AN ANALYSIS ON EFFICIENT POLYNOMIAL MULTIPLICATION ALGORITHMS FOR CRYPTOGRAPHIC PURPOSES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MURAT BURHAN İLTER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
CRYPTOGRAPHY

SEPTEMBER 2016

Approval of the thesis:

AN ANALYSIS ON EFFICIENT POLYNOMIAL MULTIPLICATION ALGORITHMS FOR CRYPTOGRAPHIC PURPOSES

submitted by **MURAT BURHAN İLTER** in partial fulfillment of the requirements for the degree of **Master of Science in Department of Cryptography, Middle East Technical University** by,

Prof. Dr. Bülent Karasözen
Director, Graduate School of **Applied Mathematics** _____

Prof. Dr. Ferruh Özbudak
Head of Department, **Cryptography** _____

Assoc. Prof. Dr. Murat Cenk
Supervisor, **Cryptography, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Ali Doğanaksoy
Mathematics, METU _____

Assoc. Prof. Dr. Murat Cenk
Cryptography, METU _____

Assist. Prof. Dr. Fatih Sulak
Mathematics, ATILIM University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: MURAT BURHAN İLTER

Signature :

ABSTRACT

AN ANALYSIS ON EFFICIENT POLYNOMIAL MULTIPLICATION ALGORITHMS FOR CRYPTOGRAPHIC PURPOSES

İLTER, Murat Burhan
M.S., Department of Cryptography
Supervisor : Assoc. Prof. Dr. Murat Cenk

September 2016, 36 pages

The idea of Public Key Cryptography showed up after the studies conducted by W. Diffie and M. Hellman in 1976. In the light of these works, RSA, the first Public Key Cryptography algorithm, came into play. In this algorithm, modular exponentiation is highly costly. In addition to this, key sizes of public key cryptography algorithms has become longer in order to ensure the security as the time passes. For these reasons, the speed of algorithms is relatively slower when it is compared to the speed of ones in Symmetric Key Cryptography algorithms. However, Public Key Cryptography algorithms have a wide area of utilization. Thus, it is highly crucial to accelerate these algorithms. Making use of fast multiplication algorithms is an effective way to reduce the cost.

In this thesis, the main point is to analyze some multiplication algorithms with respect to their time complexities. Before the invention of Karatsuba method, time complexity of multiplying two polynomials was known to be $O(n^2)$. After this invention lots of research has been done in this field. For example, Fast Fourier Transform is used for designing fast multiplication algorithms. However, this method is not efficient enough for cryptographic demands in today's world. For this reason, existing methods such as Karatsuba and its variations, Montgomery, and hybrid methods are investigated.

Keywords : Karatsuba, Refined Karatsuba, Optimized Karatsuba, Montgomery's method

ÖZ

KRİPTOGRAFİK AMAÇLAR İÇİN VERİMLİ POLİNOM ÇARPMASI ÜZERİNE BİR ANALİZ

İLTER, Murat Burhan
Yüksek Lisans, Kriptografi Bölümü
Tez Yöneticisi : Doç. Dr. Murat Cenk

Eylül 2016, 36 sayfa

Açık Anahtarlı Kriptografi fikri W. Diffie ve M. Hellman'ın 1976 yılında yürüttüğü çalışmalardan sonra ortaya çıktı. Bu çalışmaların ışığında, ilk Açık Anahtarlı Kriptografi algoritması olan RSA ortaya çıktı. Bu algoritmada, modüler üst alma oldukça maliyetlidir. Buna ek olarak zaman geçtikçe güvenliği sağlamak için Açık-Anahtar Kriptosistem algoritmalarının anahtar uzunlukları artmaktadır. Bu sebeplerden ötürü, Simetrik Anahtarlı Kriptografi algoritmalarının hızıyla karşılaştırıldığında Açık Anahtarlı Kriptografi algoritmalarının hızı daha yavaştır. Buna karşın Açık Anahtarlı Kriptografi algoritmalarının geniş bir kullanım alanı vardır. Bu yüzden bu algoritmaları hızlandırmak oldukça önemlidir. Hızlı çarpma algoritmalarından yararlanmak maliyeti azaltmak için verimli bir yoldur. Bu tezde, ana fikir olarak bazı çarpma algoritmalarını zaman karmaşıklıklarına göre analiz etmektir. Karatsuba metodunun icadından önce iki polinomun çarpımının zaman karmaşıklığı $O(n^2)$ olarak biliniyordu. Bu icattan sonra, bu alanda birçok çalışma yapıldı. Örnek olarak, Hızlı Fourier Dönüşümü hızlı çarpma algoritmalarını tasarlamak için kullanılmıştır. Buna karşılık, bu metot günümüz dünyasında kriptografik talepler için yeterince verimli değildir. Bu sebeple, Karatsuba ve varyasyonları, Montgomery ve hibrid metodları gibi mevcut olan metotlar incelenmektedir.

Anahtar Kelimeler : Karatsuba, İncelikli Karatsuba, Optimize Karatsuba, Montgomery metodu

To My Family

ACKNOWLEDGMENTS

I would like to thank my thesis supervisor, Assoc. Prof. Dr. Murat Cenk, for his support, motivation and guidance. Without his help this thesis would not be achieved. It is a chance for me to work under his guidance. I would like to also thank my thesis jury members, Assoc. Prof. Dr. Ali Doğanaksoy and Assist. Prof Dr. Fatih Sulak for their meaningful inputs.

I am grateful to my professor Assist. Prof Dr. Mehmetçik Pamuk from the Department of Mathematics for his constant support and motivation throughout my undergraduate and graduate education.

I would like to thank Tayfun Doğrar, H. Bartu Yünüak, Dr. Onur Koçak and H. Ali Şahin for their help while writing this thesis.

I would like to thank my friends Sura İmren, Hakan Genç, Cemre Aydın and my friends from ”Room 001” for their kind friendship and endless support.

I would like to thank my parents Altan İlter and Reha İlter and my brother Umut İlter for always encouraging and motivating me.

My deepest thanks belong to Cansu Aktepe who was always there with love and motivation when I needed her.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

$\mathbb{R}$	Real Numbers
$\forall$	For all
$\exists$	There exists

CHAPTER 1

INTRODUCTION

The history of cryptography dates back to ancient Egypt nearly 1900 B.C., where it was used to hide the meaning of messages written on hieroglyphs [13]. Apart from Egypt, the ancient Greeks also utilized cryptography for different purposes on a basic level.

The need for cryptography gained a lot of importance during Second World War when great progress was made in terms of mechanical cipher machines. At the present time, on the other hand, it has become more of an issue to enhance security with the increasing number of devices connected to internet, mobile communication, and on-line banking. For example, in TLS protocol used for the purpose of secure internet connection, many cryptographic algorithms such as RSA [24] and AES [8] are being operated. In the future, it is planned that many devices we use today in our daily life are going to be connected to the internet with the concept of Internet of Things (IoT). This means there will be much more demand to cryptography so as to reinforce security and privacy.

In modern cryptography, there are two encryption types namely Symmetric Key Cryptography and Asymmetric Cryptography, also known as Public Key Cryptography. More than 4000 years symmetric Key Cryptography has been used for secure communication [23]. Before the invention of the Public Key Cryptography, in order to utilize the cryptographic algorithms, people needed to exchange keys through a secure channel. In their works, Diffie and Hellman [9] suggested exponentiation in a finite field to exchange keys which made it possible to utilize in an insecure channel. In 1977, Ron Rivest, Adi Shamir, and Leonard Adleman discovered RSA algorithm [24]. Following the invention of both Diffie-Hellman key exchange and RSA algorithm, researchers tended towards Public Key Cryptography and more algorithms were discovered.

Although a great number of problems is solved by using public key cryptography, compared to symmetric key cryptography, it has some disadvantages. First of all, with the purpose of ensuring the equivalent security, key sizes of the asymmetric algorithms should be longer than the ones in symmetric algorithms. As claimed by National Institute of Standards and Technology (NIST), suggested key sizes for equivalent security of RSA algorithm should be at least 2048-bit but the key size of AES algorithm is at least 128- bit [1]. To obtain the identical security level as block ciphers, it is not possible to use shorter keys in public key cryptography algorithms. In 2010, Klein-

jung et al. [16] showed that 768-bit RSA modulus can be factorized in a practical time. In other words, using RSA in 768 or shorter key lengths is not secure anymore. The longer key length is one of the reasons why asymmetric key algorithms are slower than symmetric key algorithms. Another reason is that operations in Public Key Cryptography is more costly. For instance, multiplication operation in RSA algorithm has high cost. For elliptic curve, point doubling takes a long time to compute.

In this thesis, the main goal is to design efficient integer multiplication algorithms by introducing a faster multiplication algorithms and using them hybridly in recursive approach.

1.1 Public Key Cryptography

In this section, Public Key Cryptography (PKC) algorithms are introduced. Integer factorization, discrete logarithm, and elliptic curve are three main schemes used for public key cryptography. Key distribution problem in an insecure channel and a number of keys $\left(\frac{n(n-1)}{2}\right.$ number of keys for n separate user$\left.\right)$ solved by public key cryptography. However, keys of PKC algorithms are longer than symmetric algorithms. The following table is taken from [23].

Table 1.1: Bit lengths of different algorithms for different security levels

Algorithm	Security level as bits			
RSA	1024	3072	7680	15360
ECDH	160	256	384	512
AES	80	128	192	256

In the Table 1.1, the same cryptographic strength with different key lengths is shown explicitly. As observed from this table, to guarantee the same security level, key lengths should be longer in RSA algorithm than AES algorithm.

The timing comparison of AES and RSA algorithms is taken from [19].

Table 1.2: Comparison of RSA and AES of Encryption and Decryption Time

Algorithm	Packet Size (KB)	Encryption Time (sec)	Decryption Time (sec)
AES	153	1.6	1
RSA		7.3	4.9
AES	196	1.7	1.4
RSA		8.5	5.9
AES	312	1.8	1.6
RSA		7.8	5.1
AES	868	2.0	1.8
RSA		8.2	5.1

As it can be observed from the Table 1.2 even for small packet sizes, RSA algorithm is slower than AES algorithm with respect to time. Therefore, in lots of studies have been investigated to speed up the PKC algorithms. One of the main objects of these studies is to decrease the cost of multiplication because the cost of multiplication in RSA algorithm is considerably high. Fast multiplication algorithms are functional to accelerate this algorithm.

Before 1960's, naive multiplication algorithm (Schoolbook method) had been used to multiply polynomials. In 1963, A. A. Karatsuba and Y. Ofman discovered a new algorithm which is known today as Karatsuba method [14] to multiply polynomials. Following this method Toom-Cook [27] [6] algorithm was suggested.

Discrete Fourier and Fast Fourier Transform methods are used to discover new multiplication algorithms. These methods are efficient to multiply very large numbers. In other words, these techniques includes the multiplication algorithms which are seen to be the fastest, but solely when the number of binary digits is in the span 8,000- 10,000 their advantage in terms of swiftness over the schoolbook method $O(n^2)$ emerges. Depending on the desires of cryptographic systems, such numbers are greater than the span which is imagined in the implementation domain of functions [30].

1.2 About this Thesis

In this thesis the main concern is to present efficient integer multiplication algorithms to speed-up public key cryptography algorithms on cryptographic sizes. This thesis is organized as 4 chapters. In Chapter 2, an introduction of Public Key Cryptography, definitions of some important subjects about algorithms and complexity, and integer multiplication algorithms are mentioned. In Chapter 3, complexity analysis of integer multiplication algorithms is indicated. In Chapter 4, the conclusion of this thesis is given.

CHAPTER 2

PRELIMINARIES

In this chapter a brief introduction of public key cryptography is presented. RSA algorithm and elliptic curves which are used in public key cryptography, and their definitions are discussed. With the purpose of examining the cost of an algorithm some crucial concepts such as Big-O notation and the Master theorem are provided. Furthermore, a history of multiplication algorithms are presented in the Table 2.1 and their main properties are examined.

