Implementation of Hermite Normal Form in NTRU Matrix Formulation Algorithm

Khushboo Thakur[1]***, B.P. Tripathi***[2]***, and B.K. Sharma***[1]

[1]School of Studies in Mathematics, Pt. Ravishankar Shukla University,
Raipur(C.G.), India

[2]Department of Mathematics, Govt. N.P.G. College of Science,
Raipur, Chhattisgarh, 492010, India

ABSTRACT: In this paper, we have shown that applying Hermite Normal Form (HNF) for the matrix formulation algorithm in NTRU public key cryptosystem substantially Increases it efficiency as compare to other PKC, such as Nayak et al.

KEYWORDS: NTRU, Hermite Normal Form, Private Key, Public Key, Encryption, Decryption.

1 INTRODUCTION

Lattices were first studied by mathematician Joseph Louis Lagrange and Carl Friedrich Gauss. Later lattices have been used in public key cryptosystems by Ajtai Dwork (Ajtai and Dwork 1997), Goldreich Goldwasser Halevi (Goldreich et al. 1997) and NTRU (Hoffstein et al.1998)cryptosystem. Lattices have been used recently in computer algorithms and in cryptanalysis. In 1996, Mikls Ajtai showed in a seminal result the use of lattices as a cryptography primitive. NTRU the best among the other lattice based cryptosystems. The NTRU PKC Nayak et al. [7] was designed with invertible matrix. In this paper we use and introduce NTRU cryptosystem for Hermite Normal Form. We also find Key generation, Encryption and Decryption by upper triangular matrix. This cryptosystem is a new design of Matrix formulation algorithm.

NTRU allegedly stands for “Nth Degree Truncated Polynomial Ring Units".NTRU is a public key cryptosystem presented by J. Hoff stein, J. pipher and J. Silverman [4]. The first version of the NTRU encryption system was presented at the crypto 96 conference [4]. The computational basis of the NTRU lies in polynomial algebra and it is a relatively new cryptosystem. NTRU is based on lattice-based cryptography it has different cryptographic properties from RSA and ECC [3]. The strength of cryptographic NTRU performs valuable private key operations much faster in comparison to RSA. Polynomial algebra is the basic building block of the NTRU Encryption system. The truncated polynomials given in J. Silverman [6], P.Prapoorna [5] in the ring $\mathrm{R} = \mathrm{Z[X]/(X^N -1)}$ are basic objects and the reduction of polynomials with respect to relatively prime modulo i.e., p and q are the basic tools. NTRU polynomials a(x) are frequently reduced to modulo p and q, the small and large modulo. The large modulus q is an integer, so reduction of $\mathrm{a(x)} = \mathrm{a}_0 + a_1x + a_2x^2 + -------a_{N-1}x^{N-1}$ (mod q) means just reduction of each a_i modulo q. The small modulo p can also be an integer. It is required that p and q are relatively prime i.e. gcd (p, q) = 1. The main objects in such systems are use of small polynomials; i.e. polynomials with small coefficients. The public key h is defined by an equation f*h = p*g (mod q), where f and g are small polynomials. The polynomial f should always have inverses modulo p and q [1], f*fp = I (mod p) and f*fq = I (mod q). Moreover, the parameters N, p and q are also public, and can be used as common domain parameters for all users. Polynomials f and g are private to the key owner. The polynomial g is needed only in key generation. Firstly Bob chooses two small polynomials f and g in the ring of truncated polynomials and keeps f and g private. He then computes inverse of f (mod p) [fp] and inverse of f (mod q) [fq], where p and q are relatively prime to each other. He then computes h = p*fq*g (mod q), which becomes the public key for Alice and the pair of polynomials f and fp forms his private key pair. The message is also represented in the form of a truncated polynomial. Let it be m. The sender Alice encrypts using the public key i.e. h as e = h*r + m (mod q), where r is a random polynomial basically used to obscure the message. This encrypted message may be sent in a public channel. Alice decrypts the encrypted message using his private key pair by performing the following operations:

a = f * e (mod q)
b = a (mod p)
c = fp* b (mod p) where c is the original message:
c = m mod p.

Recently, Nayak et al. [7] have proposed taking invertible or non singular matrix in NTRU cryptosystem [4]. They have given a PKC by method, which is suitable to send in the key generation phase of large message in the form of matrices. Now in this paper, we consider Hermite Normal Form during key generation replacing the invertible matrix of Nayak et al. [7] design. In our opinion Hermite Normal Form makes the PKC more efficient as compare to invertible matrix because HNF is always upper triangular matrix hence easily reducible. This makes the PKC more efficient and more secure as well as time HNF always provides non commutative matrix where as invertible matrix can be commutative also.

