

$$0 < \mathcal{H}(\mathcal{B}) = \left(\frac{\det L}{\|v_1\| \, \|v_2\| \cdots \|v_n\|} \right)^{1/n} \leq 1$$

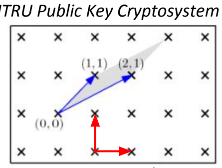

Figure 2: Integral Lattice (by Oded Regev)

- Shortest Vector Problem (SVP):
  - Find a shortest nonzero vector in a lattice *L* (min L2 norm)
  - The **Gaussian expected shortest length** for *L* of dimension *n*:

$$\|v_{\text{shortest}}\| \approx \sigma(L) = \sqrt{\frac{n}{2\pi e}} (\det L)^{1/n}$$

  - Becomes more computationally expensive as the dimension of lattice grows: **NP-hard**
  - The shortest vector in a basis with $\mathcal{H}(\mathcal{B}) \approx 1$ (a "good" basis) will be the solution for SVP.
- The NTRU Lattice:
  - Reformat public key $h(x) = h_0 + h_1 x + \cdots + h_{N-1} x^{N-1}$ into a **2N-dimensional lattice** spanned by the rows of

$$M_h^{\text{NTRU}} = \left(\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & h_0 & h_1 & \cdots & h_{N-1} \\ 0 & 1 & \cdots & 0 & h_{N-1} & h_0 & \cdots & h_{N-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & h_1 & h_2 & \cdots & h_0 \\ \hline 0 & 0 & \cdots & 0 & q & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & q & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & q \end{array} \right)$$

  - We have $f(x) \star h(x) \equiv g(x) \pmod{q}$ so we can find $u(x) \in R$ such that $f(x) \star h(x) = g(x) + q u(x)$ then the **private key vector** (f, g) will be inside the lattice as it can be written as a linear combination of the rows of M_h^{NTRU} :

$$(f, -u) M_h^{\text{NTRU}} = (f, g)$$

  - (f, g) will be one of the shortest vectors in the lattice.

Lattice - Based Cryptography

> Conjectured security against quantum attack

> Algorithmic simplicity, efficiency and parallelism

> Strong security guarantees from worst-case hardness

> Constructions of versatile and powerful cryptographic objects

Figure 3: Attractive Features of LBC

LLL Lattice Reduction algorithm

- Turning any random ("bad") basis into a "better" basis:
  - Algorithm summarized in *Figure 4*.
  - Vectors as short as possible: start with shortest vector, then small length increment until the last vector in the basis.
  - Basis vectors have **Hadamard ratio** close to 1 (**orthogonal**).
- Taking a **Gram-Schmidt orthogonal basis** $\mathcal{B}^* = \{v_1^*, v_2^*, \ldots, v_n^*\}$ as reference., the basis $\mathcal{B} = \{v_1, v_2, \ldots, v_n\}$ of lattice L is said to be **LLL-reduced** if it satisfies:
  - **Size Condition** $\quad |\mu_{i,j}| = \dfrac{|v_i \cdot v_j^*|}{\|v_j^*\|^2} \leq \dfrac{1}{2} \quad$ for all $1 \leq j < i \leq n$.
  - **Lovasz Condition** $\quad \|v_i^*\|^2 \geq \left(\dfrac{3}{4} - \mu_{i,i-1}^2\right) \|v_{i-1}^*\|^2 \quad$ for all $1 < i \leq n$.

[1]	Input a basis $\{v_1, \ldots, v_n\}$ for a lattice L
[2]	Set $k = 2$
[3]	Set $v_1^* = v_1$
[4]	Loop while $k \leq n$
[5]	Loop down $j = k-1, k-2, \ldots, 2, 1$
[6]	Set $v_k = v_k - \lfloor u_{k,j} \rceil v_j$ **[Size Reduction]**
[7]	End j loop
[8]	If $\|v_k^*\|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) \|v_{k-1}^*\|^2$ **[Lovasz Condition]**
[9]	Set $k = k+1$
[10]	Else
[11]	Swap v_{k-1} and v_k **[Swap Step]**
[12]	Set $k = max(k-1, 2)$
[13]	End If
[14]	End k Loop
[15]	Return LLL reduced basis $\{v_1, \ldots, v_n\}$

Figure 4: LLL lattice reduction algorithm

[1]	Input a basis $B = \{v_1, \ldots, v_n\}$
[2]	Input $m, n \in Z : 1 \leq m < n \leq 2N$
[3]	Set $a = det(B)$
[4]	Set $i = 1$
[5]	Loop while $i < n$
[6]	Set $B' = \{v_1, \ldots, v_{n-1}, birotate_i(v_m), v_{n+1}, \ldots v_{2N}\}$
[7]	If $det(B') = \pm a$ and $v_n \neq birotate_i(v_m)$:
[8]	Return B'
[9]	Return False

