NOTE TO USERS

Page(s) not included in the original manuscript and are unavailable from the author or university. The manuscript was scanned as received.

This reproduction is the best copy available.

UMI[®]

THE AVERAGE COMPLEXITY OF THE EUCLIDEAN ALGORITHM AND
HYPERELLIPTIC CRYPTOGRAPHY

by

Stephen Savard

A thesis submitted in conformity with the requirements
for the degree of Master of Science
School of Mathematics and Statistics
Carleton University

Copyright © 2003 by Stephen Savard

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-79818-6

Canada

The undersigned recommended to
the Faculty of Graduate Studies and Research
acceptance of the thesis
The Average Complexity of the Euclidean Algorithm
and Hyperelliptic Cryptography
submitted by Stephen Savard
in partial fulfilment of the requirements for
the degree of Master of Science

Thesis Supervisor

Director, School of Mathematics and Statistics

Carleton University

2003

Abstract

This thesis deals with the average complexity of the Euclidean algorithm and hyperelliptic cryptography. The Euclidean algorithm is used to find the greatest common divisor as well as algebraic inverses. Hyperelliptic cryptography is a generalization of elliptic curve cryptography, which has come to the forefront of encryption technologies in recent years.

The most costly operation for hyperelliptic cryptography is the use of the extended Euclidean algorithm. We first use known results to establish the expected number of field operations for this algorithm when classical arithmetic is employed. This leads to the computations for the average complexity of hyperelliptic cryptography.

To improve the exact and asymptotic analysis, Karatsuba multiplication is considered and analyzed. This analysis is then used to improve the complexity of the extended Euclidean algorithm.

Acknowledgements

This thesis is special for many reasons. It is the epitome of an epic struggle that symbolizes my many years at Carleton University. This thesis would not have been possible without the help of countless individuals who have aided my journey throughout university. I consider myself lucky to have had the opportunity to work and study with such talented and passionate individuals.

My supervisor Daniel Panario was nothing short of phenomenal to work under. His guidance in solving challenging problems is inspiring. There were more than a few occasions when we arrived at an impasse in this thesis. I will certainly miss our weekly meetings.

Andreas Enge was particularly helpful in completing this thesis. Andreas kindly answered several questions via email about his original research that this thesis is an extension of. His suggestions on solving some of the more intricate details are greatly appreciated.

Finally, I want to thank the long-standing support of my family. Many kind words can also be said of Rishi Datta for encouragement and advice during harrowing times.

Contents

1	Introduction	1
2	Mathematical Preliminaries	3
2.1	Algebraic Foundations	3
2.2	Finite Fields	4
2.3	Combinatorics	6
2.4	Combinatorial Equation Derivations	7
2.5	Probability	10
2.6	Series	11
3	The Euclidean Algorithm and its Analysis	13
3.1	The Euclidean Algorithm	13
3.2	The Extended Euclidean Algorithm	14
3.3	The Expected Number of Field Operations for the Euclidean Algorithm .	17
4	Elliptic and Hyperelliptic Curves	26
4.1	Elliptic Curves	27
4.2	Hyperelliptic Curves	29
4.3	Polynomial and Rational Functions	30
4.4	Zeros and Poles	32
4.5	Divisors	33

4.6	Semi-Reduced and Reduced Divisors	35
4.7	Hyperelliptic Cryptography	36
4.7.1	Composition Algorithm	37
4.7.2	Reduction Algorithms	39
5	The Average Cost of Hyperelliptic Cryptography	42
5.1	Heuristic Assumption on Probabilities	42
5.2	The Average Cost	44
6	Karatsuba Multiplication and the Euclidean Algorithm	49
6.1	Basic Method	49
6.2	The Euclidean Algorithm with Karatsuba Approximations	55
7	Conclusion	59
7.1	Future Work	59
	Bibliography	61
A	Identities	64
A.1	Series	64
A.2	Binomial Identities	65
A.3	Binomial Summations	65
A.4	Binomial Absorptions	66
B	Proof for Classical Multiplication	67
B.1	Proof of Theorem 3.3.1	67
C	Proof for Karatsuba Multiplication	79
C.1	Proof of Lemma 6.2.1	79

Chapter 1

Introduction

This thesis deals with the average complexity of the extended Euclidean algorithm and hyperelliptic cryptography. The Euclidean algorithm is used to find the greatest common divisor as well as algebraic inverses. Hyperelliptic cryptography is an extension of the methods used for elliptic curve cryptography. In order to apply cryptographic techniques to the group of points in these types of curves, there needs to be a clearly defined operator to add two elements in the group. For an elliptic curve defined over a finite field, there is a beautiful geometric approach that is simple and very effective. A hyperelliptic curve is essentially a higher-order elliptic curve, but the same geometric argument does not apply to hyperelliptic curves. Instead, we look to the Jacobian of a hyperelliptic curve where we can define an addition operator.

When adding two elements of the Jacobian, the most costly operation in terms of multiplications and inversions is the use of the extended Euclidean algorithm. In order to fully analyze the complexity of hyperelliptic cryptography, we examine the extended Euclidean algorithm in great detail with respect to the number of field operations.

This work is based on results obtained by Andreas Enge [7]. The aim of this thesis is to produce a more exact and accurate asymptotic analysis by improving his results. To accomplish this, we consider and implement a recursive multiplication algorithm discov-

ered by Karatsuba. We prove that for given bounds, the use of Karatsuba multiplication results in less field operations when executing the extended Euclidean algorithm than classical multiplication. As an example demonstrated in Chapter 6, using Karatsuba multiplication in the extended Euclidean algorithm for two polynomials of degree 160 will result in savings of over 700 multiplications (see Tables 6.1 and 6.2).

A great deal of emphasis has been incorporated into the writing of this thesis to be self-contained. All of the required mathematics to compute the results are contained within this thesis. Chapter 2 is an overview of the algebraic, combinatorial, and discrete mathematics used in all derivations. The Euclidean algorithm is the focus of Chapter 3. After a brief review of the algorithm, we work out the expected number of field operations the algorithm uses upon execution. For Chapter 4, we switch focus to elliptic and hyperelliptic cryptography, and discuss the composition and reduction formulas for adding two elements in the Jacobian of a hyperelliptic curve. We use our model in Chapter 5 to determine the average cost of hyperelliptic cryptography when classical arithmetic is used. In Chapter 6 we present and analyze Karatsuba's multiplication algorithm. This algorithm is then applied to the extended Euclidean algorithm to obtain faster results. Further conclusions are drawn in Chapter 7.

This thesis has both a theoretical and computational aspect. Many of the derivations of formulas are very complicated. To increase readability, these results are placed in the appendices at the end of this thesis. Appendix A is a reference table of all formulas and identities that appeared in the derivations. Appendices B and C contain all of the proofs for the long derivations.

Chapter 2

Mathematical Preliminaries

This chapter introduces the various types of mathematics that are required for this thesis. The breadth of this thesis is far reaching, and requires tools from several different branches of advanced mathematics. This chapter is more computational by nature, and contains all relevant material needed to physically compute the results contained in this thesis. As a companion to this material, Appendix A contains a complete list of all the formulas that are derived in this thesis. Only general cases are discussed here.

We start with the basics, and continue to derive many complicated results. Full details are given for many derivations, as understanding the proofs of these results is essential to understanding the material. For more detail on this type of material, consult *The Art of Computer Programming* by Knuth [11] or *Concrete Mathematics* by Graham, Knuth, and Patashnik [9].

2.1 Algebraic Foundations

We do not review the basics of group and ring theory, but we briefly state some of the algebraic properties that are relevant to cryptography.

Given a finite group G , we denote the number of elements in a group by $|G|$ and call this the cardinality or the order of the group. A group G is cyclic if there exists an

element $a \in G$ such that for any $b \in G$ there is some integer i with $b = a^i$. The element g is called a generator of the group. Every subgroup of a cyclic group is itself a cyclic group. Every element $g \in G$ generates a subgroup H of G , and is denoted by $\langle g \rangle$.

Theorem 2.1.1 (*Lagrange*) *If G is a finite group and H is a subgroup of G , then $|H|$ divides $|G|$. Furthermore, for $g \in G$, then the order of g divides $|G|$.*

Let $\mathbb{F}$ be a field. Then a polynomial in the indeterminate variable x over the field $\mathbb{F}$ is an expression of the form

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$$

where $a_i \in \mathbb{F}$ and $n \geq 0$. The polynomial ring is then denoted as $\mathbb{F}[x]$, with the two ring operations being the standard polynomial addition and multiplication. The polynomial ring $\mathbb{F}[x]$ is a unique factorization domain, such that every polynomial of positive degree in $\mathbb{F}[x]$ can be written in essentially one and only one way as the product of irreducible polynomials. A polynomial ring is not necessarily required to be univariate, and such $\mathbb{F}[x, y]$ would denote the polynomials with the indeterminates x and y over the field $\mathbb{F}$.

2.2 Finite Fields

The standard reference for finite fields is the book by Lidl and Niederreiter [15]. This section is a brief overview on the field theory required for this thesis.

A field is a commutative ring where all non-zero elements have multiplicative inverses. Some examples of fields include $\mathbb{R}$, $\mathbb{C}$, $\mathbb{Q}$, and $\mathbb{Z}_p$ which is the set of integers modulo a prime number p . The field $\mathbb{Z}_p$ is often denoted as $\mathbb{F}_p$ or $GF(p)$ which stands for a Galois Field with p elements.

We are interested in two types of finite fields. Let p be a prime number. Let $\mathbb{F}_p = \{0, 1, 2, \dots, p-1\}$ be the set of integers modulo p . If p is prime, then $\mathbb{F}_p$ is always a field. There are $p-1$ non-zero elements in $\mathbb{F}_p$, and these elements form an abelian group with

respect to multiplication. A generator in $\mathbb{F}_p$ has order $p - 1$ and generates every non-zero element. The field $\mathbb{F}_p$ has a total of $\varphi(p - 1)$ different generators, where φ is the Euler totient function. This function represents the number of elements in $\mathbb{F}_p$ relatively prime to $p - 1$.

The second type of field are of the form $\mathbb{F}_q$, where $q = p^n$ for any positive integer n and $\mathbb{F}_q$ has characteristic p and p^n elements. Very often we set $p = 2$, as this case is ideally suited for computer implementation. Consider the polynomial ring $\mathbb{F}_p[x]$. There exists an irreducible polynomial $f \in \mathbb{F}_p[x]$ of degree n such that (f) is a maximal ideal of $\mathbb{F}_p[x]$. It then follows that there exists an isomorphism such that

$$\mathbb{F}_q \cong \frac{\mathbb{F}_p[x]}{(f)}$$

is a field with p^n elements.

For the remainder of this thesis, let $\mathbb{F}$ denote a field. We call the characteristic of a field the smallest positive integer p such that $p \cdot 1 = 0$, where for $p \cdot 1$ we mean $1 + 1 + \dots + 1$, p times if such p exists. Otherwise we say that $\mathbb{F}$ has characteristic zero.

Definition 2.2.1 *Let $\mathbb{F}$ be a field. Then $\mathbb{F}$ is said to be algebraically closed if every polynomial of degree ≥ 1 with coefficients in $\mathbb{F}$ splits completely into linear factors. Equivalently, $\mathbb{F}$ is said to be algebraically closed if every polynomial of degree n has n roots in $\mathbb{F}$.*

The smallest algebraically closed extension field of $\mathbb{F}$ is called the algebraic closure of $\mathbb{F}$ and is denoted as $\overline{\mathbb{F}}$. For example, given the field of real numbers, its algebraic closure is the field of complex numbers. The algebraic closure of a field is an infinite set. Given the field $\mathbb{F}_q$, then its algebraic closure $\overline{\mathbb{F}_q}$ can be viewed as the union of $\mathbb{F}_{q^k}$ for $k \geq 1$.

2.3 Combinatorics

We begin by reviewing basic combinatorics. The most common notation to denote a binomial number is either $C(n, k)$ or $\binom{n}{k}$. In factorial notation, we have

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \quad \text{with } n \geq k \geq 0, \text{ for } n, k \in \mathbb{N}.$$

The symbol $\binom{n}{k}$ is also called the binomial coefficient. These are used in the binomial theorem where

$$(x+y)^n = \sum_{k=0}^n \binom{n}{k} x^k y^{n-k} \quad \text{for } n \geq 0.$$

The binomial theorem gives us a powerful tool to easily compute powers of $(x+y)$. Some important properties of binomial numbers are

$$\binom{n}{k} = \binom{n}{n-k} \quad \text{for } n \geq 0 \text{ and } n, k \in \mathbb{N}, \quad (2.1)$$

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1} \quad \text{for all } k \in \mathbb{N}. \quad (2.2)$$

Very often, as we shall see, we want to have the ability to move values in and out of the binomial coefficient. This process is called *absorption*. This is accomplished by converting to the factorial notation such that

$$\begin{aligned} \binom{n}{k} &= \frac{n!}{k!(n-k)!} = n \frac{(n-1)!}{k!(n-k)!} = \frac{n}{k} \frac{(n-1)!}{(k-1)!(n-k)!} \\ &= \frac{n}{k} \binom{n-1}{k-1}. \end{aligned}$$

The following equivalent absorption identity is very important in this thesis:

$$k \binom{n}{k} = n \binom{n-1}{k-1}. \quad (2.3)$$

The addition formula (2.2) is quite useful. Recursively applying the addition formula to the first term of $\binom{n}{k}$, we get the *upper summation* formula:

$$\begin{aligned} \sum_{n=0}^s \binom{n}{k} &= \binom{0}{k} + \binom{1}{k} + \binom{2}{k} + \cdots + \binom{s-1}{k} + \binom{s}{k} \\ &= \binom{s+1}{k+1}. \end{aligned} \quad (2.4)$$

Often we want to absorb a value into the binomial, but Equation (2.3) requires the value to be equal to the lower index. The following two formulas that are immediate from the addition formula are somehow the opposite. The addition formula expresses a coefficient as the sum of smaller values, whereas these two formulas express the coefficient as the difference of larger values:

$$\binom{n}{k} = \binom{n+1}{k+1} - \binom{n}{k+1}, \quad (2.5)$$

$$\binom{n}{k} = \binom{n+1}{k} - \binom{n}{k-1}. \quad (2.6)$$

2.4 Combinatorial Equation Derivations

In the last section we give known results about binomial coefficients. We can now apply these identities to build techniques for practical benefits. In this section, we want to give an overview of how to derive the formulas needed for this thesis.

First, we want to discuss a method to rewrite a summation that appears frequently. For example, take $\sum_{\mu=0}^{d_1-k+1} \binom{d_1-\mu-1}{k-2}$. Examining the binomial, when $\mu = 0$ is $\binom{d_1-1}{k-2}$, and when $\mu = d_1 - (d_1 - k + 1) - 1 = k - 2$ is $\binom{k-2}{k-2}$. Now if we sum the summation backwards, going from $k - 2$ down to $d_1 - 1$, we get

$$\binom{k-2}{k-2} + \binom{k-3}{k-2} + \cdots + \binom{d_1-2}{k-2} + \binom{d_1-1}{k-2} = \sum_{\mu=k-2}^{d_1-1} \binom{\mu}{k-2}.$$

This is essentially upper summation Equation (2.4). This particular example summation would derive to $\binom{d_1}{k-1}$.

There are several types of summations which we need to work with. Starting with the most basic formula, let's try to find a closed form for

$$\sum_{k=3}^{d_1+1} (q-1)^{k-1} \binom{d_1}{k-2}.$$

We want to present a general formula for solving these types of problems. The first thing we should notice is that $(q-1)$ is a constant, so this is not very worrying. Set

$\alpha = (q - 1)$. We would like to make the exponent of α a simpler number, so we can take α as a common factor to get

$$\sum_{k=3}^{d_1+1} \alpha^{k-1} \binom{d_1}{k-2}.$$

If we expand the summation, we now have

$$\alpha \left[\alpha \binom{d_1}{1} + \alpha^2 \binom{d_1}{2} + \dots + \alpha^{d_1-2} \binom{d_1}{d_1-2} + \alpha^{d_1-1} \binom{d_1}{d_1-1} \right].$$

This is closer to the binomial theorem, except we are missing some terms. If we add and subtract the missing terms, we derive

$$\alpha \left[\alpha^0 \binom{d_1}{0} + \alpha^1 \binom{d_1}{1} + \dots + \alpha^{d_1-1} \binom{d_1}{d_1-1} + \alpha^{d_1} \binom{d_1}{d_1} - \alpha^0 \binom{d_1}{0} - \alpha^{d_1} \binom{d_1}{d_1} \right].$$

Now we can clearly see the binomial theorem, and contracting the terms gives

$$\alpha \left[(\alpha + 1)^{d_1} - \alpha^0 \binom{d_1}{0} - \alpha^{d_1} \binom{d_1}{d_1} \right].$$

We know that $\binom{d_1}{0} = \binom{d_1}{d_1} = 1$, so substituting our constant α back in, we get our final answer of

$$(q - 1) [q^{d_1} - 1 - (q - 1)^{d_1}].$$

Often, a summation will have the index k being summed in the denominator. The first of two occasions would be in a sum like $\sum_{k=3}^{d_1+1} \frac{(q-1)^{k-1}}{q^k} \binom{d_1}{k-2}$. Once again, we want to fix the $k - 1$ exponent, which is accomplished by factoring in a $(q - 1)$. Once we have

$$\frac{1}{(q - 1)} \sum_{k=3}^{d_1+1} \frac{(q - 1)^k}{q^k} \binom{d_1}{k-2} = \frac{1}{(q - 1)} \sum_{k=3}^{d_1+1} \left(\frac{q - 1}{q} \right)^k \binom{d_1}{k-2},$$

we see that $(q - 1)/q$ is constant, so this time we set $\alpha = (q - 1)/q$. We have now reduced this problem to our basic technique, which we certainly know how to solve.

Similarly, the denominator may contain a term such as $k - 1$. This is actually easier to solve than the previous problem. Take the summation $\sum_{k=3}^{d_1+1} \frac{(q-1)^{k-1}}{k-1} \binom{d_1}{k-2}$. Since

the term in the denominator is one greater than the lower index in the binomial, this problem is a prime candidate for the absorption identity (2.3). Applying the identity then derives $(d_1 + 1) \sum_{k=3}^{d_1+1} (q-1)^{k-1} \binom{d_1+1}{k-1}$.

A few words need to be said about absorptions. We know that we can solve $\binom{d_1}{k} k = d_1 \binom{d_1-1}{k-1}$ by Equation (2.3). We can also solve $\binom{d_1}{k-1} k$, where the binomial's lower index is less than the outside term. First we need to use identity (2.5) to increase the lower index. This gives $\binom{d_1}{k-1} k = [\binom{d_1+1}{k} - \binom{d_1}{k}] k$, which is now the same as the previous problem and is solvable by Equation (2.3).

Absorbing a k^2 term is a bit more complicated. However, upon noticing that $k^2 = k(k-1) + k$, we have now created two subcases that we know how to solve. Similarly,

$$\begin{aligned} k^3 &= k(k-1)(k-2) + 3k^2 - 2k \\ &= k(k-1)(k-2) + 3(k(k-1) + k) - 2k \\ &= k(k-1)(k-2) + 3k(k-1) + k. \end{aligned}$$

A k^2 term will only appear if the lower index of a binomial is $k-2$, whereas a k^3 will require the lower index to be $k-3$.

Finally, there is one combinatorial summation that is particularly painful to solve. Take the following summation for example

$$\sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+2} \binom{d_1-\mu-1}{k-3} q^\mu.$$

This summation appears quite often, and sometimes the final term is μq^μ instead of q^μ . The same technique solves both summations.

The trick with this summation is that we want to rewrite the summation backwards. To do this, we first need to expand the summation into horizontal rows based on

the value of k , and we proceed to sum on the vertical columns. For example

$$\begin{aligned}
&= (q-1)^2 \binom{d_1-1}{0} + (q-1)^2 \binom{d_1-2}{0} q + (q-1)^2 \binom{d_1-3}{0} q^2 + \dots + (q-1)^2 \binom{0}{0} q^{d_1-1} \\
&+ (q-1)^3 \binom{d_1-1}{1} + (q-1)^3 \binom{d_1-2}{1} q + (q-1)^3 \binom{d_1-3}{1} q^2 + \dots + (q-1)^3 \binom{0}{1} q^{d_1-1} \\
&\vdots \\
&+ (q-1)^{d_1-1} \binom{d_1-1}{d_1-3} + (q-1)^{d_1-1} \binom{d_1-2}{d_1-3} q + (q-1)^{d_1-1} \binom{d_1-3}{d_1-3} q^2 \\
&+ (q-1)^{d_1} \binom{d_1-1}{d_1-2} + (q-1)^{d_1} \binom{d_1-2}{d_1-2} q.
\end{aligned}$$

The first two columns are easy to sum, as they are both of the same size. The first column sums to

$$\sum_{i=0}^{d_1-2} \binom{d_1-1}{i} (q-1)^{i+2}$$

while the second column sums to

$$\sum_{i=0}^{d_1-2} \binom{d_1-2}{i} q (q-1)^{i+2}.$$

All of the remaining elements are arranged in a triangle formation, and can be written as a double summation such that

$$\sum_{j=3}^{d_1} \sum_{i=0}^{d_1-j} \binom{d_1-j}{i} q^{j-1} (q-1)^{i+2}.$$

The original summation has now been rewritten into three solvable summations. These summations are solved in Section 3.3 and Appendix B.

