UNIVERSITY OF CALGARY

Unconditional Class Group Tabulation to 2^{40}

by

Anton S. Mosunov

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

July, 2014

Abstract

In this thesis, we aim to tabulate the class groups of binary quadratic forms for all fundamental discriminants $\Delta < 0$ satisfying $|\Delta| < 2^{40}$. Our computations are performed in several stages. We first multiply large polynomials in order to produce class numbers $h(\Delta)$ for $\Delta \not\equiv 1 \pmod 8$. This stage is followed by the resolution of class groups Cl_Δ with the Buchmann-Jacobson-Teske algorithm, which can be significantly accelerated when the class numbers are known. In order to determine Cl_Δ for $\Delta \equiv 1 \pmod 8$, we use this algorithm in conjunction with Bach's conditional averaging method and the Eichler-Selberg trace formula, required for unconditional verification. Our novel class group tabulation method allowed us to gather unconditional numerical evidence in support of certain hypotheses, such as the Littlewood's bounds on $L(1, \chi_\Delta)$ and the Cohen-Lenstra heuristics.

Acknowledgements

I am grateful to my supervisor, Dr. Michael J. Jacobson, Jr., who suggested this exciting project to me when I was still a Baccalaureate student. Without him, this project would not have started, and most certainly would not be finished, as his wise advice were of enormous help to my research. I thank Prof. Richard K. Guy and Dr. Renate Scheidler for their comments on my thesis.

I would also like to thank "Alberta Innovates Technology Futures" for their generous support of my project over the past two years. I hope that they will find it worth their investments.

All of my computations were performed on one of WestGrid's supercomputers, and I would like to acknowledge an outstanding work of their support personnel, namely Douglas Phillips, Masao Fujinaga, and especially Belaid Moa, who spent countless hours over the phone helping to debug my program.

Perhaps, the most warm and tender thank you goes to my dear family, who always supported me when I was eight thousand kilometres away from home. They made my arrival to Canada possible, and I will always be thankful for this great opportunity.

Finally, I thank all of my friends on both sides of the Atlantic Ocean, especially my fellow graduate students Sebastian Lindner, Longsheng Zhou and Ostap Orobets for making office time pass joyfully.

Table of Contents

List of Tables

List of Figures and Illustrations

List of Algorithms

Chapter 1

INTRODUCTION

The class group of binary quadratic forms is a mathematical object that was thoroughly studied by many mathematicians over the past two centuries. First discovered by Gauß in 1798, this group consists of equivalence classes of *binary quadratic forms*, i.e. integer variable functions of the form $ax^2 + bxy + cy^2$ with $a, b, c \in \mathbb{Z}$. Quadratic forms in every equivalence class of a certain class group possess the same discriminant $\Delta = b^2 - 4ac$. For that reason, this finite abelian group is denoted by Cl_Δ, in order to indicate that the structure of a group varies depending on Δ.

Many things are known about the class group. For example, if we know its *class number* $h(\Delta)$, which is merely the size of Cl_Δ, we can find a non-trivial factor of Δ. Also, from the prime factorization of Δ we can determine the parity of $h(\Delta)$, as well as the rank of the 2-Sylow subgroup. Reduction and composition algorithms, such as NUCOMP developed by Shanks [Sha89], allow us to efficiently manipulate the elements of Cl_Δ. However, the number of open questions about Cl_Δ most certainly exceeds the number of answered. For example, the structure computation of Cl_Δ is believed to be hard. The computation of the class number $h(\Delta)$, which is merely the size of Cl_Δ, is considered to be a difficult problem as well. However, the heuristics of Cohen and Lenstra allow us to make certain predictions regarding the structure of the class group, and divisibility properties of $h(\Delta)$ [CL83, CL84]. Also, Littlewood's bounds allow to deduce an estimate on the size of $h(\Delta)$ [Lit28]. However, Littlewood's result depends on the Extended Riemann Hypothesis, and might potentially be incorrect. Both Cohen-Lenstra heuristics and the Littlewood's bounds require careful examination, which we do in this thesis by providing an extensive computational evidence in their support.

This chapter is organized as follows. In Section 1.1, we give a more detailed motivation for studying class groups and hypotheses associated to them. In Section 1.2, we describe previous work in this field and our contribution. Finally, in Section 1.3 we outline the organization of the thesis.

1.1 Motivation

One of the major questions that one can ask about Cl_Δ concerns its structure, which is widely believed to be hard to compute. Also, some of the problems that are hard for finite abelian groups in general, such as the discrete logarithm problem or the Diffie-Hellman problem, are considered to be hard for Cl_Δ well. Nevertheless, there exist several results which allow one to predict particular properties of Cl_Δ. As mentioned earlier, it is rather easy to derive certain information regarding the 2-Sylow subgroup of Cl_Δ. Also, the heuristic argument of Cohen and Lenstra gives certain predictions regarding the structure of the odd part of Cl_Δ [CL83, CL84]. In 2006, Ramachandran gathered an extensive computational evidence in support of the Cohen-Lenstra heuristics by tabulating all class groups to $2 \cdot 10^{11}$ [Ram06]. One of the main goals of this thesis is to extend the result of Ramachandran, and to provide additional numerical evidence in support of the hypothesis of Cohen and Lenstra to 2^{40}.

Though easier than the class group structure resolution, the class number computation problem is also considered to be hard. One possible question of interest is to find out whether $h(\Delta)$ is divisible by a given odd prime p. The Cohen-Lenstra heuristics address the question of divisibility properties of $h(\Delta)$. Another question of interest is to provide a good estimate of $h(\Delta)$. Such an estimate was given, for example, by Littlewood [Lit28]. However, this result is *conditional*; that is, it depends on the *Extended Riemann Hypothesis* (ERH), formulated below. We thoroughly examine Littlewood's results in our thesis.

Conjecture 1.1.1. *Let $\chi(n)$ denote a Dirichlet character* [JW09, Definition 8.4]*, and*

$$L(s,\chi) = \sum_{n=1}^{\infty} \frac{\chi(n)}{n^s}$$

be the corresponding L-function of a complex variable s with real part $\Re(s) > 1$. Consider its analytic continuation to the whole complex plane $\mathbb{C}$. Then the equality $L(s,\chi) = 0$ for $0 < \Re(s) < 1$ implies $\Re(s) = 1/2$.

When $\Delta > 0$, the analysis of Cl_Δ is very much different from that for negative values of Δ, if not more sophisticated. Gauß conjectured that for $\Delta > 0$ there are infinitely many Cl_Δ of class number one, and to this day the problem is still open. When $\Delta < 0$, it is known that for a given positive integer k, there are finitely many class groups which satisfy $h(\Delta) = k$. The famous Gauß class number problem asks to tabulate all Δ satisfying this equality. Significant progress on this problem was made by Watkins, who provided complete tables of Δ for each $k \leq 100$ [Wat03].

As a final motivation, we would like to mention that class groups have certain applications in cryptography. The monograph of Jacobson and Williams gives a survey on encryption schemes and protocols based on class groups [JW09, Chapter 14]. Since the security of each such cryptosystem resides in the difficulty of some hard mathematical problem, it is possible to choose a hard problems specific to the class group. For example, the discrete logarithm problem in Cl_Δ can potentially be hard, and one can make a heuristic argument that with a high probability it is invulnerable to Pohlig-Hellman algorithm. Of course, in order to make sure that a particular problem in the class group is difficult, one has to be able to predict that Cl_Δ possesses certain desired properties. The heuristics of Cohen and Lenstra allow us to make these predictions. Having a lot of unconditional computational evidence in support of this result would give us more certainty that the cryptosystem is secure.

1.2 Previous Work and Contribution

Many researchers worked on class group tabulation. The first exhaustive and unconditionally correct computation of that kind was done in 1976 by Buell up to $|\Delta| < 4000000$ [Bue76]. In this work, Buell presented a table of "exotic" groups with p-rank greater than 1. In his next tabulation to 25000000 [Bue87], Buell also gathered statistics on the heuristics of Cohen and Lenstra that were introduced in the middle of the 1980s [CL83, CL84]. His "last exhaustive computation of class groups" was done in 1999 up to $2.2 \cdot 10^9$ [Bue99], where statistics on the bounds of Littlewood and Bach were gathered [Lit28, Bac90]. Before Buell, Littlewood's bounds were also systematically examined by Shanks [Sha73].

The next notable work on class group tabulation is due to Jacobson, Ramachandran and Williams [JRW06], who computed statistics on Cl_Δ with $|\Delta| < 10^{11}$ using the Buchmann-Jacobson-Teske Algorithm [BJT97]. Later their result was extended to $2 \cdot 10^{11}$ by Ramachandran in her Master's thesis [Ram06]. Despite the fact that their computations were only correct under the assumption of the ERH, they were able to apply the Eichler-Selberg trace formula for the unconditional verification of their result [SvdV91]. This important technique was also used in our work.

In this thesis, we push the feasibility limit further by computing class groups for $|\Delta| < 2^{40} = 1.09951\ldots \times 10^{12}$. Such a notable increase in the order of magnitude became possible due to the novel *class number tabulation* technique we apply prior to the class group tabulation. This technique is based on the computational method of Hart et al. [HTW10], who used multiplication of large polynomials for the purpose of tabulating all *congruent numbers*[1] to 10^{12}. By applying Hart's methods, we were able to produce class numbers *unconditionally* for all fundamental $\Delta \not\equiv 1 \pmod 8$ which do not exceed 2^{40} in their absolute value. The precomputation of class numbers allowed us not only to avoid the verification step for $\Delta \not\equiv 1 \pmod 8$, but also to achieve a significant speedup of the class group computation in this

[1] A congruent number is an integer that is also the area of a right triangle with sides being three rational numbers. See the monograph of Koblitz for more details [Kob84].

case (see Chapter 5 for more details). Class groups for $\Delta \equiv 1 \pmod 8$ got computed using the verification-based technique of Jacobson et al. [JRW06]. In the end, we observed that class groups with $\Delta \not\equiv 1 \pmod 8$ got computed over 4.72 times faster than class groups with $\Delta \equiv 1 \pmod 8$.

In Chapter 6, we present our extensive computational data in order to support certain heuristics. Unfortunately, the speedup of our computations did not come for free, as it no longer allowed us to test Bach's bound on the number of generators required to produce the whole class group [Bac90], which was previously done by Buell, Jacobson et al., and Ramachandran [Bue99, JRW06, Ram06]. However, we were still able to examine Littlewood's bounds, as well as various heuristics of Cohen and Lenstra. We also extended the table of groups with a p-rank exceeding 1, which Buell referred to as "exotic" [Bue99]. These groups are quite rare, and many researchers elaborated on the construction of class groups with high p-ranks. For example, Diaz y Diaz et al. listed 119 negative discriminants, which correspond to Cl_Δ with 3-rank exceeding 4 [DyDSW79], and Schoof presented 75 examples with 5-rank ≥ 3 [Sch83]. However, the construction of such groups in general does not allow one to say whether a particular Δ is the smallest in its absolute value discriminant such that Cl_Δ has a certain p-rank. As an exceptional example, see the work of Belabas [Bel04], who demonstrated that Cl_Δ with the smallest 3-rank equal to 5 corresponds to $\Delta = -90338558707159$. In our work, we find first occurrences of class groups using the brute force approach by iterating over all the class groups Cl_Δ for negative Δ, satisfying $|\Delta| < 2^{40}$. Our tables of exotic groups can be found in Chapter 6.

