A Novel Modular Multiplication Algorithm and its Application to RSA Decryption

G.A.V. Rama Chandra Rao ¹, P.V. Lakshmi ² and N. Ravi Shankar ³

¹Dept. of computer Science & Technology, Sri Chaitanya Engg. College Visakhapatnam, Andhra Pradesh, India
²Dept. of Information Technology, GIT, GITAM University Visakhapatnam, Andhra Pradesh, India
³Dept. of Applied Mathematics, GIS, GITAM University Visakhapatnam, Andhra Pradesh, India

Abstract

The services such as electronic commerce, internet privacy, authentication, confidentiality, data integrity and non repudiation are presented by public key cryptosystems. The most popular of public key cryptosystems is RSA cryptosystem. RSA is widely used for digital signature and digital envelope, which provide privacy and authentication. The basic operation of RSA cryptosystem is modular exponentiation which is achieved by repeated modular multiplications. RSA can be speeded up by using the Chinese Remainder Theorem (CRT) and using strong prime criterion. In this paper, we present an efficient modulo n multiplication algorithm with reasonable factors of $2^n$ and $2^{n+2}$. In this paper we discuss decryption techniques in RSA cryptosystem. We show that this new technique can speed up the decryption process and it can reduce the computational time compared to the methods of traditional, CRT, and Hwang et al.[10].

Keywords: RSA cryptosystem, modular multiplication, Chinese Remainder theorem, prime factors.

1. Introduction

Modular multiplication contributes a vital role to more than a few public-key cryptosystems such as the RSA cryptosystem [1]. Hayashi et al. [2] and Rao et al.[3] proposed a new modular multiplication method to reduce the computational time of RSA cryptosystem. In the paper[2], modular exponentiation with the modulus n transforms into replacement operations with modulii n+1 and n+2. However, n is a odd number, we cannot factor n+2 easily in the method proposed in [2]. In [3], modular exponentiation with the modulus n converts into substitute operations with modulii 2n+1 and 2n+2. It happened that 2n+1 and 2n+2 can easily be factorized, even if n is a prime or difficult to be factorized into prime factor. However, in the method of [3], 2n+1 cannot be factored easily when n is odd number but advantage of method in [3] over the method in [2] is that 2n+1 is larger than n+1. In this paper, we propose an efficient modulo n multiplication method with factors of 2n and 2n+2. The advantage of this method is that we can easily factorize 2n and 2n+2 because they are even when n is even or odd. The proposed method renovates a modular operation with the modulus n into two alternate operations with moduli 2n+2 and 2n. The idea is that fast Chinese Remainder Theorem (CRT) assessment may work for moduli 2n+2 and 2n, even if it does not work for modulus n. The algorithm can improve the performance of RSA decryption and reduce the time complexity of RSA encryption. The security of RSA cryptosystem is based on the difficulty of factoring problem. So, the prime factors of modulus of RSA algorithm must be strong primes. The large modular exponentiation result can be generated from small exponents and moduli. Based on the strong prime [4-8] of RSA principle, abusers can employ the proposed algorithm to improve the performance of RSA decryption.

In section 2, we review RSA algorithm. Section 3 introduces our proposed algorithm for modular multiplication. Computational complexity of our proposed algorithm for decryption in public key cryptosystem (RSA) is presented in section 4 before we originate conclusions in section 5.
2. RSA algorithm

RSA algorithm is a classic public-key cryptosystem for encryption and decryption. It is a fundamental procedure of various security protocols. It can be illustrated in brief as follows:

(i) Select two large strong prime numbers, \( p \) and \( q \). Let \( n = p \times q \).

(ii) Compute Euler’s totient value for \( n : \phi(n) = (p - 1) \times (q - 1) \).

(iii) Find a random number \( e \) satisfying \( 1 < e < \phi(n) \) and relatively prime to \( \phi(n) \) i.e., \( \gcd(e, \phi(n)) = 1 \).

(iv) Calculate a number \( d \) such that \( d = e^{-1} \mod \phi(n) \).

(v) Encryption: Given a plain text \( m \) satisfying \( m < n \), then the cipher text \( c = m^e \mod n \).