2.5 Probability

The analysis of the algorithms in this thesis use various tools from probability. A useful statistic on a random variable X is the average or expected value. The expected value of a random variable X is defined as

$$E[X] = \sum_x x \text{Prob}\{X = x\}.$$

We say that a finite sample space S has a uniform probability distribution on S if every element in S has a probability of $1/|S|$ of occurring.

Next let the random variable X be the number of successes in n trials. Let p be the probability of success and $q = 1 - p$ be the probability of failure. Then X has values in the range $\{0, 1, \dots, n\}$. For $k = 0, 1, \dots, n$, there are $\binom{n}{k}$ ways where k of the n trials are successful. Each success occurs with probability $p^k q^{n-k}$. A probability distribution that satisfies

$$\text{Prob}\{X = k\} = \binom{n}{k} p^k q^{n-k}$$

is called a *binomial* distribution.

2.6 Series

Frequently in this thesis, we want to know the value of a summation, especially when we want to compute the values of various probabilities that occur. In general, working with infinite series makes life easier. But cryptography is a finite discipline, and this causes us to use harder finite formulas. Proofs of most formulas can be found in any introductory calculus or discrete mathematics textbook. All these finite formulas can be proven by the technique of mathematical induction.

A very common series that occurs in this thesis is the geometric series. This series sums the powers of a constant. If the constant is called q , a geometric series only converges if $|q| < 1$. We can now give the infinite and finite values for the geometric series:

$$\begin{aligned} \sum_{\mu=0}^{\infty} q^{\mu} &= \frac{1}{1-q} \\ \sum_{\mu=0}^n q^{\mu} &= \frac{q^{n+1} - 1}{q - 1} \quad \text{or} \quad \sum_{\mu=1}^n q^{\mu} = \frac{q^{n+1} - q}{q - 1}. \end{aligned}$$

Of a similar nature, the two following series occur frequently in this thesis

$$\begin{aligned}\sum_{\mu=1}^n \mu q^\mu &= \frac{q^{n+1}((n+1)q - n - 1 - q) + q}{(q-1)^2} \\ \sum_{\mu=1}^n \mu^2 q^\mu &= \frac{1}{(q-1)^3} (q^{n+1}(-2(n+1)q^2 + 2(n+1)q + q^2 + q \\ &\quad + (n+1)^2 q^2 - 2(n+1)^2 q + (n+1)^2)) - (q^2 - q)) .\end{aligned}$$

It can be useful at times if we can approximate complicated formulas. Let us examine $\sum_{\mu=0}^{\infty} \mu q^\mu$, and try to find a closed form as we sum it to infinity. The trick is that we have to make the summation look like a derivative. If we take a constant q out of the summation, then we derive the sum $q \sum_{\mu=0}^{\infty} \mu q^{\mu-1}$. However, the summation of $\sum_{\mu=0}^{\infty} \mu q^{\mu-1}$ is equal to the first derivative of the geometric sum $\sum_{\mu=0}^{\infty} q^\mu$. So we can finally derive

$$q \left(\frac{1}{q-1} \right)' = \frac{q}{(q-1)^2}.$$

These types of derivations are used many times in this thesis.

Chapter 3

The Euclidean Algorithm and its Analysis

This chapter discusses the Euclidean algorithm, and then continues to derive various probabilities in finite fields which are used to establish the expected number of field operations required to perform the algorithm.

This chapter is an extension of previous work completed by Andreas Enge [7]. All computations in this chapter use classical operations. This restriction will be lifted in Chapter 6, where we give a more accurate accounting of the Euclidean algorithm's behavior as used in practice.

3.1 The Euclidean Algorithm

The Euclidean algorithm first appeared in the seventh book of Euclid's Elements, around 300 B.C. The algorithm itself almost certainly predates this publication. It is based on a simple observation. If $a = qb + r$, then $\gcd(a, b) = \gcd(b, r)$. We can iteratively reduce the size of the input without changing the greatest common divisor.

The Euclidean algorithm works for both integers and polynomials with minor modifications, but we are only interested in polynomials over finite fields. The algorithm

generates a sequence of polynomials $(r_0, r_1, \dots, r_k)$ where r_k is the greatest common divisor of r_0 and r_1 .

Algorithm 1: The Euclidean Algorithm

Input: Two polynomials $f, g \in \mathbb{F}_p[x]$ with $f(x) \geq g(x)$

Output: The greatest common divisor of f and g

$r_0 := f(x)$

$r_1 := g(x)$

$k := 1$

while $(r_k \neq 0)$

$r_{k+1} := r_{k-1} \bmod r_k$

$k := k + 1$

Return r_{k-1}

There are many well known facts about the Euclidean algorithm. It was established by Lamé that the worst case over the integers consists of two consecutive Fibonacci numbers. Interestingly enough, this was the first practical application of the Fibonacci sequence. The best case is when $f(x)$ is a multiple of the greatest common divisor.

3.2 The Extended Euclidean Algorithm

The extended Euclidean algorithm (EEA) was developed for integers around 500 AD in an astrological book written by Aryabhata. The polynomial version of the EEA was first published by Sir Isaac Newton around 1707. The extended Euclidean algorithm computes the greatest common divisor as well as two multipliers $u(x)$ and $v(x)$. Given two polynomials $f(x)$ and $g(x)$, their greatest common divisor can always be written as a linear combination where the $\gcd(f(x), g(x)) = u(x)f(x) + v(x)g(x)$.

We now use this opportunity to fix some notation for the remainder of this thesis. Let r_0 and r_1 be two polynomials such that $\deg r_0 \geq \deg r_1$. Instead of writing $\deg r_0$

and $\deg r_1$, we denote the degree of a polynomial as d_0 and d_1 respectively. Let k be the variable that counts the number of iterations for the extended Euclidean algorithm. Let t be the index such that r_t is the GCD and d_t is the degree of the GCD. We denote $d_k = -\infty$ for $k > t$ so we can clearly see any computations done after the GCD. This is a convention that will aid our computations later. We could take $d_k = -1$ for $k > t$, for example, and nothing would change.

Upon executing the Euclidean algorithm, it generates the sequence of polynomials $(r_0, r_1, \dots, r_t)$ based on the two input polynomials. The EEA generates two further sequences $(u_0, u_1, \dots, u_t)$ and $(v_0, v_1, \dots, v_t)$, which represent the polynomials for the multipliers u and v respectively.

We can use our notation for $k = 1, \dots, t$ and rewrite our multiplier identity as $r_k = u_k r_0 + v_k r_1$. This now gives the appropriate notation to present the extended Euclidean algorithm.

Algorithm 2: The Extended Euclidean Algorithm

Input: Two polynomials $f, g \in \mathbb{F}_p[x]$ with $f \geq g$

Output: The greatest common divisor of f and g , and the multipliers u and v .

$r_0 := f(x)$

$r_1 := g(x)$

$k := 1$

$u_0 := 1, u_1 := 0, v_0 := 0, v_1 := 1$

while $(r_k \neq 0)$

$r_{k-1} := q_k r_k + r_{k+1}$

$u_{k+1} := u_{k-1} - q_k u_k$

$v_{k+1} := v_{k-1} - q_k v_k$

$k := k + 1$

return r_{k-1}

Example 3.2.1 Find the greatest common divisor and the multipliers of $f(x) = x^5 + 2x^4 + 3x^3 + 2x^2$ and $g(x) = x^4 + 2x^3 + 2x^2 + x$ over the field $\mathbb{F}_5$

k	r_k	q_k	u_k	v_k
1	$x^5 + 2x^4 + 3x^3 + 2x^2$		1	0
2	$x^4 + 2x^3 + 2x^2 + x$	x	0	1
3	$x^3 + x^2$	$x + 1$	1	$4x$
4	$x^2 + x$	x	$4x + 4$	$x^2 + x + 1$
5	0			

Table 1: The Execution of the Extended Euclidean Algorithm.

The extended Euclidean algorithm gives $x^2 + x = (x^5 + 2x^4 + 3x^3 + 2x^2)(4x + 4) + (x^4 + 2x^3 + 2x^2 + x)(x^2 + x + 1)$. ■

We want to point out that the degrees of q_k , u_k , and v_k can be written with respect to the degree of r_k . This is accomplished by

$$\deg q_k = d_{k-1} - d_k \quad \text{for } k \geq 1, \quad (3.1)$$

$$\deg u_k = d_1 - d_{k-1} \quad \text{for } k \geq 2, \quad (3.2)$$

$$\deg q_k = d_0 - d_{k-1} \quad \text{for } k \geq 2. \quad (3.3)$$

Finally, it is noteworthy to mention that the Euclidean algorithm and its extension as presented here are not necessarily the most efficient methods to compute greatest common divisors. Stein [27] discovered an analogous algorithm to compute GCDs over binary numbers. This method is called the binary Euclidean algorithm. The algorithm is efficient because it replaces costly divisions by shifting bits of the binary representation. This binary Euclidean algorithm and its binary extended Euclidean algorithm would yield slightly better results in $\mathbb{F}_2$.

3.3 The Expected Number of Field Operations for the Euclidean Algorithm

We want to determine the expected number of field operations in $\mathbb{F}_q$ used by the extended Euclidean algorithm. The first required step needed is to compute various probabilities incurred by the two input polynomials. Most of this section follows the outline established by Enge [7]. Since addition and subtraction are relatively inexpensive compared to multiplication and inversion, we only compute the expected values for the number of inversions and multiplications in $\mathbb{F}_q$.

For this chapter, we use the same assumptions of Enge on the cost of the polynomial arithmetic. For a large q , we assume that all polynomials have non zero coefficients. This assumption is reasonable because the number of non-zero coefficients of a random polynomial of degree n over $\mathbb{F}_q$ is $\text{Binomial}(n, 1 - 1/q)$ distributed. As q grows larger, $1 - 1/q$ approaches 1. The loss of precision of this assumption is negligible. This assumption does not hold true for $q = 2$, where each coefficient of a polynomial has a 50% probability of being non-zero.

In order to compute probabilities in $\mathbb{F}_q$ some mechanisms must first be implemented. The computation is accomplished with the aid of some conditional probabilities. The following lemma helps establish the probability of successive steps in the Euclidean algorithm.

Let $f_1 \in \mathbb{F}_q[x]$ be uniformly distributed over the polynomials of degree α_1 , $f_2 \in \mathbb{F}_q[x]$ be fixed of degree $\alpha_2 \leq \alpha_1$, with s the quotient and f_3 the remainder of f_1 divided by f_2 , that is, $f_1 = s \cdot f_2 + f_3$. Then s is uniformly distributed over the polynomials of degree $\alpha_1 - \alpha_2$ and f_3 is uniformly and independently of s distributed over the polynomials of degree less than α_2 .

Lemma 3.3.1 *With the previous notation, it follows that for $\mu > \nu \geq 0$*

$$\begin{aligned} P(d_k = \nu \mid d_{k-1} = \mu) &= \frac{\text{number of polynomials of degree } \nu}{\text{number of polynomials of degree less than } \mu} \\ &= \frac{(q-1)q^\nu}{q^\mu} \\ P(d_k = -\infty \mid d_{k-1} = \mu) &= \frac{1}{q^\mu}. \end{aligned}$$

Proof. For the first probability, assuming the leading coefficient is not zero, then there are $(q-1)q^\nu$ polynomials of degree ν . The second probability is easier to see. There are q^μ polynomials of degree less than μ . Only one of these can satisfy the condition that $d_k = -\infty$. ■

The previous lemma gives us a powerful tool that allows us to compute the total probability of d_k . The following lemma gives the probability that $d_k = -\infty$ as well as the probability of d_k for any value of k in the sequence.

Lemma 3.3.2

$$P(d_k = -\infty) = \begin{cases} \frac{1}{q^{d_1}} \sum_{i=0}^{k-2} \binom{d_1}{i} (q-1)^i & \text{for } 2 \leq k \leq d_1 + 1, \\ 1 & \text{for } k \geq d_1 + 2. \end{cases}$$

$$P(d_k = \nu) = \begin{cases} \frac{(q-1)^{k-1}}{q^{d_1}} q^\nu \binom{d_1-\nu-1}{k-2} & \text{for } 2 \leq k \leq d_1 + 1 \text{ and } 0 \leq \nu \leq d_1 - k + 1, \\ 0 & \text{otherwise.} \end{cases}$$

Proof.

$$\begin{aligned} P(d_{k+1} = -\infty) &= \sum_{\mu=0}^{d_1-k+1} P(d_k = \mu) P(d_{k+1} = -\infty \mid d_k = \mu) + P(d_k = -\infty) \\ &= \sum_{\mu=0}^{d_1-k+1} \frac{(q-1)^{k-1}}{q^{d_1}} q^\mu \binom{d_1-\mu-1}{k-2} \frac{1}{q^\mu} + \frac{1}{q^{d_1}} \sum_{i=0}^{k-2} \binom{d_1}{i} (q-1)^i \\ &= \frac{(q-1)^{k-1}}{q^{d_1}} \sum_{\mu=k-2}^{d_1-1} \binom{\mu}{k-2} + \frac{1}{q^{d_1}} \sum_{i=0}^{k-2} \binom{d_1}{i} (q-1)^i \\ &= \frac{(q-1)^{k-1}}{q^{d_1}} \binom{d_1}{k-1} + \frac{1}{q^{d_1}} \sum_{i=0}^{k-2} \binom{d_1}{i} (q-1)^i, \end{aligned}$$

where we used a derivation from Section 2.4. For the second probability, let $0 \leq \nu \leq d_1 - k$ which gives

$$\begin{aligned}
 P(d_{k+1} = \nu) &= \sum_{\mu=\nu+1}^{d_1-k+1} P(d_k = \mu) P(d_{k+1} = \nu \mid d_k = \mu) \\
 &= \sum_{\mu=\nu+1}^{d_1-k+1} \frac{(q-1)^{k-1}}{q^{d_1}} q^\mu \binom{d_1 - \mu - 1}{k-2} \frac{(q-1)q^\nu}{q^\mu} \\
 &= \frac{(q-1)^k}{q^{d_1}} q^\nu \sum_{\mu=k-2}^{d_1-\nu-2} \binom{\mu}{k-2} \\
 &= \frac{(q-1)^k}{q^{d_1}} q^\nu \binom{d_1 - \nu - 1}{k-1}.
 \end{aligned}$$

■

It is very noteworthy to mention some issues with respect to the boundary conditions. If we want the probability that $d_k = \nu$ where k is the final index such that d_k is the greatest common divisor, then the above formula works perfectly. If however, we want to compute the probability for $d_{k-1} = \nu$, we still need to stay within the boundary conditions. This is accomplished by subtracting the appropriate value from k in the equation. In the case of $d_{k-1} = \nu$, this would yield

$$\frac{(q-1)^{k-2}}{q^{d_1}} q^\nu \binom{d_1 - \nu - 1}{k-3},$$

whereas $d_{k-\ell} = \nu$ for an integer ℓ would be

$$\frac{(q-1)^{k-1-\ell}}{q^{d_1}} q^\nu \binom{d_1 - \nu - 1}{k-2-\ell}.$$

The next lemma gives the probability distribution of t . An alternative proof can be found in Ma and von zur Gathen [16].

Lemma 3.3.3

$$P(t = k) = \begin{cases} \binom{d_1}{k-1} \frac{(q-1)^{k-1}}{q^{d_1}} & \text{for } 1 \leq k \leq d_1 + 1, \\ 0 & \text{otherwise.} \end{cases}$$

Proof.

$$\begin{aligned}
P(t = k) &= P(d_{k-1} \neq -\infty)P(d_k = -\infty \mid d_{k-1} \neq -\infty) \\
&= \sum_{\mu=0}^{\infty} P(d_{k-1} = \mu)P(d_k = -\infty \mid d_{k-1} = \mu) \\
&= \sum_{\mu=0}^{\infty} \frac{(q-1)^{k-1}}{q^{d_1}} q^{\mu} \binom{d_1 - \mu - 1}{k-2} \frac{1}{q^{\mu}} \\
&= \frac{(q-1)^{k-1}}{q^{d_1}} \sum_{\mu=0}^{\infty} \binom{d_1 - \mu - 1}{k-2} \\
&= \frac{(q-1)^{k-1}}{q^{d_1}} \sum_{\mu=k-2}^{d_1-1} \binom{\mu}{k-2} \\
&= \frac{(q-1)^{k-1}}{q^{d_1}} \binom{d_1}{k-1}.
\end{aligned}$$

■

We need one final set of probabilities before we can reach this chapter's main theorem. In order to work with hyperelliptic cryptosystems, we need to compute the multipliers u and v . So we need to know the values of d_t with respect to u and v .

Lemma 3.3.4

$$\begin{aligned}
P(d_{t-1} = \mu \text{ AND } d_t = \nu) &= \begin{cases} \frac{1}{q^{d_1}} & \text{for } \mu = d_0 \text{ and } \nu = d_1, \\ \frac{q-1}{q^{d_1}} & \text{for } \mu = d_0 \text{ and } 0 \leq \nu < d_1, \\ \frac{(q-1)^2}{q^{\mu+1}} & \text{for } 1 \leq \mu < d_1 \text{ and } 0 \leq \nu < \mu, \\ 0 & \text{otherwise;} \end{cases} \\
P(d_{t-1} = \mu) &= \begin{cases} \frac{1}{q^{d_1}} & \text{for } \mu = d_0, \\ d_1 \frac{q-1}{q^{d_1}} & \text{for } \mu = d_1, \\ \mu \frac{(q-1)^2}{q^{\mu+1}} & \text{for } 1 \leq \mu < d_1, \\ 0 & \text{otherwise;} \end{cases} \\
P(d_t = \nu) &= \begin{cases} \frac{1}{q^{d_1}} & \text{for } \nu = d_1, \\ \frac{(q-1)}{q^{\nu+1}} & \text{for } 0 \leq \nu < d_1, \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

NOTE TO USERS

Page(s) not included in the original manuscript and are unavailable from the author or university. The manuscript was scanned as received.

This reproduction is the best copy available.

UMI[®]

the number of multiplications

$$\sum_{k=1}^t (d_k + 1)(d_{k-1} - d_k + 1),$$

since in step k the degree of the polynomials are d_{k-1} and d_k . Counting the number of inversions is simpler, as there is 1 inversion per iteration which derives $|k = 1, \dots, t : d_t \neq 0|$ inversions. The extended Euclidean algorithm requires some additional multiplications in the computation of the multipliers u and v . The multipliers u_k requires

$$\sum_{k=4}^t (d_{k-2} - d_{k-1} + 1)(d_1 - d_{k-2} + 1)$$

multiplications whereas the multipliers v_k requires

$$\sum_{k=3}^t (d_{k-2} - d_{k-1} + 1)(d_1 - d_{k-2} + 1)$$

multiplications. While all three summations are similar in nature, they all have different lower bounds. The variable k is bounded by $1 \leq k \leq d_1 + 1$, but some values of k are always the same in the beginning of the extended Euclidean algorithm. It always holds that $u_2 = 1$, $u_3 = -q_2$, and $v_2 = -q_1$ are guaranteed to be computed by the algorithm, and so the lower indices of the summations can be adjusted accordingly. We use three formulas from Section 3.2 which can express the degrees of q_k , u_k , and v_k in terms of d_k .

Having counted the number of multiplications and inversions for simple cases, we next want to do this for all possible combinations of outputs for the extended Euclidean algorithm. Thus, the expected number of field operations are given in the following lemma.

Lemma 3.3.6 *Let $E_i(d_0, d_1)$ and $E_m(d_0, d_1)$ be the expected number of inversions and multiplications in $\mathbb{F}_q$ performed by the non-extended Euclidean algorithm. Denote by $E^u(d_0, d_1)$ and $E^v(d_0, d_1)$ the expected number of additional multiplications needed to compute the multipliers u and v in the extended Euclidean algorithm. The expected num-*

ber of field operations are given by

$$\begin{aligned}
E_i(d_0, d_1) &= \sum_{\tau=1}^{d_1+1} \tau P(t = \tau) - P(\deg \gcd(d_0, d_1) = 0) \\
E_m(d_0, d_1) &= (d_1 + 1)(d_0 - d_1 + 1) + \sum_{\mu=1}^{\infty} P(d_2 = \mu)(\mu + 1)(d_1 - \mu + 1) \\
&\quad + \sum_{k=3}^{\infty} \sum_{\mu=0}^{\infty} \sum_{\nu=1}^{\infty} P(d_{k-1} = \mu \text{ AND } d_k = \nu)(\nu + 1)(\mu - \nu + 1) \\
E^u(d_0, d_1) &= \sum_{k=4}^{\infty} \sum_{\mu=0}^{\infty} \sum_{\nu=0}^{\infty} P(d_{k-2} = \mu \text{ AND } d_{k-1} = \nu \text{ AND } d_k \neq -\infty) \\
&\quad \times (\mu - \nu + 1)(d_1 - \mu + 1) \\
E^v(d_0, d_1) &= \sum_{k=3}^{\infty} \sum_{\mu=0}^{\infty} \sum_{\nu=0}^{\infty} P(d_{k-2} = \mu \text{ AND } d_{k-1} = \nu \text{ AND } d_k \neq -\infty) \\
&\quad \times (\mu - \nu + 1)(d_0 - \mu + 1).
\end{aligned}$$

■

Given the degrees of any two polynomials, we can compute the number of multiplications required for the result. After executing the extended Euclidean algorithm, we know the probability that the polynomial result is of a certain degree. So we can compute the expected value by summing all possible combinations of polynomials times their probability. This is represented by the formulas above.