2.1 Public Key Cryptography

In this section a brief introduction of PKC is given. In symmetric key cryptography, one key is used for encryption and decryption. In PKC; however, there are two keys which are called as public key and private key. Although there exists a mathematical relation between public and private key, having knowledge of only public key is not enough to extract private keys.

An illustration can be given about how this algorithm works: Assume that Alice wants to send a secret message to Bob by using PKC algorithms. Firstly, Alice takes Bob's public key in order to encrypt the message and send it to Bob. He decrypts the encrypted message with his private key and gets the original message. In order to sign message Bob uses his private key, and Alice checks this signature by using Bob's public key.

2.1.1 RSA

RSA [24] algorithm was discovered by Ron Rivest, Adi Shamir, and Leonard Adleman in 1977 . The security of RSA algorithm depends on integer factorization problem. This problem can be described as follows: Given a composite integer P, find a non-trivial factor x of P.

Modular exponentiation is used during the encryption, and the decryption processes. In order to have an fast RSA system, taking exponentiation have to be done with efficient methods in the encryption and the decryption processes.

In following sections, key generation, encryption and decryption, and speed-up techniques of RSA are introduced.

2.1.1.1 Key Generation of RSA

As mentioned before in RSA algorithm, there are two keys namely public and private. To generate these keys following produce should be followed [20].

1) Choose two distinct odd primes p and q where $p \neq q$

2) Calculate $n = p \cdot q$

3) Calculate $\phi(n) = (p-1)(q-1)$

4) Select a random integer e where $1 < e < \phi(n)$ such that $gcd(e, \phi(n)) = 1$

5) Calculate the multiplicative inverse of e in mod $\phi(n)$ in other words find d such that $e \cdot d = 1 \mod \phi(n)$ and keep d as a private key

6) The private key is d and the public key is the pair (n, e) and . Note that e is referred to the public exponent and d is called private exponent.

2.1.1.2 Encryption of RSA

Alice would like to transmit a secret message to Bob. She should follow the steps below:

1) Download Bob's authentic public key pair (n, d)

2) Demonstrate the plaintext as an integer m in the interval $[0, n-1]$

3) Calculate $c = m^e \mod n$ where c is ciphertext

4) Send this ciphertext c to Bob

2.1.1.3 Decryption of RSA

Bob takes this ciphertext and he wants to understand the original message. For this reason:

1) He uses his private key d to obtain m where $m = c^d \mod n$

A proof why decryption works: As known from the key generation part $ed \equiv 1(\ mod\phi)$ there exists an integer k where $ed = 1 + k\phi$. From the Fermat's little theorem if $gcd(m, p) = 1$, then

$$m^{p-1} \equiv 1 \mod p$$

We take $k(q-1)$ power of of both sides

$$m^{k(q-1)(p-1)} \equiv 1 \mod p$$

and multiply both sides by m so that we get

$$m^{k(q-1)(p-1)} \equiv 1 \mod p$$

Otherwise, if $gcd(m,p) = p$, then the last congruence still holds since both sides are congruent to 0 in $\mod p$. Therefore, for both cases

$$m^{ed} \equiv 1 \mod p.$$

By the same logic,

$$m^{ed} \equiv 1 \mod q.$$

Since $p \neq q$ we get

$$m^{ed} \equiv 1 \mod n,$$

Therefore

$$c^d \equiv (m^e)^d \equiv m \mod n.$$

2.1.1.4 Techniques for Speeding-up RSA

In this section speed-up techniques for the RSA algorithm is represented. In encryption and decryption part, modular exponentiation is used. The complexity of direct computation is $d-1$ multiplications where d is the degree of exponentiation. On the other hand, we have repeated squaring algorithm. In this algorithm in average case $(\lceil \log d \rceil - 1)(\text{Squaring} + \frac{1}{2}\text{Multiplication})$ is needed. It can be concluded that Repeated Squaring algorithm is better than the direct computation since in Repeated Squaring Algorithm the number of multiplications are less than the ones in direct computation. Moreover, short public exponents for fast encryption and fast decryption with Chinese remainder theorem are two techniques to speed up RSA algorithm.

Theorem 2.1. *For the pairwise relatively prime integers $m_1, m_2, \cdots m_t$, the system of congruences*

$$\begin{aligned} k &\equiv a_1 \mod m_1 \\ k &\equiv a_2 \mod m_2 \\ &\vdots \\ k &\equiv a_n \mod m_t \end{aligned} \tag{2.1}$$

has a unique solution in $m \equiv \mod m_1 m_2 \cdots m_t$.

For the fast decryption with Chinese remainder Theorem 2.1, firstly we compute

$$\begin{aligned} x_p &\equiv x \mod p \\ x_q &\equiv x \mod q \end{aligned}$$

and then we calculate

$$y_p \equiv x_p^{d_p} \mod p$$
$$y_q \equiv x_q^{d_q} \mod q$$

where $d_p \equiv d \mod (p-1)$ and $d_q \equiv d \mod (q-1)$. By using Theorem 2.1

$$y \equiv [qc_p]y_p + [pc_q]y_q \mod n$$

2.1.2 Elliptic curves

Elliptic curve cryptography, a branch of public key cryptography, was suggested independently by Victor S. Miller [21] in 1985 and Neal Koblitz [18] in 1987. Elliptic curve cryptography is widely applied in banking and TLS 1.2 protocol. The security of elliptic curve cryptography is based on the discrete logarithm problem(DLP). Before we explain the DLP, a necessary definition is provided below:

Definition 2.1. An abelian group $(G, *)$ in which G is a set with a binary operation $* : GxG \to G$ satisfying the following conditions:

1) $x * (y * z) = (x * y) * z$,$\forall x, y, z \in G$

2) There exists an element $e \in G$ which is called as identity element such that $x * e = e * x = x$,$\forall x \in G$

3) For each $x \in G$, there exist an element $t \in G$,namely inverse of x, where $x * t = t * x = e$ s

4) $x * y = y * x$, $\forall x, y \in G$

Now, the DLP is stated in the following definition:

Definition 2.2. Discrete Logarithm Problem (DLP) in $\mathbb{Z}_p^*$ is for a given finite cyclic group which has a order $p-1$ and a primitive element $\alpha \in \mathbb{Z}_p^*$ and another element $\beta \in \mathbb{Z}_p^*$ to determine the integer x where $1 \leq x \leq p-1$ such that

$$\alpha^x \equiv \beta \mod p.$$

Definition 2.3. An elliptic curve E over a field K is defined by an equation

$$E : y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

where $a_1, a_2, a_3, a_4, a_6 \in K$ and $\Delta \neq 0$, where Δ is the discriminant of E and defined as follows:

$$\begin{aligned}
\Delta &= -d_2^2d_8 - 8d_4^3 - 27d_6^2 + 9d_2d_4d_6 \\
d_2 &= a_1^2 + 4a_2 \\
d_4 &= 2a_4 + a_1a_3 \\
d_6 &= a_3^2 + 4a_6 \\
d_8 &= a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2
\end{aligned}$$

In elliptic curve, point addition and point doubling defined in characteristic is not equal to 2 or 3 curves as follows:

1) Point addition: Let $P = (x_1, y_1) \in E(K)$ and $Q = (x_2, y_2) \in E(K)$, where $P \neq \pm Q$. Then $P + Q = (x_3, y_3)$, where

$$x_3 = \Big(\frac{y_2 - y_1}{x_2 - x_1}\Big)^2 - x_1 - x_2 \qquad y_3 = \Big(\frac{y_2 - y_1}{x_2 - x_1}\Big)(x_1 - x_3) - y_1$$

2) Point doubling: Let $P = (x_1, y_1) \in E(K)$, where $P \neq -P$. Then $2P = (x_3, y_3)$, where

$$x_3 = \Big(\frac{3x_1^2 + a}{2y_1}\Big)^2 - 2x_1 \qquad y_3 = \Big(\frac{3x_1^2 + a}{2y_1}\Big)(x_1 - x_3) - y_1$$

Point multiplications occupy a wide place in the execution times of elliptic curve cryptographic schemes [11].

2.2 Algorithms and Complexity

Informally, algorithm can be defined as a sequence of computational steps which convert input to output. Also it can be seen as a tool to solve a specified computational problem. There are various methods to solve problems. One of the widely used method is the divide-and-conquer. In this method, the problem is firstly divided into subproblems with the same properties as the original problem. After that, in conquer part, problems are solved by the idea of recursion. Finally, solutions of subproblems are combined for the solution of original problem.

The time complexity of the algorithm can be described as a function of the size of a problem in terms of the time needed by algorithm. The boundary behavior of the complexity as size grows is called the asymptotic time complexity.

Definition 2.4. $O(h(n)) = \{g(n)$: for positive constants k and n_0 where

$0 \leq g(n) \leq ch(n)$ for all $n \geq n_0$ $\}$ [7]

O-notation is used to offer an upper bound on a function.

If an algorithm operates inputs of size n in time kn^3 for a constant k, then the time complexity of that algorithm is $O(n^3)$.

Theorem 2.2 (Master Theorem)**.** *Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $M(n)$ be defined on the nonnegative integers by the recurrence*

$$T(n) = aT(n/b) + f(n) \tag{2.2}$$

where n/b is either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $M(n)$ has following asymptotic bounds.

1) If $f(n) = O(n^{\log_b a - \epsilon})$ *for some constant* $\epsilon > 0$*, then* $M(n) = \Theta(n^{\log_b a})$

2) If $f(n) = \Theta(n^{\log_b a})$*, then* $M(n) = \Theta(n^{\log_b a} \log_b a)$

3) If $f(n) = O(n^{\log_b a + \epsilon})$ *for some constant* $\epsilon > 0$*, and if* $af(n/b) \leq cf(n)$ *for some constant* $c < 1$ *and all sufficiently large n, then* $M(n) = \Theta(f(n))$ *[7]*

To compute for a given non-recursive solution to inductive expression in order to evaluate complexities.

Lemma 2.3. *Let a,b and i be positive integers. Let* $n = b^i$ *and* $a = b$ *and* $a \neq 1$*. The solution of inductive relation*

$r_1 = e$

$r_n = ar_{n/b} + cn + d$

$$r_n = \Big(e + \frac{bc}{a-b} + \frac{d}{a-1}\Big) n^{log_b(a)} - \Big(\frac{bc}{a-b}\Big) n - \frac{d}{a-1} \text{ [5]}$$

2.3 Multiplication Algorithms

In this section , we will put an emphasize on fast multiplication algorithms for polynomials. Until 1960s, complexity of multiplication of two polynomials was $O(n^2)$. However, Karatsuba and Ofman [14] discovered a new multiplication algorithm where the complexity is $O(n^{1.58})$ in 1963. With this invention, a variety of scientific analysis have been done in the literature. The following table is taken from [12] displays a brief history of studies done regarding on this subject.

Table 2.1: Historical overview of integer multiplication algorithms

Date	Algorithm	Time Complexity
$<$ 3000 BC	Schoolbook	$O(n^2)$
1962	Karatsuba	$O(n^{\log_2 3})$
1963	Toom	$O(n2^{5\sqrt{\log_2 n}})$
1966	Schöngage	$O(n2^{\sqrt{2\log_2 n}}(logn)^{\frac{3}{2}})$
1969	Knuth	$O(n2^{\sqrt{2\log_2 n}}(logn)$
1971	Schönage-Strassen	$O(nlognloglogn)$
2007	Fürer	$O(nlogn2^{O(log^* n)})$
2014	Harvey	$O(nlogn8^{log^* n})$

2.3.1 Schoolbook

Schoolbook algorithm is a classical way to multiply two polynomials. Consider two polynomials $A(x)$ and $B(x)$ of degree $n-1$:

$$A(x) = \sum_{i=0}^{n-1} a_i x^i, B(x) = \sum_{i=0}^{n-1} b_i x^i \tag{2.3}$$

We can calculate the product $C(x) = A(x)B(x)$ as

$$C(x) = \sum_{i=0}^{n-1} x^i \left(\sum_{j+k=i}^{n-1} a_j b_k \right) \tag{2.4}$$

In the schoolbook algorithm we have n^2 multiplications and $(n-1)^2$ additions. The complexity of this algorithm is $O(n^2)$.