(vi) Decryption: The cipher text is decrypted by \( m = c^d \mod n \).

3. A novel algorithm for modular multiplication

In this section, we present a well-organized modulo \( n \) multiplication algorithm with reasonable factors of \( 2n+2 \) and \( 2n \). Let \( a, b, n \) be three \( n \)-bit positive binary integers where \( a, b < n \) and \( \gcd(2, 2n) = 2 \) (a prime). Assume \( 2n \) and \( 2n+2 \) can be decomposed into products of mutually prime factors, i.e., \( 2n = 2^s \times (l_1 l_2 ... l_i) \), \( 2n+2 = 2^s \times (m_1 m_2 ... m_i) \), where \( m_i \) and \( m_i \) are relatively prime to each other for \( 1 \leq i \leq j \leq s \), and \( 2n+2 = 2(n+1) = 2^s \times (m_1 m_2 ... m_i) \), where \( m_i \) and \( m_i \) are relatively prime to each other for \( 1 \leq i \leq j \leq t \).

A novel modular multiplication algorithm

**Input:** \( a, b, n, l_1, l_2, ..., l_i, m_1, m_2, ..., m_i \)

**Output:** \( g = ab \mod n \)

**Step 1:** Calculate \( k = (n^2-1)/2 \).

**Step 2:** Compute \( s = ab \).

**Step 3:** If \( s \geq k \), then \( q = 1 \), else \( q = 0 \).

**Step 4:** Compute \( r_j = s \mod l_j, j = 1, ..., s \).

**Step 5:** Use CRT algorithm to compute \( y_1 \).

**Step 6:** Compute \( y_i = s \mod m_i, k = 1, ..., t \).

**Step 7:** Use CRT algorithm to compute \( y_2 \).

**Step 8:** Compute \( g = 2^s (y_1 + y_2 - q') \mod n \) where \( q' = q \) if \( y_1 > y_2 \), otherwise \( q' = q+1 \).

**Step 9:** Return \( (g) \)

**CRT algorithm**

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

\[
x = x_1 \mod m_1 \]
\[
x = x_2 \mod m_2 \]
\[
\vdots
\]
\[
x = x_t \mod m_t
\]

\( x \) can be computed as

\[
x = (x_1 m_2 m_3 ... m_t + x_2 m_3 m_4 ... m_t + ... + x_t m_1 m_2 ... m_{t-1}) \mod M
\]

where \( M_i = (m_1 m_2 ... m_{i-1} / m_i \) and \( u_i = M_i^{-1} \mod m_i \).

The following theorem exhibits that the effect of \( \text{"ab mod n"} \) can be produced from \( \text{"(ab) mod (2n)"} \) and \( \text{"(ab) mod (2n+2)"} \). The new modular algorithm produces the exact value of \( \text{"a(b) mod n"} \) at Step 8.

**Theorem 3.1:** Given \( y_1 = x \mod (2n), y_2 = x \mod (2n+2) \), such that \( 0 \leq x \leq 4n(n+1) \) and \( \gcd(2, 2n) = 2 \), then \( g = x \mod (2n+1) = 2^s(y_1+y_2)(x \geq 2n(n+1)) - (y_1+y_2) \mod (2n+1) \), where Knuth’s bracket notation \([9]\) for a Boolean-valued expression \( E \) i.e., \([E] \) is 0 if \( E \) is false and 1 if \( E \) is true.

**Proof:** Since \( 0 \leq x \leq 4n(n+1) \), we write \( x = a \cdot (2n+1) + g \), where \( 0 \leq a \leq 2n \) and \( 0 \leq s \leq 2n \).

If \( g+a \leq 2n \), then \( x = a(2n+2)(g+a) \), so \( y_1 = x \mod (2n) = g + a \).

If \( g + a \geq 2n+1 \), then \( x = (a+1)(2n+2)(g+a-2n) \), so \( y_2 = x \mod (2n) = g+a-2n \).

Similarly, if \( g - a \leq 0 \) then \( x = r(2n+2)(g-a) \), so \( y_2 = x \mod (2n+2) = g-a \).