1.3 Organization of the Thesis

Our thesis is organized as follows. Chapter 2 contains an introduction to the theory of binary quadratic forms. It covers their subdivision into equivalence classes, and demonstrates how one can put a group structure onto the set of all equivalence classes of forms. We also give an

overview to the theory of binary quadratic forms with even middle coefficient. Gauß applied this theory in order to prove a very important result, which we outline in Theorem 3.2.3. This result connects *ternary quadratic forms*, which we also discuss in Chapter 2, to class numbers of binary quadratic forms with even middle coefficient.

Chapter 3 addresses the problem of class number tabulation for $\Delta \not\equiv 1 \pmod 8$. In this chapter, we outline Gauß's proof of our main result stated in Theorem 3.2.3, and derive formulas suitable for the tabulation of class numbers.

Chapter 4 is dedicated to the out-of-core polynomial multiplication technique of Hart et al. [HTW10], which we utilize in our program to compute class numbers $h(\Delta)$. We present pseudocode for algorithms, which constitute polynomial multiplication, and study their asymptotic time complexity. Additionally, we discuss the choice of computational parameters required for the correct computation of $h(\Delta)$.

Chapter 5 contains a description of the Buchmann-Jacobson-Teske algorithm, which we utilize for the class group tabulation [BJT97, Algorithm 4.1]. In Section 5.1, we also outline Bach's conditional approach for the computation of a lower bound h^* such that $h^* \leq h(\Delta) \leq 2h^*$, which we require in order to tabulate Cl_Δ with $\Delta \equiv 1 \pmod 8$. In order to eliminate the ERH dependency, we apply the Eichler-Selberg trace formula [SvdV91], which was previously used by Jacobson et al. for class group tabulation purposes [JRW06].

In Chapter 6, we discuss the performance of our class number and class group tabulation programs, and present several tables and figures, which demonstrate their timings and scalability. We also present numerical data on Littlewood's bounds, Cohen-Lenstra heuristics and first occurrences of non-cyclic p-groups.

Chapter 7 concludes this thesis by giving a brief discussion of future work that will further allow to increase the scale of class group computations.

Chapter 2

NUMBER THEORY BACKGROUND

This chapter is dedicated to the theory of quadratic forms, which was first established by Gauß in "Disquisitiones Arithmeticae" [Gau98], and later refined by Dirichlet in his "Vorlesungen über Zahlentheorie" [Dir63]. In Section 2.1, we look at binary quadratic forms from the modern perspective, and consider some fundamental questions, such as the question of equivalence of forms. We also explain how the set of all equivalence classes of forms with the same discriminant Δ forms a group Cl_Δ.

Since its development by Gauß and Dirichlet, the theory of binary quadratic forms significantly evolved. One fundamental difference is that Gauß and Dirichlet studied binary quadratic forms with *even* middle coefficient. Because certain class number formulas were demonstrated from the perspective of Gauß and Dirichlet, we dedicate Sections 2.2 and 2.3 to describe their approach, and find its connection to the modern theory.

In the final section of this chapter, we state certain results from the theory of ternary quadratic forms, which was also established by Gauß [Gau98, §266]. These results are crucial to the proof of our main Theorem 3.2.3, which connects the number of solutions of the Diophantine equation $n = x^2 + y^2 + z^2$ to the class number $h(-n)$ or $h(-4n)$, depending on the congruence class of n modulo 4.

For those who would like to learn about more profound results of the theory of binary quadratic forms, we recommend the monograph of Buchmann and Vollmer [BV07]. Alternatively, a textbook of Crandall and Pomerance gives a nice introduction to this theory [CP05, Section 5.6].

2.1 Binary Quadratic Forms

We begin with the following definitions.

Definition 2.1.1. A function $f(x, y)$ of the form

$$f(x, y) = ax^2 + bxy + cy^2,$$

where a, b, c are fixed integers and x, y are integer variables, is called a *binary quadratic form*. Hereafter, the binary quadratic form $ax^2 + bxy + cy^2$ will be denoted by the ordered triple (a, b, c).

Definition 2.1.2. The integer $\Delta = b^2 - 4ac$ is called the *discriminant* of the form (a, b, c). The discriminant Δ is called *fundamental* if it is not divisible by an odd square, and satisfies the congruence $\Delta \equiv 1, 5, 8, 9, 12, 13 \pmod{16}$.

Here we have to emphasize that both Gauß and Dirichlet considered quadratic forms of a slightly different representation. In particular, they studied forms with *even* middle coefficient, i.e. $ax^2 + 2bxy + cy^2$. In this section, we prefer to stick to a modern approach, considered, for example, in the monograph of Buchmann and Vollmer [BV07]. Nevertheless, we cannot avoid the discussion of forms with even middle coefficient, as some fundamental formulas that we use for class number tabulation are based on the theory of Gauß and Dirichlet. We discuss this theory in the following section.

Let us now consider the fact that some forms can be transformed from one into another. Indeed, if we have a form (a, b, c), then by means of the substitution

$$x = \alpha x' + \beta y', \;\; y = \gamma x' + \delta y', \tag{2.1.1}$$

with α, β, γ and δ being integers, we obtain a new quadratic form of variables x' and y':

$$a(\alpha x' + \beta y')^2 + b(\alpha x' + \beta y')(\gamma x' + \delta y') + c(\gamma x' + \delta y')^2 = a'x'^2 + b'x'y' + c'y'^2 = (a', b', c').$$

It is not hard to verify that (a, b, c) and (a', b', c') have the same discriminant Δ. Also, if an integer n can be represented by the form (a, b, c), i.e.

$$n = ax_0^2 + bx_0y_0 + cy_0^2$$

for some integers x_0 and y_0, then n can be represented by (a', b', c') as well. However, the converse is not true in general; in fact, it holds only when $\alpha\delta - \beta\gamma = \pm 1$. We conclude that some forms of the same discriminant are alike [BV07, Chapter 2].

Definition 2.1.3. Two quadratic forms, (a, b, c) and (a', b', c'), are called *equivalent*[1], if there exist integers α, β, γ and δ, satisfying $\alpha\beta - \gamma\delta = 1$, such that by means of substitution (2.1.1) (a, b, c) can be transformed into (a', b', c').

Definition 2.1.4. The set of all quadratic forms, equivalent to some given form (a, b, c), is called an *equivalence class*, and is denoted by $[(a, b, c)]$. Any form (a', b', c'), which belongs to this set, is called a *representative* of $[(a, b, c)]$.

Of course, computationally we deal with certain representatives of an equivalence class. A common approach is to use those representatives which have "smallest" coefficients. Gauß demonstrated that, irrespective of the sign of Δ, every equivalence class contains forms with the property described below.

Definition 2.1.5. The form (a, b, c) of discriminant Δ is called *reduced*, if its coefficients satisfy the inequality

$$\left|\sqrt{|\Delta|} - 2|a|\right| < b < \sqrt{|\Delta|}. \tag{2.1.2}$$

This definition applies to any Δ, whether it is positive or negative. When $\Delta > 0$, a single equivalence class might have several reduced forms. However, when $\Delta < 0$, an equivalence class can possess only one reduced form (a, b, c), or two reduced forms $(a, \pm b, c)$. In this case,

[1]This kind of equivalence is usually referred to as *proper equivalence*, so our terminology differs from the one that can be found in literature, for example in [BV07, Definition 2.2.3].

the condition (2.1.2) can be simplified to

$$-a < b \leq a < c, \text{ or } 0 \leq b \leq a = c. \tag{2.1.3}$$

The condition (2.1.3) ensures that every equivalence class for $\Delta < 0$ has exactly one reduced form.

It is easy to demonstrate that the form (a, b, c) of discriminant $\Delta < 0$ can represent only positive or only negative integers, depending on the sign of the first coefficient a. That is why such forms are called *positive* or *negative definite* (otherwise they are called *indefinite*). In this thesis, we restrict our attention solely to *positive definite* forms, i.e. forms of discriminant $\Delta < 0$ with their first coefficient being positive.

In order to obtain a reduced form equivalent to some given form (a, b, c), there exist a reduction algorithm, discovered by Gauß [BV07, Algorithm 5.3]. In order to confirm that two forms of discriminant $\Delta < 0$ are equivalent, it is sufficient to verify that their respective equivalent reduced forms match.

As we have seen, two forms belonging to the same equivalence class must have the same discriminant. However, the converse is not true, as the following example demonstrates.

Example 2.1.6. Consider two binary quadratic forms

$$(a, b, c) = x^2 + 5y^2, \quad (a', b', c') = 2x^2 + 2xy + 3y^2.$$

Both of these forms have the discriminant $\Delta = -20$. However, 1 is representable by (a, b, c), but not representable by (a', b', c'). Therefore, (a, b, c) and (a', b', c') are not equivalent.

Definition 2.1.7. The total number of equivalence classes of binary quadratic forms with the same discriminant Δ is called the *class number* with respect to Δ, and denoted by $h(\Delta)$.

Example 2.1.8. In Example 2.1.6, we have $h(-20) = 2$, so any form of discriminant $\Delta = -20$ is equivalent to either $(1, 0, 5)$ or $(2, 2, 3)$.

Gauß proved that $h(\Delta)$ is always finite [Gau98, Chapter 5]. The class number is a very important invariant, associated to the discriminant Δ. For example, it allows one to find a non-trivial factor of Δ [CP05, Section 5.6.1]. However, there exists a lot of computational evidence that the calculation of $h(\Delta)$ is a harder problem than integer factorization. The famous Shanks baby-step giant-step algorithm (BSGS), which requires $O\left(|\Delta|^{\frac{1}{4}+\varepsilon}\right)$ group operations, was originally developed for the purpose of computation of $h(\Delta)$ [Sha71].

Aside from $h(\Delta)$, one other important invariant is associated with the discriminant Δ.

Definition 2.1.9. Consider an arbitrary negative discriminant $\Delta = g^2 e$, where e is the largest in its absolute value fundamental discriminant which divides Δ. Define

$$h_\omega(\Delta) = \begin{cases} h(\Delta), & \text{if } \Delta < -4; \\ 1/2, & \text{if } \Delta = -4; \\ 1/3, & \text{if } \Delta = -3. \end{cases} \tag{2.1.4}$$

Then the number

$$H(\Delta) = \sum_{t \mid g} h_\omega\left(\frac{\Delta}{t^2}\right) \tag{2.1.5}$$

is called the *Hurwitz class number* with respect to Δ.

Note that the equality $H(\Delta) = h(\Delta)$ holds for every fundamental $\Delta < -4$. In this thesis, Hurwitz class numbers appear in two different contexts. First of all, they are a part of the formula which we utilize for class number tabulation. Second, their sum is a part of the (modified) Eichler-Selberg trace formula, which we use in order to eliminate the dependency of our results on the ERH [SvdV91].