Having established some formulas to determine the expected number of field operations, we can compute such formulas to determine the cost of each operation. First, we must know the upper bounds of the formulas in Lemma 3.3.6.

Lemma 3.3.7 *The random variable k representing the number of iterations of the Euclidean algorithm is bounded by $1 \leq k \leq d_1 + 1$. The random variable μ comes from Lemma 3.3.2 as well as the probability that $d_k = \mu$, where μ is bounded by $0 \leq \mu \leq d_1 - k + 1$. The random variable ν is bounded by $0 \leq \nu \leq \mu - 1$.*

Proof. Given two input polynomials r_0 and r_1 of degrees d_0 and d_1 respectively, if the polynomials r_i were to monotonically decrease by 1 degree each iteration, we would

reach the constant polynomial of degree zero and greatest common divisor in d_1 iterations. Including the computation for d_1 itself increases one to this total and is then bounded by $d_1 + 1$.

The upper bound for μ comes from Lemma 3.3.2. Please note the discussion after Lemma 3.3.2 with regards to the bounds for d_{k-1} and $d_{k-\ell}$. Given that k can be at most $d_1 + 1$, it works out nicely that any value of k greater than $d_1 + 1$ makes the binomial in Lemma 3.3.2 equal to 0, and such does not add any values to the summation.

Finally, we know that the degree of the remainder is less than the degree of the divisor, and hence $\mu > \nu \geq 0$. In order to sum all of the possible outcomes, ν is bounded by $\mu - 1$. ■

Upon substituting the upper bounds into the formulas in Lemma 3.3.6, we can now compute the expected number of field operations using classical arithmetic. In Chapter 6, we do the same process only we consider Karatsuba multiplication for better asymptotic results.

Theorem 3.3.1 (*Enge*). *Let r_0 and r_1 be uniformly distributed over the polynomials over $\mathbb{F}_q$ of degrees $d_0 \geq d_1 \geq 1$, respectively. Denote by $E_i(d_0, d_1)$ and $E_m(d_0, d_1)$ the expected number of inversions and multiplications in $\mathbb{F}_q$ performed by the non-extended Euclidean algorithm. Let $E^u(d_0, d_1)$ and $E^v(d_0, d_1)$ be the expected number of additional multiplications needed to compute the multipliers u and v , respectively. Then*

$$\begin{aligned}
 E_i(d_0, d_1) &= d_1 - \frac{d_1 - 1}{q} \\
 E_m(d_0, d_1)(q - 1)^2 q^{d_1} &= (d_0 d_1 + d_0 + d_1 - 1) q^{d_1 + 2} \\
 &\quad - \left(2d_0 d_1 + \frac{1}{2} d_1^2 + 2d_0 + \frac{7}{2} d_1 - 3 \right) q^{d_1 + 1} + (d_0 d_1 + d_1^2 + d_0 + 3d_1 - 2) q^{d_1} \\
 &\quad - \left(\frac{1}{2} d_1^2 + \frac{1}{2} d_1 - 1 \right) q^{d_1 - 1} + q^2 - 2q.
 \end{aligned}$$

$$\begin{aligned}
E^u(d_0, d_1)(q-1)^2 q^{d_1} &= (d_1^2 - d_1 - 2) q^{d_1+2} - \left(\frac{5}{2} d_1^2 + \frac{5}{2} d_1 - 10 \right) q^{d_1+1} \\
&+ (2d_1^2 + 5d_1 - 2) q^{d_1} - \left(\frac{1}{2} d_1^2 + \frac{3}{2} d_1 \right) q^{d_1-1} + \left(\frac{1}{2} d_1^2 + \frac{3}{2} d_1 \right) q^3 \\
&+ (2d_1^2 + 5d_1 - 2) q^2 + \left(\frac{5}{2} d_1^2 + \frac{5}{2} d_1 - 10 \right) q - (d_1^2 - d_1 - 2) \\
E^v(d_0, d_1)(q-1)^2 q^{d_1} &= (2d_0 - d_1)(d_1 - 1) q^{d_1+2} - \left(5d_0 d_1 - \frac{5}{2} d_1^2 - 2d_0 + \frac{9}{2} d_1 - 7 \right) q^{d_1+1} \\
&+ (4d_0 d_1 - 2d_1^2 - 5d_1 - 1) q^{d_1} - \left(d_0 d_1 - \frac{1}{2} d_1^2 + \frac{3}{2} d_1 \right) q^{d_1-1} \\
&+ \left(d_0 d_1 - \frac{3}{2} d_1^2 + 2d_0 - \frac{7}{2} d_1 \right) q^2 - (2d_0 d_1 - 3d_1^2 + 2d_0 - 2d_1 + 7) q \\
&+ \left(d_0 d_1 - \frac{3}{2} d_1^2 + \frac{3}{2} d_1 + 1 \right)
\end{aligned}$$

Before presenting the heavily computational proof, we wish to mention that the formulas above for the multiplications are presented with the division of the very common term $(q-1)^2 q^{d_1}$. This is done to make the formulas more presentable.

Proof. The proof of this theorem consists of mathematical derivations of the above formulas. It is very long and involves many pages. The proofs can be found in Appendix B. ■

The original paper by Enge does not include the proof of this theorem in detail.

Chapter 4

Elliptic and Hyperelliptic Curves

We now turn our attention to algebraic curves. As we shall see, there exists a beautiful geometric method to add points on an elliptic curve. This gives the points on an elliptic curve a group structure. A hyperelliptic curve is a generalization of an elliptic curve. However, the geometric method does not work on a hyperelliptic curve. Using algebraic geometry, we can create a group operation by considering the Jacobian of a hyperelliptic curve.

This chapter begins with a brief review of elliptic and hyperelliptic curves. Following this, we introduce some algebra to define the Jacobian of an algebraic curve. Finally we discuss the composition and reduction algorithms required for hyperelliptic cryptography.

There is a lot of literature on elliptic curves. An excellent elementary introduction to elliptic curves is written by Charlap and Robbins [4]. Menezes [17] as well as Blake, Seroussi, and Smart [2] discuss elliptic curve cryptography, where as Koblitz [14] has the best overview of hyperelliptic cryptography. A great book on how to implement elliptic curve cryptography in C is by Rosing [23]. Finally, consult Silverman [26] for an advanced reading on the theory on elliptic curves.

4.1 Elliptic Curves

Elliptic curves were first proposed independently for use in cryptography by Koblitz [12] and Miller [18]. Let K denote a finite field of characteristic p with p^n elements. Let $\overline{K}$ be its algebraic closure. The set $K \times K$ is called the affine plane over K .

Definition 4.1.1 *An elliptic curve E over K is the set of solutions in $\overline{K} \times \overline{K}$ to the Weierstrass equation*

$$E : Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6 \quad (4.1)$$

with $a_1, \dots, a_6 \in K$ and at least one of the two partial derivatives is non-zero at P . There also exists a special point O which denotes the point at infinity in union with the set of solutions in $\overline{K} \times \overline{K}$.

When a curve E is defined over the Weierstrass equation with $a_1, \dots, a_6$ in K , then we denote this by E/K . If we consider a curve with points in K instead of $\overline{K}$, then the k -rational points is denoted by $E(K)$.

The Weierstrass Equation (4.1) works for fields of all characteristics. We can however find simpler isomorphic curves for different characteristics, but for this we need the notion of an admissible change of variables. This change of variables maps a curve E_1 to E_2 . If the characteristic of K for the curve E/K is not equal to 2 or 3, then the admissible change of variables

$$(x, y) \rightarrow \left(x, y - \frac{a_1}{2}x - \frac{a_3}{2} \right)$$

maps the Equation (4.1) to

$$E'/K : Y^2 = X^3 + b_2X^2 + b_4X + b_6 \quad \text{with } b_2, b_4, b_6 \in K.$$

Applying the following change of variables

$$(x, y) \rightarrow \left(\frac{x - 3b_2}{36}, \frac{y}{216} \right)$$

to E' maps the Equation (4.1) to

$$E''/K : Y^2 = X^3 + aX + b \quad \text{with } a, b \in K,$$

which is referred to as the short Weierstrass equation. In practice, fields of characteristic 2 are best suited for computer implementation. The admissible change of variables

$$(x, y) \rightarrow \left(a_1^2 x + \frac{a_3}{a_1}, a_1^3 y + \frac{a_1^2 a_4 + a_3^2}{a_1^3} \right)$$

maps Equation (4.1) to

$$E'''/K : Y^2 + XY = X^3 + a_2 X^2 + a_6 \quad \text{with } a_2, a_6 \in K. \quad (4.2)$$

Having defined what an elliptic curve is, we next want to give it a group structure. This is done by defining a group law that allows the addition of two points on a curve. Given two points P and Q on E , a line through these two points intersects the curve E at exactly one other point R . If a second line is drawn between R and the point of infinity O , this is represented as a vertical line through R which crosses the elliptic curve at one point R' . In general, the inverse of a point $P = (x, y)$ on an elliptic curve is $(x, -y)$, and is denoted as $-P$. This now derives $P + Q = R'$. Adding a point to itself is called doubling a point. This is accomplished by taking the tangent to the point to be doubled, and then following the above procedure in which case $P + P = R'$.

The group law requires a rigorous definition. We do this only for fields of characteristic two, as these are the main interest for implementable reasons. Let E be an elliptic curve defined over K such that

$$E/K : Y^2 + XY = X^3 + a_2 X^2 + a_6 \quad (4.3)$$

with $a_2, a_6 \in K$. Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be two solutions to E/K such that $P, Q \neq O$, then

1. $P + O = O + P = P$
2. $P + (-P) = O$

3. If $P \neq Q$, then $P + Q = R'$ for $R' = (x_3, y_3)$ where

$$\begin{aligned}\lambda &= \frac{y_1 + y_2}{x_1 + x_2} \\ x_3 &= \lambda^2 + \lambda + x_1 + x_2 + a_2 \\ y_3 &= (x_1 + x_3)\lambda + x_3 + y_1\end{aligned}$$

4. If $P = Q$, then $P + Q = P + P = R'$ for $R' = (x_3, y_3)$ where

$$\begin{aligned}\lambda &= \frac{y_1}{x_1} + x_1 \\ x_3 &= \lambda^2 + \lambda + a_2 \\ y_3 &= (x_1 + x_3)\lambda + x_3 + y_1\end{aligned}$$

4.2 Hyperelliptic Curves

Unfortunately, a given line on a hyperelliptic curve may intersect the curve at more than 3 points, and as a result the geometric method for elliptic curves is rendered useless for hyperelliptic curves. However, we can use tools from algebraic geometry to define the Jacobian of an algebraic curve, in which case we can then create an addition operator for elements in the Jacobian.

Definition 4.2.1 *A hyperelliptic curve C of genus $g \geq 1$ over K is an equation of the form*

$$C : Y^2 + H(X)Y = F(X) \text{ in } K[X, Y] \quad (4.4)$$

where $H(X) \in K[X]$ is a polynomial of degree at most g and $F(X) \in K[X]$ is a monic polynomial of degree $2g + 1$. A hyperelliptic curve has no singular points $(x, y) \in K \times K$ that simultaneously satisfy Equation (4.4) and the two partial derivatives $2Y + H(X) = 0$ and $H'(X)Y - F'(X) = 0$.

For convenience and readability, we let $H(X) = h$ and $F(X) = f$ when using algorithms. This gives us $C : Y^2 + hY = f$.

An elliptic curve is a hyperelliptic curve of genus 1. All of Section 4.1 would apply to a genus 1 hyperelliptic curve. There are hyperelliptic curves for each genus $g \geq 1$.

If $P = (a, b)$ is a finite point on C , we say that the opposite of P is another point on the curve $\tilde{P} = (a, -b - h(a))$. The opposite of the point at infinity O is defined to be itself such that $\tilde{O} = O$. A point P is called *special* if $\tilde{P} = P$, whereas a point P is called *ordinary* if $\tilde{P} \neq P$.

Next we present an example over the field $\mathbb{F}_{11}$.

Example 4.2.1 Consider the hyperelliptic curve $C : Y^2 + (X^2 + 5)Y = X^5 + 2X + 1$ over the finite field $\mathbb{F}_{11}$. This curve has genus $g = 2$ with $H(X) = X^2 + 5$ and $F(X) = X^5 + 2X + 1$. The two partial derivatives of C are non-zero, and such there are no singular points other than O . The rational points $C(\mathbb{F}_{11})$ are

$$C(\mathbb{F}_{11}) = \{O, (1, 2), (1, 3), (3, 0), (3, 8), (5, 7), (6, 6), (6, 8)\},$$

where $P = (5, 7)$ is a special point as $\tilde{P} = (a, -b - H(a)) = (5, -7 - (5^2 + 5)) = (5, 7)$.

4.3 Polynomial and Rational Functions

The theory of polynomial and rational functions comes from algebraic geometry. A multivariate polynomial in the field $K[X, Y]$ is an expression of the form

$$C(X, Y) = \sum_{i,j} a_{ij} X^i Y^j, \quad a_{ij} \in K.$$

The degree of each individual term $a_{ij} X^i Y^j$ is $i + j$. The degree of the polynomial C is the maximum of the degrees of all terms. A root or zero of C is a point $P = (x, y) \in \overline{K} \times \overline{K}$ that satisfies $C(P) = 0$.

Definition 4.3.1 A polynomial on C is an element of $\overline{K}[X, Y]$ denoted by $\overline{K}[H]$.

Definition 4.3.2 The coordinate ring of C over K is the quotient group $K[C] = K[X, Y] / (C(X, Y))$. This definition can equally be stated for the algebraic closure of K such that $\overline{K}[C] = \overline{K}[X, Y] / (C(X, Y))$.

Given any polynomial $f(x, y)$ on C , there exists a unique canonical form $f(x, y) = s(x) + Yt(x)$ where $s(x)$ and $t(x)$ are univariate polynomials in the indeterminate X .

Next we want to define a rational function, where a rational function is a quotient of two polynomials. To do this, we need to define a field of fractions for the coordinate ring.

Definition 4.3.3 *A rational function on C is an equivalence class of formal quotients of polynomials f/g (with g not identically zero), where we identify f/g with h/s if $fs = gh$ as polynomials on C . The set of rational functions on C is a field denoted by $K(C)$.*

We observe that the ring of polynomials $K[C]$ is a subring of the ring of rational functions $K(C)$, and as such every polynomial function is a rational function. Next we want to define the value of rational functions on C .

Definition 4.3.4 *If r is a rational function on C and P is a finite point in C , we say that r is finite at P if there exists a representation $r = f/g$ where f and g are polynomials on C and $g(P) \neq 0$. If r is finite at P , we define $r(P) = f(P)/g(P)$.*

As we have just shown, the value of a rational function for finite points is easily defined. The value of a rational function at infinity is a bit more complicated to define, and requires the notion of the degree of a polynomial on C .

Definition 4.3.5 *Let $f(x, y) = s(x) + Yt(x)$ be a non-zero polynomial function on C . The degree of f is defined by*

$$\deg(f) = \max\{2 \cdot \deg_x(s), 3 + 2 \cdot \deg_x(t)\}.$$

Now we can define the value of a rational function at the point of infinity.

Definition 4.3.6 *Let $r = f/g$ be a rational function on C . Then*

- 1) *if $\deg(f) < \deg(g)$, the value of r at ∞ is defined to be $r(\infty) = 0$;*
- 2) *if $\deg(f) > \deg(g)$, the value of r is not defined at ∞ ;*

3) if $\deg(f) = \deg(g)$, then writing f and g in canonical form we derive either ax^d and bx^d if the degree of f is even, or ayx^d and byx^d if the degree of f is odd. In this case we define the value of r at ∞ to be $r(\infty) = a/b$, which is the ratio of the leading coefficients for f and g .

4.4 Zeros and Poles

Having defined what rational functions are, we next need to develop some theory to suit rational functions to our needs. First we need to define a zero and a pole.

Definition 4.4.1 *Let r be a rational function on C and $P = (x, y) \in C$. Then r has a zero at P if $r(P) = 0$ and r has a pole at P if $r(P) = \infty$.*

In order to compute the multiplicity, we need the following theorem.

Definition 4.4.2 *Let P be a point on C . There exists a rational function u that is zero at P , and has the property that if r is any rational function not identically zero, then $r = u^d s$, for some integer d and some rational function s that is finite and non-zero at P . Furthermore, the number d does not depend on the choice of the rational function u . The function u is called a uniformizer or a uniformizing parameter for P .*

Now we can compute the order or the multiplicity of a pole or a zero.

Definition 4.4.3 *If r is a rational function and $r = u^d s$ where u is a uniformizing parameter at P , then the order of r at P is d where d is defined as*

$$\text{ord}_P(r) = d.$$

If r has a zero or pole at P , we define its multiplicity as $|\text{ord}_P(r)|$.

Definition 4.4.4 *Let r be a rational function on C . Then*

$$\sum_{P \in C} \text{ord}_P(r) = 0.$$

The following example is expanded on from [17].

Example 4.4.1 Consider the elliptic curve $E : Y^2 = X^3 - X$ over $\mathbb{F}_q$ where the characteristic of the field is greater than 3. Let $f = (X^2 - X)/Y$ be a non-zero rational function. Take the point $P = (1, 0)$ which is clearly on E . If we consider f as a quotient of polynomials, then for the point P we get

$$f = \frac{X^2 - X}{Y} = \frac{1^2 - 1}{0}$$

which is undefined in the denominator for the point P . If we consider f as a rational function, then multiplying f by Y/Y derives

$$f = \frac{X^2 - X}{Y} = \frac{(X^2 - X)Y}{Y^2} = \frac{(X^2 - X)Y}{X^3 + X} = \frac{(X)(X - 1)Y}{X(X - 1)(X + 1)} = \frac{Y}{X + 1}.$$

Next plug in P to get

$$f = \frac{Y}{X + 1} = \frac{0}{1 + 1} = \frac{0}{2} = 0.$$

Clearly $f(P) = 0$, and such f has a pole at P .

4.5 Divisors

We need a device to encode the multiplicities of poles or zeros for a rational function, and this is accomplished with divisors. Arithmetic on hyperelliptic curves is done with the Jacobian of a curve, which is introduced in this section.

Definition 4.5.1 A divisor is a formal sum of points on C of the form

$$D = \sum_{P \in C} n_P(P),$$

where $n_P \in \mathbb{Z}$, and only finitely many of these are non-zero. We denote the degree of D by $\deg D$, and the degree of D is an integer where $\deg D = \sum_{P \in C} n_P$. The order of D at P is the integer n_P .

The set of all points where $n_P \neq 0$ is called the support of the divisor. The set of all divisors form an additive group under the addition law

$$\sum_{P \in C} n_P P + \sum_{P \in C} m_P P = \sum_{P \in C} (n_P + m_P) P.$$

This group is called the divisor group of C , and is denoted as $\text{Div}(C)$. The set of divisors of degree 0 is denoted as $\text{Div}^0(C)$, and is a subgroup of $\text{Div}(C)$.

For any rational function r on C , the divisor of r is

$$\text{div}(r) = \sum_{P \in C} \text{ord}_P(r)(P).$$

The following two definitions define the Jacobian of a curve.

Definition 4.5.2 *A divisor $D \in \text{Div}(C)$ is principal if it is the divisor of some function r such that $D = \text{div}(r)$. The group of principal divisors is denoted by $\text{Prin}(C)$. Two divisors D_1 and D_2 are linearly equivalent if $D_1 - D_2$ is principal. We denote linearly equivalent divisors as $D_1 \sim D_2$. The divisor class group or Picard group of C is denoted as $\text{Pic}(C)$ and is defined by the quotient group*

$$\text{Pic}(C) = \frac{\text{Div}(C)}{\text{Prin}(C)}$$

Definition 4.5.3 *The Jacobian of a curve C is the degree zero part of the Picard group, and is denoted by $\text{Pic}^0(C)$. Thus the Jacobian is the quotient group*

$$\text{Pic}^0(C) = \frac{\text{Div}^0(C)}{\text{Prin}(C)}$$

Finally, we can define the greatest common divisor of two divisors. The GCD of two distinct divisors is always a degree 0 divisor.