Thus, there are four cases to consider in computing \( y_1 + y_2 \).

**Case (i):** \( y_1 + y_2 = g + a + g - a = 2g \).

Then \( g = 2^s(y_1+y_2) \mod 2n+1 \).

Since \( s + a \leq 2n \) and \( g - a \geq 0 \), we obtain \( x < 2n(n+1) \) and

\[
y_1 + y_2
\]

**Case (ii):** \( y_1 + y_2 = g + a + g + a + 2n + 2 = 2g + 2n + 2 = 2(g + n + 1) \).

Then \( g = 2^s(y_1 + y_2 - 1) \mod 2n+1 \).

Since \( g + a \leq 2n \) and \( g - a \leq -1 \), we obtain \( x < 2n(n+1) \) and

\[
y_1 + y_2
\]

**Case (iii):** \( y_1 + y_2 = g + a - 2n - 1 + 1 + g + a = 2g - 2n = 2(g - n) \).

Then \( g = 2^s(y_1 + y_2 - 1) \mod 2n+1 \).
Since \( g + a \geq 2n + 1 \) and \( g - a \geq 0 \), we obtain \( x \geq 2n(n+1) \) and \( y_1 > y_2 \).

**Case (iv):** \( y_1 + y_2 = g + a - 2n - 1 + 1 + g - a + 2n + 2 = 2g + 2 \).

Then \( g = 2^{-1}(y_1 + y_2) \mod 2n + 1 \).

Since \( g + a \geq 2n + 1 \) and \( g - a \leq -1 \), we obtain \( x \geq 2n(n+1) \) and \( y_1 < y_2 \).

Thus, \( g = 2^{-1}(y_1 + y_2) - q \) \mod \( 2n + 1 \), where \( q \) is either 0 or 1.

In the modular multiplication computation, the remainder with modulus \( n \) can be derived from both the remainder with \( 2n \) and the remainder with \( 2n+2 \) by the previous theorem. If \( 2n \) and \( 2n+2 \) can be decomposed into products of mutually prime factors then a computation with numbers of smaller scale must be faster. The computations of the remainder with modulus \( 2n \) and the remainder with modulus \( 2n+2 \) consist of several independent parts, so that these computations can also be performed in parallel.

Additionally, \( 2x(2n+2)/2 = 2n+2 \equiv 1 \mod n \), so \( (2n+2)/2 \) is the multiplicative inverse of 2 modulo \( n \). Therefore, the inverse value of Step 8 can be computed efficiently. It is clear that the proposed modular multiplication algorithm is more efficient than direct modular multiplication.

**Lemma 1:** Given \( y_1 = x \mod(2n) \) and \( y_2 = x \mod(2n+2) \) such that \( 0 \leq x < 2n(n+1) \) and \( n \) is a prime, then \( x = (2n+2)y_1/2 - (2n)y_2/2 + (2n)(2n+2)q/2 \), where \( q \) is either 0 or 1.

**Proof:**

\[
\begin{align*}
y_1 &= x \mod(2n) \quad (A) \\
y_2 &= x \mod(2n+2) \quad (B) \\
\text{From (A) and (B) we get,} \\
x &= y_1 + 2n q_1 \quad (1) \\
x &= y_2 + (2n+2) q_2 \quad (2)
\end{align*}
\]

where \( q_1 \) and \( q_2 \) are two positive integers.

Multiplying Eq. (1) by \( 2n+2 \) we get

\[
x(2n+2) = (2n+2)y_1 + (2n)(2n+2)q_1 \quad (3)
\]

Multiplying Eq. (2) by \( 2n \) we get

\[
x(2n) = (2n)y_2 + (2n)(2n+2)q_2 \quad (4)
\]

Using Eq.(3) and Eq.(4),

\[
x = (2n+2)y_1 - (2n)y_2 + (2n)(2n+2)q_1 - (2n)(2n+2)q_2 \quad (5)
\]

Eq.(5) can be rewritten as

\[
x = (n+1)y_1 - n y_2 + (2n)(n+1)q_1 \quad (6)
\]

We will prove \( q \) is either 0 or 1 as follows:

**Case (i):** Assume \( q < 0 \).

Since \( y_1 \leq (2n-1) \) and \( y_2 \geq 0 \), we get

\[
x \leq (n+1)(2n-1) - (2n)(n+1) = -1 < 0
\]

i.e., \( x < 0 \).