2.3.2 Karatsuba

Famous Russian mathematician A.A. Kolmogorov stated that the cost of multiplying two polynomials of degree $n-1$ was $O(n^2)$ in 1956 [15]. On the other hand, a new algorithm with better complexity was discovered by A. A. Karatsuba in 1963 [14]. Karatsuba algorithm yields the complexity $O(n^{\log_2 3})$ where $\log_2 3 = 1.58$. Therefore, this algorithm gives better results than the Schoolbook algorithm. The following algorithm is taken from [28].

Input: $A, B \in R[x]$ of degrees less than n, where R is a commutative ring with unity 1 and n a power of 2
Output: $A \cdot B \in R[x]$
1) If n = 1 then return $A \cdot B \in R$
2) Suppose $A = A_1 x^{\frac{n}{2}} + A_0$ and $B = B_1 x^{\frac{n}{2}} + B_0$, with $A_0, A_1, B_0, B_1 \in R[x]$ of degrees less than $\frac{n}{2}$
3) Compute A_0B_0, A_1B_1 and $(A_0 + A_1)(B_0 + B_1)$ by a recursive call
4) Return $A_1B_1x^n + ((A_0 + A_1)(B_0 + B_1) - A_0B_0 - A_1B_1)x^{\frac{n}{2}} + A_0B_0$

2.3.3 Fast Fourier Transform and Interpolation methods

Fast Fourier transform is used for many mathematical applications. In this chapter, only one application, namely the multiplication of integers and polynomials is discussed. Schöngage [25], Knuth [17], Schöngage- Strassen [26], Fürer [10], and Harvey's [12] algorithms depend on FFT transform.

However, as Bernstein et al. mentioned in software implementations of cryptography it is hard to find integers large enough to apply FFT or Toom, and even Karatsuba's method is commonly used in implementations of RSA in their paper [3].

In interpolation method, $a(x) = a_0 + \cdots + a_{d-1}x^{d-1}$ and $b(x) = b_1 + \cdots b_{d-1}x^{d-1}$ are defined. We want to compute

$$c(x) = a(x)b(x) = c_0 + \cdots + c_{2d-2}x^{2d-2}.$$

For the computation of c_i's evaluation of $a(x)b(x) = c(x)$ at 2d-1 points and solving the system of linear equation for c_i's.

$$\begin{array}{lll} x = w_0 & \rightarrow a(w_0)b(w_0) & = c_0 + c_1w_0 + \cdots + c_{2d-2}w_0^{2d-2} \\ x = w_1 & \rightarrow a(w_1)b(w_1) & = c_0 + c_1w_1 + \cdots + c_{2d-2}w_1^{2d-2} \\ & & \vdots \\ x = w_{2d-2} & \rightarrow a(w_{2d-2})b(w_{2d-2}) & = c_0 + c_1w_{2d-2} + \cdots + c_{2d-2}w_{2d-2}^{2d-2} \end{array}$$

Toom-Cook [27] [6] is a method to multiply integers by using interpolation methods.

For this method $u, v \in R$. The product $u \cdot v = w \in R$ is computed by five steps. In splitting step, a proper basis $B \in R$ is fixed and the two operands are presented by two polynomials in $a, b \in R[x]$ of degree $d = n - 1$ as

$$a(x) = \sum_{i=0}^{d} a_i x^i \quad \text{and} \quad b(x) = \sum_{i=0}^{d} b_i x^i$$

such as $u = a(x)|_{x=B}$ and $u = b(x)|_{x=B}$. One can assume that the factors gave equal degrees, otherwise, pad the lower-degree one with zero cofficients. For evaluation step, choose $2n - 1$ values $v_i \in R$ and evaluate both operand on all of them, gaining $a(v_i), b(v_i)$. In recursion step, compute $w_i = a(v_i) \cdot b(v_i)$ recursively. Let $w = (w_i)$ be obtained values vector. In interpolation step, solve the interpolation problem $c(v_i) = w_i$ inverting the Vandermonde matrix A_n produced by the v_i values and evaluating $c = A_n^{-1}w$, where $c = (c_i)$ is the vector of $c(x)$ coefficients. Lastly for recomposition part, as soon as all the coefficient are computed, it's enough to evaluate back $w = c(x)|_{x=B}$. [4]

CHAPTER 3

SCHOOLBOOK, KARATSUBA AND VARIANTS, MONTGOMERY, AND HYBRID ALGORITHMS

In this chapter the schoolbook, Karatsuba [14], the refined Karatsuba [2], [3], Hybrid algorithms [5], and Montgomery's algorithms [22] for multiplication of polynomials are introduced. For each of these algorithms, time complexities are computed and they are compared in a detailed table.

3.1 2-way Multiplication Algorithms

In this section, 2-way multiplication algorithms are discussed. There exist four 2-way multiplication algorithms the Schoolbook,the Karatsuba,the refined Karatsuba and the optimized Karatsuba.

3.1.1 School Book 2-way Multiplication Algorithm

The schoolbook algorithm is also known as the naive algorithm in the literature. For this multiplication algorithm, suppose that $A(x)$ and $B(x)$ are two polynomials with degree $n-1$ such that

$$\begin{aligned} A(x) &= a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1}, \\ B(x) &= b_0 + b_1x + b_2x^2 + \cdots + b_{n-1}x^{n-1}. \end{aligned} \tag{3.1}$$

The schoolbook 2-way algorithm is used recursively as follows: $A(x)$ and $B(x)$ are divided into 2 parts. $A(x) = A_0 + A_1y$ where

$$\begin{aligned} A_0 &= a_0 + a_1x + \cdots + a_{\frac{n}{2}-1}x^{\frac{n}{2}-1}, \\ A_1 &= a_{\frac{n}{2}} + a_{\frac{n}{2}+1}x + \cdots + a_{n-1}x^{\frac{n}{2}-1}. \end{aligned} \tag{3.2}$$

and $y = x^{\frac{n}{2}}$. By the same argument, $B(x) = B_0 + B_1y$ where

$$\begin{aligned} B_0 &= b_0 + b_1 x + \cdots + b_{\frac{n}{2}-1} x^{\frac{n}{2}-1} \\ B_1 &= b_{\frac{n}{2}} + b_{\frac{n}{2}+1} x + \cdots + b_{n-1} x^{\frac{n}{2}-1} \end{aligned} \tag{3.3}$$

Then, the multiplication of $A(x)$ and $B(x)$ is expressed as:

$$A(x)B(x) = A_0B_0 + y[A_1B_0 + A_0B_1] + y^2 A_1B_1.$$

The number of multiplication and addition should be counted to obtain the cost of the Schoolbook 2-way algorithm. Note that there are four multiplications of size $\frac{n}{2}$, $A_0B_0, A_1B_0, A_0B_1,$ and A_1B_1.

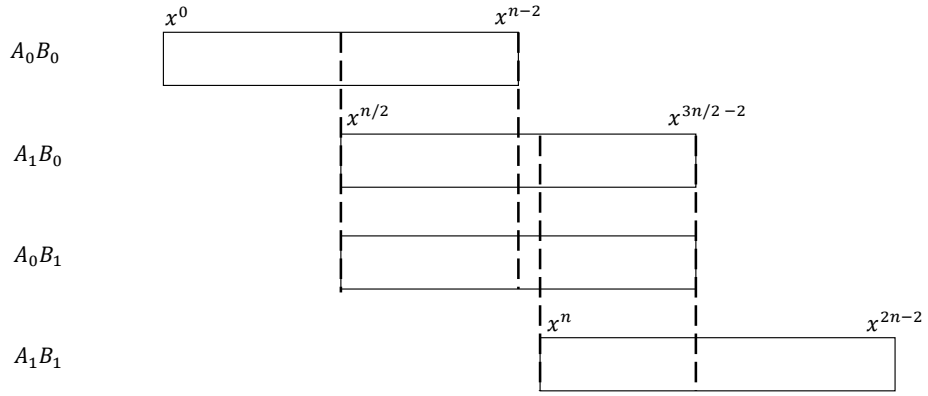

Figure 3.1: School Book 2-way multiplication algorithm

In the computation of construction, as can be seen from the Figure 3.1, there exist three overlaps between A_0B_0 and A_1B_0, A_1B_0 and A_0B_1, and A_0B_1 and A_1B_1.

Note that the half of A_0B_0 is added to the half of the A_1B_0, and this requires $\frac{n}{2} - 1$ additions. Similarly,this situation is also valid for A_1B_0 and A_0B_1, A_0B_1 and A_1B_1.

Therefore, we have totally $2(\frac{n}{2} - 1) + n - 1 = 2n - 3$ additions. As a result we obtain

$$M(n) = 4M(\frac{n}{2}) + 2n - 3 \text{ and } M(1) = 1$$

.

Assume that $n = 2^k$. Then, we write

$$
\begin{aligned}
M(n) &= 4M\left(\tfrac{n}{2}\right) + 2n - 3 \\
&= 4[4M(\tfrac{n}{4}) + 2(\tfrac{n}{2}) - 3] + 2n - 3 \\
&= 4^2 M(\tfrac{n}{4}) + 4 \cdot 2(\tfrac{n}{2}) + 2n - 3 \cdot 4 - 3 \\
&= 4^2[M(\tfrac{n}{8}) + 2(\tfrac{n}{4}) - 3]) + 4 \cdot 2(\tfrac{n}{2}) + 2n - 3 \cdot 4 - 3 \\
&= 4^3(\tfrac{n}{8}) + 4^2 \cdot 2(\tfrac{n}{4}) - 3 \cdot 4^2 + 4 \cdot 2(\tfrac{n}{2}) - 3 \cdot 4 + 2n - 3 \\
&= 4^3(\tfrac{n}{8}) + 2n[4^2(\tfrac{1}{2^2}) + 4(\tfrac{1}{2}) + 4^0] - 3[4^2 + 4^1 + 4^0] \\
&\qquad\qquad \vdots \\
&= 4^k M(\tfrac{2^k}{2^k}) + 2^k \cdot 2[(\tfrac{4^{k-1}}{2^{k-1}}) + (\tfrac{4^{k-2}}{2^{k-2}}) + \cdots + (\tfrac{4^0}{2^0})]) \\
&\quad -3[4^{k-1} + 4^{k-2} + \cdots + 4^0] \\
&= 4^k M(1) + 2 \cdot 2^k[2^{k-1} + 2^{k-2} + \cdots + 2^0] - 3[4^{k-1} + 4^{k-2} + \cdots + 4^0] \\
&= 4^k M(1) - 2 \cdot 2^k \cdot (\tfrac{2^k - 1}{2-1}) - 3 \cdot (\tfrac{4^k - 1}{4-1}) \\
&= 4^k + 2 \cdot 2^k(2^k - 1) - (4^k - 1) = n^2 + 2n(n-1) - (n^2 - 1) \\
&= n^2 + 2n^2 - 2n - n^2 + 1 \\
&= 2n^2 - 2n + 1.
\end{aligned}
$$

$2n^2 - 2n + 1 \in O(n^2)$, the complexity of School book 2-way algorithm is $O(n^2)$. We can also compute multiplication and addition complexities separately.

$$
\begin{cases}
M(n) & = 4M(\tfrac{n}{2}) + 2n - 3 \text{ and } M(1) = 1 \\
M_{\otimes}(n) & = 4M(\tfrac{n}{2}) \text{ and } M_{\otimes}(1) = 1 \\
M_{\oplus}(n) & = 4M(\tfrac{n}{2}) + 2n - 3 \text{ and } M_{\oplus}(1) = 0.
\end{cases}
$$

By using Lemma 2.3, we can calculate complexities explicitly:

$$
\begin{cases}
M(n) & = 2n^2 - 2n + 1 \\
M_{\otimes}(n) & = n^2 \\
M_{\oplus}(n) & = n^2 - 2n + 1
\end{cases}
$$

3.1.2 Karatsuba 2-way Multiplication Algorithm

For the Karatsuba 2-way multiplication algorithm [14], let $A(x)$ and $B(x)$ be the polynomials as defined in Equation 3.1

To calculate the multiplication $A(x)B(x)$ using Karatsuba 2-way algorithm, we divide the polynomials $A(x)$ and $B(x)$ into 2 parts:

$A(x) = A_0 + A_1 y$ and $B(x) = B_0 + B_1 y$

as defined in Equation 3.2 and Equation 3.3.