2 PRELIMINARIES

In this section, we briefly describe the basic definitions of Hermite Normal Form.

Hermite Normal Form

For any n × m integer matrix A the Hermite normal form (HNF) of A is the unique matrix $\mathrm{H} = (h_{i,j})$ such that there is a unimodular n×n matrix U with UA = H, and such that H satisfies the following two conditions:

a) there exist a sequence of integers $j_1 \prec .... \prec j_n$ such that for all 0 ≤ i ≤ n we have $h_{i,j} = 0$ for all $j \prec j_i$ (row echelon structure),
b) for $0 \le k. < i \le n$ we have $0 \le h_{k,j_i} \prec h_{i,j_i}$ (the pivot element is the greatest along its column and the coe_cients are nonnegative).

Thus the Hermite normal form is a generalization over Z of the reduced row echelon form of a matrix over Q. Just as computation of echelon forms is a building block for many algorithms for computing with vector spaces, Hermite normal form is a building block for algorithms for computing with modules over Z (see, e.g.,[Coh93, Chapter 2]). Hermite Normal Form (HNF) matrices are a standard form of integer matrices used in application such as lattice based cryptography and integer programming.

3 BRIEF REVIEW OF NAYAK ET AL. [7] MATRIX FORMULATION FOR NTRU CRYPTOSYSTEM

3.1 KEY GENERATION

Key is Bob creates a public/private key pair. He first randomly chooses two matrices X and Y , where matrix X should be an invertible matrix (modulo p). Bob keeps the matrices X and Y private, since any one who knows either one of them will be able to decrypt messages sent to Bob. Bob's next step is to compute the inverse of X modulo q and the inverse of X modulo p. Thus he computes matrix Xq and Xp which satisfies X *Xq = I (modulo q) and X *Xp = I (modulo p).

(Bob can ensure the existence of inverse of matrix X by checking X is non-singular and X is invertible mod p (i.e.[det[X]](mod p ≠0).Otherwise he needs to go back and choose another matrix X). Now Bob`s computes the product H = p* Xq*Y (modulo q). Bob's private key is the pair of matrices X and Xp and his public the matrix H.

3.2 ENCRYPTION

Alice wants to send a message to Bob using Bob's public key H. She first puts her message in the form of a binary matrix M, (which is a matrix of same order as X and Y) whose elements are chosen modulo p. Next, she randomly chooses another matrix R of the same order as X. This is the "blinding value", which is used to obscure the message (similar to the way that the ElGamal algorithm uses a onetime random value when encrypting).

To send message M, Alice chooses a random matrix R (which is of same order as matrix X), and Bob's public key H to compute the matrix.

E = R*H + M (modulo q).

The matrix E is the encrypted message which Alice sends to

3.3 DECRYPTION

Now Bob has received Alice's encrypted message E and he decrypt it. He begins by using his private matrix X to compute the matrix. A = X*E (modulo q).

Bob next computes the matrix B = A (modulo p).

That is, he reduces each of the coefficients of A (modulo p). Finally Bob uses his other private matrix *Xp* to compute C = *Xp**B (modulo p). The matrix C will be Alice's original message M.

4 PROPOSED ALGORITHM

Bob creates a public/private key pair. He first randomly chooses two matrices f and g, where matrix f must have the following property.

1. f is upper triangular matrix.
2. f is unique matrix.
3. f is non singular matrix.
4. f is upper triangular matrix mod p.
5. f is upper triangular matrix mod q.

Now in this section, we define that f is an upper triangular matrix mod p and mod q Because, during the key generation process, we throw f away if the inverse does not exist. In commercial application, we use an alternative way of choosing f. We take

f= XU,

Where X is non singular matrix or invertible matrix and U is unimodular matrix. unimodular matrix is a square integer matrix with determinant equal to ±1, so that U^{-1} is also an integer matrix.

The proposed cryptosystem is divided into three parts: Key generation, Encryption, and Decryption.

4.1 KEY GENERATION

Step1: Bob`s randomly chooses two matrices f and g, where matrix f should be an upper triangular matrix (mod p) that is determinant of f should be 1.Here the matrices f and g is private.

Step2: Bob's next step is to compute the inverse of f mod q and the inverse of f mod p. Thus he computes matrix fq and fp which satisfies,

f * fq = I (mod q)

and

f * fp = I (mod p), where I is an identity matrix.

by checking, the existence of inverse of matrix f is non-singular and f mod p is upper triangular, because f = XU is always upper triangular matrix**.**

Step3: Now Bob`s computes the product H = p*(XU)q*g (mod q). Bob's private key is the pair of matrices f and fp and his public key is the matrix H.

4.2 ENCRYPTION

Step 1: Alice wants to send a message to Bob using Bob's public key H. She first put her message in the form of binary matrix M and its size is same as private key f and g.