Now, let us turn our attention to the *structure* that the set of all equivalence classes of a fixed discriminant Δ possesses. When studying the problem of integer factorization, Fermat noted that for any $m = x^2 + y^2$ and $n = x'^2 + y'^2$, the following equality holds:

$$mn = (x^2 + y^2)(x'^2 + y'^2) = (xx' - yy')^2 + (xy' + yx')^2.$$

That is, if m and n can both be represented by the form $(1, 0, 1)$, then mn is also representable by this form. In "Disquisitiones Arithmeticae", Gauß made a more general observation for arbitrary quadratic forms of the same discriminant [Gau98, §234]. That is, given two forms (a, b, c) and (a', b', c') of the same discriminant Δ, one can produce another quadratic form (a'', b'', c'') with discriminant Δ. The values a'', b'' and c'' can be computed as follows:

$$a'' = \frac{aa'}{d^2}, \;\; b'' = b' + 2\frac{a'}{d}\left(\frac{b - b'}{2}v - c_2 w\right), \;\; c'' = \frac{b''^2 - \Delta}{4a''},$$

where $d = \gcd(a, a', (b + b')/2)$, and u, v, w are three arbitrary integers which satisfy the equation $ua + va' + w(b + b')/2 = d$ [CP05, Algorithm 5.6.7][2]. Note that the form (a'', b'', c'') represents all the possible products mn, where m and n are integers representable by (a, b, c) and (a', b', c'), respectively. This operation is called *composition of forms*, and if we consider the set of all equivalence classes Cl_Δ equipped with the operation of composition, then Cl_Δ forms a *finite abelian group*. Equivalence classes $\left[\left(1, 0, -\frac{\Delta}{4}\right)\right]$ and $\left[\left(1, 1, \frac{1-\Delta}{4}\right)\right]$ play the role of units in Cl_Δ when $\Delta \equiv 0$ and $\Delta \equiv 1 \pmod 4$, respectively. The forms (a, b, c) and $(a, -b, c)$ are called *opposite*, i.e. $(a, -b, c) = (a, b, c)^{-1}$.

The goal of this thesis is to compute the *structure* of each Cl_Δ for $|\Delta| < N$, where we chose N to be 2^{40}. Recall that according to the fundamental theorem of finitely generated abelian groups, Cl_Δ can be decomposed into the direct product of cyclic subgroups, i.e. $Cl_\Delta \cong C(m_0) \times \ldots \times C(m_{r-1})$ for some integer $r > 0$, where $m_{j+1} \,|\, m_j$ for $0 \leq j < r - 1$, and $C(x)$ denotes the cyclic group of order x [JW09, Section 7.1]. To compute $m_0, \ldots, m_{r-1}$, we utilize the Buchmann-Jacobson-Teske algorithm [BJT97, Algorithm 4.1], discussed in Chapter 5. We were also interested in gathering some statistics on Cl_Δ in order to provide evidence in support of certain heuristics, such as the Cohen-Lenstra heuristics [CL83, CL84], and the Littlewood's bounds [Lit28]. These are discussed in Chapter 6 of this thesis.

[2] In practice, we utilize an improved version of this algorithm developed by Shanks [Sha89], which is called NUCOMP [JW09, Section 5.4].

2.2 Binary Quadratic Forms with Even Middle Coefficient

In the previous section, we discussed *binary quadratic forms*, i.e. functions of the form $ax^2 + bxy + cy^2$, where a, b, c are some fixed integers, and x, y are integer variables. However, historically, both Gauß and Dirichlet studied quadratic forms with *even* middle coefficient, $ax^2 + 2bxy + cy^2$ [Gau98, Chapter 5] [Dir63, Chapters 4, 5]. The number $D = b^2 - ac$ was referred by both mathematicians as the *determinant* of a form. We shall denote these forms by $(a, 2b, c)$. Note that unlike the discriminant $\Delta = b^2 - 4ac$, which could only take values $\equiv 0, 1 \pmod 4$, the value of D can potentially be equal to *any* integer. Gauß introduced the following classification of quadratic forms.

Definition 2.2.1. Consider a quadratic form $(a, 2b, c)$ and its *divisor*[3] $\delta = \gcd(a, b, c)$. Then $(a, 2b, c)$ is called *primitive* if $\delta = 1$, and *derived* otherwise.

Definition 2.2.2. A primitive quadratic form $(a, 2b, c)$ is called *proper* if $\gcd(a, 2b, c) = 1$, i.e. its coefficients a and c are not both even; it is called *improper* otherwise[4]. A derived form of divisor δ is proper when $(a/\delta, 2b/\delta, c/\delta)$ is proper; otherwise it is improper.

This classification can also be extended to *equivalence classes* of quadratic forms. That is, if an equivalence class $[(a, 2b, c)]$ contains a proper (or improper) quadratic form $(a, 2b, c)$, then one can show that *all* of its forms are proper (or improper).

Definition 2.2.3. If an equivalence class contains properly primitive quadratic forms, it is called *properly primitive*. If every primitive representative $(a, 2b, c)$ of an equivalence class has even coefficients a and c, it is called *improperly primitive*. *Properly* and *improperly derived* classes can be defined analogously. The sets of all properly and improperly primitive classes of determinant $D = -n$ are called *properly* and *improperly primitive orders*, respectively.

[3] Our terminology differs from Dirichlet, who used the term *divisor* with respect to the value $\sigma = \gcd(a, 2b, c)$ [Dir63, §61].

[4] Kronecker referred to proper forms as *uneven*, and to improper forms as *even* [Kro60, Kro62]. This terminology is primarily used in the theory of elliptic functions (see, for example, [Hum07, Smi94, Wat35]).

We denote their cardinalities by $\tilde{F}(n)$ and $\tilde{F}_1(n)$, respectively. Analogously, we can define *properly* and *improperly derived orders*. Following Kronecker [Kro60], we indicate the total number of proper and improper (primitive or derived) classes of determinant $D = -n$ by $F(n)$ and $F_1(n)$, respectively.

It is possible to express $F(n)$ in terms of $\tilde{F}(n)$, and $F_1(n)$ in terms of $\tilde{F}_1(n)$. Consider an arbitrary positive integer $n = g^2 e$, where e is squarefree. Then the following equalities hold:

$$F(n) = \sum_{t \mid g} \tilde{F}\left(\frac{n}{t^2}\right), \quad F_1(n) = \sum_{t \mid g} \tilde{F}_1\left(\frac{n}{t^2}\right). \tag{2.2.1}$$

Indeed, when $\gcd(a, b, c) = t > 1$, from every properly primitive form $(a/t, 2b/t, c/t)$ of determinant $-n/t^2$ we can obtain every properly derived form $(a, 2b, c)$ of determinant $D = -n$ and divisor $\delta = t$. By adjoining all properly primitive forms with all properly derived ones, we obtain $F(n)$. A similar reasoning allows us to deduce the formula for $F_1(n)$.

Example 2.2.4. Table 2.1 demonstrates the subdivision of equivalence classes of forms with determinant $D = -5^2 \cdot 7$ into four different orders. The binary quadratic forms listed in Table 2.1 are reduced representatives of each equivalence class.

Table 2.1: Subdivision of classes of forms with determinant $D = -5^2 \cdot 7$ into orders

	proper	improper
primitive	$(1, 2\cdot 0, 135), (5, 2\cdot 5, 32),$ $(8, \pm 2\cdot 1, 17), (9, \pm 2\cdot 3, 16)$	$(2, 2\cdot 1, 68), (4, \pm 2\cdot 1, 34)$ $(8, \pm 2\cdot 3, 18), (10, 2\cdot 5, 16)$
derived	$(5, 2\cdot 0, 35)$	$(10, 2\cdot 5, 20)$

In this case, we have

$$F(n) = \tilde{F}(135) + \tilde{F}(7) = 6 + 1 = 7;$$
$$F_1(n) = \tilde{F}_1(135) + \tilde{F}_1(7) = 6 + 1 = 7.$$

Note that $\tilde{F}(7)$ and $\tilde{F}_1(7)$ count only properly and improperly derived forms, which have the divisor 5.

The operation of composition of equivalence classes of forms with even middle coefficient is well defined [Gau98, §234] [Dir63, §146]. If we have two quadratic forms $(a, 2b, c)$ and $(a', 2b', c')$ of the same determinant D such that $\gcd(a, a', b + b') = 1$, then we can produce another form $(a'', 2b'', c'')$ of determinant D for some integers $a'' = aa'$, b'' and c''. Gauß referred to those forms as *composable* (composita), and for any two equivalence classes there always exist quadratic forms that are composable [Dir63, §147]. It is also important to mention that whenever two forms $(a, 2b, c)$ and $(a', 2b', c')$ are composable, $\sigma = \gcd(a, 2b, c)$ and $\sigma' = \gcd(a', 2b', c')$ have to be coprime, and the resulting form $(a'', 2b'', c'')$ will have $\sigma\sigma' = \gcd(a'', 2b'', c'')$ [Dir63, §147]. From here it follows that the set of all properly primitive classes forms a group, because all quadratic forms in these classes have $\sigma = 1$. In contrast, the set of improperly primitive classes does not form a group.

In the remainder of this section, we aim to prove the following theorem, which describes the connection between regular binary quadratic forms and binary quadratic forms with even middle coefficient.

Theorem 2.2.5. *For an arbitrary negative discriminant Δ, the following relation holds:*

$$h(\Delta) = \begin{cases} \tilde{F}(n), & \text{when } \Delta \equiv 0, 1, 4 \pmod 8 \text{ or } \Delta = -3; \\ \tilde{F}(n)/3, & \text{when } \Delta \equiv 5 \pmod 8 \text{ and } \Delta \neq -3, \end{cases} \tag{2.2.2}$$

where

$$n = \begin{cases} -\Delta/4, & \text{if } \Delta \equiv 0 \pmod 4; \\ -\Delta, & \text{if } \Delta \equiv 1 \pmod 4. \end{cases} \tag{2.2.3}$$

Proof. Consider an arbitrary positive integer n. We begin by demonstrating that for any such n the value $\tilde{F}_1(n)$, if it is non-zero, divides $\tilde{F}(n)$. The proof of this fact was first demonstrated by Gauß [Gau98, §256]. An alternative approach was given by Dirichlet [Dir63, §94, §150], which he stated for an arbitrary non-trivial $\sigma = \gcd(a, 2b, c)$ of a quadratic form $(a, 2b, c)$. Of course, we are interested in the case $\sigma = 2$, i.e. when $(a, 2b, c)$ is an improperly primitive quadratic form.

We begin with a simple observation that every improperly primitive form $(a, 2b, c)$ of determinant $D = -n$ must have b odd, for otherwise a, b and c would have 2 as a divisor. This implies that $D = b^2 - ac \equiv 1 \pmod 4$, since $b^2 \equiv 1 \pmod 4$. Therefore, for any $D \not\equiv 1 \pmod 4$, the number of improperly primitive classes $\tilde{F}_1(n)$ is zero.

Next, recall that the set of improperly primitive classes does not form a group under composition. Despite that, it is still possible to compose improperly primitive classes with properly primitive ones. In fact, every improperly primitive class K can be obtained from a (not necessarily unique) properly primitive class H by means of the relation $SH = K$, where $S = \left[\left(2, 2{\cdot}1, \frac{1-D}{2}\right)\right]$ denotes the trivial improperly primitive class [Dir63, §148, 3]. Now, let $K = S$. If any two properly primitive classes R and R' satisfy $SR = S$ and $SR' = S$ respectively, then $S(RR') = S$. Therefore, the set $\mathfrak{R}$ of all properly primitive R satisfying $SR = S$ forms a subgroup of a group of all properly primitive classes. We conclude that its cardinality, which we shall denote by r, has to divide $\tilde{F}(n)$, so $\tilde{F}(n) = rk$ for some integer k. Note that for a fixed properly primitive H we have $SRH = SH$ for any $R \in \mathfrak{R}$, and SH is improperly primitive. On the other hand, if $SH' = SH$ for some properly primitive H and H', then $R = H'H^{-1}$ has to be in $\mathfrak{R}$, hence $H' = RH \in \mathfrak{R}H$. So the total number of improperly primitive classes $\tilde{F}_1(n)$ is exactly k, and therefore $\tilde{F}(n) = r\tilde{F}_1(n)$.