Which is a contradiction.

Therefore, \( q \geq 0 \).

**Case (ii):** Assume that \( q > 1 \).

Since \( y_1 \geq 0 \) and \( y_2 \leq (2n+1) \), we get

\[
x \geq 2n \cdot 2n + (2n)(2n+2)q_2 > 2n(n+1)
\]

This result contradicts the given condition \( x < 2n(n+1) \).

Hence, \( q \) is not larger than 1.

By above two cases, \( q \) must be either 0 or 1.

**Theorem 3.2:** Given \( y_1 = x \mod(2n) \) and \( y_2 = x \mod(2n+2) \) such that \( 0 \leq x < 2n(n+1) \) and \( n \) is a prime, then \( x = 2^{-1}(y_1 + y_2 - q) \mod(2n+1) \), where \( q = 0 \) if \( y_1 \geq y_2 \); otherwise \( q = 1 \).

**Proof:** Using Lemma 1,

\[
x = (n+1)y_1 - n y_2 + (2n)(2n+1)q_1 \quad (7)
\]

where \( q \) is either 0 or 1.

In addition, we will demonstrate the conditions of \( q = 0 \) and \( q = 1 \) in Eq. (7).

From Eq.(2) and Eq.(1), we get

\[
y_2 = y_2 - y_1 = (2n)q_1 - (2n+2) q_2 (2n)(q_1 - q_2) - 2q_2 \quad (8)
\]

where \( y_1 \geq 0 \), \( y_2 \geq 0 \) and \( 2n+1 > 0 \) and both \( q_1 \) and \( q_2 \) are two positive integers.

Let \( q = q_1 - q_2 \). Then
\[ y_2 - y_1 = (2n)q - 2q \]

By Eq. (2) and \( 0 \leq x < 2(n+1), 0 \leq y_2 \leq (2n+1) \), we get \( 0 \leq y_2 + (2n+2)q \leq 2n(n+1) \)

Hence, \( 0 \leq q_2 < n \)

Now by Eq. (8),

\[ y_2 - y_1 > (2n)q - 2n = 2n(q - 1) \]

(9)

Case (i): Let \( y_1 \geq y_2 \).

Therefore, \( y_2 - y_1 \leq 0 \).

By Eq. (9), we get \( (2n)(q - 1) < 0 \).

Since \( 2n > 0 \), we have \( q - 1 < 0 \Rightarrow q < 1 \).

By Lemma 1, \( q \) must be equal to \( 0 \) when \( y_1 \geq y_2 \).

Case (ii): Let \( y_1 < y_2 \).

we get \( y_2 - y_1 > 0 \).

By Eq. (9), we get \( (2n)(q - 1) \geq 0 \Rightarrow q \geq 1 \).

By Lemma 1, \( q = 1 \) when \( y_1 < y_2 \).

Hence \( x = 2^{-1}(y_1 + y_2 - q) \mod (2n+1) \), where \( q=0 \) if \( y_1 \geq y_2 \); otherwise \( q=1 \).

2. MUL \((x)\), ADD \((x)\) and MOD \((x)\) denotes the computational complexity of multiplication, addition and modulus operations with the bit length of operand is \( x \).

(iii) LEN \((x)\) denotes the bit length of \( x \).

(iv) SH denotes computational complexity of the shift operator.