Figure 5: BIROTATION algorithm

BIROT and GAME algorithm

- **BIROTATION** algorithm *(Figure 5)*:
  - based on the **cyclic automorphisms** of the NTRU lattice
  - let ***v*** = $(v_1, v_2, \ldots, v_{2N})$ then ***v*** **birotated** by k position, *birotate*$_k$***(v)*** $= (v_{1+k \bmod N}, \ldots, v_{N+k \bmod N}, v_{N+(1+k \bmod N)}, \ldots, v_{N+(N+k \bmod N)})$
  - **Thesis**: For NTRU lattice L, ***v*** ∈ L i.i.f *birotate*$_k$***(v)*** ∈ L
  - replacing **basis vector** v_n with *birotate*$_k$(v_m) returns a **sublattice** L'

    ⇒ a **new basis** for L is created if the basis vectors of L' is **linearly independent** and they form the same span: |det(L')| = |det(L)|
  - We have **BIROT reduced basis** (a hit) if $||v_m|| <= ||v_n||$, v_n != *birotate*$_k$(v_m) and |det(L')| = |det(L)|
- **GAME** algorithm (combining **LLL** and **BIROT**)
  - **LLL** alone produces the same output when applied once or multiple consecutive times with same parameter

    ⇒ change **LLL**'s output lattice with **BIROT** so running **LLL** again will result in further reduction
  - a **round** of GAME starts with **LLL** followed by trying **BIROT** until there is a hit (*n loop down in step 10*) as seen in *Figure 6*:

[1]	Input a basis $B = \{v_1, \ldots, v_{2N}\}$	[9]	Set m = 1
[2]	Input r = max rounds of BIROT	[10]	Loop down n = 2N, ..., m + 1:
[3]	Loop for i from 1 to r:	[11]	Set L2 = BIROT(L, m, n) **[BIROT]**
[4]	Set L = LLL(L) **[LLL]**	[12]	If L2 is not False: # **a hit**
[5]	Sort L by L2 norm	[13]	Set L = L2
[6]	If $birotate_i(v_1) == v_1$:	[14]	Break loop n
[7]	Set m = 2	[15]	If L == const: # **no hit**
[8]	Else:	[16]	Break loop i

Figure 6: GAME algorithm

Experimentations and Results

- For all experiments (*Figure 7-9*), we used:
  - **Independent variables**: *N* as safe primes (11, 23, 47)
  - **Constants**: *max rounds of GAME*: 10; *number of trials*: 20

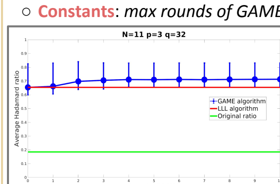

Figure 7: (N = 11) Both LLL and GAME (all 10 hits) performed well. GAME improved Hadamard ratio up to 9.25%. With the best basis, the shortest vector is 85.7% the length of (f, g).

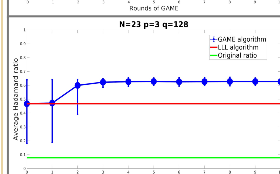

Figure 8: (N = 23) GAME (all 10 hits) shows significant improvements of 34.12% compared to LLL. With the best basis, the shortest vector is 81.1% the length of (f, g).

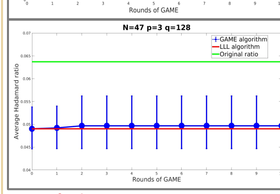

Figure 9: (N = 47) Both LLL and GAME (only 1 hit) can no longer produce a "better" basis. GAME performs 1.36% better than LLL. With the returned basis, the shortest vector is 1700% the length of (f, g).

- **Conclusion**:
  - In most cases, BIROT can complement LLL to improve the Hadamard ratio of NTRU bases.
  - For small N (<47), BIROT significantly improves the Hadamard ratio of NTRU bases, increasing the chances of breaking NTRU
  - For large N (>=47), LLL worsens the Hadamard ratio of NTRU bases more than the improvement made by BIROT, thus BIROT cannot complement LLL to break NTRU for large N.

Acknowledgement

This project is our experimentations with the ideas presented in:

1. D. Socek (2002). *Deterministic and Non-deterministic basis reduction techniques for NTRU lattices* (Master's thesis,, Florida Atlantic University).
2. J. Hoffstein, J. Pipher, J. Silverman (2014). *An Introduction to Mathematical Cryptography* (Second Edition)