To find the value of r, we refer to the theorem of Dirichlet [Dir63, §150]. It states that $\mathfrak{R}$ contains only those properly primitive classes whose quadratic forms can represent either 1 or 4. There are only three forms of that kind, namely $(1, 2{\cdot}0, -D)$ and $\left(4, \pm 2{\cdot}1, \frac{1-D}{4}\right)$. Now there are only two cases left to consider. Whenever $D \equiv 1 \pmod 8$, $\frac{1-D}{4}$ has to be even, and therefore only the first form is proper, so $r = 1$ and $\tilde{F}(n) = \tilde{F}_1(n)$. However, when $D \equiv 5 \pmod 8$, the last two forms are both proper as well, and we need to understand whether some of them are equivalent. In fact, these three forms belong to the same equivalence class only if $D = -3$, which implies $\tilde{F}(3) = \tilde{F}_1(3)$. For any other $D \equiv 5 \pmod 8$, these forms will always belong to three different equivalence classes, and therefore $\tilde{F}(n) = 3\tilde{F}_1(n)$ [Dir63,

§151]. We obtain the following formula as a result:

$$\tilde{F}_1(n) = \begin{cases} \tilde{F}(n), & \text{when } n \equiv 7 \pmod 8 \text{ or } n = 3; \\ \tilde{F}(n)/3, & \text{when } n \equiv 3 \pmod 8 \text{ and } n \neq 3; \\ 0, & \text{when } n \not\equiv 3 \pmod 4. \end{cases} \tag{2.2.4}$$

Now we can finally write down the formula for $h(\Delta)$ with respect to $\tilde{F}(n)$, where $n = -D$ and $D < 0$. When $D \equiv 1 \pmod 4$, each improperly primitive form $ax^2 + 2bxy + cy^2$ of determinant D, where a, c are even and b odd, can be transformed into a form $(a/2)x^2 + bxy + (c/2)y^2$ with discriminant $\Delta \equiv 1 \pmod 4$. This map is bijective, since every form $ax^2 + bxy + cy^2$ of discriminant Δ with odd b corresponds to an improperly primitive form $2ax^2+2bxy+2cy^2$ of determinant D. We conclude that $h(\Delta) = \tilde{F}_1(n)$. When $D \not\equiv 1 \pmod 4$, there are no improperly primitive forms, and a properly primitive form $ax^2+2bxy+cy^2$ with determinant D already has a discriminant $\Delta = 4D$, so $h(\Delta) = \tilde{F}(n)$. There are no other forms of discriminant Δ, besides those of determinant D. In the end, we obtain the relation (2.2.2). □

2.3 Subdivision of Classes Into Genera

Recall that in Section 2.1 we subdivided all the binary quadratic forms into *equivalence classes*. After that, in Section 2.2, we demonstrated that every binary quadratic form with even middle coefficient $(a, 2b, c)$ belongs to one of the four orders, depending on the parity of its coefficients a and c, and the value of its divisor $\delta = \gcd(a, b, c)$. In this section, we introduce one more classification given by Gauß, who subdivided these forms into various *genera*. The observations of this section are crucial for the proof of the class number tabulation formula, which we introduce in Chapter 3.

Let $(a, 2b, c)$ be a form of determinant D, and p be an odd prime divisor of D. Then integers not divisible by p and representable by $(a, 2b, c)$ are either all quadratic residues or all quadratic non-residues of p [Gau98, §229]. The proof of this fact is rather simple: if two

integers, m and m', are representable by $(a, 2b, c)$, i.e.

$$m = ag^2 + 2bgh + ch^2, \;\; m' = ag'^2 + 2bg'h' + ch'^2,$$

then

$$mm' = (agg' + b(gh' + hg') + chh')^2 - D(gh' - hg')^2.$$

Now, $\left(\frac{mm'}{p}\right) = \left(\frac{m}{p}\right) \cdot \left(\frac{m'}{p}\right) = 1$, which implies that $\left(\frac{m}{p}\right) = \left(\frac{m'}{p}\right) = \pm 1$, where $\left(\frac{m}{p}\right)$ denotes the Kronecker symbol. Following the terminology of Gauß, we say that a form $(a, 2b, c)$ corresponds to a *particular character* Rp (or Np) whenever all integers not divisible by p and representable by $(a, 2b, c)$ are quadratic residues (or non-residues) of p, where p is an odd prime divisor of D.

Thus, every quadratic form corresponds to a set of particular characters for all odd prime divisors of D, which Gauß referred to as the *complete character*. Any other quadratic form F', equivalent to a given form F, will have the same complete character as F, since F and F' represent the same set of integers. Therefore, every equivalence class possesses a certain complete character. This allows to divide equivalence classes of a fixed determinant D into *genera* by putting classes with the same complete character into a single *genus*. Just like orders, Gauß subdivided genera into proper and improper, primitive and derived.

Example 2.3.1. Let $D = -3^2 \cdot 5 \cdot 7$. In Table 2.2, we demonstrate the subdivision of equivalence classes of forms into various genera. The binary quadratic forms listed in Table 2.2 are reduced representatives of each equivalence class.

We use Example 2.3.1 to demonstrate three important observations:

(a) Note that in Example 2.3.1, proper and improper primitive orders contain the same number of genera, which we denote by g. In fact, this holds for any $D \equiv 1 \pmod 4$, such that $D < -3$. If we consider a form ϕ, composed of two primitive forms f and f', of which one is properly primitive, then one can determine a complete character of ϕ.

Table 2.2: Subdivision of classes of forms with determinant $D = -3^2 \cdot 5 \cdot 7$ into genera

	proper		improper	
	form	complete character	form	complete character
primitive	$(1, 2\cdot 0, 315), (4 \pm 2\cdot 1, 79)$	R3 R5 R7	$(10, 2\cdot 5, 34)$	R3 R5 N7
	$(7, 2\cdot 0, 45), (13, \pm 2\cdot 6, 27)$	R3 N5 N7	$(14, 2\cdot 7, 26)$	R3 N5 R7
	$(9, 2\cdot 0, 35), (11, \pm 2\cdot 2, 29)$	N3 R5 R7	$(18, 2\cdot 9, 22)$	N3 R5 N7
	$(5, 2\cdot 0, 63), (17, \pm 2\cdot 5, 25)$	N3 N5 N7	$(2, 2\cdot 1, 158)$	N3 N5 R7
derived	$(15, 2\cdot 0, 21), (9, \pm 2\cdot 3, 36)$	R5 R7	$(6, 2\cdot 3, 54)$	R5 N7
	$(3, 2\cdot 0, 105), (12, \pm 2\cdot 9, 33)$	N5 N7	$(18, 2\cdot 3, 18)$	N5 R7

In particular, if we fix an odd prime p which divides D, and both f and f' correspond to the same particular character Rp or Np, then ϕ has to be in Rp. Alternatively, if one of these forms is in Rp, and another is in Np, then ϕ has to be in Np [Gau98, §246].

Now, from the previous section we know that each improperly primitive class K is derived from some (not necessarily unique) properly primitive class H by means of the relation $SH = K$, where $S = \left[\left(2, 1, \frac{1-D}{2}\right)\right]$. If we now take representatives $H_1, \ldots, H_g$ from each of the g genera in a properly primitive order, and compose them with S, we obtain improperly primitive classes $K_1, \ldots, K_g$. These classes must belong to pairwise distinct genera, for otherwise some H_i, H_j for $1 \leq i < j \leq g$ would have to belong to the same genus. We conclude that for $D \not\equiv 1 \pmod 4$ the number of genera in properly and improperly primitive orders match.

(b) Observe that in Example 2.3.1, as in general, for any possible complete character there always exists a primitive form of that complete character [Gau98, §261 – §264]. In total, there are precisely 2^μ complete characters, where μ is the number of odd prime divisors of D (not counting multiplicity). Whenever $D \equiv 1 \pmod 4$, the total 2^μ complete characters are equally distributed among properly and improperly primitive orders, so each primitive order contains $g = 2^{\mu-1}$ genera. In contract, when $D \not\equiv 1 \pmod 4$, an improperly primitive order does not exist, so the properly primitive one contains all

$g = 2^\mu$ complete characters. We can write the precise formula for g as follows:

$$g = \begin{cases} 2^{\mu-1}, & \text{if } D \equiv 1 \pmod 4; \\ 2^\mu, & \text{if } D \not\equiv 1 \pmod 4, \end{cases} \tag{2.3.1}$$

where μ is the number of odd prime divisors of D.

(c) Finally, note that each properly and improperly primitive genus contains the same number of classes k and k', respectively. In fact, this is true for *any* determinant $D = -n$ [Gau98, §252]. We can therefore conclude that

$$\tilde{F}(n) = gk \text{ and } \tilde{F}_1(n) = gk', \tag{2.3.2}$$

where g is defined in (2.3.1). As we observed previously, $k' = 0$ when $D \not\equiv 1 \pmod 4$. Otherwise, k and k' are connected by means of the relation

$$k' = \begin{cases} k, & \text{when } n \equiv 7 \pmod 8 \text{ or } n = 3; \\ k/3, & \text{when } n \equiv 3 \pmod 8 \text{ and } n \neq 3; \\ 0, & \text{when } n \not\equiv 3 \pmod 4, \end{cases}$$

derived from (2.2.4).

In order to conclude our investigations, let us introduce the notions of a *quadratic residue* of a binary quadratic form, and its *characteristic number*.

Definition 2.3.2. [Gau98, §233] If the primitive form $(a, 2b, c)$ of determinant $D = -n$ is such that there exists a pair of numbers (v, w) and an integer M coprime to n, satisfying

$$v^2 \equiv aM, \;\; vw \equiv bM, \;\; w^2 \equiv cM \pmod n,$$

then the binary quadratic form $M(a, 2b, c)$ is called a *quadratic residue* of n, and (v, w) is called a *value* of the expression $\sqrt{M(a, 2b, c)} \pmod n$. The number M is called a *characteristic number* of $(a, 2b, c)$.

In total, there exist 2^μ values (v, w) modulo n, where μ is the number of odd prime divisors of n (not counting multiplicity). Also, each form has many characteristic numbers. These numbers possess a beautiful property that they fully determine the complete character of a form [Gau98, §233.IV]. That is, if a form $(a, 2b, c)$ has characteristic number M and M is a quadratic residue of an odd prime $p \mid D$, then $(a, 2b, c)$ corresponds to a particular character Rp. An analogous statement holds when M is a quadratic non-residue of p, which implies that $(a, 2b, c)$ is in Np. Evidently, forms with the same characteristic number must belong to a single genus.

Example 2.3.3. We extend Example 2.3.1 by listing the smallest positive characteristic numbers and one of the values $\sqrt{M(a, 2b, c)} \pmod{315}$ of each properly and improperly primitive binary quadratic form in Table 2.2. We indicate in bold the genus with characteristic number -1. As we shall observe in the following section, each form in this genus can be transformed into the ternary quadratic form $x^2 + y^2 + z^2$.