Then, the multiplication of $A(x)$ and $B(x)$ is demonstrated as:

$$A(x)B(x) \quad = A_0B_0 + y[(A_0 + A_1)(B_0 + B_1) - A_0B_0 - A_1B_1] + A_1B_1y^2$$

The number of multiplication and addition should be counted to obtain the cost of the Karatsuba 2-way algorithm. Indicate that there are three multiplications of size $\frac{n}{2}, A_0B_0, (A_0 + A_1)(B_0 + B_1),$ and A_1B_1 and there are two additions of size $\frac{n}{2}, A_0 + A_1$ and $B_0 + B_1$.

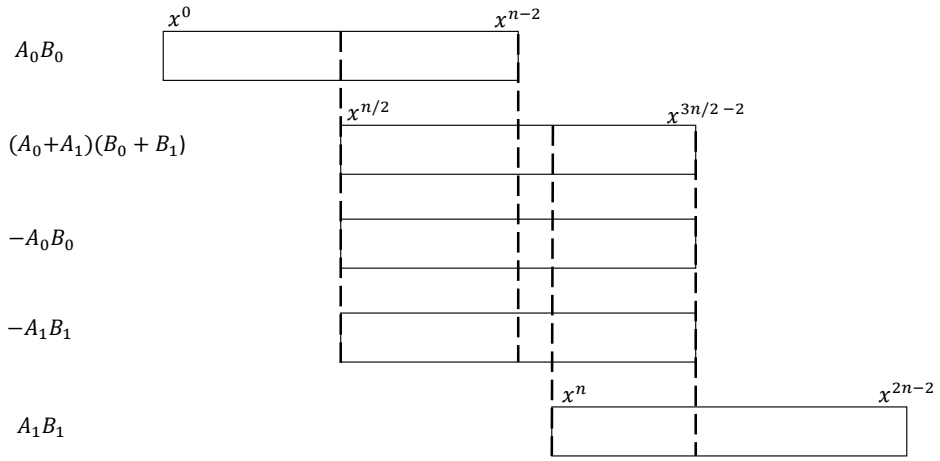

Figure 3.2: Karatsuba 2-way multiplication algorithm

In the computation of construction, as one can be see from the Figure 3.2, there exist four overlaps between A_0B_0 and $(A_0 + A_1)(B_0 + B_1)$, $(A_0 + A_1)(B_0 + B_1)$ and A_0B_0, A_0B_0 and A_1B_1, and A_1B_1 and A_1B_1.

Note that the half of A_0B_0 is added to the half of the $(A_0 + A_1)(B_0 + B_1)$, and this requires $\frac{n}{2} - 1$ additions. Similarly,this situation is also valid for A_1B_1 and A_1B_1. On the other hand,$(A_0 + A_1)(B_0 + B_1)$ is added to the A_0B_0, and this requires $n - 1$ additions. Likewise, this case is also valid for A_0B_0 and A_1B_1.

In construction part we get $3n-4$ additions, in addition part we have n addition, totally we get $4n - 4$ additions. Consequently, we get:

$$M(n) = 3M(\frac{n}{2}) + 4n - 4 \text{ and } M(1) = 1$$

By using Lemma 2.3

$$
\begin{aligned}
M(n) &= 3M(\tfrac{n}{2}) + 4n - 4 \text{ and } M(1) = 1 \\
e &= 1 \\
a &= 3 \\
b &= 2 \\
c &= 4 \\
d &= -4 \\
r_n &= (1 + \tfrac{8}{3-2} - \tfrac{4}{2})n^{log_2 3} - (\tfrac{8}{3-2})n - \tfrac{-4}{3-1} \\
&= 7log_2 3 - 8n + 2
\end{aligned}
$$

$7n^{log_2 3} - 8n + 2 \in O(n^{log_2 3}) = O(n^{1.58})$, the complexity of Karatsuba 2-way algorithm is $O(n^{1.58})$

$$
\begin{cases}
M(n) &= 3M(\tfrac{n}{2}) + 4n - 4 \text{ and } M(1) = 1 \\
M_{\otimes}(n) &= 3M(\tfrac{n}{2}) \text{ and } M_{\otimes}(1) = 1 \\
M_{\oplus}(n) &= 3M(\tfrac{n}{2}) + 4n - 4 \text{ and } M_{\oplus}(1) = 0.
\end{cases}
$$

By using Lemma 2.3, we can calculate complexities explicitly:

$$
\begin{cases}
M(n) &= 7n^{log_2 3} - 8n + 2 \\
M_{\otimes}(n) &= n^{log_2 3} \\
M_{\oplus}(n) &= 6n^{log_2 3} - 8n + 2
\end{cases}
$$

3.1.3 Refined Karatsuba 2-way Multiplication Algorithm

For the Refined Karatsuba 2-way multiplication algorithm [2], suppose $A(x)$ and $B(x)$ are the polynomials as defined in Equation 3.1.

To calculate $A(x)B(x)$ by using Refined Karatsuba 2-way algorithm, $A(x)$ and $B(x)$ are divided into 2 parts:

$A(x) = A_0 + A_1 y$ and $B(x) = B_0 + B_1 y$

as defined in Equation 3.2 and Equation 3.3.

$P_1 = A_0 B_0$

$P_2 = (A_0 + A_1)(B_0 + B_1)$

$P_3 = A_1 B_1$

Then the multiplication of $A(x)B(x)$ can be shown as:

$$A(x)B(x) = (x^{\frac{n}{2}} - 1)(x^{\frac{n}{2}} P_3 - P_1) + P_2 x^{\frac{n}{2}}$$

The number of multiplication and addition should be counted to obtain the cost of the Refined Karatsuba 2-way algorithm. Note that there are three multiplications of size $\frac{n}{2}, A_0B_0, (A_0 + A_1)(B_0 + B_1),$ and A_1B_1 and there are two additions of size $\frac{n}{2}, A_0 + A_1$ and $B_0 + B_1$.

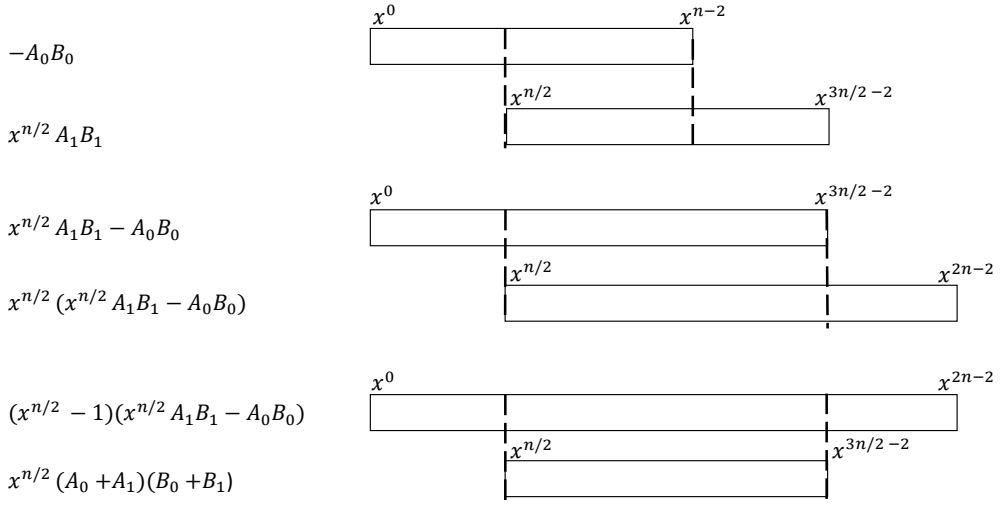

Figure 3.3: Refined Karatsuba 2-way multiplication algorithm

M_i's, where $1 \leq i \leq 3$, are defined as:

$$\begin{aligned} M_1 &= x^{\frac{n}{2}} P_3 - P_1 \\ M_2 &= x^{\frac{n}{2}} M_1 - M_1 \\ M_3 &= M_2 + P_2. \end{aligned}$$

In construction part, for M_1 we get $\frac{n}{2} - 1$ additions. For M_2 and M_3 we get $n - 1$ additions.

Note that, in construction part we have $\frac{n}{2} - 1 + 2(n - 1) = \frac{5n}{2}$ additions, and in addition part we have n addition, therefore, totally we get $\frac{7n}{2} - 3$ additions. As a result we obtain,

$$M(n) = 3M(\frac{n}{2}) + \frac{7n}{2} - 3 \text{ and } M(1) = 1.$$

By using Lemma 2.3

$$
\begin{aligned}
M(n) &= 3M(\tfrac{n}{2}) + \tfrac{7n}{2} - 3 \text{ and } M(1) = 1 \\
e &= 1 \\
a &= 3 \\
b &= 2 \\
c &= \tfrac{7}{2} \\
d &= -3 \\
r_n &= (1 + \tfrac{7}{3-2} - \tfrac{3}{2})n^{log_2 3} - (\tfrac{7}{3-2})n - \tfrac{-3}{3-1} \\
&= 6.5log_2 3 - 7n + \tfrac{3}{2}
\end{aligned}
$$

$O(6.5n^{log_2 3} - 7n + \frac{3}{2}) \in O(n^{log_2 3}) = O(n^{1.58})$, the complexity of this algorithm is $O(n^{1.58})$

$$
\begin{cases}
M(n) &= 3M(\tfrac{n}{2}) + \tfrac{7n}{2} - 3 \text{ and } M(1) = 1 \\
M_{\otimes}(n) &= 3M(\tfrac{n}{2}) \text{ and } M_{\otimes}(1) = 1 \\
M_{\oplus}(n) &= 3M(\tfrac{n}{2}) + \tfrac{7n}{2} - 3 \text{ and } M_{\oplus}(1) = 0.
\end{cases}
$$

By using Lemma 2.3, we can calculate complexities explicitly:

$$
\begin{cases}
M(n) &= 6.5n^{log_2 3} - 7n + \tfrac{3}{2} \\
M_{\otimes}(n) &= n^{log_2 3} \\
M_{\oplus}(n) &= 5.5n^{log_2 3} - 7n + \tfrac{3}{2}.
\end{cases}
$$

3.1.4 Optimized Karatsuba 2-way Multiplication Algorithm

In Optimized Karatsuba 2-way multiplication algorithm [31] suppose $A(x)$ and $B(x)$ are the polynomials as defined in Equation 3.1.

Multiplication of $A(x)B(x)$ is calculated by using Optimized Karatsuba 2-way algorithm. In order to use this algorithm the polynomials $A(x)$ and $B(x)$ are divided into 2 parts:

$A(x) = A_0 + A_1 y$ and $B(x) = B_0 + B_1 y$

as defined in Equation 3.2 and Equation 3.3.

$P_1 = A_0 B_0$

$P_2 = (A_0 + A_1)(B_0 + B_1)$

$P_3 = A_1 B_1$

and P_{iL} where $i = 1, 2, 3$ is the lower part of P_i and in lower parts there exist $\frac{n}{2}$ terms. P_{iH} where $i = 1, 2, 3$ is the higher part of P_i, in which the number of terms is $\frac{n}{2} - 1$. $\frac{n}{2}$

additions are required for the sum $P_{iL} + P_{jL}$. On the other hand, $\frac{n}{2} - 1$. additions are needed for the sums $P_{iL} + P_{jH}$ and $P_{iH} + P_{jH}$.