Step 2: Create an encrypted message she chooses a Random matrix R of size f and g, this matrix is based on blinding value.

Step 3: Alice computes encrypted message using R and Bob's public key as follows.

E = R* H +M (mod q)

The matrix E is the encrypted message which Alice sends to Bob.

4.3 DECRYPTION

Step 2: Bob next computes the matrix B = A (mod p).

Step 3: Finally Bob uses his other private matrix fp to compute

C = fp* B (mod p).

The matrix C will be Alice's original message M.

4.4 CORRECTNESS OF ALGORITHM:

Theorem 1. The equation C = M (mod p) is correct.

C = fp *B
C = fp * A (mod p)
C = fp * f * E (mod p)
C = fp *f *(R *H +M) (mod p)
C = [fp* f *R *H + fp* f *M] (mod p)
C = fp *f *R *p * fq *g (mod p) + fp *f *M (mod p) [Since fp *f (mod p)=I
C = R*p* fq *g (mod p)+ M (mod p) [Since p (mod p)=0]
C = M (mod p)

5 EFFICIENCY

Table 1 defines parameter value for encryption and decryption procedure in our paper.

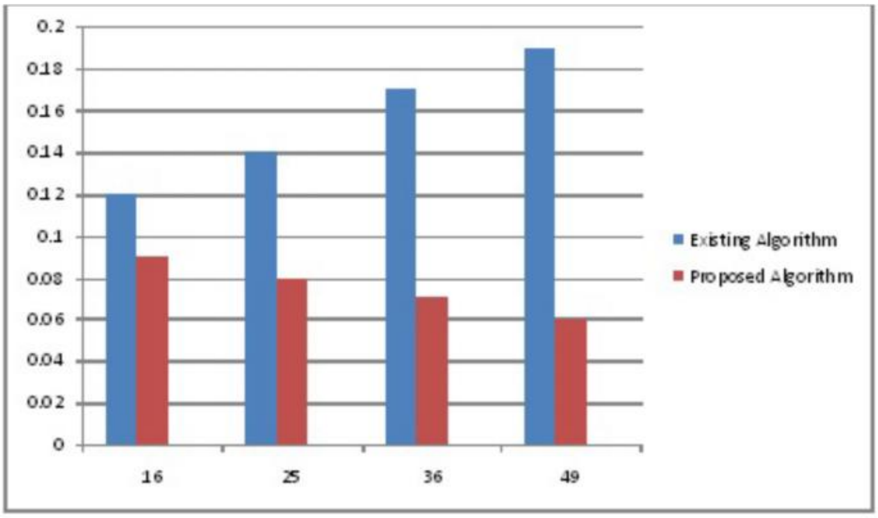
Table 2 shows the efficiency comparison of our proposed algorithm with the algorithm of Nayak et al.[5]. We have successfully performed the encryption and decryption operations for message. The codes of the algorithm given by Nayak et al. [7] and proposed by us have been computed and run out on Mathematica 7.0 [8]. Also we have estimated time for different size of degree of polynomials for p =3, q = 256.

Table 1. parameter values for encryption and decryption

Degree of Polynomial	q	p
36	256	3

Table 2. Comparison of Time

Degree of Polynomial	Existing Algorithm	Proposed Algorithm
16	0.12 Sec	0.09 Sec
25	0.14 Sec	0.08 Sec
36	0.17 Sec	0.07 Sec
49	0.19 Sec	0.06 Sec



6 Example

6.1 Key generation Example

Let F = 6×6 matrix, g = 6×6 matrix. Let parameters p = 3, q = 256 and N = 36

Let: $X = \begin{bmatrix} 1 & -1 & 0 & -1 & 1 & 1 \\ -1 & 0 & 1 & 1 & -1 & -1 \\ 0 & 0 & 1 & -1 & -1 & 1 \\ -1 & -1 & 0 & 0 & 1 & 1 \\ 0 & -1 & 1 & 1 & 0 & -1 \\ 0 & 1 & -1 & -1 & 1 & -1 \end{bmatrix}$ And $U = \begin{bmatrix} -2 & -3 & 1 & 0 & 2 & 0 \\ -10 & -12 & 5 & 2 & 8 & 1 \\ -9 & -11 & 5 & 2 & 8 & 1 \\ -5 & -6 & 2 & 1 & 4 & 0 \\ -8 & -10 & 4 & 2 & 7 & 1 \\ -4 & -5 & 2 & 1 & 3 & 0 \end{bmatrix}$

