A New Modular Multiplication Method in Public Key Cryptosystem

G.A.V.Rama Chandra Rao[1], P.V.Lakshmi[2], and N.Ravi Shankar[3]
(Corresponding author: G.A.V.Rama Chandra Rao)

Dept. of Computer science & Technology, Sri Chaitanya Engg. College, Visakhapatnam, India[1]
Dept. of Information of technology, GIT, GITAM University, Visakhapatnam, India[2]
Dept. of Applied Mathematics, GIS, GITAM University, Visakhapatnam, India[3]
(e-mail: drravi68@gmail.com)

(Received Spt. 20, 2011; revised and accepted Dec. 1, 2011)

Abstract

Modular multiplication is an important operation in several public key cryptosystems. This paper brings a novel idea based on the idea in the exponentiation computation. In this paper a new algorithm for modular multiplication for public key cryptosystem is presented. This proposed method is based on the following two ideas.(i) The remainder in regard to n can be constructed from the remainder with modulus (*2n+1*) and the remainder with modulus (*2n+2*). (ii) It often happens that *2n+1* and *2n+2* can easily be factorized, even if n is a prime number or difficult to be factorized into prime factors. The performance of new algorithm using estimation of the computational complexity is observed, and it is shown that the procedure is much faster than the previous algorithms.

Keywords: El Gamal cryptosystem, modular multiplication, pubic key cryptosystem, RSA cryptosystem

1 Introduction

A public key cryptosystem that is essentially a variant of Diffie-Hellman scheme [3] was introduced by El-Gamal [4]. The algorithm performs as follows: suppose $GF(q)$ is known by public. User "*A*" selects a generator $g \in GF(q)$, and an integer '*a*'. It then publishes (g, g^a) as the public key and keeps '*a*' secret. User "*B*," who requires to send a message $m \in GF(q)$ to "*A*," selects an integer $2 \le k \le q\text{-}2$ randomly, computes *m*. $(g^a)^k = m\, g^{ak}$, and sends the pair $(g^k, m\, g^{ak})$ to "*A*". User "*A*" who knows a, recovers m by computing $m.g^{ak}\left(g^k\right)^{-a} = m.g^{ak}\, g^{-ak}$.

Two public key cryptosystems with their security based on intractability of integer factorization are RSA [13] and Rabin public key encryption [12]. Aggarwal et al. [1] have proved that for almost all possible distributions of integer *n*, the problem of factoring integer *n* can be efficiently reduced to solving the strong RSA problem on Z_n in the generic ring model of computation, where an algorithm can perform ring operations, inverse ring operations, and test equality. The overview of major attacks on the RSA encryption and signatures are presented in [7].

Another important public key cryptosystem is Eliptic curve cryptosystem. The first elliptic curve scheme was proposed by Koblitz [8] and Miller [9] independently. The elliptic curve systems are based on a group of points on an elliptic curve which are defined over a finite field. A new approach on multi-core implementation has been proposed in the paper [2] using elliptic curve cryptosystem.

The method proposed in this paper is based on an idea that is different from those methods, and can be combined with the latter. The proposed method is based on the idea that the remainder for modulus n is constructed from the remainders with moduli different from n. In the already known method [5] the remainder with modulus *n* is constructed from the remainder with modulus *n+1* and the remainder with modulus *n+2*. The method proposed in this paper differs in the remainder with modulus *n* is constructed from the remainder with modulus *2n+1* and the remainder with modulus *2n+2*. It may be sensible to consider what the advantage is in using different moduli. In RSA cryptography, the modulus n is not a prime number, but its prime factorization is not known except for the decipherer. In El Gamal cryptography, the modulus is a prime number. If *2n+1* and *2n+2* can easily be prime factorized, the remainder operation can be speed up by applying the Chinese remainder theorem. In fact, n is set so that it is difficult to be prime factorized in RSA cryptography, it is applied to *n+1* and *n+2* in the paper [5]. Hwang et al. [6] have proposed an efficient modulo *p* multiplication algorithm with moderate factors of *p+1*

and *p-1* by assuming *p-1* and *p+1* can be decomposed into products of mutually prime factors. But such a consideration is not applied to *2n+1* and *2n+2*. It may be possible to set *n* so that *2n+1* and *2n+2* can easily be prime factorized, although the prime factorization of *n* is difficult.