Definition 4.5.4 *Let $D_1 = \sum_{P \in C} n_P P$ and $D_2 = \sum_{P \in C} m_P P$ be two divisors. The greatest common divisor of D_1 and D_2 is defined to be*

$$\text{gcd}(D_1, D_2) = \sum_{P \in C} \min(n_P, m_P) P - \left(\sum_{P \in C} \min(n_P, m_P) P \right) \infty.$$

4.6 Semi-Reduced and Reduced Divisors

Next we want to define a semi-reduced divisor, and then find a method to build a polynomial representation for semi-reduced divisors.

Definition 4.6.1 *A semi-reduced divisor is a divisor of the form*

$$D = \sum m_i P_i - \left(\sum m_i \right) \infty,$$

where each $m_i \geq 0$ and the P_i 's are finite points such that when $P_i \in \text{supp}(D)$, one has $P_i \notin \text{support}(D)$, unless $P_i = \tilde{P}_i$ in which case $m_i = 1$.

For every degree 0 divisor, there exists a semi-reduced divisor $D_1 \in \text{Div}^0(C)$ such that $D \sim D_1$.

We have a nice formal definition of a semi-reduced divisor. We need to establish a polynomial representation for a semi-reduced divisor that we can do computations with. The following lemma is first needed to accomplish this.

Lemma 4.6.1 *Let $P = (x, y)$ be an ordinary point on C . Then for each $k \geq 1$ there exists a unique polynomial b such that*

- 1) $\deg_x b < k$
- 2) $b(x) = y$
- 3) $b^2 + bh \equiv f \pmod{(X - x)^k}$

Any semi-reduced divisor can be uniquely represented as the gcd of two polynomial functions. The above lemma is used to compute the second argument in the polynomial representation. The following theorem is crucial to the computational aspect of hyperelliptic cryptography.

Theorem 4.6.1 *Let $D = \sum m_i P_i - (\sum m_i) \infty$ be a semi-reduced divisor with $P_i = (x_i, y_i)$. Let $A(X) = \prod (X - x_i)^{m_i}$. There exists a unique polynomial $B(X)$ satisfying 1) $\deg_x b < \deg_x a$, 2) $B(x_i) = y_i$ for all i for which $m_i \neq 0$, and 3) $A(X)$ divides $B(X)^2 + B(X)h(X) - f(X)$. Then $D = \gcd(\text{div}(A(X)), \text{div}(B(X) - Y))$.*

It should be noted that a divisor $\text{div}(a, b)$ is semi-reduced if and only if $a \mid (b^2 + bh - f)$. Finally, this leads us to the notion of a reduced divisor. Its definition follows.

Definition 4.6.2 *Let $D = \sum m_i P_i - (\sum m_i) \infty$ be a semi-reduced divisor. If $\sum m_i \leq g$, then D is called a reduced divisor.*

As we shall see in the next section, we use a composition and reduction algorithm to compute reduced and semi-reduced divisors. See examples 4.7.1 and 4.7.2 for computation examples on the material from this section.

4.7 Hyperelliptic Cryptography

Hyperelliptic curves were first proposed as an application in cryptography in 1989 by Koblitz [13]. The problem with a hyperelliptic curve is that any given line on a hyperelliptic curve may intersect more than 3 points on the curve. Due to this, the geometric method that is used in elliptic curves can not be applied to hyperelliptic curves. In order to use hyperelliptic curves for cryptography, we need to define a different group structure where we can add elements of the group. We do this in the quotient group called the Jacobian that we defined in Section 4.5. In order to add elements in the Jacobian, we need the algebra presented in the previous sections in this chapter.

The Jacobian can be uniquely represented by reduced divisors. There are two steps to adding elements in the Jacobian. Given two divisors, we first perform a composition step to compute a semi-reduced divisor. We then perform a reduction step to compute a reduced divisor. There is only one composition algorithm, but there are three reduction algorithms due to Gauss, Cantor, and Lagrange. As we shall prove in Chapter 5, Lagrange reduction is the most efficient reduction algorithm.

Hyperelliptic cryptography is considerably slower than elliptic curve cryptography, and in general is very unpractical. The slowest part about hyperelliptic cryptography is

the use of the extended Euclidean algorithm to compute inverses. In order for hyperelliptic curves to be seriously considered for cryptographic purposes, more research needs to be performed to find ways to further increase the arithmetic capabilities and efficiency of hyperelliptic curves. An implementation and discussion on the practical aspects of a hyperelliptic cryptosystem in $\mathbb{F}_p$ and $\mathbb{F}_2^n$ is written by Sakai and Sakurai [24].

Mainstream cryptographic protocols such as the Diffie-Hellman [5] public key exchange and the ElGamal [6] message transmission can be used in the Jacobian in the same manner as they are used for elliptic curves.

The security of hyperelliptic cryptography is based on the presumed intractability of the discrete logarithm problem. Consider two divisors D_1 and D_2 from the Jacobian of a hyperelliptic curve. The hyperelliptic discrete logarithm problem is to determine if an integer n exists such that $D_2 = n D_1$. As in the case for elliptic curve cryptography, subexponential algorithms such as the index calculus [20] are not applicable to the Jacobian of a hyperelliptic curve in general.

There are however some implementation requirements that must be followed for security reasons. When defining the finite field, the prime p must be greater than 2^{160} . This is to protect against square root attacks such as Pollard's Rho [22] or Shank's Baby-Step Giant-Step algorithm [25]. The number of elements in the Jacobian should be divisible by a prime number of at least 40 digits to prevent the Pohlig-Hellman attack [21]. Finally, the genus g of a hyperelliptic curve should not exceed 4. When the genus becomes large with respect to the field size, then there are several variants of subexponential attacks that take advantage of this structure.

4.7.1 Composition Algorithm

This composition algorithm can be traced back to Gauss [8], who used a similar method to compose binary quadratic forms. Cantor [3] used this method of a composition and reduction step to add elements in the Jacobian. This became known as

Cantor's Algorithm. Koblitz [13] modified the formula when introducing hyperelliptic cryptography, and Enge [7] further modified the formula for a field of any characteristic.

INPUT: Two semi-reduced divisors $D_1 = \text{div}(a_1, b_1)$ and $D_2 = \text{div}(a_2, b_2)$.

OUTPUT: A semi-reduced divisor $D_3 = \text{div}(a_3, b_3)$ over F such that $D_3 \sim D_1 + D_2$.

1. Use the extended Euclidean algorithm to find polynomials $\alpha, e_1, e_2 \in F[X]$ such that

$$\alpha = \gcd(a_1, a_2) = e_1 a_1 + e_2 a_2.$$

2. Use the extended Euclidean algorithm to find polynomials $d, c_1, c_2 \in F[X]$ such that

$$d = \gcd(\alpha, b_1 + b_2 + h) = c_1 \alpha + c_2 (b_1 + b_2 + h).$$

3. Let $s_1 = c_1 e_1$, $s_2 = c_1 e_2$, and $s_3 = c_2$ so that $d = s_1 a_1 + s_2 a_2 + s_3 (b_1 + b_2 + h)$.

4. $a_3 = (a_1 a_2) / d^2$

5. $b_3 = b_1 + [s_1 a_1 (b_2 - b_1) + s_3 (f - b_1^2 - b_1 h)] / d \pmod{a_3}$

Steps 1 through 3 explicitly computes $d = \gcd(a_1, a_2, b_1 + b_2 + h)$ which requires two extended Euclidean algorithms. This computes d, s_1, s_2 , and s_3 . In the worst case, a_3 will be of degree $2g$ if $\deg a_1 = \deg a_2 = g$.

Doubling a point is more efficient than the above algorithm. Let $a_1 = a_2$ and $b_1 = b_2$ for a doubled point. In this case, $s_2 = 0$. If we examine fields of characteristic 2 such as $\mathbb{F}_2$ with $h(x) = 1$, then we can further simplify with $d_1 = 1$, $s_1 = s_2 = 0$, and $s_3 = 1$. This transforms the hyperelliptic curve to $Y^2 + Y = f(X)$. Under these conditions, we can double a point as follows.

INPUT: Two semi-reduced divisors $D_1 = \text{div}(a_1, b_1)$ and $D_2 = \text{div}(a_2, b_2)$.

OUTPUT: A semi-reduced divisor $D_3 = \text{div}(a_3, b_3)$ over F such that $D_3 \sim D_1 + D_2$.

1. Set $a_2 = a_1^2$

2. $b_2 = b_1^2 + f \pmod{a}$

The proof of correctness for the composition algorithm can be found in the appendix of the book by Koblitz [14]. The following two examples are from the same source.

Example 4.7.1 Consider the hyperelliptic curve $C : Y^2 + (X^2 + X)Y = X^5 + X^3 + 1$ which has genus $g = 2$ and the values $h = X^2 + X$ and $f = X^5 + X^3 + 1$. Consider the curve C over the field $\mathbb{F}_{2^5}$ with $X^5 + X^2 + 1$ as the irreducible polynomial used to define the field. Take an ordinary point $P = (\alpha^{30}, 0)$ where the opposite of P is $\tilde{P} = (\alpha^{30}, \alpha^{16})$. Let $Q_1 = (0, 1)$ and $Q_2 = (1, 1)$ which are special points.

Let $D_1 = P + Q_1 - 2\infty = (\alpha^{30}, 0) + (0, 1) - 2\infty$ and $D_2 = \tilde{P} + Q_2 - 2\infty = (\alpha^{30}, \alpha^{16}) + (1, 1) - 2\infty$ be two reduced divisors. Compute the composition of the semi-reduced divisor $D = \text{div}(a_3, b_3) = D_1 + D_2$.

To begin this, we first need to compute the divisors D_1 and D_2 . We do this by using Theorem 4.6.1. If we consider our divisor to have the form $D = \sum m_i P_i - (\sum m_i) \infty$, then we compute $a(X)$ by $a(X) = \prod (X - x_i)^{m_i}$ and the unique polynomial $b(x)$ by the conditions required in Theorem 4.6.1. Recall that we are working over characteristic two. So $D_1 = \text{div}(a_1, b_1)$ where $a_1 = (X - \alpha^{30})(X - 0) = (X + \alpha^{30})X$ and $b_1 = \alpha X + 1$. Similarly, $D_2 = \text{div}(a_2, b_2)$ where $a_2 = (X - \alpha^{30})(X - 1) = (X + \alpha^{30})(X + 1)$ and $b_2 = \alpha^{23}X + \alpha^{12}$. Having computed the divisors, we can now apply the composition algorithm to obtain the semi-reduced divisor.

1. $\alpha = \gcd(a_1, a_2) = X - \alpha^{30} = a_1 + a_2$
2. $d = \gcd(\alpha, b_1 + b_2 + h) = X + \alpha^{30} = 1 \cdot d_1 + 0 \cdot (b_1 + b_2 + h)$
3. $d = a_1 + a_2 + 0 \cdot (b_1 + b_2 + h)$
4. $a_3 = (a_1 a_2) / d^2 = X^2 + X$
5. $b_3 \equiv 1 \pmod{a_3}$

which finishes the computation for $D = \text{div}(a_3, b_3)$. ■

4.7.2 Reduction Algorithms

There exists three algorithms we can use for reduction. As we shall see in Chapter 5, the Lagrange reduction is the fastest for most situations. We only discuss two of them: Gauss reduction and Lagrange reduction.

Gauss Reduction

Gauss reduction works by replacing the divisor $\text{div}(a, b)$ by the equivalent divisor $\text{div}(b - Y) - \text{div}(a, b)$, which usually is two degrees smaller than $\text{div}(a, b)$. Gauss reduction's inefficiency is in line 4, where it computes an inverse for each iteration in the for loop. This effectively computes t costly inversions.

INPUT: A semi-reduced divisor $D = \text{div}(a, b)$.

OUTPUT: The unique reduced divisor $D_k = \text{div}(a_k, b_k)$.

1. $a_0 = a$
 2. $b_0 = b$
 3. for $k = 1, \dots, t$ where $t \geq 0$ is the smallest index such that $\deg a_t \leq g$
 4. $a_k = (f - b_{k-1}h - b_{k-1}^2)/a_{k-1}$
 5. $b_k = -h - b_{k-1} \pmod{a_k}$
-

The proof of correctness can once again be found in [14].

Example 4.7.2 Consider the hyperelliptic curve $C : Y^2 + (X^2 + X)Y = X^5 + X^3 + 1$ which has genus $g = 2$ and the values $h = X^2 + X$ and $f = X^5 + X^3 + 1$. Consider the curve C over the field $\mathbb{F}_{2^5}$ with $X^5 + X^2 + 1$ as the irreducible polynomial used to define the field. Consider the semi-reduced divisor $D = (0, 1) + (1, 1) + (\alpha^5, \alpha^{15}) - 3\infty$. We first need to transfer the divisor form to $D = \text{div}(a_0, b_0)$ which is done in the same manner as Example 4.7.1 and Theorem 4.6.1. Upon executing the Gauss reduction algorithm yields

1. $a_0 = (X - 0)(X - 1)(X - \alpha^5) = X(X + 1)(X + \alpha^5) = X^3 + \alpha^2 X^2 + \alpha^5 X$
2. $b_0 = \alpha^{17} X^2 + \alpha^{17} X + 1$
4. $a_1 = [(X^5 + X^3 + 1) - (\alpha^{17} X^2 + \alpha^{17} X + 1)(X^2 + X) - (\alpha^{17} X^2 + \alpha^{17} X + 1)^2] / [X^3 + \alpha^2 X^2 + \alpha^5 X]$
 $= X^2 + \alpha^{15} X + \alpha^{26}$
5. $b_1 = (-X^2 + X) - (\alpha^{17} X^2 + \alpha^{17} X + 1) \pmod{X^3 + \alpha^2 X^2 + \alpha^5 X} = \alpha^{23} X + \alpha^{21}$

This gives the divisor $D = \text{div}(a_1, b_1)$ which is equivalent to $D = (\alpha^{28}, \alpha^7) + (\alpha^{29}, 0) - 2\infty$.

Lagrange Reduction

Lagrange reduction is an improvement on Gauss reduction, as it remedies Gauss reduction's inefficiency in computing t inversions. Lagrange reduction computes the inverse for a_0 , and afterwards the formula for a_k is rewritten using previously computed information.

INPUT: A semi-reduced divisor $D = \text{div}(a, b)$.

OUTPUT: The unique reduced divisor $D_k = \text{div}(a_k, b_k)$.

1. $a_0 = a$
 2. $b_0 = b$
 3. $a_1 = (f - b_0h - b_0^2)/a_0$
 4. $b_1 = -b_0 - h - q_1a_1$ with $\deg b_1 < \deg a_1$
 5. for $k = 2, \dots, t$ where $t \geq 0$ is the smallest index such that $\deg a_t \leq g$
 6. $a_k = a_{k-2} + q_{k-1}(b_{k-1} - b_{k-2})$
 7. $b_k = -b_{k-1} - h - q_ka_k$ with $\deg b_k < \deg a_k$
-

Chapter 5

The Average Cost of Hyperelliptic Cryptography

Having analyzed the Euclidean algorithm for the expected number of field operations and described the operations to add elements in the Jacobian of a hyperelliptic curve, this chapter proceeds to analyze the average cost of adding two such elements. We first analyze the various composition and reduction algorithms, and then compute the average cost for hyperelliptic cryptography. All results are given with respect to the genus g of the curve.

5.1 Heuristic Assumption on Probabilities

Before we can begin to compute the average cost of adding elements in the Jacobian, we first need to state a heuristic that for a random divisor $\text{div}(a, b)$, the polynomials a and b are uniformly distributed over all polynomials of fixed degrees. However, most pairs of (a, b) do not define a divisor, so the curve must vary. The heuristic states that if a and b are not uniformly distributed, the effect on the analysis is negligible. Heuristic 5.1.1 below and the proof are from Enge [7].

Heuristic 5.1.1 *Let the hyperelliptic curve H of fixed genus g be chosen randomly according to a uniform distribution on the defining pairs of suitable polynomials (h, f) . If $\text{div}(a, b)$ is a uniformly selected element of $J(H)$, then we can assume that a varies uniformly over all polynomials of degree g and b over all polynomials of degree $g - 1$. Likewise, if $\text{div}(a_1, b_1)$ and $\text{div}(a_2, b_2)$ are two uniformly selected elements of $J(H)$, then we can assume that a_1 and a_2 vary uniformly and independently over all polynomials of degree g and b_1 and b_2 over all polynomials of degree $g - 1$.*

Proof. We make two assumptions for the heuristic. First, assume that any two polynomials h of degree at most g and f monic of degree $2g + 1$ define a hyperelliptic curve, while ignoring the negligible proportion that define a singular curve. Second, we assume that all Jacobians have cardinality q^g , so there are $(q - 1)q^g$ pairs of polynomials (a, b) that define a Jacobian element as we do not require a to be monic.

Let (a_1, b_1, a_2, b_2) be a quadruple of polynomials such that $\deg b_1 < \deg a_1 \leq g$ and $\deg b_2 < \deg a_2 \leq g$. We know that two polynomials are relatively prime with probability $1 - 1/q$. Since q is large, we can assume that the GCD of a_1 and a_2 is 1. If we assume that h is fixed, then there are $q^{2g+1-\deg(a_1, a_2)}$ curves admitting $\text{div}(a_1, b_1)$ and $\text{div}(a_2, b_2)$ simultaneously as elements of their Jacobians.

Now if we let h vary then there are $q^{3g+2-\deg(a_1, b_1)}$ curves containing both of the divisors in the Jacobians, and the probability of selecting one of them is $q^{-\deg(a_1, a_2)}$. When a curve is fixed, the conditional probability of choosing (a_1, b_1, a_2, b_2) is given by $1/((q - 1)q^g)^2$. So the quadruple occurs with a total probability of $q^{-2g-\deg a_1-\deg a_2}(q - 1)^{-2}$.

All of the quadruples of polynomials with $\deg a_1 = \deg a_2 = g$ have the same probability of $q^{-4g}(q - 1)^{-2}$. So the event that $\deg a_1 = \deg a_2 = g$ and $\deg b_1 = \deg b_2 = g - 1$ occurs with probability

$$\frac{(q - 1)^4 q^{4g-2}}{(q - 1)^2 q^{4g}} = \left(1 - \frac{1}{q}\right)^2$$

■

5.2 The Average Cost

For this analysis, there are several cases that we need to consider based on the characteristic p of a field and the hyperelliptic curve. If the field characteristic is odd, then $h = 0$. If the field characteristic is even, then $h \neq 0$ and the exact cost depends on the degree of h . Two cases that are efficient for implementation are curves with $h = 1$ and with $\deg h = 1$. In the latter case, an affine change of variable results in $h = X$.

Squaring has a different cost between even and odd characteristics. Assuming field arithmetic is implemented with a normal base [1, 23], squaring a polynomial is done with a simple bit shift, and this squaring is essentially done for free. In odd characteristic, squaring requires $\frac{1}{2}(n+1)(n+2)$ field multiplications.

Now we can proceed to analyze the algorithms in Section 4.7. The proofs for the following theorems are all very similar, and hence we give only one in full detail.

Composition. We assume that the composition algorithm of Section 4.7.1 is applied to uniformly distributed polynomials a_1, a_2, b_1 , and b_2 of degrees $g, g, g-1, g-1$ respectively.

Theorem 5.2.1 *On average, two distinct reduced divisors can be composed with*

- $8g^2 + 5g - 2 + \frac{1}{q} O(g^2)$ multiplications and $g + 2 + \frac{1}{q} O(g)$ inversions for p odd.
- $7g^2 + 7g - 1 + \frac{1}{q} O(g^2)$ multiplications and $g + 1 + \frac{1}{q} O(g)$ inversions for $p = 2$ with $h \in \{1, X\}$.

On average, a reduced divisor can be composed with itself with

- $7g^2 + 7g - 1 + \frac{1}{q} O(g^2)$ multiplications and $g + 1 + \frac{1}{q} O(g)$ inversions for p odd.
- $4g + 2$ multiplications and 1 inversions for $p = 2, h = 1$.
- $4g^2 + 5g + 2 + \frac{1}{q} O(g^2)$ multiplications and 2 inversions for $p = 2, h = X$.