Table 2.3: Characteristic numbers of forms with determinant $D = -3^2 \cdot 5 \cdot 7$

proper			improper		
$(a, 2b, c)$	M	$\sqrt{M(a, 2b, c)}$	$(a, 2b, c)$	M	$\sqrt{M(a, 2b, c)}$
$(1, 2{\cdot}0, 315), (4, \pm 2{\cdot}1, 79)$	1	$(1, 0), (2, \pm 157)$	$(10, 2{\cdot}5, 34)$	19	(190, 284)
$(7, 2{\cdot}0, 45), (13, \pm 2{\cdot}6, 27)$	13	$(91, 135), (13, \pm 6)$	$\mathbf{(18, 2{\cdot}9, 22)}$	**22**	**(9, 22)**
$(9, 2{\cdot}0, 35), (11, \pm 2{\cdot}2, 29)$	11	$(162, 140), (11, \pm 2)$	$(14, 2{\cdot}7, 26)$	26	(7, 26)
$(5, 2{\cdot}0, 63), (17, \pm 2{\cdot}5, 20)$	17	$(155, 126), (17, \pm 5)$	$(2, 2{\cdot}1, 158)$	2	(2, 1)

2.4 Ternary Quadratic Forms

So far, we have been looking at *binary* quadratic forms, which are functions of two integer variables x and y. Of course, nothing prevents us from looking at quadratic forms of three or more variables, though we would expect the theory to get more sophisticated as the number of variables increases. Though Gauß was the pioneer in his attempt to systematize the knowledge about ternary quadratic forms, he was mostly interested not in this theory itself,

but in its connection and similarities with the theory of quadratic forms in two variables. In "Disquisitiones Arithmeticae", he demonstrated two important results. First of all, ternary quadratic forms can also be subdivided into equivalence classes, and the number of those equivalence classes for a fixed determinant D is finite [Gau98, §276]. Second, and much more important for us, he established that the number of solutions $r_3(n)$ to the Diophantine equation $n = x^2 + y^2 + z^2$ has a connection to the number of equivalence classes of binary quadratic forms with determinant $D = -n$.

In this section, we outline a simple procedure which Gauß developed to connect a three squares representation of an integer $n = x_0^2 + y_0^2 + z_0^2$ to some binary quadratic form of determinant $D = -n$. We begin with the following definitions.

Definition 2.4.1. A function $f(x, y, z)$ of the form

$$f(x, y, z) = ax^2 + a'y^2 + a''z^2 + 2byz + 2b'xz + 2b''xy,$$

where a, a', a'', b, b', b'' are fixed integers, and x, y, z are integer variables, is called a *ternary quadratic form.*

Definition 2.4.2. The integer $D = ab^2+a'b'^2+a''b''^2-aa'a''-2bb'b''$ is called the *determinant* of the ternary quadratic form $f(x, y, z)$.

Definition 2.4.3. Given a ternary quadratic form $f(x, y, z)$, the integer n is *representable* by f if there exist integers x_0, y_0 and z_0, such that $f(x_0, y_0, z_0) = n$. The triple x_0, y_0, z_0 is called the *representation of n by $f(x, y, z)$*. This representation is *primitive* if $\gcd(x_0, y_0, z_0) = 1$, and *derived* otherwise.

Example 2.4.4. The most simple example of a ternary quadratic form is $f = x^2 + y^2 + z^2$, where $a = a' = a'' = 1$, $b = b' = b'' = 0$, and $D = -1$.

Note that any ternary quadratic form can be transformed into a binary quadratic form of integer variables t, u by means of the substitution

$$x = mt + nu, \;\; y = m't + n'u, \;\; z = m''t + n''u, \tag{2.4.1}$$

where m, m', m'', n, n', n'' are fixed integers [Gau98, §278]. Such a binary form is called *representable* by a given ternary form. Moreover, *every* representation of an integer n by a ternary form f can be deduced from some binary form of determinant $D = -n$. As a special case, Gauß considered the ternary quadratic form $f = x^2 + y^2 + z^2$. If an integer n is representable by f for some $x = x_0$, $y = y_0$ and $z = z_0$, then for integers m, m', m'', n, n', n'', satisfying

$$\begin{cases} m'n'' - m''n' & = x_0; \\ m''n - mn'' & = y_0; \\ mn' - m'n & = z_0, \end{cases} \tag{2.4.2}$$

the binary form $\phi(t, u)$ obtained by means of the substitution (2.4.1) has a determinant $D = -n$. The algorithm for solving the system above is described in the monograph of Gauß [Gau98, §279].

Example 2.4.5. Consider the ternary form $f = x^2 + y^2 + z^2$. Let us take one representation of the number $n = 822$ by f, for example $x_0 = 5$, $y_0 = 11$ and $z_0 = 26$. Then the integers $m = -11$, $m' = 5$, $m'' = 0$, $n = 74$, $n' = -36$, $n'' = 1$ satisfy the system of equations (2.4.2), so the substitution (2.4.1) results in the quadratic form $146t^2 - 1988tu + 6773u^2$, which has determinant $D = -n = -822$.

Now it would be interesting to consider the opposite question; that is, given a properly primitive form $(a, 2b, c)$ of determinant $D = -n$ and a ternary form f of determinant D', what are representations of $(a, 2b, c)$ by f? Gauß discovered that these representations exist if and only if D' is a characteristic number of $(a, -2b, c)$ [Gau98, §282]. In other words, $D'(a, -2b, c)$ has to be a quadratic residue of n, and if this is the case then every value $(v, w) = \sqrt{D'(a, -2b, c)} \pmod{n}$ produces 48 representations. For the form $f = x^2 + y^2 + z^2$, these representations of $(a, 2b, c)$ by f can be converted into 48 three squares representations of an integer n using the system (2.4.2). In the following chapter, we present Gauß's formula for the number of primitive representations of an integer n as a sum of three squares.

Chapter 3

THE THETA SERIES

In this chapter, our goal is to prove Theorem 3.2.3, which allows us to deduce nice formulas, suitable for the tabulation of class numbers $h(\Delta)$ with $\Delta \not\equiv 1 \pmod 8$ up to some given bound N. We describe the connection that exists between the *theta series* $\vartheta_3(q)$, the *sum of squares function* $r_3(n)$, and the class number $h(\Delta)$, where Δ and n are connected by means of the relation (2.2.3).

3.1 Jacobi Theta Functions

Let τ be a fixed complex number with positive imaginary part, and $q = e^{\pi i \tau}$. Consider the following four functions:

$$\begin{aligned}
\vartheta_1(z,q) &= 2\sum_{n=0}^{\infty}(-1)^n q^{\left(n+\frac{1}{2}\right)^2}\sin\left((2n+1)z\right);\\
\vartheta_2(z,q) &= 2\sum_{n=0}^{\infty} q^{\left(n+\frac{1}{2}\right)^2}\cos\left((2n+1)z\right);\\
\vartheta_3(z,q) &= 1+2\sum_{n=1}^{\infty} q^{n^2}\cos\left(2nz\right);\\
\vartheta_4(z,q) &= 1+2\sum_{n=1}^{\infty}(-1)^n q^{n^2}\cos\left(2nz\right).
\end{aligned} \tag{3.1.1}$$

First studied by Jacobi, these functions are now known as his famous *theta functions* [Jac29]. Due to the fact that $|q| < 1$, each $\vartheta_k(z,q)$ (where $1 \le k \le 4$) is uniformly convergent as a series of analytic functions on any bounded domain of z [WW02, Chapter 21]. In other words, theta functions are *analytic* at all finite points of the complex plane $\mathbb{C}$.

Theta functions have other beautiful properties, such as *quasi doubly-periodicity*. That is, any theta function $\vartheta_k(z,q)$ satisfies $\vartheta_k(z+\pi,q) = c_1\vartheta_k(z,q)$ and $\vartheta_k(z+\pi\tau,q) = c_2\,\vartheta_k(z,q)$ for some complex constants c_1, c_2 not both equal to 1, known as *periodicity factors*. Functions which satisfy these relations for $c_1 = c_2 = 1$ are known as *doubly-periodic*, or *elliptic*. Elliptic functions were of primary concern in Jacobi's original work, and today these functions are

known to have deep connections to important fields of mathematics, such as the *theory of elliptic curves* and the *theory of modular forms* [Kob84]. However, in this thesis we prefer a purely algebraic approach to the study of theta functions. Those who are keen to investigate the subject from other perspectives we refer to the monographs [AE06] and [DS05]. The first one gives a thorough explanation of how theta functions arise in the theory of elliptic functions, while the second one (see Section 1.2) demonstrates, for example, that $\vartheta_3^4(z,q)$ is a *modular form*.

We now leave these sophisticated theories behind, focusing our attention on some algebraic properties of theta functions. Using $\vartheta_k(q)$ to denote $\vartheta_k(0,q)$, consider the following two infinite series:

$$\vartheta_2(q) = 2\sum_{n=0}^{\infty} q^{\left(n+\frac{1}{2}\right)^2} = 2q^{\frac{1}{4}} + 2q^{\frac{9}{4}} + 2q^{\frac{25}{4}} + 2q^{\frac{49}{4}} + \dots \ ;$$

$$\vartheta_3(q) = 1 + 2\sum_{n=1}^{\infty} q^{n^2} = 1 + 2q + 2q^4 + 2q^9 + \dots \ .$$

We refer to $\vartheta_k(q)$ as the *k-th theta series*. We shall often drop the number k for brevity, when there is no ambiguity involved. In $\vartheta_k(q)$, we are not really concerned with what q is; we simply assume that it allows the series to converge. We shall also utilize the notation $\vartheta_{k,N}(q)$ to denote the polynomial which can be obtained by truncation of a k-th theta series to degree $N-1$.

Now, consider the following definitions.

Definition 3.1.1. For two positive integers, k and n, the *sum of squares function* $r_k(n)$ counts the number of solutions to the Diophantine equation $n = x_1^2 + x_2^2 + \ldots + x_k^2$. The solution $x_1, \ldots, x_k$ is called *primitive* whenever $\gcd(x_1, \ldots, x_k) = 1$; it is called *derived* otherwise[1]. We denote the function which counts all the primitive solutions by $\tilde{r}_k(n)$.

There exists a straightforward connection between $r_k(n)$ and $\tilde{r}_k(n)$. In particular, if we

[1]Gauß referred to these solutions as *proper* and *improper*, respectively.

write $n = g^2 e$ where e is squarefree, then

$$r_k(n) = \sum_{t|g} \tilde{r}_k\left(\frac{n}{t^2}\right). \tag{3.1.2}$$

Indeed, in order to find all the integer solutions to $n = x_1^2 + \ldots + x_k^2$, we first need to count the primitive solutions. Then we add up the number of primitive solutions to the equations $n/t^2 = (x_1/t)^2 + \ldots + (x_k/t)^2$, where t runs over all the divisors of g. We can easily notice that such primitive solutions induce derived solutions to our original equation, where x, y and z have the greatest common divisor t.

Next, observe that the equality $\vartheta_3(q) = \sum_{n=0}^{\infty} r_1(n) q^n$ holds for any n. Indeed, when $n = m^2$ for some integer m, there are exactly two representations as a square, namely $n = m^2$ and $n = (-m)^2$, unless $n = 0$, when these two representations coincide. This result can be generalized as follows.

Theorem 3.1.2. [Dir63, §92] *For any positive integer k, the following equality holds:*

$$\vartheta_3^k(q) = \sum_{n=0}^{\infty} r_k(n) q^n. \tag{3.1.3}$$

Proof. Consider the series $\sum_{x_1,\ldots,x_k \in \mathbb{Z}} q^{x_1^2+\ldots+x_k^2}$. The coefficient of q^n is equal to $r_k(n)$, so $\sum_{x_1,\ldots,x_k \in \mathbb{Z}} q^{x_1^2+\ldots+x_k^2} = \sum_{n=0}^{\infty} r_k(n) q^n$. On the other hand, the following equality holds:

$$\sum_{x_1,\ldots,x_k \in \mathbb{Z}} q^{x_1^2+\ldots+x_k^2} = \sum_{x_1 \in \mathbb{Z}} q^{x_1^2} \times \sum_{x_2 \in \mathbb{Z}} q^{x_2^2} \times \ldots \times \sum_{x_k \in \mathbb{Z}} q^{x_k^2} = \left(\sum_{x \in \mathbb{Z}} q^{x^2}\right)^k.$$

But the series $\sum_{x \in \mathbb{Z}} q^{x^2}$ is exactly $\vartheta_3(q)$, and therefore $\vartheta_3^k(q) = \sum_{n=0}^{\infty} r_k(n) q^n$. □

Of course, the simplicity of our proof relies on the fact that representation $x_1^2 + \ldots + x_k^2$ does not contain mixed products $x_i x_j$ for $1 \le i < j \le k$. Construction of a similar formula for different representations, for example $x^2 + xy + y^2$, would require more deep investigations. These formulas, along with a question of representability of a number n by an *arbitrary* quadratic form, were studied by Chandrasekharan [Cha85, Chapter 11].