Section 2 presents the Chinese remainder theorem, the exponential modular computation, RSA and Rabin cryptosystems. Section 3 presents the new modular multiplication theorem that gives the basis for the new remainder computation together with numerical examples. Section 4 presents the **mulmod** computation based on the modular computation derived in Section 3. The **mulmod** computation serves as the basis for the exponentiation computation. Section 5 gives the performance of new algorithm using estimation of the computational complexity, and it is shown that the procedure is much faster than the algorithm [5].

2 Remainder Computations and Cryptosystems

In this section we present the Chinese remainder theorem, RSA cryptosystem and Robin cryptosystem.

2.1 Chinese Remainder Theorem

Assume $m_1, m_2, \ldots m_t$ are mutually coprime. Denote $M=m_1m_2\ldots m_t$. Given $x_1, x_2, \ldots, x_t$ there exist a unique x, $0 < x < M$, such that

$$x = x_1 \bmod m_1$$
$$x = x_2 \bmod m_2$$
$$\vdots$$
$$x = x_t \bmod m_t$$

x can be computed as

$$x = \left(x_1u_1M_1 + x_2u_2M_2 + \ldots + x_tu_tM_t\right) \bmod M$$

where $M_i = \dfrac{m_1m_2\ldots m_t}{m_i}$ and $u_i = M_i^{-1} (\bmod\ m_i)$.

2.2 RSA Cryptosystem

In RSA, p and q are two relatively prime and large random numbers. A positive integer n is defined as a product of p and q.

$$\varphi(n) = (p-1)(q-1).$$

Choose e such that $1 < e < \varphi(n)$ and $gcd(e, \varphi(n)) = 1$.

Here, d is computed by $d = e^{-1} \bmod \varphi(n)$.

In RSA, e and n are public keys and d and (p, q) are private keys so the plaintext M is encrypted by: $1 < M < n$ and $C = M^e \bmod n$. And the cipher text C is decrypted by $M = C^d \bmod n$

2.3 Rabin Cryptosystem

Key generation:
Private keys: random primes p,q
Public key: $n = p\,q$

Encryption:
Plain text: $m \in Z_n^*$ where $Z_n^* = \{1,2,\ldots,n-1\}$
Cipher text: $c = m^2 \bmod N$.
Decryption:
Compute $c^{\frac{1}{2}} \bmod n$.

3 A New Modular Multiplication Method

In this section a new modular multiplication method is presented. The basic idea of this method is that the remainder modulo n can be transformed into remainder with modulus *2n+1* and modulus *2n+2*.

Theorem : Assume

$y = X \bmod n$, where $0 \le X \le (2n-1)^2$, (1)
$y_1 = X \bmod (2n+1)$ (2)
and
$y_2 = X \bmod (2n+2)$. (3)
Then y can be expressed as follows:
$y_1 \ge y_2, y \equiv 2y_1 - y_2 \pmod n$ (4)
$y_1 < y_2, y \equiv 2y_1 - y_2 + 2 \pmod n$. (5)