Then: $f = XU = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

Suppose: $g = \begin{bmatrix} 1 & 0 & -1 & 0 & -1 & 0 \\ -1 & 1 & 0 & 0 & 1 & 1 \\ 2 & -2 & 1 & 0 & -1 & 1 \\ 0 & 0 & -1 & 1 & -1 & 2 \\ 1 & 0 & 0 & -3 & 2 & 0 \\ -1 & 0 & 0 & 0 & 1 & 2 \end{bmatrix}$

Next Bob find inverse of f (mod q) [fq]. Thus we get: $f_q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

Now Bob generates a public key H as H = p*fq*g (mod q).

$$H = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}$$

6.2 Encryption Example

Now, suppose Alice want to send the message M to bob by using bob's public key.

$$M = \begin{bmatrix} 1 & 0 & -1 & 0 & -1 & 0 \\ -1 & 1 & 0 & 1 & 1 & 1 \\ 0 & -2 & 1 & 0 & -1 & -1 \\ 2 & 0 & 1 & 0 & -1 & 2 \\ 0 & 1 & -1 & 0 & 1 & 1 \\ 1 & 0 & -1 & 0 & 2 & -1 \end{bmatrix}$$

He first choose a random matrix

$$R = \begin{bmatrix} 1 & 1 & -1 & 1 & -1 & 1 \\ -1 & 1 & 1 & -1 & 1 & 0 \\ 0 & 1 & 1 & 2 & -1 & 1 \\ -1 & 0 & 1 & -1 & -1 & 2 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 1 & 2 & 1 \end{bmatrix}$$

Therefore encrypted message E = (R* H +M) (mod q) is computed as

$$E = \begin{bmatrix} 254 & 0 & 255 & 0 & 255 & 0 \\ 255 & 4 & 0 & 1 & 1 & 0 \\ 0 & 254 & 4 & 0 & 255 & 255 \\ 2 & 0 & 1 & 253 & 255 & 2 \\ 0 & 1 & 255 & 0 & 7 & 1 \\ 1 & 0 & 255 & 0 & 2 & 5 \end{bmatrix}$$

Alice sends this encrypted message E to Bob.

6.3 Encryption Example

Bob has received the encrypted message from Alice. He uses his private key f to compute

A = f *E (mod q)

$$A = \begin{bmatrix} 254 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & -253 & 0 & 0 \\ 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \end{bmatrix}$$

Since Bob is computing A (mod q). Next Bob reduce the coefficients of A (mod p), we get:

$$\mathrm{B} = \mathrm{A}\ (\mathrm{mod}\ \mathrm{p}) = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

The matrix will be Alice's original message M, So Bob has successfully decrypted Alice message.

7 Conclusion

In this paper we have proposed a method for choosing the different form of f = UX for encryption and decryption using by matrices of taking f as proposed in the original matrix formulation for NTRU cryptosystem. The advantage of using Hermite Normal Form matrices is that these improve the efficiency of lattice based cryptosystem in terms of key length and computation time without compromising their security. This method is more efficient and more secure as compare to the Nayak et al. [7]. There is always a matrix in NTRU Cryptosystem in the form of upper triangular matrix and invertible matrix which speeds up the key generation. Because a matrix is invertible only when it's determinant is 1.

References

[1] Wells A. L., “A polynomial form for logarithms modulo a prime", IEEE Transactions on Information Theory, pp. 845-846, 1984.

[2] Cohen H., “A Course in Computational Algebraic Number Theory", Springer-Verlag, Berlin, 1993.

[3] Coppersmith and A. Shamir, “Lattice attacks on NTRU", in Proc. of EURO-CRYPT 97, Lecture Notes in Computer Science, Springer-Verlag, 1997.

[4] Hoffstein J., Pipher J. and Silverman J.H., \NTRU": A Ring Based Public Key Cryptosystem, In Proc. Of ANTS III, volume 1423 of LNCS. Springer-Verlag, Available at http://www.ntru.com, pp. 267-288, 1998.

[5] Ho_stein J., Lieman D., Silverman J. “Polynomial Rings and Efficient Public Key Authentication", Proceeding of the International Workshop on Cryptographic Techniques and E-Commerce (CrypTEC 99), M.Blum and C.H.Lee, eds., City University of Hong Kong Press, 1999.

[6] Roja P.Prapoorna, Avadhani P.S. and Prasand E.V. “An Efficient Method of Shared Key Generation Based on Truncated Polynomials", IJCSNS International Journal of Computer Science and Network Security, VOL. 6 No. 8B, pp. 156-161, 2006.

[7] Nayak Rakesh, Sastry C.V., and Pradhan Jayaram “A Matrix Formula-tion for NTRU Cryptosystems", Proc. 16th IEEE International conference on Network(ICON-2008), New Delhi, India, pp. 12-14, 2008.

[8] User Manual of, “Mathematica 7.0".