Proof. We first need to find the greatest common divisor of a_1 , a_2 , and u_1 , where a_1 and a_2 are of degree g and u_1 is most likely of degree $g - 1$. In order to determine the number of field operations for $\gcd(a_1, a_2)$, we use Theorem 3.3.1 with the input $E_m(g, g)$ to get

$$\begin{aligned} E_m(g, g)(q-1)^2 &= (g^2 + 2g - 1)q^2 - \left(\frac{5}{2}g^2 + \frac{11}{2}g - 3\right)q + (2g^2 + 4g - 2) \\ &\quad - \frac{1}{q} \left(\frac{g^2}{2} + \frac{g}{2} - 1\right) + \frac{q^2}{q^{d_1}} - \frac{2q}{q^{d_1}} \\ &= g^2 + 2g - 1 + \frac{1}{q}O(g^2). \end{aligned}$$

$$\begin{aligned} E_u(g, g)(q-1)^2 &= (g^2 - g - 2)q^2 - \left(\frac{5}{2}g^2 + \frac{5}{2}g - 10\right)q + (2g^2 + 5g - 2) \\ &\quad - \frac{1}{q} \left(\frac{g^2}{2} + \frac{3g}{2}\right) + \frac{(g^2 + 3g)q^3}{2q^{d_1}} - \frac{(2g^2 + 5g - 2)g^2}{q^{d_1}} \\ &\quad + \frac{(5g^2 + 5g - 20)q}{2q^{d_1}} - \frac{(g^2 - g - 2)}{q^{d_1}} \\ &= g^2 - g - 2 + \frac{1}{q}O(g^2). \end{aligned}$$

Hence $E_m + E_u = 2g^2 + g - 3 + \frac{1}{q}O(g^2)$. We also have

$$E_i(g, g) = g + \frac{g-1}{q} = g + \frac{1}{q}O(g).$$

The average number of multiplications to compute $\gcd(a_1, a_2)$ and u_1 as shown above is in $2g^2 + g - 3 + \frac{1}{q}O(g^2)$ with $g + \frac{1}{q}O(g)$ inversions on average. The greatest common divisor is constant with degree 0 with probability $1 - \frac{1}{q}$. So we can set $u_3 = 0$. By Lemma 3.3.4, u_1 is most likely of degree $u_1 = g - 1$. Computing d^{-1} requires 1 inversion. Then we can compute the a formula for compositions with $a = ((d^{-1}a_1)a_2)$ with $1 + (g+1) + (g+1)^2$ multiplications and is most likely of degree $2g$. The computation of the b formula is best performed by $b_1 + ((d^{-1}u_1)(b_2 - b_1))a_1$ with $g + g^2 + (2g-2)(g+1)$ multiplications and is most likely of degree $3g - 2$. Computing b modulo a requires $(2g+1)(3g-2-2g+1) = (2g+1)(g-1)$ additional multiplications and 1 inversion. The summation of these parts is the answer required in the theorem.

A more efficient approach can be used for $p = 2$. Recall that squarings and multiplica-

tions by $h = \{1, X\}$ are essentially free in characteristic 2. We can compute $\gcd(a_2, b_1 + b_2 + h)$, d^{-1} , and u_3 in the same manner as the previous result with $2g^2 - 4 + \frac{1}{q}O(g^2)$ multiplications and $g + \frac{1}{q}O(g)$ inversions. Compute $a = ((d^{-1})a_1)a_2$ with $(g+1) + (g+1)^2$ multiplications which is most likely of degree $2g$. In order to compute b , first reduce $f \bmod a$ such that $f_1 = f \bmod a$ with $(2g+1)(2g+1-2g+1) = (2g+1)(2)$ multiplications. We can compute $b_1^2 + b_1h$ for free. Next compute $b_1 + (d^{-1}u_3)(f_1 + b_1^2 + b_1h)$ with $g + 2g$ multiplications and is most likely of degree $3g - 2$. This last computation needs to be reduced modula a with an additional $(2g+1)(g-1)$ multiplications. There is no need to find the inverse of the leading coefficient of a this time as it was previously computed. As Enge [7] points out, this approach uses fewer multiplications for $g \geq 3$. For $g = 2$, it saves 1 inversion at the expense of two multiplications.

Several simplifications can be used when doubling a point by computing $\gcd(a_1, 2b_1 + h)$. For $p = 2$ and $h = X$, the values of d and u_3 are easy to determine. If $p = 2$ and $h = 1$, then $d = u_3 = 1$ which reduces the number of operations. ■

Gauss Reduction. We assume that the composition algorithm yields a random semireduced divisor $\text{div}(a, b)$ of degree $2g$, such that a and b are uniformly distributed over all polynomials of degree $2g$ and $2g - 1$ respectively.

Theorem 5.2.2 *On average, Gauss reduction can be executed with*

- $\frac{7}{4}g^3 + \frac{33}{8}g^2 + \frac{1}{4}g + \frac{1}{q} O(g^4)$ multiplications and $\frac{1}{2}g + 1 + \frac{1}{q} O(g^2)$ inversions for p odd, g even.
- $\frac{7}{4}g^3 + \frac{39}{8}g^2 + \frac{9}{4}g + \frac{17}{8} + \frac{1}{q} O(g^4)$ multiplications and $\frac{1}{2}g + \frac{3}{2} + \frac{1}{q} O(g^2)$ inversions for p odd, g odd.
- $\frac{7}{6}g^3 + 3g^2 - \frac{1}{6}g + \frac{1}{q} O(g^4)$ multiplications and $\frac{1}{2}g + 1 + \frac{1}{q} O(g^2)$ inversions for $p = 2$, g even.

- $\frac{7}{6}g^3 + \frac{7}{2}g^2 + \frac{4}{3}g + 2 + \frac{1}{q} O(g^4)$ multiplications and $\frac{1}{2}g + \frac{3}{2} + \frac{1}{q} O(g^2)$ inversions for $p = 2$, g odd, $h \in \{1, X\}$.

Proof. The proof is similar to the composition algorithm. For $t = \lceil \frac{g}{2} \rceil$, we have $\deg a_k = 2(g - k)$ and $\deg b_k = 2(g - k) - 1$ for $k < t$ and $\deg a_t = g$. ■

Lagrange Reduction. We consider this reduction for $g \geq 3$ as this is the case where it differs from Gauss.

Theorem 5.2.3 *If $g \geq 3$, then on average, Lagrange reduction can be executed with*

- $9g^2 - 5 + \frac{1}{q} O(g^3)$ multiplications and $\frac{1}{2}g + 1 + \frac{1}{q} O(g^2)$ inversions for p odd, g even.
- $9g^2 + g - 2g + \frac{1}{q} O(g^3)$ multiplications and $\frac{1}{2}g + \frac{3}{2} + \frac{1}{q} O(g^2)$ inversions for p odd, g odd.
- $7g^2 - g - 5 + \frac{1}{q} O(g^3)$ multiplications and $\frac{1}{2}g + 1 + \frac{1}{q} O(g^2)$ inversions for $p = 2$, g even, $h \in \{1, X\}$.
- $7g^2 - 2 + \frac{1}{q} O(g^3)$ multiplications and $\frac{1}{2}g + \frac{3}{2} + \frac{1}{q} O(g^2)$ inversions for $p = 2$, g odd, $h \in \{1, X\}$.

Proof. The proof is again similar to the composition algorithm. For $t = \lceil \frac{g}{2} \rceil$, we have $\deg a_k = 2(g - k)$ and $\deg b_k = 2(g - k) - 1$ and $\deg q_k = 1$ for $k < t$. If g is odd, then $\deg a_t = g$ and $\deg q_t = 0$ with high probability. ■

With the asymptotic bounds established for the composition step and the three reduction steps, we can clearly see the following theorem about the reduction steps.

Theorem 5.2.4 *For large q , Lagrange reduction is the fastest reduction procedure.*

All the tools to compute the expected number of field operations are now available. Combining Theorems 5.2.1, 5.2.2, and 5.2.3, we can derive the following theorem for the various cases.

Theorem 5.2.5 (Enge) *On average, two distinct divisors can be added with the following number of field operations:*

	<i>Multiplications</i>	<i>Inversions</i>
$p \neq 2, g \text{ even}$	$17g^2 + 5g - 7 + \frac{1}{q}O(g^3)$	$\frac{3}{2}g + 3 + \frac{1}{q}O(g^2)$
$p \neq 2, g \text{ odd}$	$17g^2 + 6g - 4 + \frac{1}{q}O(g^3)$	$\frac{3}{2}g + \frac{7}{2} + \frac{1}{q}O(g^2)$
$p = 2, g \text{ even}$	$14g^2 + 6g - 6 + \frac{1}{q}O(g^3)$	$\frac{3}{2}g + 2 + \frac{1}{q}O(g^2)$
$p = 2, g \text{ odd}$	$14g^2 + 7g - 3 + \frac{1}{q}O(g^3)$	$\frac{3}{2}g + \frac{5}{2} + \frac{1}{q}O(g^2)$

On average, a divisor can be doubled with the following number of field operations:

	<i>Multiplications</i>	<i>Inversions</i>
$p \neq 2, g \text{ even}$	$16g^2 + 7g - 6 + \frac{1}{q}O(g^3)$	$\frac{3}{2}g + 2 + \frac{1}{q}O(g^2)$
$p \neq 2, g \text{ odd}$	$16g^2 + 8g - 3 + \frac{1}{q}O(g^3)$	$\frac{3}{2}g + \frac{5}{2} + \frac{1}{q}O(g^2)$
$p = 2, h = 1, g \text{ even}$	$7g^2 + 3g - 3 + \frac{1}{q}O(g^3)$	$\frac{1}{2}g + 2 + \frac{1}{q}O(g^2)$
$p = 2, h = 1, g \text{ odd}$	$7g^2 + 4g - 3 + \frac{1}{q}O(g^3)$	$\frac{1}{2}g + \frac{5}{2} + \frac{1}{q}O(g^2)$
$p = 2, h = X, g \text{ even}$	$11g^2 + 4g - 3 + \frac{1}{q}O(g^3)$	$\frac{1}{2}g + 3 + \frac{1}{q}O(g^2)$
$p = 2, g = X, g \text{ odd}$	$11g^2 + 5g + \frac{1}{q}O(g^3)$	$\frac{1}{2}g + \frac{7}{2} + \frac{1}{q}O(g^2)$

Proof. The proof is quite simple. For the various cases, add the appropriate matching values from Theorem 5.2.1 and Theorem 5.2.3. For example, take $p \neq 2$ with g even and count the number of multiplications. Theorem 5.2.1 tells us $8g^2 + 5g - 2 + \frac{1}{q}O(g^2)$ multiplications and Theorem 5.2.3 gives $9g^2 - 5 + \frac{1}{q}O(g^3)$ multiplications. This sums to $17g^2 + 5g - 7 + \frac{1}{q}O(g^3)$ as required. ■

With Theorem 5.2.5, we can conclude this chapter. For a large q of genus at least 2, adding two elements in the Jacobian is faster in even characteristic with $h \in \{1, X\}$. The number of field multiplications and inversions strictly increase with the genus.

Chapter 6

Karatsuba Multiplication and the Euclidean Algorithm

This chapter is dedicated to Karatsuba's recursive sub-division method for multiplying two n -degree polynomials. We also present a more implementable version of Karatsuba's algorithm along with some algorithmic analysis. Afterwards, we take the Karatsuba approximations and apply them to the results in Chapters 3. This obtains better exact and asymptotic approximations on the average cost of the Euclidean algorithm than the ones in Chapter 5 using classical arithmetic.

6.1 Basic Method

Karatsuba [10] first presented his algorithm in 1963 in a paper co-authored with Offman, who performed the algorithm analysis. Let f and g be two polynomials of degree n , where we assume for now that n is a power of 2. This is a realistic assumption as a polynomial can always be padded with zeros. Both polynomials can be split into $n/2$

halves such that

$$f = F_1x^{n/2} + F_0$$

$$g = G_1x^{n/2} + G_0$$

where F_1 , F_0 , G_1 , and G_0 are polynomials of degree less than or equal to $n/2$. Naively, we can multiply f and g by

$$fg = F_1G_1x^n + (F_0G_1 + F_1G_0)x^{n/2} + F_0G_0,$$

which requires 4 multiplications and 3 additions. However, Karatsuba noticed that the same product can be done with 3 multiplications and 6 additions by rearranging the formula to get

$$fg = F_1G_1x^n + [(F_1 + F_0)(G_1 + G_0) - F_1G_1 + F_0G_0]x^{n/2} + F_0G_0,$$

where F_1G_1 , $(F_1+F_0)(G_1+G_0)$, and F_0G_0 are computed with recursive calls to Karatsuba. The algorithm is presented below. Multiplications are computationally more expensive than additions, so Karatsuba multiplication outperforms classical multiplication for large n . In practice, Karatsuba's algorithm is better even for small values of n .

ALGORITHM 6.1 Karatsuba's Polynomial Multiplication Algorithm

Input: Two polynomials f and g of degrees less than n , where n is a power of 2.

Output: fg

1. if $n = 1$, return fg
 2. let $f = F_1x^{n/2} + F_0$ and $g = G_1x^{n/2} + G_0$ with F_1 , F_0 , G_1 , G_0 of degrees less than $n/2$.
 3. compute F_0G_0 , F_1G_1 , and $(F_1 + F_0)(G_1 + G_0)$ by a recursive call.
 4. return $fg = F_1G_1x^n + [(F_1 + F_0)(G_1 + G_0) - F_1G_1 + F_0G_0]x^{n/2} + F_0G_0$.
-

Karatsuba's multiplication algorithm is well known to be of order $O(n^{\log_2 3}) = O(n^{1.59})$, whereas classical multiplication is $O(n^2)$ [28]. There are several other well

known multiplication algorithms that are not analyzed in this thesis. Fast Fourier Transformations (FFT) are computed with roots of unity and are of order $O(n \log n \log \log n)$. Another option is to use Montgomery representations for polynomial multiplications [19].

Example 6.1.1 *Let $f = x^3 + 2x^2 + 3x + 4$ and $g = 4x^3 + 3x^2 + 2x + 1$ be two polynomials of degree $n = 3$. First we need to split f and g into two polynomials where*

$$\begin{aligned} f &= F_1x^{n/2} + F_0 = (x + 2)x^2 + (3x + 4) \\ g &= G_1x^{n/2} + G_0 = (4x + 3)x^2 + (2x + 1) \end{aligned}$$

We need to recursively compute the values of F_1G_1 , $(F_1 + F_0)(G_1 + G_0)$, and F_0G_0 . But the degree of these polynomials are now linear and we compute them classically to obtain $F_1G_1 = 4x^2 + 11x + 6$, $(F_1 + F_0)(G_1 + G_0) = 24x^2 + 52x + 24$, and $F_0G_0 = 6x^2 + 11x + 4$. We take these answers and plug them into the Karatsuba formula to obtain

$$\begin{aligned} fg &= F_1G_1x^n + [(F_1 + F_0)(G_1 + G_0) - F_1G_1 - F_0G_0]x^{n/2} + F_0G_0 \\ &= [4x^2 + 11x + 6]x^2 + [(24x^2 + 52x + 24) - (4x^2 + 11x + 6) - (6x^2 + 11x + 4)]x \\ &\quad + 6x^2 + 11x + 4 \\ &= 4x^6 + 11x^5 + 20x^4 + 30x^3 + 20x^2 + 11x + 4 \end{aligned}$$

which is the correct answer. ■

Algorithm 6.1 requires the polynomial to be padded with zeros. This can be timely when polynomial sizes are on the order of a large power of 2. The following two variants strategically split the polynomials to avoid padding zeros. The key is that the polynomials must be split evenly for efficient recursion. There are two cases to consider. First, Algorithm 6.2 multiplies two polynomials of degree n and degree greater than or equal to $n/2$ respectively. Algorithm 6.3 multiplies two polynomials of degree n and less than $n/2$ respectively.

ALGORITHM 6.2 Karatsuba's Algorithm for the Multiplication Two n -degree Polynomials

Input: A polynomial f of degree n and a polynomial g of degree greater than or equal to $n/2$

Output: fg

1. if $n = 1$, return fg
 2. if n is even, pad one 0 as the leading coefficient, split the polynomial evenly, and then factor out $x^{n/2}$ from the upper polynomial such that $f = F_1x^{n/2} + F_0$ and $g = G_1x^{n/2} + G_0$
 3. if n is odd, split the polynomial evenly and factor out $x^{(n+1)/2}$ from the upper polynomial such that $f = F_1x^{(n+1)/2} + F_0$ and $g = G_1x^{(n+1)/2} + G_0$. [Observe that $x^{(n+1)/2} = x^{\lceil n/2 \rceil}$.]
 4. compute F_0G_0 , F_1G_1 , and $(F_1 + F_0)(G_1 + G_0)$ by a recursive call
 5. return $fg = F_1G_1x^n + [(F_1 + F_0)(G_1 + G_0) - F_1G_1 + F_0G_0]x^{n/2} + F_0G_0$
-

Let α be an integer such that $2^{k-1} < \alpha \leq 2^k$. There is one initial call made to the algorithm, which represents the root of a tertiary tree. Every call to the algorithm with degree greater than 1 recursively calls the algorithm 3 times. This tertiary tree has exactly 3^{k-1} leaves of degree $n = 1$. At this point, each leaf requires 4 classical multiplication. Algorithm 6.2 requires $4 \cdot 3^{k-1}$ classical multiplications.

Given two input polynomials of degrees $d_0 \geq d_1$, the value of k is determined by $\lceil \log_2 d_0 \rceil$.

ALGORITHM 6.3 Karatsuba's Algorithm for the Multiplication of Two Polynomials of Degree n and Degree Less than $n/2$

Input: A polynomials f of degree n and a polynomial g of degree less than to $n/2$ but greater than 2.

Output: fg

1. if $n = 1$, return fg

2. if n is even, pad one 0 as the leading coefficient, split the polynomial evenly, and then factor out $x^{n/2}$ from the upper polynomial such that $f = F_1x^{n/2} + F_0$ and $g = G_0$
 3. if n is odd, split the polynomial evenly and factor out $x^{(n+1)/2}$ from the upper polynomial such that $f = F_1x^{(n+1)/2} + F_0$ and $g = G_0$. [Observe that $x^{(n+1)/2} = x^{\lceil n/2 \rceil}$.]
 4. compute F_0G_0 , and F_1G_0 by a recursive call
 5. return $fg = [F_1G_0 + 2F_0G_0]x^{n/2} + F_0G_0$
-

Algorithm 6.3 considers the case when one polynomial is less than half the size of the other. In this case, the spitting of the polynomials is not even. When splitting the lower degree polynomial g into $G_1x^{n/2} + G_0$, the term G_1 is a zero term and the formulas can be adjusted accordingly. This reduces our multiplications from 3 to 2. In a similar analysis to Algorithm 6.2, this now builds a binary tree with 2^{k-1} leaves for a total of $4 \cdot 2^{k-1}$ classical multiplications.

On a final very important note, the bound for the second polynomial in Algorithm 6.3 is $2 < \text{degree of polynomial} < n/2$. The algorithm is more efficient for multiplying two polynomials of drastically different degrees, except for polynomials of degree 2 or less, in which case classical multiplication is faster than Karatsuba multiplication.

To build an efficient multiplication algorithm, we must combine both Algorithms 6.2 and 6.3. Given two polynomials f and g with $\deg f \geq \deg g$. If f is at least double the size of g , then use Algorithm 6.3. At this point we are constructing a binary tree. While iterating down the recursion, the size of the larger polynomial is halved each iteration. Eventually, the smaller polynomial will be greater than half of the bigger one. When this occurs, switch to Algorithm 6.2. This now constructs a tertiary tree.

The implemented Karatsuba algorithm is listed below.

ALGORITHM 6.4 Karatsuba's Algorithm for the Multiplication of Two Polynomials

Input: Polynomials f and g where $f \geq g$

Output: fg

1. let $\deg f = n$ and $\deg g = m$
 2. if $n = 1$, return fg
 3. if $m \leq 2$, return fg
 4. while $m \geq n/2$ [essentially use Algorithm 6.2]
 5. if n is even, pad one 0 as the leading coefficient, split the polynomial evenly,
 and then factor out $x^{n/2}$ from the upper polynomial such that $f = F_1x^{n/2} + F_0$
 and $g = G_1x^{n/2} + G_0$
 6. if n is odd, split the polynomial evenly and factor out $x^{(n+1)/2}$ from the upper
 polynomial such that $f = F_1x^{(n+1)/2} + F_0$ and $g = G_1x^{(n+1)/2} + G_0$.
 7. compute F_0G_0, F_1G_1 , and $(F_1 + F_0)(G_1 + G_0)$ by a recursive call
 8. return $fg = F_1G_1x^n + [(F_1 + F_0)(G_1 + G_0) - F_1G_1 + F_0G_0]x^{n/2} + F_0G_0$
 9. while $m < n/2$, [essentially use Algorithm 6.3]
 10. if n is even, pad one 0 as the leading coefficient, split the polynomial evenly,
 and then factor out $x^{n/2}$ from the upper polynomial such that $f = F_1x^{n/2} + F_0$
 and $g = G_0$
 11. if n is odd, split the polynomial evenly and factor out $x^{(n+1)/2}$ from the upper
 polynomial such that $f = F_1x^{(n+1)/2} + F_0$ and $g = G_0$
 12. compute F_0G_0 , and F_1G_0 by a recursive call
 13. return $fg = [F_1G_0 + 2F_0G_0]x^{n/2} + F_0G_0$
-

The trick to the analysis for Algorithm 6.4 is to determine when the switch will be made from constructing a binary tree to a tertiary tree. Let f and g be two polynomials of degrees α and β . Then $\alpha \leq 2^a$ and $\beta \leq 2^b$ for integers a and b . Equivalently, we can define $a = \log_2 \alpha$ and $b = \log_2 \beta$. This creates a tree with $2^{[a-b]}3^{[b-1]}$ leaves. Each leaf requires 4 classical multiplications, which gives the final tally of

$$4 \cdot 2^{[a-b]}3^{[b-1]} = \frac{4}{3} \cdot 2^{[a]} \left(\frac{3}{2}\right)^{[b]} \quad (6.1)$$

Example 6.1.2 *Building upon Example 6.1.1, we can manually count that multiplying two polynomials of degree 3 requires 12 multiplications using Karatsuba multiplication.*

Classically, this would take 16 multiplications for non-sparse polynomials. We can compute $a = b = \log_2 3 = 2$ which means that $2^1 < a, b \leq 2^2$ is the boundary conditions. Next we plug our values into (6.1) to get

$$\frac{4}{3} \cdot 2^{\lceil a \rceil} \left(\frac{3}{2} \right)^{\lceil b \rceil} = \frac{4}{3} \cdot 2^2 \left(\frac{3}{2} \right)^2 = 12$$

multiplications as required.