Therefore multiplication of $A(x)B(x)$ is expressed as:

$$\begin{aligned} A(x)B(x) &= P_{1L} + x^{\frac{n}{2}}[P_{1H} + P_{2L} - P_{1L} - P_{3L}] \\ &\quad + x^n[P_{2H} - P_{1H} - P_{3H} + P_{3L}] + x^{\frac{3n}{2}}P_{3H} \\ &= P_{1L} + x^{\frac{n}{2}}[(P_{1H} - P_{3L}) + P_{2L} - P_{1L}] \\ &\quad + x^n[-(P_{1H} - P_{3L}) + P_{2H} - P_{3H}] + x^{\frac{3n}{2}}P_{3H} \end{aligned}$$

The number of multiplication and addition should be counted to obtain the cost of the Refined Karatsuba 2-way algorithm. Note that there are three multiplications of size $\frac{n}{2}, A_0B_0, (A_0 + A_1)(B_0 + B_1),$ and A_1B_1 and there are two additions of size $\frac{n}{2}, A_0 + A_1$ and $B_0 + B_1$.

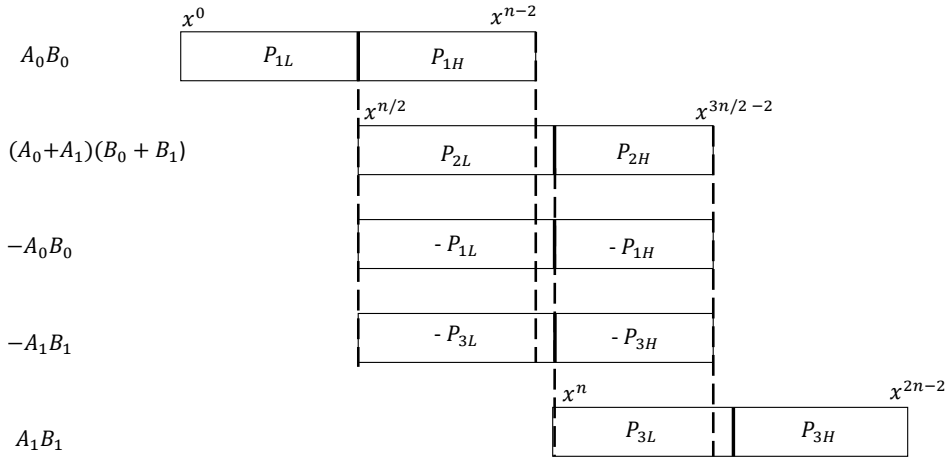

Figure 3.4: Optimized Karatsuba 2-way multiplication algorithm

In construction part, M_i's, where $1 \leq i \leq 5$, are defined as

$$\begin{aligned} M_1 &:= P_{1H} - P_{3L} \rightarrow \tfrac{n}{2} - 1 \text{ additions} \\ M_2 &:= M_1 + P_{2L} \rightarrow \tfrac{n}{2} \text{ additions} \\ M_3 &:= M_2 - P_{1L} \rightarrow \tfrac{n}{2} \text{ additions} \\ M_4 &:= -M_1 + P_{2H} \rightarrow \tfrac{n}{2} - 1 \text{ additions} \\ M_5 &:= M_4 + P_{3H} \rightarrow \tfrac{n}{2} - 1 \text{ additions} \end{aligned}$$

$$A(x)B(x) = P_{1L} + x^{\frac{n}{2}}M_3 + x^nM_5 + x^{\frac{3n}{2}}P_{3H}$$

.

In construction part, for M_1 we get $\frac{n}{2} - 1$ additions. Similarly, it's valid for M_4 and M_5. For M_2 and M_3, we get $\frac{n}{2}$ additions.

In construction part we have $\frac{n}{2} - 1 + 2(n-1) = \frac{5n}{2}$ additions, and in addition part we have n additions, therefore, totally we get $\frac{7n}{2} - 3$ additions.As a result we obtain,

$$M(n) = 3M(\frac{n}{2}) + \frac{7n}{2} - 3 \text{ and } M(1) = 1.$$

As computed in Section 3.2.3 the complexity of Optimized Karatsuba 2-way multiplication algorithm is

$O(6.5n^{log_2 3} - 7n + \frac{3}{2}) \in O(n^{log_2 3}) = O(n^{1.58})$, the complexity of Optimized Karatsuba 2-way algorithm is $O(n^{1.58})$

$$\begin{cases} M(n) & = 3M(\frac{n}{2}) + \frac{7n}{2} - 3 \text{ and } M(1) = 1 \\ M_{\otimes}(n) & = 3M(\frac{n}{2}) \text{ and } M_{\otimes}(1) = 1 \\ M_{\oplus}(n) & = 3M(\frac{n}{2}) + \frac{7n}{2} - 3 \text{ and } M_{\oplus}(1) = 0. \end{cases}$$

By using Lemma 2.3, we can calculate complexities explicitly:

$$\begin{cases} M(n) & = 6.5n^{log_2 3} - 7n + \frac{3}{2} \\ M_{\otimes}(n) & = n^{log_2 3} \\ M_{\oplus}(n) & = 5.5n^{log_2 3} - 7n + \frac{3}{2}. \end{cases}$$

3.2 3-way Multiplication Algorithms

In this section 3-way multiplication algorithms are discussed. There exist three 3-way multiplication algorithms namely Schoolbook, Karatsuba and Optimized Karatsuba.

3.2.1 Schoolbook 3-way Multiplication Algorithm

In Schoolbook 3-way multiplication algorithm suppose $A(x)$ and $B(x)$ are the polynomials as defined in Equation 3.1

To calculate $A(x)B(x)$ by using School Book 3-way algorithm. The polynomials $A(x)$ and $B(x)$ are divided into 3 parts

$A(x) = A_0 + A_1 y + A_2 y^2$ where

$$
\begin{aligned}
A_0 &= a_0 + a_1 x + \cdots + a_{\frac{n}{3}-1} x^{\frac{n}{3}-1} \\
A_1 &= a_{\frac{n}{3}} + a_{\frac{n}{3}+1} x + \cdots + a_{n-1} x^{\frac{n}{3}-1} \\
A_2 &= a_{\frac{2n}{3}} + a_{\frac{2n}{3}+1} x + \cdots + a_{n-1} x^{\frac{3n}{3}-1}
\end{aligned} \tag{3.4}
$$

and $y = x^{\frac{n}{3}}$. $B(x) = B_0 + B_1 y + B_2 y^2$ where

$$
\begin{aligned}
B_0 &= b_0 + b_1 x + \cdots + b_{\frac{n}{3}-1} x^{\frac{n}{3}-1} \\
B_1 &= b_{\frac{n}{3}} + b_{\frac{n}{3}+1} x + \cdots + b_{n-1} x^{\frac{n}{3}-1} \\
B_2 &= b_{\frac{2n}{3}} + b_{\frac{2n}{3}+1} x + \cdots + b_{n-1} x^{\frac{3n}{3}-1}
\end{aligned} \tag{3.5}
$$

Then, the multiplication of $A(x)$ and $B(x)$ is expressed as follows:

$$
\begin{aligned}
A(x)B(x) &= A_0B_0 + y[A_1B_0 + A_0B_1] + y^2(A_0B_2 + A_1B_1 + A_2B_0) \\
&\quad + y^3(A_1B_2 + A_2B_1) + y^4 A_2B_2
\end{aligned}
$$

The number of multiplication and addition should be counted to obtain the cost of the Schoobook 3-way algorithm. Note that there are nine multiplications of size $A_0B_0, A_1B_0, A_0B_1, A_1B_1, A_0B_2, A_2B_0, A_2B_1, A_1B_2,$ and A_2B_2

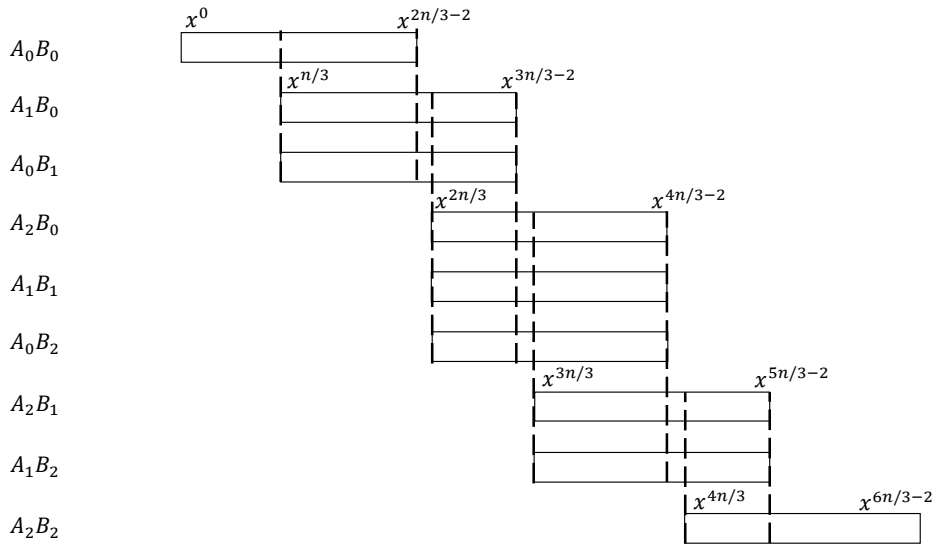

Figure 3.5: School Book 3-way multiplication algorithm

In the computation of construction, as can be seen in the Figure 3.5, three exists eight overlaps between A_0B_0 and A_1B_0, A_1B_0 and A_0B_1, A_0B_1 and A_2B_0, A_2B_0 and A_1B_1,A_1B_1 and A_0B_2, A_0B_2 and A_2B_1, A_2B_1 and A_1B_2, A_1B_2 and A_2B_2.

Indicate that the half of A_0B_0 is added to the half of the A_1B_0, and this requires $\frac{n}{3} - 1$ additions. This case is also valid for A_0B_1 and A_2B_0, A_0B_2 and A_2B_1, and A_1B_2 and

A_2B_2. On the other hand, the addition cost for A_1B_0 and A_0B_1 is $\frac{2n}{3} - 1$. Likewise, this case is also valid for A_2B_0 and A_1B_1, A_1B_1 and A_0B_2, and A_2B_1 and A_1B_2.

In total, we have $4(\frac{2n}{3} - 1) + 4(\frac{n}{3} - 1) = 3n - 8$ additions. Therefore, we get:

$$M(n) = 9M(\tfrac{n}{3}) + 3n - 8$$

By using Lemma 2.3

$$\begin{array}{ll} M(n) & = 9M(\frac{n}{3}) + 3n - 8 \text{ and } M(1) = 1 \\ e & = 1 \\ a & = 9 \\ b & = 3 \\ c & = 3 \\ d & = -8 \\ r_n & = (1 + \frac{9}{9-3} - \frac{8}{9-1})n^{log_3 9} - (\frac{9}{6})n - \frac{-8}{9-1} \\ & = 3n^2 - \frac{3}{2}n + 1 \end{array}$$

$3n^2 - \frac{3}{2}n + 1 \in O(n^2)$, the complexity of School Book 3-way algorithm is $O(n^2)$

$$\begin{cases} M(n) & = 9M(\frac{n}{3}) + 3n - 8 \text{ and } M(1) = 1 \\ M_{\otimes}(n) & = 9M(\frac{n}{3}) \text{ and } M_{\otimes}(1) = 1 \\ M_{\oplus}(n) & = 9M(\frac{n}{3}) + 3n - 8 \text{ and } M_{\oplus}(1) = 0. \end{cases}$$

By using Lemma 2.3, we can calculate complexities explicitly:

$$\begin{cases} M(n) & = 3n^2 - \frac{3}{2}n + 1 \\ M_{\otimes}(n) & = n^2 \\ M_{\oplus}(n) & = 2n^2 - \frac{3}{2}n + 1. \end{cases}$$

3.2.2 Karatsuba 3-way Multiplication Algorithm

For the Karatsuba 3-way multiplication algorithm, assume that $A(x)$ and $B(x)$ are the two polynomials as given in Equation 3.1.