By the addition chain method [9] the computational complexity of modular exponentiation is

\[ EMOD(y, n) = 1.5 \times \text{LEN}(y)[\text{MUL}(\text{LEN}(n)) + 2 \text{MOD}(\text{LEN}(n)) + 1] \]

(10)

The computational complexity of multiplication and addition operations can be expressed as follows [13]:

\[ \text{MUL}(x) = 3 \times \text{MUL}(x/2) + 5 \times \text{ADD}(x) + 2 \times \text{SH} \]

(11)

\[ \text{ADD}(x) = x/32. \]

(12)

Using divide and conquer algorithm [14], the computational complexity of modulo operation can be expressed as

\[ \text{MOD}(x) = \text{MOD}(x/2) + 4 \times \text{MUL}(x/2) + 1.5 \times \text{ADD}(x) + 3 \times \text{SH} \]

(13)

w.l.g., we assume that the computational complexities MOD \((32)\), MUL \((32)\), ADD \((32)\) and SH take one clock cycle. The clock cycles for MUL \((x)\), MOD \((x)\), and ADD \((x)\) are calculated and given in Table I.
By the equations (11), (12) and (13), we get
\[ \text{MOD (1024)} = \text{MOD (512)} + 4 \times \text{MOD (256)} + 1.5 \times \text{ADD (1024)} + 3 \times \text{SH} \]
\[ = [\text{MOD (256)} + 4 \times \text{MOD (256)} + 1.5 \times \text{ADD (512)} + 3 \times \text{SH}] + 3295 \]
\[ = [\text{MOD (128)} + 4 \times \text{MOD (128)} + 1.5 \times \text{ADD (256)} + 3 \times \text{SH}] + 4294 \]
\[ = [\text{MOD (64)} + 4 \times \text{MOD (64)} + 1.5 \times \text{ADD (128)} + 3 \times \text{SH}] + 4577 \]
\[ = [\text{MOD (32)} + 4 \times \text{MOD (32)} + 1.5 \times \text{ADD (64)} + 3 \times \text{SH}] + 4646 \]
\[ = 4657 \]

\[ \text{MOD (2048)} = \text{MOD (1024)} + 4 \times \text{MOD (1024)} + 1.5 \times \text{ADD (2048)} + 3 \times \text{SH} \]
\[ = [\text{MOD (1024)} + 4 \times (2595) + 1.5 \times (64) + 3] \]
\[ = [\text{MOD (1024)} + 10380 + 96 + 3] \]
\[ = [\text{MOD (1024)} + 10479] \]
\[ = [\text{MOD (512)} + 4 \times \text{MOD (512)} + 1.5 \times \text{ADD (1024)} + 3 \times \text{SH}] + 10479 \]
\[ = [\text{MOD (512)} + 4 \times (811) + 1.5 \times (32) + 3] + 10479 \]
\[ = [\text{MOD (512)} + 3295] + 10479 \]
\[ = [\text{MOD (512)} + 13774 \]
\[ = [\text{MOD (256)} + 4 \times \text{MOD (256)} + 1.5 \times \text{ADD (512)} + 3 \times \text{SH}] + 13774 \]
\[ = [\text{MOD (256)} + 4 \times (243) + 1.5 \times (16) + 3] + 13774 \]
\[ = [\text{MOD (256)} + 972 + 24 + 3] + 13774 \]
\[ = [\text{MOD (256)} + 14773 \]
\[ = [\text{MOD (128)} + 4 \times \text{MOD (128)} + 1.5 \times \text{ADD (256)} + 3 \times \text{SH}] + 14773 \]

= 14773

\[ \text{MOD (8192)} = \text{MOD (4096)} + 4 \times \text{MOD (4096)} + 1.5 \times \text{ADD (8192)} + 3 \times \text{SH} \]
\[ = [\text{MOD (4096)} + 4 \times (24963) + 1.5 \times (256) + 3] \]
\[ = [\text{MOD (4096)} + 99852 + 384 + 3] \]
\[ = 147998 \text{ clock cycles.} \]

By Equation (10), the traditional decryption method can be represented as
\[ \text{EMOD} (y, n) = 1.5 \times 8192 \times \text{MOD (8192)} + 4 \times \text{MOD (8192)} + 1.5 \times \text{ADD (4096)} + 3 \times \text{SH} \]
\[ = 15136 \times (4(8107) + 1.5 \times (128) + 3] \]
\[ = 15136 + 32428 + 192 + 3 \]
\[ = 47759 \text{ clock cycles.} \]