6.2 The Euclidean Algorithm with Karatsuba Approximations

We can use Karatsuba's multiplication algorithm with the extended Euclidean algorithm to reduce the number of multiplications. The extended Euclidean algorithm is explained in Section 3.2 and the expected number of field operations with classical multiplication is derived in Section 3.3.

The formulas to derive the expected number of field operations is found in Lemma 3.3.6. As explained in that lemma, the formulas for E_u and E_v are different from E_m . As a result, we can only apply Karatsuba multiplication to the multipliers u and v . Upon substituting the Karatsuba approximations in place of the classical multiplication, we derive the following formulas.

Lemma 6.2.1 *Let $E^u(d_0, d_1)$ and $E^v(d_0, d_1)$ be the expected number of multiplications needed to compute the multipliers u and v in the extended Euclidean algorithm. The*

expected number of field operations are given by

$$\begin{aligned}
 E^u(d_0, d_1) &= \sum_{k=4}^{\infty} \sum_{\mu=0}^{\infty} \sum_{v=0}^{\infty} P(d_{k-2} = \mu \text{ AND } d_{k-1} = v \text{ AND } d_k \neq -\infty) \\
 &\quad \times \frac{4}{3} 2^{\lceil \log(d_1 - \mu) \rceil} \left(\frac{3}{2} \right)^{\lceil \log(\mu - \nu) \rceil} \\
 E^v(d_0, d_1) &= \sum_{k=3}^{\infty} \sum_{\mu=0}^{\infty} \sum_{v=0}^{\infty} P(d_{k-2} = \mu \text{ AND } d_{k-1} = v \text{ AND } d_k \neq -\infty) \\
 &\quad \times \frac{4}{3} 2^{\lceil \log(d_0 - \mu) \rceil} \left(\frac{3}{2} \right)^{\lceil \log(\mu - \nu) \rceil}
 \end{aligned}$$

The new Karatsuba equations are much more complicated due to the introduction of the floor of logarithmic exponentials. All logarithms are base 2. This gives us a method of attack as we know when the exponent will increment. Unfortunately when solving the second summation, the following summation appears

$$\sum_{j=1}^l \left(\frac{3}{2} \right)^j \left(\frac{1}{q} \right)^{2^j} = \int_{x=1}^l \left(\frac{3}{2} \right)^x \left(\frac{1}{q} \right)^{2^x} dx$$

which does not appear to have a closed form. The ramification is that we can not derive a closed form for the equations in Lemma 6.2.1. However, we can derive a closed form for specific intervals of powers of two. For elliptic curve cryptography, the recommended size of a public key is 160 bits as a minimum. Our analysis is for the interval $\{128, 256\} = \{2^7, 2^8\}$.

With Lemmas 3.3.6 and 6.2.1, we can empirically compare the efficiency of the extended Euclidean algorithm with Karatsuba multiplication. For the multiplier u , the formulas do not contain the term d_0 , and as a result they are only dependant on the input polynomial d_1 . Table 6.1 compares the classical equations from Lemma 3.3.6, and the Karatsuba equations from Lemma 6.2.1. This table is done over the interval $\{150, 180\}$. As demonstrated in the table, the Karatsuba equations require less multiplications than classical multiplications.

For the multiplier ν , the equations depend on both d_0 and d_1 . Both must be varied to see the results. For Table 6.2, we choose to compute d_0 in the range $d_0 = \{160, 167\}$,

d_1	<i>Classical</i>	<i>Karatsuba</i>	d_1	<i>Classical</i>	<i>Karatsuba</i>
160	18450	18101	170	20878	20686
161	18686	18360	171	21129	20944
162	18924	18618	172	21381	21203
163	19163	18877	173	21635	21461
164	19403	19135	174	21891	21719
165	19645	19394	175	22148	21978
166	19889	19652	176	22406	22237
167	20134	19911	177	22666	22495
168	20380	20169	178	22928	22754
169	20628	20427			

Table 6.1: The Classical Multiplication and Karatsuba Multiplication Cost for the Multiplier u for the Range $d_1 = \{160, 179\}$

and we let d_1 vary from d_0 to $d_0 - 2$.

From these two preceding empirical examples, we draw the main conclusion of this thesis.

Conclusion. When using the extended Euclidean algorithm, the multiplications should be computed classically, but the multipliers u and v should be computed with Karatsuba multiplication. This reduces the number of multiplications required to execute the algorithm.

We would like to be more mathematical about this conclusion. Ideally, we want to solve the two equations in Lemma 6.2.1. This appears to be quite possible, but unfortunately is taking longer than the duration of this thesis. The two equations are much more complicated than the classical equations. This is no doubt due to the introduction of the floor of exponential logarithms, which has increased the difficulty of the discrete mathematics required to solve them.

d_0	d_1	<i>Classical</i>	<i>Karatsuba</i>	d_0	d_1	<i>Classical</i>	<i>Karatsuba</i>
161	161	18686	18360	165	165	19645	19394
161	160	18683	18359	165	164	19642	19393
161	159	18670	18357	165	163	19638	19391
162	162	18924	18618	166	166	19889	19652
162	161	18920	18617	166	165	19885	19651
162	160	18916	18615	166	164	19881	19649
163	163	19163	18877	167	167	20134	19911
163	162	19159	18877	167	166	20131	19910
163	161	19155	18874	167	165	20126	19908

Table 6.2: The Classical Multiplication and Karatsuba Multiplication Cost for the Multiplier v for the Range $d_0 = \{160, 167\}$

As of the defence of this thesis, we have successfully managed to solve the inner two summations. The mathematical derivations for this can be found in Appendix C. The many remaining terms are still complicated, but they should be solvable by the same techniques used to eliminate the inner two summations. Appendix C contains a detailed discussion on what remains to solve the summations.

Solving these equations will certainly be continued after the defence of this thesis.

Upon solving Lemma 6.2.1 for the average number of field operations, the new results can be used to find the average cost of hyperelliptic cryptography when using Karatsuba multiplication. This is accomplished in the same way as demonstrated in Chapter 5.

Chapter 7

Conclusion

This thesis has focused on the average cost of hyperelliptic cryptography and the extended Euclidean algorithm. We began in section 3.3 by stating the probabilities on the polynomials used in the extended Euclidean algorithm. This led to the derivation of the average number of field operations required for the EEA. We then continued to discuss hyperelliptic cryptography, including several algorithms in Section 4.7 for computing elements in the Jacobian of a hyperelliptic curve. Chapter 5 then determined the average cost of adding elements in the Jacobian. In Chapter 6, we developed an approximation to Karatsuba multiplication. We then used this approximation to improve the bounds on the extended Euclidean algorithm. With this, we demonstrated that for using the extended Euclidean algorithm and hyperelliptic cryptography, it is more efficient to use Karatsuba multiplication than classical multiplication for computing the multipliers μ and ν .

7.1 Future Work

The obvious first comment would be to fully solve the equations in Lemma 6.2.1. There were many hardships in solving what has been completed in Appendix C, but it appears that the most difficult of the discrete mathematics has been surpassed. Completing

the remaining summations is a matter of time. After Lemma 6.2.1 has been solved for the average number of field operations, this can be used to find the average cost of hyperelliptic cryptography when using Karatsuba multiplication. In any case, our estimations presented in Tables 6.1 and 6.2 show that in the range of practical interest, the use of Karatsuba multiplication outperforms classical arithmetic.

There are a few other areas in this work that can be improved upon. It could be possible that different multiplication algorithms could provide better results. In this sense, Montgomery multiplication is increasingly being used in the industry nowadays. Still, Karatsuba seems to be the most used in the industry.

A further practical goal would be to complete a full study over the fields of characteristic 2. These fields are most used in cryptography due to their binary properties. The results from Chapter 5 would need to be computed for the value $q = 2$. To accomplish this, the computations for the reduction and composition algorithms in Section 5.2 would need to be expanded to one more asymptotic term. Substituting $q = 2$ would then obtain the desired result.

Bibliography

- [1] D.W. Ash, I.F. Blake, and S.A. Vanstone. Low complexity normal bases. *Discrete Applied Math.*, 25:191–210, 1989.
- [2] I. Blake, G. Seroussi, and N. Smart. Elliptic curves in cryptograpy. In *London Mathematical Society Lecture Note Series*, volume 265. London Mathematical Society, 1999.
- [3] D. Cantor. Computing in the Jacobian of a hyperelliptic curve. *Mathematics of Computation*, 48:95–101, 1987.
- [4] L. Charlap and D. Robbins. An elementary introduction to elliptic curves. In *CRD Expository Report No. 31, Institute for Defence Analysis, Princeton*, 1988.
- [5] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, 22:644–654, 1976.
- [6] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Info. Theory*, 31:469–472, 1985.
- [7] Andreas Enge. The extended Euclidean algorithm on polynomials and the computational efficiency of hyperelliptic cryptography. In *Designs, Codes, and Cryptography*, volume 23, pages 53–74, 2001.
- [8] C.F. Gauss. *Disquisitiones Arithmeticae*. Braunschweig, 1801.
- [9] R. Graham, D.E. Knuth, and O. Patashnik. *Concrete Mathematics*. Addison-Wesley, Reading, MA, 2nd edition, 1994.

- [10] A. Karatsuba and Y. Ofman. Umnozhenie mnogoznachnykh chisel na abto-matakh. *Dokl. Akad. Nauk USSR*, 145:293–294, 1962. English translation: Multiplication of multidigit numbers on automata, *Soviet Physics–Doklady* 7 (1963), 595–596.
- [11] D.E. Knuth. *The Art of Computer Programming, Vol.1: Fundamental Algorithms*. Addison-Wesley, Reading MA, 3rd edition, 1997.
- [12] N. Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48:203–209, 1987.
- [13] N. Koblitz. Hyperelliptic cryptosystems. *Journal of Cryptography*, 1:139–150, 1989.
- [14] N. Koblitz. *Algebraic Aspects of Cryptography*, volume 3 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 1998.
- [15] R. Lidl and H. Niederreiter. *Introduction to Finite fields and their Applications*. Cambridge University Press, 2nd edition, 1994.
- [16] K. Ma and J. von zur Gathen. Analysis of Euclidean algorithms for polynomials over finite fields. In *Journal of Symbolic Computation*, volume 9, pages 429–455, 1990.
- [17] A.J. Menezes. *Elliptic Curve Public Key Cryptosystems*. Kluwer Academic Publishers, Boston, Dordrecht, Lancaster, 1993.
- [18] V. Miller. Use of elliptic curves in cryptography. In *Advances in Cryptology, CRYPTO 85, LNCS 218*, volume 1, pages 417–426, 1986.
- [19] P. Montgomery. Modular multiplication without trial division. *Mathematics of Computation*, 44:519–521, 1985.
- [20] A. Odlyzko. Discrete logarithms and their cryptographic significance. In *Advances in Cryptology, Proceedings of Eurocrypt 1984*, volume 209 of *Lecture Notes in Computer Science*, pages 224–314. Springer-Verlag, 1985.
- [21] S.C. Pohlig and M.E. Hellman. An improved algorithm for computing logarithms over $GF(p)$ and its cryptographic significance. *IEEE Trans. Information Theory*, 24:106–110, 1978.

- [22] J.M. Pollard. Monte carlo methods for index computation (mod p). *Mathematics of Computation*, 32:918–924, 1978.
- [23] Michael Rosing. *Implementing Elliptic Curve Cryptography*. Manning, 1999.
- [24] Y. Sakai and K. Sakurai. On the practical performance of hyperelliptic curve cryptosystems in software implementation. *IEICE Trans. Fundamentals*, E83-A:692–703, 2000.
- [25] D. Shanks. Class number, a theory of factorization, and genera. In *Number Theory Institute 1969*, volume 20 of *Proceedings of Symposia in Pure Mathematics*, pages 415–440. Amer. Math. Soc., 1969.
- [26] J.H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1986.
- [27] J. Stein. Computational problems associated with Racah algebra. In *Journal of Computational Physics*, volume 1, pages 397–405, 1967.
- [28] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1998.

Appendix A

Identities

This appendix is a compilation of all identities and equations used throughout this thesis.

A.1 Series

Let c , q , and n be constants.

$$\begin{aligned}\sum_{\mu=1}^n c &= cn \quad \text{or} \quad \sum_{\mu=0}^n c = c(n+1) \\ \sum_{\mu=1}^n \mu &= \frac{n^2 + n}{2} \\ \sum_{\mu=0}^{\infty} q^{\mu} &= \frac{1}{q-1} \\ \sum_{\mu=0}^n q^{\mu} &= \frac{q^{n+1} - 1}{q-1} \quad \text{or} \quad \sum_{\mu=1}^n q^{\mu} = \frac{q^{n+1} - q}{q-1} \\ \sum_{\mu=1}^n \mu q^{\mu} &= \frac{q^{n+1}((n+1)q - n - 1 - q) + q}{(q-1)^2} \\ \sum_{\mu=1}^n \mu^2 q^{\mu} &= \frac{q^{n+1}(-2(n+1)q^2 + 2(n+1)q + q^2 + q + (n+1)^2 q^2 - 2(n+1)^2 q + (n+1)^2))}{(q-1)^3} \\ &\quad - \frac{q^2 + q}{(q-1)^3}\end{aligned}$$

A.2 Binomial Identities

$$(x + y)^n = \sum_{k=0}^n \binom{n}{k} x^k y^{n-k}$$

$$(x + 1)^n = \sum_{k=0}^n \binom{n}{k} x^k$$

$$\binom{n}{k} = \binom{n+1}{k+1} - \binom{n}{k+1}$$

$$\binom{n}{k} = \binom{n+1}{k} - \binom{n}{k-1}$$

A.3 Binomial Summations

$$\sum_{\mu=0}^n \binom{\mu}{c} = \binom{n+1}{c+1}$$

$$\sum_{\mu=0}^n \binom{n}{\mu} (q-1)^\mu = \sum_{\mu=0}^n \binom{n}{\mu} (q-1)^\mu 1^{n-\mu} = q^n$$

$$\sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu+2}{k-1} = \sum_{\mu=k-1}^{d_1+2} \binom{\mu}{k-1} = \binom{d_1+3}{k}.$$

$$\sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu+1}{k-2} = \sum_{\mu=k-2}^{d_1+1} \binom{\mu}{k-2} = \binom{d_1+2}{k-1}.$$

$$\sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu}{k-3} = \sum_{\mu=k-3}^{d_1} \binom{\mu}{k-3} = \binom{d_1+1}{k-2}.$$

$$\sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} = \sum_{\mu=k-4}^{d_1-1} \binom{\mu}{k-4} = \binom{d_1}{k-3}.$$

$$\sum_{\mu=0}^{d_1-k+2} \binom{d_1-\mu-1}{k-3} = \sum_{\mu=k-3}^{d_1-1} \binom{\mu}{k-3} = \binom{d_1}{k-2}.$$

$$\sum_{k=3}^{d_1+1} (q-1)^{k-1} \binom{d_1}{k-2} = (q-1)[q^{d_1} - 1 - (q-1)^{d_1}]$$

A.4 Binomial Absorptions

$$\begin{aligned}
\binom{n+1}{k+1} &= \frac{n+1}{k+1} \binom{n}{k} \\
\binom{n}{k} &= \frac{n}{k} \binom{n-1}{k-1} \quad \text{or} \quad k \binom{n}{k} = n \binom{n-1}{k-1} \\
\sum_{\mu=0}^{d_1-k+2} \mu \binom{d_1-\mu-1}{k-3} &= \frac{d_1-k+2}{k-1} \binom{d_1}{k-2} = \frac{d_1-k+2}{d_1+1} \binom{d_1+1}{k-1} \\
\sum_{\mu=0}^{d_1-k+3} \mu \binom{d_1-\mu-1}{k-4} &= \frac{d_1-k+3}{k-2} \binom{d_1}{k-3} = \frac{d_1-k+3}{d_1+1} \binom{d_1+1}{k-2} \\
\sum_{\mu=0}^{d_1-k+3} \mu^2 \binom{d_1-\mu-1}{k-4} &= d_1^2 \binom{d_1}{k-3} - d_1 \binom{d_1}{k-3} - 2d_1(k-3) \binom{d_1+1}{k-2} \\
&\quad + (k-3)(k-2) \binom{d_1+2}{k-1} + \frac{d_1-k+3}{k-2} \binom{d_1}{k-3} \\
\sum_{\mu=0}^{d_1-k+3} \mu^3 \binom{d_1-\mu-1}{k-4} &= d_1^3 \binom{d_1}{k-3} - d_1 \binom{d_1}{k-3} - 3d_1^2(k-3) \binom{d_1+1}{k-2} \\
&\quad - 3d_1(k-3) \binom{d_1+1}{k-2} + 3d_1(k-3)(k-2) \binom{d_1+2}{k-1} \\
&\quad - (k-3)(k-2)(k-1) \binom{d_1+3}{k} + 3(k-3)(k-2) \binom{d_1+2}{k-1} \\
&\quad + \frac{d_1-k+3}{k-2} \binom{d_1}{k-3} \\
\sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu &= -q(q-1)^{d_1} + \frac{q^{d_1+1} d_1^2}{2} + \frac{q^{d_1+1} d_1}{2} \\
&\quad - q^{d_1} d_1^2 + \frac{q^{d_1-1} d_1^2}{2} - \frac{q^{d_1-1} d_1}{2} \\
\sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} q^\mu &= 4q^{d_1+1} - 7q^{d_1} + 3q^{d_1-1} - (q-1)^{d_1} d_1 \\
&\quad + (q-1)^{d_1} - (q-1)^{d_1+1} - q(q-1)^{d_1} + 8 \\
\sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+2} \binom{d_1-\mu-1}{k-3} \mu q^\mu &= \frac{q^{d_1+1} d_1^2}{2} - \frac{q^{d_1+1} d_1}{2} - q^{d_1} d_1^2 + q^{d_1} d_1 \\
&\quad + \frac{q^{d_1-1} d_1^2}{2} - \frac{q^{d_1-1} d_1}{2} \\
\sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+2} \binom{d_1-\mu-1}{k-3} q^\mu &= -(q-1)^{d_1} q + (q-1)^{d_1} + q^{d_1+1} d_1 \\
&\quad - 2q^{d_1} d_1 + q^{d_1-1} d_1
\end{aligned}$$

Appendix B

Proof for Classical Multiplication

This appendix contains the heavily computational proofs that is derived from Theorem 3.3.1, which is the average number of field operations required for the extended Euclidean algorithm. The proof for the expected number of multiplications is not shown here because it is not needed for this thesis.