Proof :
From Equation (2), there exists an integer q_1 such that
$X = (2n+1)q_1 + y_1$ (6)
From Equation (3), there exists an integer q_2 such that
$X = (2n+2)q_2 + y_2$ (7)
Multiplying Equation (6) by $(2n+2)$, we obtain
$(2n+2)X = (2n+1)(2n+2)q_1 + (2n+2)y_1$ (8)
Multiplying Equation (7) by $(2n+1)$, we get
$(2n+1)X = (2n+1)(2n+2)q_2 + (2n+1)y_2$ (9)
From Equation (8) and Equation (9), we obtain
$X = (2n+2)y_1 - (2n+1)y_2 + (2n+1)(2n+2)(q_1-q_2)$ (10)
First assume that $q_1 - q_2 < 0$.
Since $y_1 \le 2n$ and $y_2 \ge 0$, it follows that
$(2n+2)y_1 - (2n+1)y_2 + (2n+1)(2n+2)(q_1-q_2) \le (2n+2)(2n) - (2n+1)(2n+2)$
$= -(2n+2) < 0$.
Since $X \ge 0$, from Equation (10), q_1-q_2 cannot be negative.
Second assume that $q_1 - q_2 > 1$. Since $y_1 \ge 0$ and $y_2 \le 2n+1$, it follows that
$(2n+2)y_1 - (2n+1)y_2 + (2n+1)(2n+2)(q_1-q_2)$
$\ge -(2n+1)(2n+1) + 2(2n+1)(2n+2)$
$= (2n+1)(2n+3) > (2n-1)^2$.
Since $X \le (2n-1)^2$, from Equation (10), q_1-q_2 cannot be greater than unity.
Thus it is shown that $q_1 - q_2 = 0$ or 1.
i.e., $q_1 = q_2$ or $q_1 = q_2 + 1$.
From $q_1 = q_2$ and using Equation (10), we conclude that if $y_1 \ge y_2$, then $y \equiv 2y_1 - y_2 \pmod n$.
From $q_1 = q_2 + 1$ and using Equation (10), we conclude that if $y_1 < y_2$, then
$y \equiv 2y_1 - y_2 + 2 \pmod n$. □

Example 3.1 :
Let $n = 19$ and $X = x^2$
(i) When $x = 11$, $y_1 = X \bmod (2n+1) = 121 \bmod (39) = 4$ and $y_2 = X \bmod (2n+2) = 121 \bmod (40) = 1$. In this case, $y_1 > y_2$. Applying Equation (4), i.e., $y \equiv 2y_1 - y_2 \pmod n$,

$y \equiv 2(4)-1 \pmod{19} = 7 \bmod 19$ is obtained. This agrees with the result of direct calculation 121 mod19.
(ii) When $x = 25$, $y_1 = X \bmod (2n+1) = 625 \bmod 39 = 1$ and $y_2 = X \bmod (2n+2) = 625 \bmod 40 = 25$. In this case, $y_1 < y_2$ Applying Equation (5), $y \equiv (2(1)-25+2) \pmod{19} = (-21) \bmod 19 = 2 \bmod 19$ is obtained. This agrees with the result of direct calculation of 625 mod 19.

Example 3.2:
Consider the RSA cryptography example in the paper [13]. Let $n = 1386$ and $X = x^2$.Then $2n+1 = 2773 = 47 \times 59$ and $2n+2 = 2774 = 38 \times 73$.
When $x = 920$, $y_1 = X \bmod (2n+1) = 846400 \bmod 2773 = 635$, and $y_2 = X \bmod (2n+2) = 846400 \bmod 2774 = 330$. In this case, $y_1 \geq y_2$. Applying Equation (4),

$$y \equiv 2(635) - 330 + (\bmod 1386)$$
$$\equiv 1270 - 330 \bmod (1386)$$
$$\equiv 940 \pmod{1386}$$
$$\equiv 940 \text{ is obtained.}$$

This agrees with the result of direct calculation of $920^2 \bmod 1386 = 940$.

As is seen from the above Examples 3.1 and 3.2, two computations of mod $(2n+1)$ and mod $(2n+2)$ are used. Instead of the single computation of mod n. It may seem that the computation is made more complex, but the method has an advantage. Considering above general cases, several ways have to be thought out to simplify the algorithm and the value k in order to make the computation easy.