3.2 Relationship Between Theta Series and Class Numbers

A connection between the function $r_3(n)$ and quadratic forms with even middle coefficient of determinant $D = -n$ was first established by Gauß [Gau98, §291]. We present his results in Theorem 3.2.1.

Over sixty years later, this connection was also established by Kronecker during his study of the theory of elliptic functions [Kro60]. The proofs of Kronecker's relationship from the perspective of elliptic functions can be found in [Smi94, §130] and [AE06, Chapter 11]. In Theorem 3.2.2, we follow an alternative approach and derive Kronecker's relationship from the result of Gauß.

The formula that we use for tabulation purposes differs from both those of Gauß and Kronecker as it relates $r_3(n)$ to quadratic forms with arbitrary middle coefficient. In Theorem 3.2.3, we derive our main formula (3.2.3) from the result of Gauß. To the best of our knowledge, this formula is not present in any literature available, though its statement for the special case of squarefree $n > 4$ can be found in the monograph of Grosswald [Gro85, Chapter 4, Theorem 2].

Theorem 3.2.1. [Gau98, §291.III] *For $n \neq 1, 3$, the number of primitive solutions $\tilde{r}_3(n)$ to the Diophantine equation $n = x^2 + y^2 + z^2$ can be computed as follows:*

$$\tilde{r}_3(n) = 3 \cdot 2^{\mu+2} k, \tag{3.2.1}$$

where μ is the number of odd prime divisors of n (not counting multiplicity), and k is the number of classes of binary quadratic forms with determinant $D = -n$, contained in a primitive genus with characteristic number -1.

Proof. From Section 2.4, we know that a ternary quadratic form $f(x, y, z)$ can be transformed into binary quadratic forms of determinant $D = -n$ by means of various substitutions of the form (2.4.1). The number of these transformations is intimately connected to the number of solutions of the Diophantine equation $n = f(x, y, z)$. Following Gauß, let us count all

the possible representations of binary quadratic forms of a fixed determinant $D = -n$ by a ternary quadratic form $f = x^2 + y^2 + z^2$ with determinant $D' = -1$.

Recall from Section 2.4 that any primitive representation (2.4.1) of $(a, 2b, c)$ by f arises from some value $\sqrt{D'(a, -2b, c)} \pmod{n}$. Therefore, the determinant $D' = -1$ of f has to be a characteristic number of $(a, -2b, c)$. From Section 2.3 we know that forms which share a characteristic number must belong to the same genus.

Next, recall from Section 2.3 that there are 2^μ values $(v, w) = \sqrt{-(a, -2b, c)} \pmod{n}$, where μ is the number of odd prime divisors of n (not counting multiplicity). When $n > 2$, we are interested in only half of those values, because (v, w) and $(-v, -w)$ produce similar transformations [Gau98, §289.III]. By [Gau98, §283], a single value (v, w) corresponds to 48 representations of n. This allows us to conclude that the number of all proper representations of $(a, 2b, c)$ by f will be $48 \cdot 2^{\mu-1}$.

Consequently, since all the forms representable by f belong to the same primitive genus with k elements, there must exist $48 \cdot 2^{\mu-1}k$ representations. All these representations are guaranteed to be different [Gau98, §280.III], with the only exception corresponding to transformations (2.4.1) and

$$x = -mt - nu, \;\; y = -m't - n'u, \;\; z = -m''t - n''u$$

for $n \neq 1, 3$, which induce the same representation of n by f [Gau98, §291]. From here it follows that $\tilde{r}_3(n) = 48 \cdot 2^{\mu-1}k/2 = 3 \cdot 2^{\mu+2}k$, where k is the number of classes in a primitive genus with characteristic number -1. □

Theorem 3.2.2. [Smi94, §130] *Allow $F(n)$ and $F_1(n)$ to count the classes $[(1, 2\cdot 0, 1)]$ and $[(2, 2\cdot 1, 2)]$, and classes derived from them, as $1/2$ and $1/3$, respectively. Then*

$$r_3(n) = \begin{cases} r_3(n/4), & \text{if } n \equiv 0 \pmod 4; \\ 12\left[F(n) - F_1(n)\right], & \text{if } n \not\equiv 0 \pmod 4. \end{cases} \tag{3.2.2}$$

Proof. From Theorem 3.2.1, we know that primitive forms with characteristic number -1 must belong to the same genus. Gauß found that this genus does not exist when $n \equiv 0, 4, 7$

(mod 8) [Gau98, §288]. For other values of n this genus is proper when $n \equiv 1, 2, 5, 6 \pmod 8$, and improper when $n \equiv 3 \pmod 8$ (as we observed in Example 2.3.3). Now, consider four cases:

(a) Let $n \equiv 0 \pmod 4$, and consider the Diophantine equation $n = x^2 + y^2 + z^2$. Because $x^2 \equiv 0, 1 \pmod 4$ for any integer x, the congruence $x^2 + y^2 + z^2 \equiv 0 \pmod 4$ can be satisfied only if x, y, z are even. From here it follows that the original equation and the equation $n/4 = (x/2)^2 + (y/2)^2 + (z/2)^2$ have the same number of solutions, i.e. $r_3(n) = r_3(n/4)$;

(b) Let $n \equiv 7 \pmod 8$. Since for any integer x we have $x^2 \equiv 0, 1, 4 \pmod 8$, it is easy to verify that there are no x, y, z that can satisfy the congruence $x^2 + y^2 + z^2 \equiv 7 \pmod 8$. We conclude that $\tilde{r}_3(n) = 0$, or equivalently $\tilde{r}_3(n) = 12\left[\tilde{F}(n) - \tilde{F}_1(n)\right]$, because according to the relation (2.2.4) we have $\tilde{F}(n) = \tilde{F}_1(n)$;

(c) Let $n \equiv 1, 2, 5, 6 \pmod 8$. For $n = 1$, we have $\tilde{F}(1) = 1/2$ and $\tilde{F}_1(1) = 0$, since there exist only one primitive class $[(1, 2{\cdot}0, 1)]$. We can easily verify that formula (3.2.2) gives us the correct result $r_3(1) = 6$. For $n \neq 1$, all the quadratic forms belong to a *proper* order. From Section 2.3, we know that $\tilde{F}(n) = gk$, where $g = 2^\mu$ is the number of genera, and μ is the number of odd prime divisors of n (not counting multiplicity). Also, from Section 2.2, we know that $\tilde{F}_1(n) = 0$. We obtain $\tilde{r}_3(n) = 3 \cdot 2^{\mu+2}k = 12\left[\tilde{F}(n) - \tilde{F}_1(n)\right]$;

(d) Let $n \equiv 3 \pmod 8$. For $n = 3$, we have $\tilde{F}(3) = 1$ and $\tilde{F}_1(3) = 1/3$, since there exist only two primitive classes $[(1, 2{\cdot}0, 3)]$ and $[(2, 2{\cdot}1, 2)]$. We can easily verify that formula (3.2.2) gives us the correct result $r_3(3) = 8$. For $n \neq 3$, all the quadratic forms belong to an *improper* order. From Section 2.3, we know that $\tilde{F}_1(n) = gk$, where $g = 2^{\mu-1}$. From Section 2.2, we know that $\tilde{F}(n) = 3\tilde{F}_1(n)$. We obtain $\tilde{r}_3(n) = 3 \cdot 2^{\mu+2}k = 24\tilde{F}_1(n) = 12\left[\tilde{F}(n) - \tilde{F}_1(n)\right]$.

Now that we obtained the formula for $\tilde{r}_3(n)$, we need to generalize it to $r_3(n)$. For that

purpose, recall the formulas (2.2.1), (3.1.2), and write $n = g^2 e$ where e is squarefree. Then we can obtain the relation (3.2.2) as follows:

$$r_3(n) = \sum_{t \mid g} \tilde{r}_3\left(\frac{n}{t^2}\right) = \sum_{t \mid g} 12\left[\tilde{F}\left(\frac{n}{t^2}\right) - \tilde{F}_1\left(\frac{n}{t^2}\right)\right] =$$

$$= 12\left[\sum_{t \mid g} \tilde{F}\left(\frac{n}{t^2}\right) - \sum_{t \mid g} \tilde{F}_1\left(\frac{n}{t^2}\right)\right] = 12\left[F(n) - F_1(n)\right].$$

□

Theorem 3.2.3. [Gro85, Chapter 4, Theorem 2, generalized] *Let n be an arbitrary positive integer. Then*

$$r_3(n) = \begin{cases} r_3(n/4), & \text{if } n \equiv 0 \pmod 4; \\ 12H(-4n), & \text{if } n \equiv 1, 2 \pmod 4; \\ 24H(-n), & \text{if } n \equiv 3 \pmod 8; \\ 0, & \text{if } n \equiv 7 \pmod 8, \end{cases} \tag{3.2.3}$$

where $H(\Delta)$ denotes the Hurwitz class number of discriminant Δ.

Proof. In Theorem 3.2.2, we already considered cases $n \equiv 0, 4, 7 \pmod 8$. Now, suppose that $n \not\equiv 0, 4, 7 \pmod 8$. Analogously to Theorem 3.2.2, consider the following two cases:

(a) Let $n \equiv 1, 2, 5, 6 \pmod 8$. For $n = 1$, we can verify that the formula (3.2.3) gives us the correct result. For $n \neq 1$, all the quadratic forms belong to a *proper* order. From Section 2.2, we know that $\tilde{F}_1(n) = 0$, and $h(\Delta) = \tilde{F}(n)$, where $\Delta = -4n$ according to the relation (2.2.3). Therefore, $\tilde{r}_3(n) = 12\left[\tilde{F}(n) - \tilde{F}_1(n)\right] = 12h_\omega(-4n)$, where $h_\omega(\Delta)$ is defined in (2.1.4);

(b) Let $n \equiv 3 \pmod 8$. For $n = 3$, we can verify that the formula (3.2.3) gives us the correct result. For $n \neq 3$, all the quadratic forms belong to an *improper* order. From Section 2.2, we know that $\tilde{F}(n) = 3\tilde{F}_1(n)$, and $h(\Delta) = \tilde{F}_1(n)$, where $\Delta = -n$ according to the relation (2.2.3). Therefore, $\tilde{r}_3(n) = 12\left[\tilde{F}(n) - \tilde{F}_1(n)\right] = 24h_\omega(-n)$, where $h_\omega(\Delta)$ is defined in (2.1.4).