B.1 Proof of Theorem 3.3.1

$$\begin{aligned} E_i(d_0, d_1) &= \sum_{\tau=1}^{d_1+1} \tau P(t = \tau) - P(\deg \gcd(d_0, d_1) = 0) \\ &= \sum_{\tau=1}^{d_1+1} \tau \binom{d_1}{\tau-1} \frac{(q-1)^{\tau-1}}{q^{d_1}} - \left(1 - \frac{1}{q}\right) \\ &= \frac{1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \tau \binom{d_1}{\tau-1} (q-1)^{\tau-1} - \left(1 - \frac{1}{q}\right) \\ &= \frac{1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \tau \left[\binom{d_1+1}{\tau} - \binom{d_1}{\tau} \right] (q-1)^{\tau-1} - \left(1 - \frac{1}{q}\right) \\ &= \frac{1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \tau \binom{d_1+1}{\tau} (q-1)^{\tau-1} - \frac{1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \tau \binom{d_1}{\tau} (q-1)^{\tau-1} - \left(1 - \frac{1}{q}\right) \end{aligned}$$

$$\begin{aligned}
&= \frac{d_1 + 1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \binom{d_1}{\tau-1} (q-1)^{\tau-1} - \frac{d_1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \binom{d_1-1}{\tau-1} (q-1)^{\tau-1} - \left(1 - \frac{1}{q}\right) \\
&= \frac{d_1 + 1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \binom{d_1}{\tau-1} (q-1)^{\tau-1} 1^{d_1-(\tau-1)} \\
&\quad - \frac{d_1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \binom{d_1-1}{\tau-1} (q-1)^{\tau-1} 1^{d_1-1-(\tau-1)} - \left(1 - \frac{1}{q}\right) \\
&= \frac{d_1 + 1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \binom{d_1}{\tau-1} (q-1)^{\tau-1} 1^{d_1-(\tau-1)} \\
&\quad - \frac{d_1}{q^{d_1}} \sum_{\tau=1}^{d_1+1} \binom{d_1-1}{\tau-1} (q-1)^{\tau-1} 1^{d_1-1-(\tau-1)} - \left(1 - \frac{1}{q}\right) \\
&= \frac{d_1 + 1}{q^{d_1}} q^{d_1} - \frac{d_1}{q^{d_1}} q^{d_1-1} - \left(1 - \frac{1}{q}\right) \\
&= d_1 + 1 - \frac{d_1}{q} - 1 + \frac{1}{q} \\
&= d_1 - \frac{d_1 - 1}{q}
\end{aligned}$$

■

$$\begin{aligned}
&E_u(d_0, d_1) = \\
&= \sum_{k=4}^{\infty} \sum_{\mu=0}^{\infty} \sum_{v=0}^{\infty} P(d_{k-2} = \mu \text{ AND } d_{k-1} = v \text{ AND } d_k \neq -\infty) (\mu - v + 1) (d_1 - \mu + 1) \\
&= \sum_{k=4}^{\infty} \sum_{\mu=0}^{\infty} \sum_{v=0}^{\infty} P(d_{k-2} = \mu) P(d_{k-1} = v \mid d_{k-2} = \mu) (1 - P(d_k = -\infty \mid d_{k-1} = v)) \\
&\quad \times (\mu - v + 1) (d_1 - \mu + 1) \\
&= \sum_{k=4}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[\frac{(q-1)^{k-3}}{q^{d_1}} q^{\mu} \binom{d_1 - \mu - 1}{k-4} \right] \left[\frac{(q-1)q^v}{q^{\mu}} \right] \left[1 - \frac{1}{q^v} \right] \\
&\quad \times (\mu - v + 1) (d_1 - \mu + 1) \\
&= \sum_{k=4}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[\frac{(q-1)^{k-2}}{q^{d_1}} \binom{d_1 - \mu - 1}{k-4} \right] q^v \left[1 - \frac{1}{q^v} \right] (\mu - v + 1) (d_1 - \mu + 1) \\
&= \frac{1}{q^{d_1}} \sum_{k=4}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[(q-1)^{k-2} \binom{d_1 - \mu - 1}{k-4} \right] [q^v - 1] (\mu - v + 1) (d_1 - \mu + 1)
\end{aligned}$$

$$\begin{aligned}
&= \frac{1}{q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} (d_1-\mu+1) \sum_{v=0}^{\mu-1} [q^v - 1] (\mu - v + 1) \\
&= \frac{1}{q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} (d_1-\mu+1) \\
&\quad \times \left[\sum_{v=0}^{\mu-1} v + (\mu+1) \sum_{v=0}^{\mu-1} q^v - (\mu+1) \sum_{v=0}^{\mu-1} 1 - \sum_{v=0}^{\mu-1} v q^v \right] \\
&= \frac{1}{q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} (d_1-\mu+1) \\
&\quad \times \left[\frac{(\mu-1)^2 + (\mu-1)}{2} + (\mu+1) \frac{q^\mu - 1}{q-1} - (\mu+1)\mu - \frac{q^\mu(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&= \frac{1}{q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} (d_1-\mu+1) \\
&\quad \times \left[\frac{\mu^2 - \mu}{2} + \frac{\mu q^\mu + q^\mu - \mu - 1}{q-1} - \mu^2 - \mu - \frac{q^\mu(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&= \frac{d_1+1}{q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&\quad \times \left[\frac{\mu^2 - \mu}{2} + \frac{\mu q^\mu + q^\mu - \mu - 1}{q-1} - \mu^2 - \mu - \frac{q^\mu(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&\quad - \frac{1}{q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&\quad \times \left[\frac{\mu^2 - \mu}{2} + \frac{\mu q^\mu + q^\mu - \mu - 1}{q-1} - \mu^2 - \mu - \frac{q^\mu(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&= \frac{d_1+1}{q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&\quad \times \left[\frac{\mu^2 - \mu}{2} + \frac{\mu q^\mu + q^\mu - \mu - 1}{q-1} - \mu^2 - \mu - \frac{q^\mu(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&\quad - \frac{1}{q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&\quad \times \left[\frac{\mu^3 - \mu^2}{2} + \frac{\mu^2 q^\mu + \mu q^\mu - \mu^2 - \mu}{q-1} - \mu^3 - \mu^2 - \frac{q^\mu(\mu^2 q - \mu^2 - \mu q) + \mu q}{(q-1)^2} \right]
\end{aligned}$$

$$\begin{aligned}
&= \left[\frac{1}{q^{d_1}} + \frac{1}{(q-1)q^{d_1}} + \frac{1}{2q^{d_1}} - \frac{d_1+1}{q^{d_1}} + \frac{d_1+1}{2q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&+ \left[\frac{q}{(q-1)^2 q^{d_1}} + \frac{1}{(q-1)q^{d_1}} - \frac{d_1+1}{q^{d_1}} - \frac{d_1+1}{(q-1)q^{d_1}} - \frac{d_1+1}{2q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \\
&\quad \times \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&+ \left[-\frac{q}{(q-1)^2 q^{d_1}} - \frac{1}{(q-1)q^{d_1}} + \frac{(d_1+1)}{(q-1)^2 q^{d_1}} - \frac{q(d_1+1)}{(q-1)^2 q^{d_1}} + \frac{d_1+1}{(q-1)q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \\
&\quad \times \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \left[\frac{q(d_1+1)}{(q-1)^2 q^{d_1}} + \frac{d_1+1}{(q-1)q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} q^\mu \\
&- \left[\frac{q(d_1+1)}{(q-1)^2 q^{d_1}} + \frac{d_1+1}{(q-1)q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&+ \left[\frac{1}{q^{d_1}} - \frac{1}{2q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^3 \\
&+ \left[-\frac{1}{(q-1)^2 q^{d_1}} + \frac{q}{(q-1)^2 q^{d_1}} - \frac{1}{(q-1)q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 q^\mu \\
&= \left[\frac{(q-1)q - d_1/2(q-1)^2}{(q-1)^2 q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&+ \left[\frac{q - d_1(q-1) - (3d_1+3)/2(q-1)^2}{(q-1)^2 q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&+ \left[\frac{d_1+1 - 2q - qd_1 + d_1(q-1)}{(q-1)^2 q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \left[\frac{q(d_1+1) + (q-1)(d_1+1)}{(q-1)^2 q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} q^\mu \\
&- \left[\frac{q(d_1+1) + (q-1)(d_1+1)}{(q-1)^2 q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&+ \left[\frac{(q-1)^2}{2(q-1)q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^3
\end{aligned}$$

$$\begin{aligned}
&= \left[(q-1)q - d_1/2(q-1)^2 \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \left[d_1^2 \binom{d_1}{k-3} - d_1 \binom{d_1}{k-3} \right. \\
&\quad \left. - 2d_1(k-3) \binom{d_1+1}{k-2} + (k-3)(k-2) \binom{d_1+2}{k-1} + \frac{d_1-k+3}{k-2} \binom{d_1}{k-3} \right] \\
&+ \left[\frac{q-d_1(q-1)}{(q-1)^2 q^{d_1}} - \frac{3d_1+3}{2q^{d_1}} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \frac{d_1-k+3}{d_1+1} \binom{d_1+1}{k-2} \\
&+ [d_1+1-2q-qd_1+d_1(q-1)] \left[-q(q-1)^{d_1} + \frac{q^{d_1+1}d_1^2}{2} + \frac{q^{d_1+1}d_1}{2} - q^{d_1}d_1^2 \right. \\
&\quad \left. + \frac{q^{d_1-1}d_1^2}{2} - \frac{q^{d_1-1}d_1}{2} - (q-1)(d_1q^{d_1}) \right] \\
&+ [q(d_1+1) + (q-1)(d_1+1)] (3q^{d_1+1} - 6q^{d_1} + 3q^{d_1-1} - (q-1)^{d_1}d_1 + (q-1)^{d_1} \\
&\quad - (q-1)^{d_1+1} - q(q-1)^{d_1} + 8) \\
&- [q(d_1+1) + (q-1)(d_1+1)] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \binom{d_1}{k-3} \\
&+ \left[\frac{(q-1)^2}{2} \right] \sum_{k=4}^{d_1+1} (q-1)^{k-2} \left[d_1^3 \binom{d_1}{k-3} - d_1 \binom{d_1}{k-3} - 3d_1^2(k-3) \binom{d_1+1}{k-2} \right. \\
&\quad - 3d_1(k-3) \binom{d_1+1}{k-2} + 3d_1(k-3)(k-2) \binom{d_1+2}{k-1} - (k-3)(k-2)(k-1) \binom{d_1+3}{k} \\
&\quad \left. + 3(k-3)(k-2) \binom{d_1+2}{k-1} + \frac{d_1-k+3}{k-2} \binom{d_1}{k-3} \right]
\end{aligned}$$

$$\begin{aligned}
E^u(d_0, d_1)(q-1)^2 q^{d_1} &= (d_1^2 - d_1 - 2) q^{d_1+2} - \left(\frac{5}{2}d_1^2 + \frac{5}{2}d_1 - 10 \right) q^{d_1+1} + (2d_1^2 + 5d_1 - 2) q^{d_1} \\
&- \left(\frac{1}{2}d_1^2 + \frac{3}{2}d_1 \right) q^{d_1-1} + \left(\frac{1}{2}d_1^2 + \frac{3}{2}d_1 \right) q^3 + (2d_1^2 + 5d_1 - 2) q^2 \\
&+ \left(\frac{5}{2}d_1^2 + \frac{5}{2}d_1 - 10 \right) q - (d_1^2 - d_1 - 2)
\end{aligned}$$

■

$$\begin{aligned}
E_v(d_0, d_1) &= \\
&= \sum_{k=3}^{\infty} \sum_{\mu=0}^{\infty} \sum_{v=0}^{\infty} P(d_{k-2} = \mu \text{ AND } d_{k-1} = v \text{ AND } d_k \neq -\infty) (\mu - v + 1) (d_0 - \mu + 1) \\
&= \sum_{k=3}^{\infty} \sum_{\mu=0}^{\infty} \sum_{v=0}^{\infty} P(d_{k-2} = \mu) P(d_{k-1} = v \mid d_{k-2} = \mu) (1 - P(d_k = -\infty \mid d_{k-1} = v)) \\
&\quad \times (\mu - v + 1) (d_0 - \mu + 1) \\
&= \sum_{k=3}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[\frac{(q-1)^{k-3}}{q^{d_1}} q^{\mu} \binom{d_1 - \mu - 1}{k-4} \right] \left[\frac{(q-1)q^v}{q^{\mu}} \right] \left[1 - \frac{1}{q^v} \right] \\
&\quad \times (\mu - v + 1) (d_0 - \mu + 1) \\
&= \sum_{k=3}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[\frac{(q-1)^{k-2}}{q^{d_1}} \binom{d_1 - \mu - 1}{k-4} \right] q^v \left[1 - \frac{1}{q^v} \right] (\mu - v + 1) (d_0 - \mu + 1) \\
&= \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[(q-1)^{k-2} \binom{d_1 - \mu - 1}{k-4} \right] [q^v - 1] (\mu - v + 1) (d_0 - \mu + 1) \\
&= \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} (d_0 - \mu + 1) \sum_{v=0}^{\mu-1} [q^v - 1] (\mu - v + 1) \\
&= \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} (d_0 - \mu + 1) \\
&\quad \times \left[\sum_{v=0}^{\mu-1} v + (\mu + 1) \sum_{v=0}^{\mu-1} q^v - (\mu + 1) \sum_{v=0}^{\mu-1} 1 - \sum_{v=0}^{\mu-1} v q^v \right] \\
&= \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} (d_0 - \mu + 1) \\
&\quad \times \left[\frac{(\mu-1)^2 + (\mu-1)}{2} + (\mu+1) \frac{q^{\mu} - 1}{q-1} - (\mu+1)\mu - \frac{q^{\mu}(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&= \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} (d_0 - \mu + 1) \\
&\quad \times \left[\frac{\mu^2 - \mu}{2} + \frac{\mu q^{\mu} + q^{\mu} - \mu - 1}{q-1} - \mu^2 - \mu - \frac{q^{\mu}(\mu q - \mu - q) + q}{(q-1)^2} \right]
\end{aligned}$$

$$\begin{aligned}
&= \frac{d_0+1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&\quad \times \left[\frac{\mu^2-\mu}{2} + \frac{\mu q^\mu + q^\mu - \mu - 1}{q-1} - \mu^2 - \mu - \frac{q^\mu(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&\quad - \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&\quad \times \left[\frac{\mu^2-\mu}{2} + \frac{\mu q^\mu + q^\mu - \mu - 1}{q-1} - \mu^2 - \mu - \frac{q^\mu(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&= \frac{d_0+1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&\quad \times \left[\frac{\mu^2-\mu}{2} + \frac{\mu q^\mu + q^\mu - \mu - 1}{q-1} - \mu^2 - \mu - \frac{q^\mu(\mu q - \mu - q) + q}{(q-1)^2} \right] \\
&\quad - \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&\quad \times \left[\frac{\mu^3-\mu^2}{2} + \frac{\mu^2 q^\mu + \mu q^\mu - \mu^2 - \mu}{q-1} - \mu^3 - \mu^2 - \frac{q^\mu(\mu^2 q - \mu^2 - \mu q) + \mu q}{(q-1)^2} \right] \\
&= \frac{d_0+1}{2q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&\quad - \frac{d_0+1}{2q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&\quad + \frac{d_0+1}{(q-1)q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&\quad + \frac{d_0+1}{(q-1)q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} q^\mu \\
&\quad - \frac{d_0+1}{(q-1)q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&\quad - \frac{d_0+1}{(q-1)q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&\quad - \frac{d_0+1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&\quad - \frac{d_0+1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu
\end{aligned}$$

$$\begin{aligned}
&= -\frac{q(d_0+1)}{(q-1)^2 q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \frac{(d_0+1)}{(q-1)^2 q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \frac{q(d_0+1)}{(q-1)^2 q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} q^\mu \\
&- \frac{q(d_0+1)}{(q-1)^2 q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&- \frac{1}{2q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^3 + \frac{1}{2q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&- \frac{1}{(q-1)q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 q^\mu \\
&- \frac{1}{(q-1)q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \frac{1}{(q-1)q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&+ \frac{1}{(q-1)q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&+ \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^3 + \frac{1}{q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&+ \frac{q}{(q-1)^2 q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 q^\mu \\
&- \frac{1}{(q-1)^2 q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 q^\mu \\
&- \frac{q}{(q-1)^2 q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \frac{q}{(q-1)^2 q^{d_1}} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu
\end{aligned}$$

$$\begin{aligned}
&= \left[\frac{d_0+1}{2q^{d_1}} - \frac{d_0+1}{q^{d_1}} + \frac{1}{2q^{d_1}} + \frac{1}{(q-1)q^{d_1}} + \frac{1}{q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&- \left[\frac{d_0+1}{2q^{d_1}} + \frac{d_0+1}{(q-1)q^{d_1}} + \frac{d_0+1}{q^{d_1}} - \frac{1}{(q-1)q^{d_1}} - \frac{q}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \\
&\quad \times \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&+ \left[\frac{d_0+1}{(q-1)q^{d_1}} + \frac{(d_0+1)}{(q-1)^2 q^{d_1}} - \frac{q(d_0+1)}{(q-1)^2 q^{d_1}} - \frac{1}{(q-1)q^{d_1}} - \frac{q}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \\
&\quad \times \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \left[\frac{d_0+1}{(q-1)q^{d_1}} + \frac{q(d_0+1)}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} q^\mu \\
&- \left[\frac{d_0+1}{(q-1)q^{d_1}} + \frac{q(d_0+1)}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&+ \left[\frac{1}{q^{d_1}} - \frac{1}{2q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^3 \\
&+ \left[\frac{q}{(q-1)^2 q^{d_1}} - \frac{1}{(q-1)q^{d_1}} - \frac{1}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 q^\mu \\
&= \left[\frac{d_0+2}{2q^{d_1}} - \frac{d_0}{q^{d_1}} + \frac{1}{(q-1)q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&- \left[\frac{d_0+1}{2q^{d_1}} + \frac{d_0}{(q-1)q^{d_1}} + \frac{d_0+1}{q^{d_1}} - \frac{q}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&+ \left[\frac{d_0}{(q-1)q^{d_1}} + \frac{d_0+1-qd_0-2q}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \left[\frac{d_0+1}{(q-1)q^{d_1}} + \frac{q(d_0+1)}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} q^\mu \\
&- \left[\frac{d_0+1}{(q-1)q^{d_1}} + \frac{q(d_0+1)}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&+ \left[\frac{1}{q^{d_1}} - \frac{1}{2q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^3
\end{aligned}$$

$$\begin{aligned}
&= \left[\frac{d_0 - d_0 q + 2q}{2(q-1)q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^2 \\
&- \left[\frac{3q^2 d_0 + 3q^2 - 4qd_0 - 8q + d_0 + 3}{2(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu \\
&+ \left[\frac{1-2q}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu q^\mu \\
&+ \left[\frac{2qd_0 + 2q - d_0 - 1}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} q^\mu \\
&- \left[\frac{2qd_0 + 2q - d_0 - 1}{(q-1)^2 q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \\
&+ \left[\frac{1}{2q^{d_1}} \right] \sum_{k=3}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} \mu^3 \\
&= (q-1) \frac{d_0 - d_0 q + 2q}{2} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \left[d_1^2 \binom{d_1}{k-3} - d_1 \binom{d_1}{k-3} \right. \\
&\quad \left. - 2d_1(k-3) \binom{d_1+1}{k-2} + (k-3)(k-2) \binom{d_1+2}{k-1} + \frac{d_1-k+3}{k-2} \binom{d_1}{k-3} \right] \\
&- \frac{3q^2 d_0 + 3q^2 - 4qd_0 - 8q + d_0 + 3}{2} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \frac{d_1-k+3}{d_1+1} \binom{d_1+1}{k-2} \\
&+ (1-2q) \left[-q(q-1)^{d_1} + \frac{q^{d_1+1} d_1^2}{2} + \frac{q^{d_1+1} d_1}{2} - q^{d_1} d_1^2 + \frac{q^{d_1-1} d_1^2}{2} - \frac{q^{d_1-1} d_1}{2} \right] \\
&+ (2qd_0 + 2q - d_0 - 1) [4q^{d_1+1} - 7q^{d_1} + 3q^{d_1-1} - (q-1)^{d_1} d_1 + (q-1)^{d_1} \\
&\quad - (q-1)^{d_1+1} - q(q-1)^{d_1} + 8] \\
&- (2qd_0 + 2q - d_0 - 1) \sum_{k=3}^{d_1+1} (q-1)^{k-2} \binom{d_1}{k-3} \\
&+ \frac{(q-1)^2}{2} \sum_{k=3}^{d_1+1} (q-1)^{k-2} \left[d_1^3 \binom{d_1}{k-3} - d_1 \binom{d_1}{k-3} - 3d_1^2(k-3) \binom{d_1+1}{k-2} \right. \\
&\quad \left. - 3d_1(k-3) \binom{d_1+1}{k-2} + 3d_1(k-3)(k-2) \binom{d_1+2}{k-1} - (k-3)(k-2)(k-1) \binom{d_1+3}{k} \right. \\
&\quad \left. + 3(k-3)(k-2) \binom{d_1+2}{k-1} + \frac{d_1-k+3}{k-2} \binom{d_1}{k-3} \right]
\end{aligned}$$