4 New Remainder Multiplication Algorithm

The exponentiation computation can be composed of the remainder multiplication computation:

$$Y = xu \bmod n \qquad (11)$$

It was shown in section 3, that $Y = X \bmod n$ can be derived from $y_1 = X \bmod (2n+1)$ and $y_2 = X \bmod (2n+2)$. In the following, the calculation method **mulmod** for y_1 and y_2 is shown, which is needed in applying the algorithm based on theorem in section 2 to the calculation of Eq. (11) in the exponentiation computation. In this algorithm, the Chinese remainder algorithm is used to derive y_1 and y_2. This helps to improve the speed, but the computational complexity is discussed in the next section.

As the preliminary computation,$(2n+1)$ and$(2n+2)$ are decomposed into products of mutually prime factors. This need not be the prime factorization.

$$2n+1 = \prod_{i=1}^{m_1} p_i \qquad (12)$$

$$2n+2 = \prod_{i=1}^{m_2} q_i \qquad (13)$$

Assuming that moduli $(2n+1)$ and $(2n+2)$ are decomposed as above, the next algorithm receives x such that $0 \leq x \leq (2n-1)^2$, and outputs $y = xu \bmod \left(p_1, \ldots, p_{m_1}\right)$.

Algorithm **mulmod**$(x, \boldsymbol{p}, y)$

Input : $x, u, 0 \leq x \leq (2n-1)^2$,
$0 \leq u \leq (2n-1)^2, \boldsymbol{p} = \left(p_1 \ldots p_{m_1}\right)$

Output: $y = xu \bmod \left(p_1 \ldots p_{m_1}\right)$

Step 1 : Calculate $x_i = x \bmod p_i$, $u_i = u \bmod p_i$, $i = 1,\ldots, m_1$.
Step 2 : Calculate $a_i = x_i u_i$,
$i = 1,\ldots, m_1$.
Step 3 : Calculate $a_i = a_i \bmod p_i$, $i = 1,\ldots, m_1$.
Step 4: Calculate y by Chinese remainder theorem $(\boldsymbol{a}, \boldsymbol{p}, y)$

The algorithm **mulmod** is used. $y_1 = xu \bmod (2n+1)$ and $y_2 = xu \bmod (2n+2)$ are obtained by **mulmod** $(\boldsymbol{x}, \boldsymbol{p}, y_1)$ and **mulmod** (x, q, y_2), respectively.

Example 4.1: consider the same RSA cryptography as in Example 3.2
Let $n = 1386$ as preliminary computations $2n+1$ and $2n+2$ are decomposed.
i.e $2n+1 = 2773 = 47 \times 59$ and $2n + 2 = 2774 = 38 \times 73$.
Let $p_1 = 47, p_2 = 59, q_1 = 38$, and $q_2 = 73$.
As the exponentiation computation assume that $x = 920$ and $y = x^2 \bmod n$ is to be calculated.
The computation procedure for **mulmod** $(920, (47\times59), y_1)$ is shown in the following.
Step 1: $x_1 = 920 \bmod 47 = 27$
$x_2 = 920 \bmod 59 = 35$
Step 2: $a_1 = 27^2 = 729$
$a_2 = 35^2 = 1225$
Step 3: $a_1 = 729 \bmod 47 = 24$
$a_2 = 1225 \bmod 59 = 45$
Step 4: Solving the following system of congruence equations $y_1 = 635$ (in Example 2) is obtained and $y_1 \equiv 24 \bmod 47$ and $y_1 \equiv 45 \bmod 59$ similarly $y_2 = 330$ is obtained and $y_2 \equiv 26 \bmod 38$ and $y_2 \equiv 38 \bmod 73$ from **mulmod** $(920, (38\times73), y_2)$ from $y_1 = 635$ and $y_2 = 330$ hence $y_1 \geq y_2$ then apply the equation $Y \equiv 2y_1 - y_2 + \pmod n$ implies

$$Y \equiv 2(635) - 330 + \pmod{1386}.$$

Therefore, $Y \equiv 940 \bmod 1386 = 940$ is obtained. It was already seen in Example 2.