If we now apply the formula (3.1.2), by Definition 2.1.9 we obtain two more relations from (3.2.3):

$$r_3(n) = \sum_{t \mid g} \tilde{r}_3\left(\frac{n}{t^2}\right) = \begin{cases} 12 \sum_{t \mid g} h_\omega\left(\frac{-4n}{t^2}\right) = 12H(-4n), & \text{if } n \equiv 1, 2 \pmod 4; \\ 24 \sum_{t \mid g} h_\omega\left(\frac{-n}{t^2}\right) = 24H(-n), & \text{if } n \equiv 3 \pmod 8. \end{cases}$$

□

Corollary 3.2.4. [Kro60] *Define* $E(n) = F(n) - F_1(n)$ *for* $n \not\equiv 0$ *(mod 4),* $E(4n) = E(n)$ *for* $n \neq 0$*, and E(0) = 1/12; allow* $F(n)$ *and* $F_1(n)$ *to count the classes* $[(1, 2 \cdot 0, 1)]$ *and* $[(2, 2 \cdot 1, 2)]$*, and classes derived from them, as* $1/2$ *and* $1/3$*, respectively. Then*

$$\vartheta_3^3(q) = 12 \sum_{n=0}^{\infty} E(n) q^n. \tag{3.2.4}$$

Proof. This corollary follows immediately from Theorems 3.1.2 and 3.2.2. □

In Theorem 3.1.2, we established the equality $\vartheta_3^3(q) = \sum_{n=0}^{\infty} r_3(n) q^n$. It now follows from Theorem 3.2.3 that by cubing $\vartheta_{3,N}(q)$ we can tabulate all the class numbers for fundamental discriminants $\Delta < 0$, satisfying $\Delta \not\equiv 1 \pmod 8$ and $|\Delta| < N$. Unfortunately, using the formula (3.2.4) would be quite inefficient, as its computation involves a lot of overhead. The reason is that we are interested only in fundamental discriminants, which correspond to coefficients of the form $16k + 4$, $16k + 8$, $8k + 3$ and $8k + 7$ for some non-negative integer k. Therefore, out of N first coefficients of $\vartheta_{3,N}^3(q)$, there are only 0.375N of *potential* interest. Not all of them correspond to a fundamental discriminant, and in fact, we expect to have around $(3/\pi^2)N \approx 0.30396N$ fundamental discriminants, satisfying $|\Delta| < N$ [Coh93, Section 5.10]. In the end, we are interested in less than a third of all the coefficients that get computed. We address this problem in the following section.

3.3 Class Number Tabulation Formulas

As mentioned in the previous section, we do not want to use the formula (3.2.4) for class number tabulation purposes, as we are interested in less than a third of its coefficients. For-

tunately, three alternative formulas can be deduced from it, which will allow us to compute $h(-n)$ for $n = 16k + 4$, $n = 16k + 8$ and $n = 8k + 3$ separately. In order to obtain these formulas, we follow the reasoning of Bell [Bel24]. Note that the case $n = 8k + 7$ will be handled separately in Chapter 5.

Consider the following three cases:

(a) If $n \equiv 2 \pmod 4$, then for $n = x^2 + y^2 + z^2$ one of the coefficients x, y, z has to be even, and the other two must be odd. Using techniques as in the proof of Theorem 3.1.2, one can show that the n-th coefficient of $\vartheta_2^k(q^4)$ counts the number of representations of a number n as a sum of k odd squares. Also, $\vartheta_3(q^4) = 1 + 2q^4 + 2q^{16} + 2q^{36} + \ldots$ counts even squares. Therefore, the n-th coefficient of $\vartheta_2^2(q^4)\vartheta_3(q^4)$ corresponds to a number of representations of n as as a sum of two odd and one even square. We can therefore conclude that

$$12\sum_{k=0}^{\infty} E(4k+2)q^{4k+2} = 3\vartheta_2^2(q^4)\vartheta_3(q^4),$$

where $E(n)$ is defined in Corollary 3.2.4. Note that the series $\vartheta_2^2(q^4)\vartheta_3(q^4)$ on the right side, unlike the series on the left, does not take into account the position of the even term in the representation of $4k+2$ as a sum of three squares. That is, if $4n+2 = x_0^2+y_0^2+z_0^2$, and x_0 is even, then solutions

$$\begin{aligned} x = x_0, &\quad y = y_0, \quad z = z_0; \\ x = z_0, &\quad y = x_0, \quad z = y_0; \\ x = y_0, &\quad y = z_0, \quad z = x_0 \end{aligned}$$

are considered the same for the series on the right, but different for the series on the left. For that reason, we have to multiply the right side by 3 in order to equate it to the left side. Since $E(n) = F(n)$ by relation (2.2.4), we obtain

$$\sum_{k=0}^{\infty} F(4k+2)q^{4k+2} = \frac{1}{4}\vartheta_2^2(q^4)\vartheta_3(q^4),$$

which can be further simplified by substituting q with $q^{\frac{1}{4}}$, yielding

$$\sum_{k=0}^{\infty} F(4k+2)q^k = \left[\frac{1}{2}\vartheta_2(q)q^{-\frac{1}{4}}\right]^2 \vartheta_3(q) = \nabla^2(q^2)\vartheta_3(q), \tag{3.3.1}$$

where

$$\nabla(q) = \frac{1}{2}\vartheta_2(\sqrt{q})q^{-\frac{1}{8}} = \frac{1}{2}\cdot 2\sum_{k=0}^{\infty}\sqrt{q}^{\left(k+\frac{1}{2}\right)^2}\cdot q^{-\frac{1}{8}} = \sum_{k=0}^{\infty} q^{\frac{k(k+1)}{2}} = 1+q+q^3+q^6+q^{10}+\dots\ .$$

The formula (3.3.1) allows us to compute all $h(\Delta) = F(4k+2)$ for $\Delta = -4(4k+2)$. In order to tabulate values of $h(\Delta)$ up to some given positive bound N, it is sufficient to compute only the first $\lfloor N/16 \rfloor$ coefficients of the right hand side of (3.3.1).

(b) If $n \equiv 1 \pmod 4$, then for $n = x^2+y^2+z^2$ one of the coefficients x, y, z has to be odd, and the other two must be even. The formula

$$4\sum_{k=0}^{\infty} F(4k+1)q^{4k+1} = \vartheta_2(q^4)\vartheta_3^2(q^4)$$

can be obtained by reasoning similar to (a). If we now substitute q with $q^{\frac{1}{4}}$, our final relation will take the form

$$2\sum_{k=0}^{\infty} F(4k+1)q^k = \left[\frac{1}{2}\vartheta_2(q)q^{-\frac{1}{4}}\right]\ \vartheta_3^2(q) = \nabla(q^2)\vartheta_3^2(q). \tag{3.3.2}$$

With the formula above we can tabulate $h(\Delta) = F(4k+1)$ for $\Delta = -4(4k+1)$. For a given positive bound N it is sufficient to compute the first $\lfloor N/16 \rfloor$ elements of the right hand side of (3.3.2) in order to obtain all $h(\Delta)$ for $|\Delta| < N$.

(c) If $n \equiv 3 \pmod 8$, then every representation $n = x^2+y^2+z^2$ must have odd x, y, z, since the square of any odd number is $\equiv 1 \pmod 8$. Therefore, $12E(n)$ in the formula (3.2.4) counts the number of representations of a number n as a sum of three odd squares, which means that

$$12\sum_{k=0}^{\infty} E(8k+3)q^{8k+3} = \vartheta_2^3(q^4).$$

Since $E(n) = F(n) - F_1(n)$, and $F_1(n) = F(n)/3$ by the relation (2.2.4), we obtain Kronecker's formula:

$$\sum_{k=0}^{\infty} F(8k+3)q^{8k+3} = \frac{1}{8}\vartheta_2^3(q^4).$$

This formula gives us no advantage over (3.2.4), but we can simplify it by replacing q with $q^{\frac{1}{8}}$, resulting in

$$\sum_{k=0}^{\infty} F(8k+3)q^k = \left[\frac{1}{2}\vartheta_2(\sqrt{q})q^{-\frac{1}{8}}\right]^3 = \nabla^3(q). \tag{3.3.3}$$

The formula (3.3.3) is computationally more efficient than (3.2.4), since it allows us to tabulate all class numbers $h(-8k-3) = F(8k+3)/3$ for $8k+3 < N$ by computing the first $\lfloor N/8 \rfloor$ coefficients of $\nabla^3(q)$.

Formulas introduced above allow us to tabulate class numbers $h(\Delta)$ for negative Δ, satisfying $\Delta \not\equiv 1 \pmod 8$. Computationally, this can be done by converting the series on the right hand sides of (3.3.1), (3.3.2) and (3.3.3) into polynomials via truncation of each series to degree $N-1$ for some fixed integer N. In our work, we chose $N = 2^{40}$, and used the FLINT library, which implements subroutines for polynomial multiplication [Har14].

Finally, in this section we also like to note that there exist other formulas suitable for class number tabulation, though computationally inefficient in comparison to those introduced above. As an example, consider [Kro60, Formula IX]:

$$\sum_{n=0}^{\infty} F(n)q^n = \frac{q^{\frac{1}{4}}}{\vartheta_2(q)} \sum_{n=0}^{\infty} \frac{q^{n^2+3n+1}}{(1-q^{2n+1})^2}$$

and [Kro62, Formulas (1), (2)]:

$$\sum_{n=0}^{\infty} F(n)q^n = \frac{1}{2\vartheta_3(q)} \sum_{n=0}^{\infty} \frac{n}{q^n - q^{-n}} \left(q^{n^2+n} - 2 + q^{n^2-n}\right),$$

$$\sum_{n=0}^{\infty} F(n)q^n = \frac{1}{2\vartheta_4(q)} \sum_{n=0}^{\infty} n(-q)^{n^2} \frac{q^n - q^{-n}}{q^n + q^{-n}}.$$

The main advantage of these formulas is that they allow us to extract $h(\Delta)$ all at once for all congruence classes of Δ, without breaking the computation into four separate cases.

Surprisingly, the large series on the right hand sides of these equations, though needing to be handled carefully, do not cause significant difficulties. The main challenge is to compute an inverse of the k-th theta series in the denominator.

Chapter 4

OUT-OF-CORE MULTIPLICATION

In the previous chapter, we developed several formulas which allow us to produce class numbers $h(\Delta)$ by means of polynomial multiplication. However, these formulas cannot be applied directly, as the polynomials we aim to multiply are too large to fit into memory all together. In this chapter, we give a description to *out-of-core multiplication* — the computational technique for multiplying large polynomials, which we use to tabulate class numbers $h(\Delta)$ satisfying $\Delta \not\equiv 1 \pmod 8$. In Section 4.1, we give an overview to out-of-core multiplication. Section 4.2 contains pseudocode of the algorithms that we use for tabulation. Section 4.3 discusses computational parameters, such as the bound on the size of polynomial coefficients. Finally, Section 4.4 concludes the chapter with the complexity analysis of our algorithm.

4.1 Out-of-Core Multiplication with Chinese Remainder Theorem

In order to compute $h(\Delta)$ for all fundamental discriminants $\Delta < 0$, satisfying $|\Delta| < N$ and $\Delta \not\equiv 1 \pmod 8$, we aim to compute relations (3.3.1), (3.3.2) and (3.3.3) to degrees $\lfloor N/16 \rfloor$, $\lfloor N/16 \rfloor$ and $\lfloor N/8 \rfloor$, respectively. All the polynomials, as well as squares of polynomials, can be rapidly initialized, block by block, using Algorithms 4.4 – 4.9. That is why, even though each formula contains two polynomial multiplications, only a single multiplication has to be performed.

In our computations, N was chosen to be 2^{40}. If we assume that each coefficient of some polynomial $f(x)$ of degree $\lfloor N/8 \rfloor$ corresponds to `unsigned int` on 64-bit systems, i.e. it is of size 4 bytes, then it would require 512 Gb to fit $f(x)$ into memory. Hence, in order to store two polynomials, $f(x)$ and $g(x)$, as well as the resulting polynomial $h(x)$, we would need

1.5 Tb, not to mention that the Fast Fourier Transform (FFT), which we use to multiply polynomials, is a costly procedure, requiring a lot of memory for intermediate results. Such an intensive memory requirement forces us to perform polynomial multiplication *out-of-core*, i.e. with the usage of the hard disk.