The multiplication of $A(x)B(x)$ is calculated by using Karatsuba 3-way algorithm. In order to use Karatsuba 3-way algorithm we divide $A(x)$ and $B(x)$ polynomials into 3 parts

$$A(x) = A_0 + A_1y + A_2y^2 \text{ and } B(x) = B_0 + B_1y + B_2y^2$$

as defined in equations 3.4 and 3.5

Then, the multiplication of $A(x)$ and $B(x)$ is expressed as follows:

$$\begin{aligned} A(x)B(x) &= A_0B_0 + y[(A_0 + A_1)(B_0 + B_1) - A_0B_0 - A_1B_1] \\ &\quad + y^2((A_0 + A_2)(B_0 + B_2) - A_0B_0 - A_2B_2 + A_1B_1) \\ &\quad + y^3((A_1 + A_2)(B_1 + B_2) - A_1B_1 - A_2B_2) + y^4A_2B_2 \end{aligned}$$

The number of multiplication and addition should be counted to get the cost of the Karatsuba 3-way algorithm. Indicate that there are six multiplications of size $\frac{n}{3}$, A_0B_0, $(A_0+A_1)(B_0+B_1)$, A_1B_1, $(A_0+A_2)(B_0B_2)$, A_2B_2 and $(A_1+A_2)(B_1+B_2)$ and there are six additions of size $\frac{n}{3}$, A_0+A_1, B_0+B_1, A_0+A_2, B_0+B_2, A_1+A_2 and B_1+B_2.

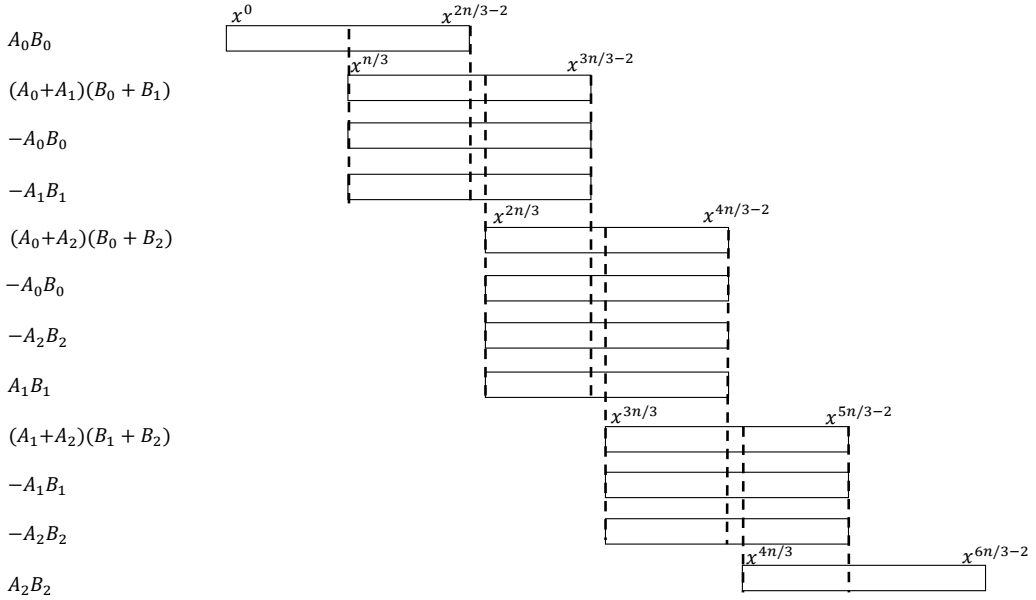

Figure 3.6: Karatsuba 3-way multiplication algorithm

In the computation of construction, as one can see in the figure 3.6, 11 overlaps between A_0B_0 and $(A_0+A_1)(B_0+B_1)$, $(A_0+A_1)(B_0+B_1)$ and A_0B_0, A_0B_0 and A_1B_1, A_1B_1 and $(A_0+A_2)(B_0+B_2)$, $(A_0+A_2)(B_0+B_2)$ and A_0B_0, A_0B_0 and A_2B_2, A_2B_2 and A_1B_1, A_1B_1 and $(A_1 + A_2)(B_1 + B_2)$, $(A_1 + A_2)(B_1 + B_2)$ and A_2B_2, A_2B_2 and A_1B_1, A_1B_1 and A_1B_1 and A_2B_2 .

The half of A_0B_0 is added to the half of the $(A_0 + A_1)(B_0 + B_1)$, and this requires $\frac{n}{3} - 1$ additions. This case is also valid for A_1B_1 and $(A_0 + A_2)(B_0 + B_2)$, A_1B_1 and $(A_1 + A_2)(B_1 + B_2)$, and A_1B_1 and A_2B_2. However, the addition cost for $(A_0 + A_1)(B_0+B_1)$ and A_0B_0 is $\frac{2n}{3}-1$. Likewise, this case is also valid for A_0B_0 and A_1B_1, $(A_0+A_2)(B_0+B_2)$ and A_0B_0, A_0B_0 and A_2B_2, A_2B_2 and A_1B_1, $(A_1+A_2)(B_1+B_2)$ and A_2B_2, and A_2B_2 and A_1B_1.

In construction part we get $7(\frac{2n}{3} - 1) + 4(\frac{n}{3} - 1) = 6n - 11$ additions, in addition part we have $6(\frac{n}{3}) = 2n$ additions. Therefore, we get $\frac{7n}{2} - 3$ additions.

$$M(n) = 6M(\frac{n}{3}) + 8n - 11 \text{ and } M(1) = 1$$

By using Lemma 2.3,

$$\begin{array}{ll} M(n) & = 6M(\frac{n}{3}) + 8n - 11 \text{ and } M(1) = 1 \\ e & = 1 \\ a & = 6 \\ b & = 3 \\ c & = 8 \\ d & = -11 \\ r_n & = (1 + \frac{24}{6-3} - \frac{11}{6-1})n^{log_3 6} - (\frac{24}{3})n - \frac{-11}{6-1} \\ & = \frac{34}{5}n^{log_3 6} - 8n + \frac{11}{5} \end{array}$$

$\frac{34}{5}n^{log_3 6} - 8n + \frac{11}{5} \in O(n^{log_3 6})$, the complexity of Karatsuba 3-way algorithm is $O(n^{1.63})$

$$\begin{cases} M(n) & = 6M(\frac{n}{3}) + 8n - 11 \text{ and } M(1) = 1 \\ M_{\otimes}(n) & = 6M(\frac{n}{3}) \text{ and } M_{\otimes}(1) = 1 \\ M_{\oplus}(n) & = 6M(\frac{n}{3}) + 8n - 11 \text{ and } M_{\oplus}(1) = 1 \end{cases}$$

By using Lemma 2.3, we can calculate complexities explicitly:

$$\begin{cases} M(n) & = \frac{34}{5}n^{log_3 6} - 8n + \frac{11}{5} \\ M_{\otimes}(n) & = n^{log_3 6} \\ M_{\oplus}(n) & = \frac{29}{5}n^{log_3 6} - 8n + \frac{11}{5}. \end{cases}$$

3.2.3 Optimized Karatsuba 3-way Multiplication Algorithm

For the Optimized Karatsuba 3-way multiplication algorithm, assume that $A(x)$ and $B(x)$ are the polynomials as defined in Equation 3.1.

We want to calculate $A(x)B(x)$ by applying Optimized Karatsuba 3-way algorithm. In order to use Optimized Karatsuba 3-way algorithm we divide $A(x)$ and $B(x)$ polynomials into 3 parts

$$A(x) = A_0 + A_1 y + A_2 y^2 \text{ and } B(x) = B_0 + B_1 y + B_2 y^2$$

as defined in equations 3.4 and 3.5

$$
\begin{aligned}
P_1 &= A_0B_0 \\
P_2 &= (A_0 + A_1)(B_0 + B_1) \\
P_3 &= A_1B_1 \\
P_4 &= (A_0 + A_2)(B_0 + B_2) \\
P_5 &= A_2B_2 \\
P_6 &= (A_1 + A_2)(B_1 + B_2)
\end{aligned}
$$

As stated Section 3.1.4, P_{iL} where $i = 1, \cdots, 6$ is the lower part of P_i where the number of terms are $\frac{n}{3}$, P_{iH} where $i = 1, \cdots, 6$ is the higher part of P_i where the number of terms are $\frac{n}{3} - 1$. For the addition of $P_{iL} + P_{jL}$ costs $\frac{n}{3}$, $P_{iL} + P_{jH}$ and $P_{iH} + P_{jH}$ costs $\frac{n}{3} - 1$ because of their term sizes.

Then, multiplication of $A(x)B(x)$ is expressed as follows:

$$
\begin{aligned}
A(x)B(x) &= P_{1L} + x^{\frac{n}{3}}[P_{1H} + P_{2L} - P_{1L} - P_{3L}] \\
&\quad + x^{\frac{2n}{3}}[P_{2H} - P_{1H} - P_{3H} + P_{4L} - P_{1L} - P_{5L} + P_{3L}] \\
&\quad + x^{\frac{3n}{3}}[P_{4H} - P_{1H} - P_{5H} + P_{3H} + P_{6L} - P_{3L} - P_{5L}] \\
&\quad + x^{\frac{4n}{3}}[P_{6H} - P_{3H} - P_{5H} + P_{5L}] + x^{\frac{5n}{3}}P_{5H} \\
&= P_{1L} + x^{\frac{n}{3}}[(P_{1H} - P_{3L}) + P_{2L} - P_{1L}] \\
&\quad + x^{\frac{2n}{3}}[-(P_{1H} - P_{3L}) - P_{3H} + P_{2H} + P_{4L} - P_{1L} + P_{5L}] \\
&\quad + x^{\frac{3n}{3}}[(P_{3H} - P_{5L}) + P_{4H} - P_{1H} - P_{5H} + P_{6L} - P_{3L}] \\
&\quad + x^{\frac{4n}{3}}[-(P_{3H} - P_{5L}) + P_{6H} - P_{5H}] + x^{\frac{5n}{3}}P_{5H}
\end{aligned}
$$

The number of multiplication and addition should be counted to get the cost of the Optimized Karatsuba 3-way algorithm. Indicate that there are six multiplications of size $\frac{n}{3}, A_0B_0, (A_0+A_1)(B_0+B_1), A_1B_1, (A_0+A_2)(B_0B_2), A_2B_2$ and $(A_1+A_2)(B_1+B_2)$ and there are six additions of size $\frac{n}{3}, A_0 + A_1, B_0 + B_1, A_0 + A_2, B_0 + B_2, A_1 + A_2$ and $B_1 + B_2$.