5 Performance

In this section, the computational complexity for new algorithm is discussed. The additions and subtractions are not counted, and the computational complexity is considered only for the multiplications and divisions. It is assumed that parallel computation is not used in each multiplication or division, and only ordinary straightforward computation is applied. The standard bit computational complexity is considered. The computational complexity for the multiplication is given by
Mul (a,b) = computational complexity for $a \times b$ bit number = $a(b+a)$.

Table 1: Example of a table caption

Traditional method	Hayashi [5] method	Hwang et al. [6]	Proposed Method
1. x^2 mod n 2. It involves multiplication and division. 3. computational complexity : 3 a^2	1. sqmod 2. It involves multiplication and division. 3. computational complexity : 3a (a+b)	1. x^y mod z 2. It involves multiplication and addition. 3.computational complexity : *1.5l*(y)*[M*(l(z)+*2*Mod(l(z))+*1]*	1. **mulmod** 2.It involves multiplication and division. 3. computational complexity : 3b (a+b)

Div(a,b) = computational complexity for a + b bit number = b(a-b).

Here, the individual multiplication and division are separated. In this case, partial computations can be executed in parallel. The computational complexity in the preliminary computation is not included. In the following, n is assumed to consist of a bits, and $p_1,...,p_h$ and $q_1,...,q_s$ of b bits at the maximum.

Computational complexity in ordinary direct method

In the ordinary calculation of x^2 mod n, the multiplication of two a-bit numbers and the division of a 2a-bit number by a a-bit number are required. The computational complexity for this is $C_0 = Mul(a, a) + Div(2a, a) = 3a^2$.

Computational complexity in Hayashi [5] method

In solving the system of congruence equations, the multiplication of a- bit number and b-bit number, as well as the division of (a+b) bit number by b-bit number, are required. The computational complexity of the method is 3a (a+b).

Computational complexity in Hwang et al. [6] method

The computational complexity of modular exponentiation (x^y mod z) is

1.5 l(y) *[M*(l(z)+*2*Mod (l(z))+*1]*

where l (.), M(.), and Mod(.) are length, computational complexity of multiplication, and computational complexity of modular exponentiation respectively.

Computational complexity in proposed method

The computational complexity of the **mulmod** algorithm is examined as follows. In order to calculate the right-hand side of the system of congruence equations, the division of a-bit number by b-bit number is required in stage 1, the multiplication of two b-bit numbers is required in stage 2, and the division of 2b-bit number by b-bit number is required in stage 3. In total, the computational complexity is $Div(a, b)+ Mul(b, b)+ Div(2b, b) = b(a- b)+ 2b^2+ b^2 = b(a+2b)$,which is needed for each i. The computational complexity of the proposed method is $C = 3b(a+b)$. The ratio of the computational complexity of the two methods is $\frac{C}{C_0}$.

6 Conclusion

In this paper, it is noted that the prime factorization of (*2n + 1*) and (*2n + 2*) is not always difficult, and it is shown that the remainder for mod n can be determined from the remainders y_1 and y_2 with those as the modulus. It is shown that the Chinese remainder theorem can be applied to the calculation of y_1 and y_2, and the remainder multiplication can be realized with less computational complexity. It is shown that the proposed computation method is useful in improving the speed of the exponentiation-based cryptosystem, such as RSA cryptosystem and El Gamal cryptosystem.

The effectiveness of the proposed method depends strongly on whether or not (*2n + 1*) and (*2n + 2*) can be decomposed into smaller prime factors. If n is set so that it is decomposed into three or more prime factors, the computational complexity will further be decreased. When n is the public key of RSA cryptosystem, if (*2n + 1*) and (*2n + 2*) are publicized in the factorized forms, the preliminary computation in ciphering is made unnecessary.