If the decryption method based on CRT only and the bit lengths of p and q are equal, the operation of the decryption method can be represented as 2 \times \text{EMOD} (y/2, n/2) + 3 \times \text{ADD (4096)} + 2 \times \text{MUL (4096)} + \text{MOD (4096)}. By this case, the decryption method takes \[ 1480580885 \text{ clock cycles.} \]

In Hwang et al. [10] method, p - 1, p + 1, q - 1 and q + 1 can be factored into at least three numbers. Without loss of generality, in the research papers [5,6, 7, 12, 15] assumed that the bit length of the largest prime factor is about \( \{\text{LEN}(n)\}/4 \) and others are about \( \{\text{LEN}(n)\}/8 \). The total number of operations of Hwang et al. [10] method is 4 \times \text{EMOD} (d/4, n/4) + 8 \times \text{EMOD}...
(d/8, n/8) + 4[ADD(2048) + 2MUL(2048) + MOD(2048)] + 4[ADD(1024) + 2MUL(1024) + MOD(1024)] + 2[ADD(4096) + MUL(4096) + MOD(4096)] + ADD(4096) + 2MUL(4096) + MOD(4096). It takes 618359917 clock cycles.

In our proposed method, 2n, 2n+2, 2m and 2m+2 (i.e., 2n, 2(n+1), 2m, 2(m+1)) can be factored into at least two numbers because 2 is a prime number and n may be prime or not a prime and (n+1) can be expressed as product of largest prime factor as mentioned by Hwang et al. [10]. Without loss of generality, in the research papers [5, 6, 7, 12, 15] assumed that the bit length of the largest prime factor is about \(\frac{\text{LEN}(n)}{8}\) and others are about \(\frac{\text{LEN}(n)}{16}\). The total number of operations of our proposed method is 8 EMOD (d/8, n/8) + 16 EMOD (d/16, n/16) + 4[ADD(256) + 2MUL(256) + MOD(256)] + 4[ADD(128) + 2MUL(128) + MOD(128)] + 2[ADD(512) + MUL(512) + MOD(512)] + ADD(512) + 2MUL(512) + MOD(512). It takes 2715648 clock cycles.

<table>
<thead>
<tr>
<th>Bit length of operand x</th>
<th>8192</th>
<th>4096</th>
<th>2048</th>
<th>1024</th>
<th>512</th>
<th>256</th>
<th>128</th>
<th>64</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>MUL (x) (clock cycles)</td>
<td>76171</td>
<td>24963</td>
<td>8107</td>
<td>2595</td>
<td>811</td>
<td>243</td>
<td>67</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>MOD (x) (clock cycles)</td>
<td>147998</td>
<td>47759</td>
<td>15136</td>
<td>4657</td>
<td>1362</td>
<td>363</td>
<td>80</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>ADD (x) (clock cycles)</td>
<td>256</td>
<td>128</td>
<td>64</td>
<td>32</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
5. Conclusion

Security has become more important technique in many applications including electronic commerce, secure internet access, and virtual private network. RSA cryptosystem is widely used for digital signature which provide privacy and authentication. The basic operation of RSA cryptosystem is modular exponentiation which is achieved by repeated modular multiplications. RSA can be speeded up by using the Chinese Remainder Theorem (CRT) and using strong prime criterion. In this paper, we have proposed an efficient modular multiplication algorithm with reasonable factors of $2n$ and $2n+2$. In this paper we have discussed decryption techniques in RSA cryptosystem. We have compared our technique with methods of traditional, CRT, and Hwang et al. The speed of proposed method is faster than the decryption method on CRT and Hwang et al. This new method can be applied not only on decryption operation but also signing phase of digital signature.

This paper proposes an efficient modular multiplication algorithm. The proposed algorithm is based on the idea that the remainder for modulus n can be generated from the remainder with modulus $2n$ and the remainder with modulus $2n+2$. The proposed algorithm greatly enhances the performance of RSA decryption, in addition to reducing the computational time of RSA.

References


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 & Head in the Department of information technology, GIT, GITAM University. Her current research interests include Bioinformatics, cryptography, and Network security. She had published more than 30 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 45 research papers on the above research fields. He has more than 20 years experience in teaching.