In construction part M_i's defined $1 \leq i \leq 11$

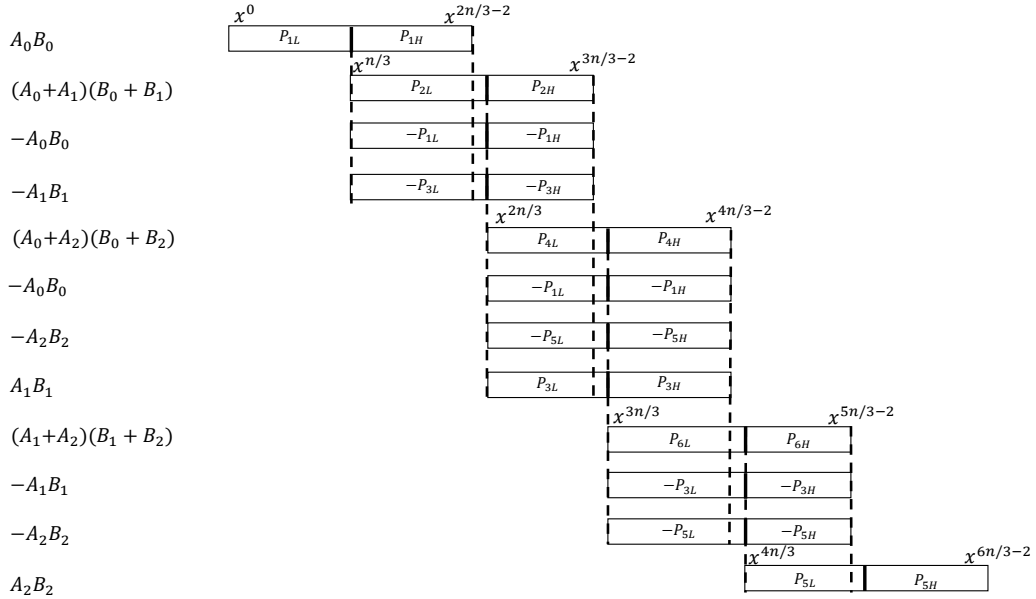

Figure 3.7: Optimized Karatsuba 3-way multiplication algorithm

$$
\begin{aligned}
M_1 &= P_{1H} - P_{3L} \rightarrow \tfrac{n}{3} - 1 \text{ additions} \\
M_2 &= M_1 + P_{2L} \rightarrow \tfrac{n}{3} \text{ additions} \\
M_3 &= M_2 - P_{1L} \rightarrow \tfrac{n}{3} \text{ additions} \\
M_4 &= -M_1 - P_{3H} \rightarrow \tfrac{n}{3} - 1 \text{ additions} \\
M_5 &= M_4 + P_{2H} \rightarrow \tfrac{n}{3} - 1 \text{ additions} \\
M_6 &= M_5 + P_{4L} \rightarrow \tfrac{n}{3} \text{ additions} \\
M_7 &= M_6 - P_{1L} \rightarrow \tfrac{n}{3} \text{ additions} \\
M_8 &= M_7 - P_{5L} \rightarrow \tfrac{n}{3} \text{ additions} \\
M_9 &= P_{3H} - P_{5L} \rightarrow \tfrac{n}{3} - 1 \text{ additions} \\
M_{10} &= M_9 + P_{4H} \rightarrow \tfrac{n}{3} - 1 \text{ additions} \\
M_{11} &= M_{10} - P_{1H} \rightarrow \tfrac{n}{3} - 1 \text{ additions} \\
M_{12} &= M_{11} + -P_{5H} \rightarrow \tfrac{n}{3} - 1 \text{ additions} \\
M_{13} &= M_{12} + P_{6L} \rightarrow \tfrac{n}{3} \text{ additions} \\
M_{14} &= M_{13} + -P_{3L} \rightarrow \tfrac{n}{3} \text{ additions} \\
M_{15} &= -M_9 + P_{6H} \rightarrow \tfrac{n}{3} - 1 \text{ additions} \\
M_{16} &= M_{15} - P_{5H} \rightarrow \tfrac{n}{3} - 1 \text{ additions}
\end{aligned}
$$

Then, we can express the multiplication of $A(x)$ and $B(x)$ as follows:

$$A(X)B(X) = P_{1L} + x^{\frac{n}{3}} M_3 + x^{\frac{2n}{3}} M_8 + x^{\frac{3n}{3}} M_{14} + x^{\frac{4n}{3}} M_{16} + x^{\frac{5n}{3}} P_{5H}$$

In construction part we get $9(\frac{n}{3} - 1) + 7(\frac{n}{3}) = \frac{16n}{3} - 9$ additions, in addition part we have $6(\frac{n}{3}) = 2n$ additions,totally we have $\frac{22n}{3} - 9$ additions.

$$M(n) = 6M(\tfrac{n}{3}) + \tfrac{22n}{3} - 9 \text{ and } M(1) = 1$$

By using Lemma 2.3,

$$\begin{array}{ll} M(n) & = 6M(\frac{n}{3}) + \frac{22n}{3} - 9 \text{ and } M(1) = 1 \\ e & = 1 \\ a & = 6 \\ b & = 3 \\ c & = \frac{22}{3} \\ d & = -9 \\ r_n & = (1 + \frac{22}{6-3} - \frac{9}{6-1})n^{log_3 6} - (\frac{22n}{3}) - \frac{-9}{6-1} \\ & = \frac{98}{15}n^{log_3 6} - (\frac{22}{3})n + \frac{9}{5} \end{array}$$

$\frac{98}{15}n^{log_3 6} - (\frac{22}{3})n + \frac{9}{5} \in O(n^{log_3 6})$, the complexity of Optimized Karatsuba 3-way algorithm is $O(n^{1.63})$

$$\begin{cases} M(n) & = 6M(\frac{n}{3}) + \frac{22n}{3} - 9 \text{ and } M(1) = 1 \\ M_{\otimes}(n) & = 6M(\frac{n}{3}) \text{ and } M_{\otimes}(1) = 1 \\ M_{\oplus}(n) & = 6M(\frac{n}{3}) + \frac{22n}{3} - 9 \text{ and } M_{\oplus}(1) = 1 \end{cases}$$

By using Lemma 2.3, we can calculate complexities explicitly:

$$\begin{cases} M(n) & = \frac{98}{15}n^{log_3 6} - (\frac{22}{3})n + \frac{9}{5} \\ M_{\otimes}(n) & = n^{log_3 6} \\ M_{\oplus}(n) & = \frac{83}{15}n^{log_3 6} - (\frac{22}{3})n + \frac{9}{5} \end{cases}$$

3.3 Comparison of Complexities

The following table is given to compare addition and multiplication complexities of 2-way and 3-way algorithms.

As it can be observed from Table 3.1, in 2-way multiplication algorithms the best complexities can be achieved by Refined and Optimized Karatsuba 2-way multiplication algorithm. Moreover, Optimized Karatsuba 3-way algorithm has the least complexity among 3-way multiplication algorithms.

3.4 Montgomery's Work

In this section, an article written by Peter L. Montgomery is analyzed [22]. New formulas were represented to multiply polynomials for 5-way, 6-way, and 7-way. In this paper, it is mentioned that this formulas use fewer multiplications with respect to generalized Karatsuba [29] algorithms.

Table 3.1: Comparison of complexities

Algorithm	$M_{\otimes}(n)$	$M_{\oplus}(n)$	$M(n)$
Schoolbook 2-way	n^2	$n^2 - 2n + 1$	$2n^2 - 2n + 1$
Karatsuba 2-way	$n^{log_2 3}$	$6n^{log_2 3} - 8n + 2$	$7n^{log_2 3} - 8n + 2$
Refined Karatsuba 2-way	$n^{log_2 3}$	$5.5n^{log_2 3} - 7n + \frac{3}{2}$	$6.5n^{log_2 3} - 7n + \frac{3}{2}$
Optimized Karatsuba 2-way	$n^{log_2 3}$	$5.5n^{log_2 3} - 7n + \frac{3}{2}$	$6.5n^{log_2 3} - 7n + \frac{3}{2}$
Schoolbook 3-way	n^2	$2n^2 - \frac{3}{2}n + 1$	$3n^2 - \frac{3}{2}n + 1$
Karatsuba 3-way	$n^{log_3 6}$	$\frac{29}{5}n^{log_3 6} - 8n + \frac{11}{5}$	$\frac{34}{5}n^{log_3 6} - 8n + \frac{11}{5}$
Optimized Karatsuba 3-way	$n^{log_3 6}$	$\frac{83}{15}n^{log_3 6} - (\frac{22}{3})n + \frac{9}{5}$	$\frac{98}{15}n^{log_3 6} - (\frac{22}{3})n + \frac{9}{5}$

3.4.1 5-way Multiplication

The formula in 3.6 is to multiply two five-term polynomials with 13 multiplication.

$$
\begin{aligned}
&(a_0 + a_1X + a_2X + a_3X^2 + a_3X^3 + a_4X^4)\cdot \\
&(b_0 + b_1X + b_2X + b_3X^2 + b_3X^3 + b_4X^4) = \\
&(a_0 + a_1 + a_2 + a_3 + a_4)(b_0 + b_1 + b_2 + b_3 + b_4)(X^5 - X^4 + X^3)+ \\
&(a_0 - a_2 - a_3 - a_4)(b_0 - b_2 - b_3 - b_4)(X^6 - 2X^5 + 2X^4 - X^3)+ \\
&(a_0 + a_1 + a_2 - a_4)(b_0 + b_1 + b_2 - b_4)(-X^5 + 2X^4 - 2X^3 + X^2)+ \\
&(a_0 + a_1 - a_3 - a_4)(b_0 + b_1 - b_3 - b_4)(X^5 - 2X^4 + X^3)+ \\
&(a_0 - a_2 - a_3)(b_0 - b_2 - b_3)(-X^6 + 2X^5 - X^4)+ \\
&(a_1 + a_2 - a_4)(b_1 + b_2 - b_4)(-X^4 + 2X^3 - X^2)+ \\
&(a_3 + a_4)(b_3 + b_4)(X^7 - X^6 + X^4 - X^3)+ \\
&(a_0 + a_1)(b_0 + b_1)(-X^5 + X^4 - X^2 + X)+ \\
&(a_0 - a_4)(b_0 - b_4)(-X^6 + 3X^5 - 4X^4 + 3X^3 - X^2)+ \\
&a_4b_4(X^8 - X^7 + X^6 - 2X^5 + 3X^4 - 3X^3 + X^2)+ \\
&a_3b_3(-X^7 + 2X^6 - 2X^5 + X^4)+ \\
&a_1b_1(X^4 - 2X^3 + 2X^2 - X)+ \\
&a_0b_0(X^6 - 3X^5 + 3X^4 - 2X^3 + X^2 - X + 1).
\end{aligned}
\tag{3.6}
$$

3.4.2 6-way Multiplication

The formula in 3.7 is to multiply two six-term polynomials with 17 multiplication. The parameter C should be chosen to eliminate one product.

$$
\begin{aligned}
&(a_0 + a_1X + a_2X + a_3X^2 + a_3X^3 + a_4X^4 + a_5X^5)\cdot \\
&(b_0 + b_1X + b_2X + b_3X^2 + b_3X^3 + b_4X^4 + b_5X^5) = \\
&(a_0 + a_1 + a_2 + a_3 + a_4 + a_5)(b_0 + b_1 + b_2 + b_3 + b_4 + b_5)C + \\
&(a_1 + a_2 + a_4 + a_5)(b_1 + b_2 + b_4 + b_5)(-C + X^6) + \\
&(a_0 + a_1 + a_3 + a_4)(b_0 + b_1 + b_3 + b_4)(-C + X^4) + \\
&(a_0 - a_2 - a_3 + a_5)(b_0 - b_2 - b_3 + b_5)(C - X^7 + X^6 - X^5 + X^4 - X^3) + \\
&(a_0 - a_2 - a_5)(b_0 - b_2 - b_5)(C - X^5 + X^4 - X^3) + \\
&(a_0 + a_3 - a_5)(b_0 + b_3 - b_5)(C - X^7 + X^6 - X^5) + \\
&(a_0 + a_1 + a_2)(b_0 + b_1 + b_2)(C - X^7 + X^6 - 2X^5 + 2X^4 - 2X^3 + X^2) + \\
&(a_3 + a_4 + a_5)(b_3 + b_4 + b_5)(C + X^8 - 2X^7 + 2X^6 - 2X^5 + X^4 - X^3) + \\
&(a_2 + a_3)(b_2 + b_3)(-2C + X^7 - X^6 + 2X^5 - X^4 + X^3) + \\
&(a_1 - a_4)(b_1 - b_4)(-C + X^6 - X^5 + X^4) + \\
&(a_1 + a_2)(b_1 + b_2)(-C + X^7 - 2X^6 + 2X^5 - 2X^4 + 3X^3 - X^2) + \\
&(a_3 + a_4)(b_3 + b_4)(-C - X^8 + 3X^7 - 2X^6 + 2X^5 - 2X^4 + X^3) + \\
&(a_0 + a_1)(b_0 + b_1)(-C + X^7 - X^6 + 2X^5 - 3X^4 + 2X^3 - X^2 + X) + \\
&(a_4 + a_5)(b_4 + b_5)(-C + X^9 - X^8 + 2X^7 - 3X^6 + 2X^5 - X^4 + X^3) + \\
&a_0b_0(-3C + 2X^7 - 2X^6 + 3X^5 - 2X^4 + 2X^3 - X + 1) + \\
&a_1b_1(3C - X^7 - X^5 + x^4 - 3X^3 + 2X^2 - X) + \\
&a_4b_4(3C - X^9 + 2X^8 - 3X^7 + x^6 - X^5 - X^3) + \\
&a_5b_5(-3C + X^{10}) - X^9 + 2X^7 - 2X^6 + 3X^5 - 2X^4 + 2X^3).
\end{aligned}
\tag{3.7}
$$