The proposed method herein will effectively be used to improve the speed of the signal processing,where the cryptography accompanying the exponentiation computation or similar computations is being used alone or combined with other methods. This new method can improve the speed in the processes of encryption, decryption and signature in public key cryptography, such as RSA or ECC.

Acknowledgments

The authors are grateful to the anonymous reviewers for valuable comments and suggestions that improve the presentation of this paper.

References

[1] D. Aggarwal, U. Maurer, and I. Shparlinski, "The equivalence of Strong RSA and factoring in the Generic Ring Model of computation," *WCC 2011-Workshop on Coding and Cryptography*, pp. 17-26, 2011.

[2] S. Basu, "A new parallel window based implementation of the Elliptic curve point multiplication in multi-core architectures," *International Journal of Network Security*, vol.14, no.1, pp.52-59, 2012.

[3] W. Diffie and M.Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, pp. 644-654, 1976.

[4] T.El-Gamal, "A public key cryptosystem and signature scheme based on discrete logarithms,"

IEEE Transactions on Information Theory, vol. 31, pp. 469-472, 1985.

[5] A. Hayashi, "A new fast modular multiplication method and its application to modular exponentiation based cryptography," *Electronics and Communications in Japan*, vol. 83, pp. 88-93,2000.

[6] R. J. Hwang, F. F. Su, and S. H. Shiau, "An efficient modulo *P* multiplication algorithm with moderate factors of *P*+1 and *P*-1," *Communication of Mathmatics Science*, vol. 5, no. 2, pp.383- 389, 2007.

[7] B.S. Kaliski JR. and M.Robshaw, "The secure use of RSA," *Cryptobytes*, vol.1, pp. 7-13, 1995.

[8] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203-209, 1987.

[9] V. Miller, "Uses of elliptic curves in cryptography," *Advances in cryptography – Crypto' 85*, LNCS 218, pp. 417- 426, Springer-Verlag, 1986.

[10] P. L. Montgomery, "Modular multiplication without trial division," *Mathematics of Computation*, vol. 44, pp. 519-521,1985.

[11] N. Nedjah and L. M. Mourelle, "Efficient parallel modular exponentiation algorithm," LNCS 2457, pp.405-414, Springer-Verlag, 2002.

[12] M. O. Rabin, "Digitalized signatures and public key functions as intractable as Factorization," MIT/LCS/TR-212,MIT Laboratory for computer science, 1979.

[13] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signature and public key cryptosystems," *Communications of the ACM*,vol. 21,pp. 120-126,1978.

[14] C. D. Walter, "Systolic modular multiplication," *IEEE Transactions on Computers*, vol. 42, no. 3, pp. 376-378, 1993.

G.A.V. Rama Chandra Rao received the M.Tech. degree in Computer Science and Technology from GITAM University. He is pursuing his Ph.D. in computer science and Engineering from GITAM University. His current research interests include cryptography and network security.

P.V. Lakshmi received the M.Tech. degree in Computer Science and Engineering and the Ph.D. degree in Computer Science and Engineering from Andhra University. She is now Professor & BoS Chairperson in the Department of information technology, GIT, GITAM University. Her current research interests include Bioinformatics, cryptography, and Network security. She had published more than 25 research papers on the above research fields. She has more than 15 years experience in teaching.

N. Ravi Shankar received the M.Sc. degree in Applied Mathematics, M.Tech. degree in Computer Science and technology with bioinformatics as specialization, and the Ph.D. degree in Applied Mathematics from Andhra University. He is now Associate Professor & BoS Chairman in the Department of Applied Mathematics, GIS, GITAM University. He is also on the editorial boards of several journals. His current research interests include Operations research, Group theory and its applications, fuzzy set theory, bioinformatics, rough set theory, cryptography, and graph theory. He had published more than 40 research papers on the above research fields. He has more than 19 years experience in teaching.