$$\begin{aligned}
&= \frac{1}{2}(-2 - 3d_1 + 20q - 4q^{(d_1+5)}d_0 + 24d_0q^3 + 3d_1^2 + 25q^4d_1 + 2d_0d_1q^{(d_1-1)} \\
&\quad + 16q^{(d_1+2)}d_0 + 7d_1^2q^{d_1} - 22d_0q^{(d_1+4)}d_1 - 19d_1q^{d_1} - 25q^{(d_1+3)}d_1^2 \\
&\quad + 4d_0q^{(d_1+5)}d_1 - 2d_0d_1 - 30q^3d_1 + 50q^{(d_1+3)}d_0d_1 - 16q^2d_0 + 4d_0q \\
&\quad + 3q^{(d_1-1)}d_1 - 48q^2 - 24d_0q^{(d_1+3)} - 2q^{(d_1+5)}d_1^2 + 30q^{(d_1+2)}d_1^2 + 5qd_1 \\
&\quad - 4d_0q^{(d_1+1)} - d_1^2q^{(d_1-1)} + 2q^{(d_1+5)}d_1 - 7q^5d_1 - 3q^5d_1^2 + 43q^{(d_1+3)}d_1 \\
&\quad - 60d_0q^{(d_1+2)}d_1 + 16q^{(d_1+4)}d_0 + 10q^2d_1 + 48q^{(d_1+2)} + 44q^3 - 14d_0d_1q^{d_1} \\
&\quad - 15q^{(d_1+4)}d_1 - 44q^{(d_1+3)} + 14q^{(d_1+4)} + 48q^{(d_1+1)}d_1 + 11q^{(d_1+4)}d_1^2 \\
&\quad + 40d_0q^{(d_1+1)}d_1 + 10d_0qd_1 - 20d_0q^2d_1 + 20d_0q^3d_1 - 10d_0q^4d_1 - 16d_0q^4 \\
&\quad + 4q^5d_0 + 2d_0q^5d_1 - 14q^4 - 20q^{(d_1+1)}d_1^2 + 2q^{d_1} - 20q^{(d_1+1)} - 62q^{(d_1+2)}d_1 \\
&\quad - 30q^3d_1^2 + 15q^4d_1^2 + 30q^2d_1^2 - 15qd_1^2)/(q-1)^3 \\
&= \frac{1}{2q}[2q - 9q^{d_1}q^2d_1 - 14q^2 - 10d_0q^{d_1}q^2d_1 + 14q^{d_1}q^2 - 2q^{d_1}q + 4d_0q^3 - 3qd_1^2 \\
&\quad - 3q^3d_1^2 + 4q^2d_1 - 7q^3d_1 + d_1^2q^{d_1} - 3d_1q^{d_1} + 6q^2d_1^2 - 4q^2d_0 + 3qd_1 + 2d_0qd_1 \\
&\quad + 10q^{d_1}qd_1 + 4q^{d_1}q^2d_0 - 4d_0q^{d_1}q^3 - 2q^{d_1}q^3d_1^2 - 4q^{d_1}qd_1^2 - 4d_0q^2d_1 \\
&\quad - 2d_0d_1q^{d_1} + 2d_0q^3d_1 + 5q^{d_1}q^2d_1^2 + 2q^{d_1}q^3d_1 + 8d_0q^{d_1}qd_1 + 4q^{d_1}q^3d_0d_1)] \\
&= \frac{1}{2q}[2q - 9q^{d_1+2}d_1 - 14q^2 - 10d_0q^{d_1+2}d_1 + 14q^{d_1+2} - 2q^{d_1+1} + 4d_0q^3 - 3qd_1^2 \\
&\quad - 3q^3d_1^2 + 4q^2d_1 - 7q^3d_1 + d_1^2q^{d_1} - 3d_1q^{d_1} + 6q^2d_1^2 - 4q^2d_0 + 3qd_1 + 2d_0qd_1 \\
&\quad + 10q^{d_1+1}d_1 + 4q^{d_1+2}d_0 - 4d_0q^{d_1+3} - 2q^{d_1+3}d_1^2 - 4q^{d_1+1}d_1^2 - 4d_0q^2d_1 \\
&\quad - 2d_0d_1q^{d_1} + 2d_0q^3d_1 + 5q^{d_1+2}d_1^2 + 2q^{d_1+3}d_1 + 8d_0q^{d_1+1}d_1 + 4q^{d_1+3}d_0d_1)] \\
&= \frac{1}{2}[2 - 9q^{d_1+1}d_1 - 14q - 10d_0q^{d_1+1}d_1 + 14q^{d_1+1} - 2q^{d_1} + 4d_0q^2 - 3d_1^2 \\
&\quad - 3q^2d_1^2 + 4qd_1 - 7q^2d_1 + d_1^2q^{d_1-1} - 3d_1q^{d_1-1} + 6qd_1^2 - 4qd_0 + 3d_1 + 2d_0d_1 \\
&\quad + 10q^{d_1}d_1 + 4q^{d_1+1}d_0 - 4d_0q^{d_1+2} - 2q^{d_1+2}d_1^2 - 4q^{d_1}d_1^2 - 4d_0qd_1 \\
&\quad - 2d_0d_1q^{d_1-1} + 2d_0q^2d_1 + 5q^{d_1+1}d_1^2 + 2q^{d_1+2}d_1 + 8d_0q^{d_1}d_1 + 4q^{d_1+2}d_0d_1)]
\end{aligned}$$

$$\begin{aligned}
&= \frac{1}{2}[(-4d_0 - 2d_1^2 + 2d_1 + 4d_0d_1))q^{d_1+2} + (-9d_1 - 10d_0d_1 + 14 + 4d_0 + 5d_1^2)q^{d_1+1} \\
&\quad + (-2 + 10d_1 - 4d_1^2 + 8d_0d_1)q^{d_1} + (d_1^2 - 3d_1 - 2d_0d_1)q^{d_1-1} \\
&\quad + (+4d_0 - 3d_1^2 - 7d_1 + 2d_0d_1)q^2 + (4d_1 + 6d_1^2 - 4d_0 - 4d_0d_1 - 14)q \\
&\quad + (2 - 3d_1^2 + 3d_1 + 2d_0d_1)]
\end{aligned}$$

$$\begin{aligned}
E^v(d_0, d_1)(q-1)^2q^{d_1} &= (2d_0 - d_1)(d_1 - 1)q^{d_1+2} - \left(5d_0d_1 - \frac{5}{2}d_1^2 - 2d_0 + \frac{9}{2}d_1 - 7\right)q^{d_1+1} \\
&\quad + (4d_0d_1 - 2d_1^2 - 5d_1 - 1)q^{d_1} - \left(d_0d_1 - \frac{1}{2}d_1^2 + \frac{3}{2}d_1\right)q^{d_1-1} \\
&\quad + \left(d_0d_1 - \frac{3}{2}d_1^2 + 2d_0 - \frac{7}{2}d_1\right)q^2 \\
&\quad - (2d_0d_1 - 3d_1^2 + 2d_0 - 2d_1 + 7)q + \left(d_0d_1 - \frac{3}{2}d_1^2 + \frac{3}{2}d_1 + 1\right)
\end{aligned}$$

■

Appendix C

Proof for Karatsuba Multiplication

This appendix is the proof of Lemma 6.2.1, which is the average number of field operations for the multipliers μ and ν for the extended Euclidean algorithm with Karatsuba approximations. This is not yet a theorem as it is not fully solved due to time constraints. We present in this appendix the computations completed before the thesis defence, as well as a brief discussion on how to complete the final summation. This work will certainly be continued and completed afterwards.

Only one of the two proofs will be detailed here, as they are very similar. The only difference between the two proofs is that the E_u lower bound sums from $k = 4$, whereas the E_v lower bound sums from $k = 3$. Secondly, E_v is dependent only on d_1 . E_u is dependent on both d_0 and d_1 .

C.1 Proof of Lemma 6.2.1

$$\begin{aligned} E_u(d_0, d_1) &= \\ &= \sum_{k=4}^{\infty} \sum_{\mu=0}^{\infty} \sum_{v=0}^{\infty} P(d_{k-2} = \mu \text{ AND } d_{k-1} = v \text{ AND } d_k \neq -\infty) \frac{4}{3} 2^{\lceil \log(d_1 - \mu) \rceil - \lceil \log(\mu - \nu) \rceil} 3^{\lceil \log(\mu - \nu) \rceil} \end{aligned}$$

$$\begin{aligned}
&= \sum_{k=4}^{\infty} \sum_{\mu=0}^{\infty} \sum_{v=0}^{\infty} P(d_{k-2} = \mu) P(d_{k-1} = v \mid d_{k-2} = \mu) (1 - P(d_k = -\infty \mid d_{k-1} = v)) \\
&\quad \times \frac{4}{3} 2^{\lceil \log(d_1 - \mu) \rceil - \lceil \log(\mu - \nu) \rceil} 3^{\lceil \log(\mu - \nu) \rceil} \\
&= \sum_{k=4}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[\frac{(q-1)^{k-3}}{q^{d_1}} q^{\mu} \binom{d_1 - \mu - 1}{k-4} \right] \left[\frac{(q-1)q^v}{q^{\mu}} \right] \left[1 - \frac{1}{q^v} \right] \\
&\quad \frac{4}{3} 2^{\lceil \log(d_1 - \mu) \rceil - \lceil \log(\mu - \nu) \rceil} 3^{\lceil \log(\mu - \nu) \rceil} \\
&= \sum_{k=4}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[\frac{(q-1)^{k-2}}{q^{d_1}} \binom{d_1 - \mu - 1}{k-4} \right] q^v \left[1 - \frac{1}{q^v} \right] \frac{4}{3} 2^{\lceil \log(d_1 - \mu) \rceil - \lceil \log(\mu - \nu) \rceil} 3^{\lceil \log(\mu - \nu) \rceil} \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} \sum_{\mu=0}^{d_1-k+3} \sum_{v=0}^{\mu-1} \left[(q-1)^{k-2} \binom{d_1 - \mu - 1}{k-4} \right] [q^v - 1] 2^{\lceil \log(d_1 - \mu) \rceil - \lceil \log(\mu - \nu) \rceil} 3^{\lceil \log(\mu - \nu) \rceil} \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} \sum_{v=0}^{\mu-1} [q^v - 1] 2^{\lceil \log(d_1 - \mu) \rceil - \lceil \log(\mu - \nu) \rceil} 3^{\lceil \log(\mu - \nu) \rceil} \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} \sum_{v=0}^{\mu-1} [q^v - 1] 2^{\lceil \log(d_1 - \mu) \rceil - \lceil \log(\mu - \nu) \rceil} 3^{\lceil \log(\mu - \nu) \rceil} \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} 2^{\lceil \log(d_1 - \mu) \rceil} \sum_{v=0}^{\mu-1} [q^v - 1] \left(\frac{3}{2} \right)^{\lceil \log(\mu - \nu) \rceil} \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} 2^{\lceil \log(d_1 - \mu) \rceil} \left[\left(\frac{3}{2} \right)^0 (q^{\mu-1} - 1) \right. \\
&\quad \left. + \sum_{j=1}^l \left(\frac{3}{2} \right)^j \sum_{v=\mu-2^j}^{\mu-2^{j-1}-1} (q^v - 1) + \left(\frac{3}{2} \right)^{l+1} \sum_{v=0}^{\mu-2^l-1} (q^v - 1) \right] \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} 2^{\lceil \log(d_1 - \mu) \rceil} [(q^{\mu-1} - 1) \\
&\quad + \sum_{j=1}^l \left(\frac{3}{2} \right)^j \left[\frac{q^{\mu-2^{j-1}} - q^{2^{j-1}} + 2^{j-1} - q^{\mu-2^j} - q^{2^j} + 2^j}{q-1} \right] + \left(\frac{3}{2} \right)^{l+1} \sum_{v=0}^{\mu-2^l-1} (q^v - 1)] \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1 - \mu - 1}{k-4} 2^{\lceil \log(d_1 - \mu) \rceil} [(q^{\mu-1} - 1) \\
&\quad + \frac{1}{q-1} \sum_{j=1}^l \left(\frac{3}{2} \right)^j [q^{\mu-2^{j-1}} - 2^{j-1}(q-1) - q^{\mu-2^j} - 2^j(q-1)] \\
&\quad + \left(\frac{3}{2} \right)^{l+1} \sum_{v=0}^{\mu-2^l-1} (q^v - 1)]
\end{aligned}$$

$$\begin{aligned}
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} 2^{\lceil \log(d_1-\mu) \rceil} [(q^{\mu-1} - 1) \\
&\quad + \frac{q^\mu}{q-1} \sum_{j=1}^l \left(\frac{3}{2}\right)^j \left(\frac{1}{q}\right)^{2^{j-1}} - \frac{1}{2} \sum_{j=1}^l \left(\frac{3}{2}\right)^j 2^j - \frac{q^\mu}{q-1} \sum_{j=1}^l \left(\frac{3}{2}\right)^j \left(\frac{1}{q}\right)^{2^j} - \sum_{j=1}^l \left(\frac{3}{2}\right)^j 2^j \\
&\quad + \left(\frac{3}{2}\right)^{l+1} \sum_{v=0}^{\mu-2^l-1} (q^v - 1)] \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} 2^{\lceil \log(d_1-\mu) \rceil} [(q^{\mu-1} - 1) \\
&\quad + \frac{q^\mu}{q-1} \left[\left(\frac{3}{2}\right) \frac{1}{q} + \left(\frac{3}{2}\right)^2 \frac{1}{q^2} + \left(\frac{3}{2}\right)^3 \frac{1}{q^4} + \left(\frac{3}{2}\right)^4 \frac{1}{q^8} + \left(\frac{3}{2}\right)^5 \frac{1}{q^{16}} + \left(\frac{3}{2}\right)^6 \frac{1}{q^{32}} \right. \\
&\quad + \left.\left(\frac{3}{2}\right)^7 \frac{1}{q^{64}} \right] - \frac{1}{2} \left[\frac{3^{l+1} - 3}{2} \right] - \frac{q^\mu}{q-1} \left[\left(\frac{3}{2}\right) \frac{1}{q^2} + \left(\frac{3}{2}\right)^2 \frac{1}{q^4} + \left(\frac{3}{2}\right)^3 \frac{1}{q^8} \right. \\
&\quad + \left.\left(\frac{3}{2}\right)^4 \frac{1}{q^{16}} + \left(\frac{3}{2}\right)^5 \frac{1}{q^{32}} + \left(\frac{3}{2}\right)^6 \frac{1}{q^{64}} + \left(\frac{3}{2}\right)^7 \frac{1}{q^{128}} \right] - \left[\frac{3^{l+1} - 3}{2} \right] \\
&\quad + \left(\frac{3}{2}\right)^{l+1} \frac{1}{q-1} (q^{\mu-2^l} - (\mu - 2^l)q + \mu - 2^l - 1)] \\
&= \frac{4}{3q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} 2^{\lceil \log(d_1-\mu) \rceil} \left[\frac{1}{128(q-1)} (-32q + 32 \right. \\
&\quad + 729q^{\mu-64} + 128 \cdot 3^{l+1} 2^{-l-1} \mu + 128 \cdot 3^{l+1} 2^{-l-1} q^{\mu-2^l} + 64q^{\mu-1} \\
&\quad - 2187q^{\mu-128} - 32 \cdot 3^{l+1} + 128q^\mu + 32 \cdot 3^{l+1} q - 128 \cdot 3^{l+1} 2^{-l-1} \\
&\quad \left. - 128 \cdot 3^{l+1} 2^{-l-1} q\mu + 96q^{\mu-2} + 144q^{\mu-4} + 216q^{\mu-8} + 324q^{\mu-16} + 486q^{\mu-32}) \right] \\
&= \frac{4}{3(q-1)q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} 2^{\lceil \log(d_1-\mu) \rceil} \left[-\frac{q}{4} + \frac{1}{4} \right. \\
&\quad + \frac{729q^\mu}{128q^{64}} + \mu \left(\frac{3}{2}\right)^{l+1} + \frac{q^\mu}{q^{2^l}} \left(\frac{3}{2}\right)^{l+1} + \frac{q^\mu}{2q} - \frac{2187q^\mu}{128q^{128}} - \frac{3^{l+1}}{4} + q^\mu \\
&\quad \left. + \frac{q3^{l+1}}{4} - \left(\frac{3}{2}\right)^{l+1} - q\mu \left(\frac{3}{2}\right)^{l+1} + \frac{96q^\mu}{128q^2} + \frac{144q^\mu}{128q^4} + \frac{216q^\mu}{128q^8} + \frac{324q^\mu}{128q^{16}} + \frac{486q^\mu}{128q^{32}} \right]
\end{aligned}$$

$$\begin{aligned}
&= \frac{4}{3(q-1)q^{d_1}} \sum_{k=4}^{d_1+1} (q-1)^{k-2} \sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} 2^{\lceil \log(d_1-\mu) \rceil} \left[-\frac{q}{4} + \frac{1}{4} \right. \\
&\quad + \frac{3^6 q^\mu}{2^7 q^{64}} + \mu \left(\frac{3}{2} \right)^{l+1} + \frac{q^\mu}{q^{2^l}} \left(\frac{3}{2} \right)^{l+1} + \frac{q^\mu}{2q} - \frac{3^7 q^\mu}{2^7 q^{128}} - \frac{3^{l+1}}{4} + q^\mu \\
&\quad \left. + \frac{q 3^{l+1}}{4} - \left(\frac{3}{2} \right)^{l+1} - q\mu \left(\frac{3}{2} \right)^{l+1} + \frac{3q^\mu}{4q^2} + \frac{3^2 q^\mu}{2^3 q^4} + \frac{3^3 q^\mu}{2^4 q^8} + \frac{3^4 q^\mu}{2^5 q^{16}} + \frac{3^5 q^\mu}{2^6 q^{32}} \right]
\end{aligned}$$

■

At this point we stop the proof for the time being. There are many things to be discussed at the current step. First, we can solve the second summation with current information. While the inner most parenthesis is very complicated, it can be greatly simplified with some asymptotics. This is because in the inner parenthesis, as μ gets larger, the q^μ term is the dominant term. This is not true for $\mu = 0, 1, 2$ though. We can then break the inner summation into the following two summations

$$\begin{aligned}
&= \sum_{\mu=0}^4 \binom{d_1-\mu-1}{k-4} 2^{\lceil \log(d_1-\mu) \rceil} \left[-\frac{q}{4} + \frac{1}{4} + \frac{3^6 q^\mu}{2^7 q^{64}} + \mu \left(\frac{3}{2} \right)^{l+1} + \frac{q^\mu}{q^{2^l}} \left(\frac{3}{2} \right)^{l+1} + \frac{q^\mu}{2q} \right. \\
&\quad - \frac{3^7 q^\mu}{2^7 q^{128}} - \frac{3^{l+1}}{4} + q^\mu + \frac{q 3^{l+1}}{4} - \left(\frac{3}{2} \right)^{l+1} - q\mu \left(\frac{3}{2} \right)^{l+1} + \frac{3q^\mu}{4q^2} + \frac{3^2 q^\mu}{2^3 q^4} + \frac{3^3 q^\mu}{2^4 q^8} \\
&\quad \left. + \frac{3^4 q^\mu}{2^5 q^{16}} + \frac{3^5 q^\mu}{2^6 q^{32}} \right] \\
&+ c \sum_{\mu=5}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} 2^{\lceil \log(d_1-\mu) \rceil} [q^\mu + O(q^{\mu-1})],
\end{aligned}$$

where c is a constant that is taken out of the inner parenthesis for the q^μ term. The first summation is finite and solvable by MAPLE. The second summation requires some rewriting. We make a change of variables so that $m = d_1 - \mu$. This gives

$$\sum_{\mu=0}^{d_1-k+3} \binom{d_1-\mu-1}{k-4} 2^{\lceil \log(d_1-\mu) \rceil} = \sum_{m=k-3}^{d_1} \binom{m-1}{k-4} 2^{\lceil \log m \rceil}.$$

All logarithms are base 2, so with some analysis this tells us when we switch to the next

power of 2. The above summation works out to

$$\begin{aligned} \sum_{m=k-3}^{d_1} \binom{m-1}{k-4} 2^{\lceil \log m \rceil} &= \sum_{m=k-3}^{2^{\lceil \log(k-3) \rceil}} \binom{m-1}{k-4} 2^{\lceil \log(k-3) \rceil} + \sum_{i=\lceil \log(k-3) \rceil+1}^{\lceil \log d_1 \rceil} \sum_{m=2^{i-1}+1}^{2^i} \binom{m-1}{k-4} 2^i \\ &+ \sum_{m=2^{\lceil \log d_1 \rceil}+1}^{d_1} \binom{m-1}{k-4} 2^{\lceil \log d_1 \rceil}. \end{aligned}$$

This manages to rewrite the original equation in such a way that the exponential logarithm is no longer in the summation. We need one further step than this general derivation, as the equation we want to solve contains a q^μ term. All that needs to be done is to multiply each term above by q^μ . This requires one small detail first. Recalling that our change of variable is $m = d_1 - \mu$, this is equivalent to $\mu = d_1 - m$. So each term is multiplied by q^{d_1-m} . This gives

$$\begin{aligned} \sum_{m=k-3}^{d_1} \binom{m-1}{k-4} 2^{\lceil \log m \rceil} q^\mu &= \sum_{m=k-3}^{2^{\lceil \log(k-3) \rceil}} \binom{m-1}{k-4} 2^{\lceil \log(k-3) \rceil} q^{d_1-m} \\ &+ \sum_{i=\lceil \log(k-3) \rceil+1}^{\lceil \log d_1 \rceil} \sum_{m=2^{i-1}+1}^{2^i} \binom{m-1}{k-4} 2^i q^{d_1-m} + \sum_{m=2^{\lceil \log d_1 \rceil}+1}^{d_1} \binom{m-1}{k-4} 2^{\lceil \log d_1 \rceil} q^{d_1-m}, \end{aligned}$$

which is now solvable. With some algebraic manipulation, the second summation can be solved. For the final summation, the same tricks need to be used to rewrite the original equation to obtain the end result.

The proof for the multiplier v is almost identical. The lower bound on the outer k summation is 3 instead of 4, which is negligible. However, the E_v equation is dependant on both d_0 and d_1 as the equation has a $2^{\lceil \log(d_0-\mu) \rceil}$ term and the outer summation is bounded by d_1 . This is not a great source of trouble, and is solved in the same manner as the first proof.