3.4.3 7-way multiplication

The formula in 3.8 is to multiply two seven-term polynomials with 22 multiplication. The parameter C should be chosen to eliminate one product.

$$
\begin{aligned}
&(a_0 + a_1X + a_2X + a_3X^2 + a_3X^3 + a_4X^4 + a_5X^5 + a_6X^6)\cdot\\
&(b_0 + b_1X + b_2X + b_3X^2 + b_3X^3 + b_4X^4 + b_5X^5 + b_6X^6) =\\
&(a_0 + a_1 + a_2 + a_3 + a_4 + a_5 + a_6)(b_0 + b_1 + b_2 + b_3 + b_4 + b_5 + b_6)\cdot\\
&(X^7 - X^6 + X^5)+\\
&(a_0 + a_1 + a_2 + a_3 - a_5 - a_6)(b_0 + b_1 + b_2 + b_3 - b_5 - b_6)\cdot\\
&(-C - X^9 + 2X^7 - 3X^6 + 2X^5)+\\
&(a_0 + a_1 - a_3 - a_4 - a_5)(b_0 + b_1 - b_3 - b_4 - b_5)\cdot\\
&(-C + 2X^7 - 3X^6 + 2X^5 - X^3)+\\
&(a_0 - a_2 - a_3 - a_4 + a_6)(b_0 - b_2 - b_3 - b_4 + b_6)\cdot\\
&(C + X^9 - 4X^7 + 6X^6 - 4X^5 + X^3)+\\
&(a_0 - a_2 - a_3 + a_5 + a_6)(b_0 - b_2 - b_3 + b_5 + b_6)\cdot\\
&(X^7 - 2X^6 + 2X^5 - X^3)+\\
&(a_0 + a_1 - a_3 - a_4 + a_6)(b_0 + b_1 - b_3 - b_4 + b_6)\cdot\\
&(-X^9 + 2X^7 - 2X^6 + X^5)+\\
&(a_1 + a_2 - a_4 - a_5)(b_1 + b_2 - b_4 - b_5)\cdot\\
&(C + X^9 - 3X^7 + 4X^6 - 3X^5 + X^3)+\\
&(a_0 + a_1 - a_5 - a_6)(b_0 + b_1 - b_5 - b_6)C+\\
&(a_0 + a_1)(b_0 + b_1)(X^9 - 5X^7 + 6X^6 - 4X^5 + X^3 + X)+\\
&(a_0 + a_2)(b_0 + b_2)(-C - X^9 + 4X^7 - 5X^6 + 3X^5 - X^2)+\\
&(a_0 - a_4)(b_0 - b_4)(X^7 - 2X^6 + 2X^5 - X^4)+\\
&(a_1 + a_3)(b_1 + b_3)(X^7 - X^6 + X^4 - X^3)+\\
&(a_2 - a_6)(b_2 - b_6)(-X^8 + 2X^7 - 2X^6 + X^5)+\\
&(a_3 + a_5)(b_3 + b_5)(-X^9 + X^8 - X^6 + X^5)+\\
&(a_4 - a_6)(b_4 - b_6)(-C - X^{10} + 3X^7 - 5X^6 + 4X^5 - X^3)+\\
&(a_5 + a_6)(b_5 + b_6)(X^{11} + X^9 - 4X^7 + 6X^6 - 5X^5 + x^3)+\\
&a_0b_0(-2X^7 + 3X^6 - 3X^5 + X^4 + X^2 - X + 1)+\\
&a_1b_1(X^5 - X^4 + X^2 - X)+\\
&a_2b_2(X^8 - 3X^7 + 3X^6 - 2X^5 + X^4 - X^3 + X^2)+\\
&a_3b_3(C + 2X^9 - X^8 - 5X^7 + 8X^6 - 5X^5 - X^4 + 2X^3)+\\
&a_4b_4(X^{10} - X^9 + X^8 - 2X^7 + 3X^6 - 3X^5 + X^4)+\\
&a_5b_5(-X^{11} + X^{10} - X^8 + X^7)+\\
&a_6b_6(X^{12} - X^{11} + X^{10} + X^8 - 3X^7 + 3X^6 - 2X^5).
\end{aligned}
\tag{3.8}
$$

3.5 Hybrid multiplication algorithms

For the polynomial multiplication, it can be seen that the complexity of Karatsuba algorithm gives better results then Schoolbook method. However, in some cases using combined methods, for example, using Karatsuba algorithm before Schoolbook method is the lowest number of operation to multiply two polynomials.

For example, consider the multiplication of 2 polynomials of degree 5. If we use Schoolbook algorithm 2.3.1 we have 36 multiplication and 25 additions. Therefore this operation results in 61 operations. However, using firstly Refined Karatsuba algorithm, then Schoolbook 3-way algorithm we have

$$\begin{aligned} M(n) &= 3M(\tfrac{n}{2}) + 7 \cdot \tfrac{n}{2} - 3 \\ &= 3(13) + 7 \cdot \tfrac{6}{2} - 3 \\ &= 57 \end{aligned}$$

Instead of $M(\frac{n}{2}) = 13$ is written, because the best way to multiply degree 2 polynomials is to use Schoolbook method where minimum operation for this case is 13.

As it can be seen from this example, computation of the minimum number of bit operations can be computed with hybrid method. The list of this operations for multiplying two polynomials up to degree 20 is given in the table.

Table 3.2: Minimum number of bit operations

n	Total Operation	Algorithm
1	1	Schoolbook
2	5	Schoolbook
3	13	Schoolbook
4	25	Schoolbook
5	41	Schoolbook
6	57	Refined Karatsuba, Schoolbook
7	81	n=6, Schoolbook
8	100	Refined Karatsuba, Schoolbook
9	132	n=8, Schoolbook
10	155	Refined Karatsuba, Schoolbook
11	195	n=10, Schoolbook
12	210	Refined Karatsuba, Schoolbook
13	258	n=12, Schoolbook
14	289	Refined Karatsuba, Schoolbook
15	345	n=14, Schoolbook
16	353	Refined Karatsuba, Schoolbook
17	417	n=16, Schoolbook
18	456	Refined Karatsuba, Schoolbook
19	528	n=18, Schoolbook
20	532	Refined Karatsuba, Schoolbook

In order to compute the minimum number of bit operations to multiply polynomials of degree 7 , we use firstly minimum number of operations for degree 6 polynomials and for the last term we use schoolbook method.

CHAPTER 4

CONCLUSION

After the invention of Diffie- Hellman [9] key exchange and RSA [24] algorithm the concept of public key cryptography arose. This field has some advantages in many aspects as well as some disadvantages. Number of keys in a network for separate users, digital signatures and key establishment are some examples for advantages. It can be concluded that number of problems are solved with the invention of public key cryptography. However, public key cryptography has some handicaps. As mentioned in Chapter 1, to satisfy security in asymmetric algorithms, key lengths have to be longer with respect to symmetric algorithms. In this chapter also time comparison of RSA and AES [8] algorithms are mentioned. Even for small packet sizes, RSA algorithm is slower than AES algorithm as can be understood from Table 1.2. In spite of these disadvantages, public key cryptography is still utilized in lots of areas. Today the main concern is to accelerate public key cryptography algorithms.

In this thesis, the existent integer multiplication algorithms for cryptographic sizes are analyzed with respect to their time complexities. To shorten the time required for these algorithms some methods were suggested.

In Chapter 2, an introduction of public key cryptography is given. In this chapter some definitions and theorems are provided in order to analyze the cost of algorithms. Furthermore, a brief history of integer multiplication algorithms and their details are mentioned.

In Chapter 3, School Book, Karatsuba, Refined Karatsuba and Optimized Karatsuba algorithms in 2-way and 3-way are examined. Besides, Montgomery's 5-way, 6-way and 7-way multiplication algorithms and Hybrid method are given.

REFERENCES

[1] E. Barker, L. Chen, A. Regenscheid, and M. Smid, Recommendation for pair wise key establishment schemes using integer factorization cryptography, NIST Special Publication, 2009.

[2] D. J. Bernstein, Batch binary edwards, in *Annual International Cryptology Conference*, pp. 317–336, Springer, 2009.

[3] D. J. Bernstein, C. Chuengsatiansup, and T. Lange, Curve41417: Karatsuba revisited, in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 316–334, Springer, 2014.

[4] M. Bodrato and A. Zanoni, Integer and polynomial multiplication: Towards optimal toom-cook matrices, in *Proceedings of the 2007 international symposium on Symbolic and algebraic computation*, pp. 17–24, ACM, 2007.

[5] M. Cenk, C. Negre, and M. A. Hasan, Improved three-way split formulas for binary polynomial and toeplitz matrix vector products.

[6] S. A. Cook and S. O. Aanderaa, On the minimum computation time of functions, Transactions of the American Mathematical Society, 142, pp. 291–314, 1969.

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, volume 6, MIT press Cambridge, 2001.

[8] J. Daemen and V. Rijmen, Aes proposal: Rijndael, 1999.

[9] W. Diffie and M. Hellman, New directions in cryptography, IEEE transactions on Information Theory, 22(6), pp. 644–654, 1976.

[10] M. Furer, *On the complexity of integer multiplication*, Pennsylvania State University, Department of Computer Science, 1989.

[11] D. Hankerson, A. J. Menezes, and S. Vanstone, *Guide to elliptic curve cryptography*, Springer Science & Business Media, 2006.

[12] D. Harvey, J. Van Der Hoeven, and G. Lecerf, Even faster integer multiplication, Journal of Complexity, 2016.

[13] D. Kahn, *The codebreakers*, Weidenfeld and Nicolson, 1974.

[14] A. Karatsuba and Y. Ofman, Multiplication of multidigit numbers on automata, in *Soviet physics doklady*, volume 7, p. 595, 1963.

[15] A. A. Karatsuba, The complexity of computations, Proceedings of the Steklov Institute of Mathematics-Interperiodica Translation, 211, pp. 169–183, 1995.

[16] T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, et al., Factorization of a 768-bit rsa modulus, in *Annual Cryptology Conference*, pp. 333–350, Springer, 2010.

[17] D. Knuth, The art of computer programming: Vol 2/seminumerical algorithms, chapter 3: Random numbers, 1969.

[18] N. Koblitz, Elliptic curve cryptosystems, Mathematics of computation, 48(177), pp. 203–209, 1987.

[19] P. Mahajan and A. Sachdeva, A study of encryption algorithms aes, des and rsa for security, Global Journal of Computer Science and Technology, 13(15), 2013.

[20] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography*, CRC press, 1996.

[21] V. S. Miller, Use of elliptic curves in cryptography, in *Conference on the Theory and Application of Cryptographic Techniques*, pp. 417–426, Springer, 1985.

[22] P. L. Montgomery, Five, six, and seven-term karatsuba-like formulae, IEEE Transactions on Computers, 54(3), pp. 362–369, 2005.

[23] C. Paar and J. Pelzl, *Understanding cryptography: a textbook for students and practitioners*, Springer Science & Business Media, 2009.

[24] R. L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Communications of the ACM, 21(2), pp. 120–126, 1978.

[25] D. D. A. Schönhage, Multiplikation großer zahlen, Computing, 1(3), pp. 182–196, 1966.

[26] D. D. A. Schönhage and V. Strassen, Schnelle multiplikation grosser zahlen, Computing, 7(3-4), pp. 281–292, 1971.

[27] A. L. Toom, The complexity of a scheme of functional elements realizing the multiplication of integers, in *Soviet Mathematics Doklady*, volume 3, pp. 714–716, 1963.

[28] J. Von Zur Gathen and J. Gerhard, *Modern computer algebra*, Cambridge university press, 2013.

[29] A. Weimerskirch and C. Paar, Generalizations of the karatsuba algorithm for efficient implementations., IACR Cryptology ePrint Archive, 2006, p. 224, 2006.

[30] M. Welschenbach, *Cryptography in C and C++*, Apress, 2006.

[31] G. Zhou and H. Michalik, Comments on" a new architecture for a parallel finite field multiplier with low complexity based on composite field", IEEE Transactions on Computers, 59(7), pp. 1007–1008, 2010.