One common technique for out-of-core FFT computation is to use many *Number Theoretic Transforms* (NTT) with *Chinese Remainder Theorem* (CRT) reconstitution[1]. The NTT method is a standard one for computing many digits of π [Bel10]. The idea is simple: in order to multiply two polynomials $f(x)$ and $g(x)$ with large coefficients, one chooses many primes $p_0, \ldots, p_{n-1}$, and performs reduction of coefficients of $f(x)$ and $g(x)$ modulo each p_i for $0 \leq i < n$. Depending on the amount of memory available, each p_i is chosen in such a way that the reduced polynomials can be comfortably multiplied in $\mathbb{F}_{p_i}[x]$. After that, n pairs of polynomials get multiplied (possibly in parallel) over each finite field $\mathbb{F}_{p_i}$, and as a result, each polynomial will have the coefficients of $h(x) = f(x) \times g(x)$ modulo p_i. In the end, the coefficients of $h(x)$ can be reconstructed with the Chinese Remainder Theorem, and this procedure can also be easily parallelized. Note that our intermediate results, namely the reduced polynomials and the result of the polynomial multiplications, get stored into files on the hard disk. Also, in order for this technique to work, one has to know ahead an upper bound C on the coefficients of $h(x)$, and choose primes such that $C < p_0 \cdot p_1 \cdot \ldots \cdot p_{n-1}$.

In 2010, Hart et al. used the CRT-based out-of-core FFT technique to tabulate all congruent numbers to 10^{12} [HTW10]. In their computations, many general word-sized primes were used in conjunction with Harvey's zn_poly library, designed for fast polynomial arithmetic in $\mathbb{F}_{p_i}[x]$ [Har08]. In order to construct a polynomial that can fit into memory for multiplication, *Kronecker substitution* (also commonly referred to as *Kronecker segmentation*) was used. Kronecker substitution is based on the observation that if we know ahead that each coefficient of $h(x) = f(x) \times g(x)$ would fit into s bits, then from $h(2^s) = f(2^s) \times g(2^s)$,

[1]The paper of Hart et al. contains a good survey on various out-of-core FFT methods and their applications [HTW10, Section 3].

which is merely an integer, we can extract all the coefficients of $h(x)$. This technique is demonstrated in Example 4.1.1 below. We shall further refer to the parameter s as the *bit size* parameter. As we shall see, Kronecker substitution allows us to construct polynomials of small degree with large coefficients. The NTT method can further be applied to these polynomials in order to fit them into memory.

Example 4.1.1. Let $f(x) = 1 + 3x + 5x^2 + 7x^3$ and $g(x) = 8 + 4x + x^2$. Observe that each coefficient of $h(x) = f(x) \times g(x)$ is less than 100. Then $f(100) = 7050301$, $g(100) = 10408$, and therefore

$$h(100) = f(100) \times g(100) = 7050301 \times 10408 = \underbrace{07}_{h_5}\ \underbrace{33}_{h_4}\ \underbrace{79}_{h_3}\ \underbrace{53}_{h_2}\ \underbrace{28}_{h_1}\ \underbrace{08}_{h_0} = 7 \cdot 100^5 + 33 \cdot 100^4 + \ldots,$$

where h_n for $0 \leq n \leq 5$ is the n-th coefficient of $h(x)$, i.e. $h(x) = \sum_{n=0}^{5} h_n x^n$.

In Example 4.1.1 we chose to evaluate the polynomials at 10^2 rather than, for example, at 2^7, because it is more convenient for humans to deal with base 10 representations. For a computer though, base 2 is a better option, since the data is kept in binary format, and Kronecker substitution can be performed by a sequence of logical shifts and ORs.

In their work, Hart et al. utilize a more sophisticated Kronecker substitution. Consider a polynomial

$$f(x) = f_0 + f_1 x + f_2 x^2 + \ldots + f_{N-1} x^{N-1}$$

of degree $N - 1$. Fix a *bundling parameter* B, dividing N, and let $N_0 = N/B$. Then we can find a polynomial $\hat{f}(x, y)$, satisfying $\hat{f}(x^B, x) = f(x)$, as follows:

$$\hat{f}(x, y) = \sum_{n=0}^{N_0 - 1} F_n(y) x^n = F_0(y) + F_1(y) x + F_2(y) x^2 + \ldots + F_{N_0 - 1}(y) x^{N_0 - 1},$$

where

$$F_n(y) = f_{nB} + f_{nB+1} y + \ldots + f_{(n+1)B-1} y^{B-1}. \tag{4.1.1}$$

If all the coefficients of $f(x)$ fit into s bits, we can *bundle* them by evaluating each $F_n(y)$ at

2^s, and obtain the following *bundled polynomial* $F(x)$ of degree $N_0 - 1$, where $N_0 = N/B$:

$$F(x) = \sum_{n=0}^{N_0-1} F_n\left(2^s\right) x^n. \tag{4.1.2}$$

While $f(x)$ has coefficients of size s bits and degree $N-1$, the bundled polynomial $F(x)$ has coefficients of size Bs bits and a smaller degree $N_0 - 1$. Now, in order to perform a multiplication $h(x) = f(x) \times g(x)$, one has to bundle coefficients of $g(x)$ with the same parameters B and s, and obtain a bundled polynomial $G(x)$. The polynomial $H(x) = F(x) \times G(x)$ will therefore contain information on the coefficients of $h(x)$. Note that $H(x)$ is *not* a bundled polynomial of $h(x)$, and extraction of the coefficients of $h(x)$ from $H(x)$ is not as straightforward as in the regular Kronecker substitution, as demonstrated in Example 4.1.1.

In order to understand how to perform an extraction of the coefficients of $h(x)$ from $H(x)$, let us write $F_n = F_n(2^s)$ and $G_n = G_n(2^s)$, and refer to F_n and G_n as *bundled coefficients* (or simply *coefficients*) of $F(x)$ and $G(x)$, respectively. Consider the n-th coefficient $H_n = \sum_{i=0}^{n} F_i G_{n-i}$ of $H(x)$. We first determine which coefficients of $h(x)$ are contained in H_n. Recall that

$$F_i G_{n-i} = \left(f_{iB} + f_{iB+1}2^s + \ldots + f_{iB+B-1}2^{(B-1)s}\right)\cdot\left(g_{(n-i)B} + g_{(n-i)B+1}2^s + \ldots + g_{(n-i)B+B-1}2^{(B-1)s}\right) =$$

$$= \underbrace{f_{iB}\, g_{(n-i)B}}_{FG_{nB}} + \underbrace{\left(f_{iB+1}\, g_{(n-i)B} + f_{iB}\, g_{(n-i)B+1}\right)}_{FG_{nB+1}} 2^s + \ldots + \underbrace{f_{iB+B-1}\, g_{(n-i)B+B-1}}_{FG_{nB+2B-2}} 2^{2(B-1)s}.$$

Evidently, FG_{nB+j} for $0 \le j \le 2B-2$ is one of the summands in $h_{nB+j} = \sum_{t=0}^{nB+j} f_t g_{nB+j-t}$, which is the $(nB+j)$-th coefficient of the polynomial $h(x) = f(x) \times g(x)$. We conclude that $F_i G_{n-i}$ encodes information on the coefficients from h_{nB} to $h_{nB+2B-2}$, and this result is independent of i, so we can extend it further to H_n. Independently of n, each H_n encodes information on $2B-1$ coefficients of $h(x)$ in total, and therefore the bit size of H_n is $(2B-1)s$.

Observe that H_n encodes only *partial* information on coefficients of $h(x)$. In order to recover some $h_k = \sum_{i=0}^{k} f_i g_{k-i}$, we would have $nB \le k \le nB + B - 2$ for some n, which

means that the summands of h_k are contained in both H_{n-1} and H_n. The only exceptions take place when $0 \leq k \leq B-1$ and when $k = tB-1$ for some integer $t > 1$. The first case corresponds to the coefficients $h_0, \ldots, h_{B-1}$, which can be extracted solely from H_0, while the second case corresponds to the coefficients h_{tB-1}, which can be extracted solely from H_{t-1}.

Example 4.1.2. Let us compute the first $N = 12$ terms of (3.3.3). For that purpose, define two polynomials that are equal to $\nabla(q)$ and $\nabla^2(q)$, truncated to degree $N-1$:

$$f(x) = 1 + x + x^3 + x^6 + x^{10};$$

$$g(x) = 1 + 2x + x^2 + 2x^3 + 2x^4 + 3x^6 + 2x^7 + 2x^9 + 2x^{10} + 2x^{11}.$$

We shall demonstrate our example in base 10. Suppose that we know a priori that every coefficient of $h(x) = f(x) \times g(x)$ has a single decimal digit. We set our decimal size (former bit size) parameter to $s = 1$, bundling parameter to $B = 4$, and obtain the following bundled polynomials of degree $N_0 - 1 = 2$, where $N_0 = N/B = 3$:

$$F(x) = 1011 + 100x + 100x^2;$$

$$G(x) = 2121 + 2302x + 2220x^2.$$

Consider the first $N_0 - 1$ coefficients of their product:

$$H(x) = F(x) \times G(x) = 2144331 + 2539422x + 2686720x^2 + O(x^3).$$

In order to extract the coefficients of $h(x) = f(x) \times g(x)$, we draw the following table:

Table 4.1: Extraction of coefficients from $H(x)$

	b_3	b_2	b_1	b_0	a_3	a_2	a_1	a_0
H_0	0	2	1	4	4	3	3	1
H_1	0	2	5	3	9	4	2	2
H_2	0	2	6	8	6	7	2	0

As expected, each H_n encodes partial information on exactly $2B-1=7$ coefficients. We also notice that the first $B=4$ coefficients of $h(x)$, from h_0 to h_3, can be extracted from the lower B digits of H_0, i.e.

$$h_0 = 1, \;\; h_1 = 3, \;\; h_2 = 3, \;\; h_3 = 4.$$

The other coefficients can be extracted according to the following algorithm: starting at coefficient H_t for $t=1$, we iterate from a_0 to a_{B-1}, and compute $h_{tB+i} = a_i + b_i$, where b_i is taken from the coefficient H_{t-1}:

$$h_4 = 2+4 = 6, \;\; h_5 = 2+1 = 3, \;\; h_6 = 4+2 = 6, \;\; h_7 = 9.$$

After the computation of h_{tB-1}, which according to our observations will always have $b_{B-1}=0$, we increment t, and repeat this procedure, until all N coefficients are obtained:

$$h_8 = 0+3 = 3, \;\; h_9 = 2+5 = 7, \;\; h_{10} = 7+2 = 9, \;\; h_{11} = 6.$$

We encourage the reader to check that each $h_n/3$, as expected by (3.3.3), corresponds to the Hurwitz class number $H(-8n-3)$ for $0 \leq n < N$.

4.2 Algorithms

Now we can give a description of the algorithm due to Hart et al. [HTW10], which multiplies two large polynomials $f(x)$ and $g(x)$ of degree $N-1$ in parallel, using T threads. We choose a suitable bundling parameter B which divides N, a bit size parameter s, and primes $p_0, \ldots, p_{n-1}$, such that $(2B-1)s < \log_2(p_0 \cdot \ldots \cdot p_{n-1})$ and n is a multiple of T. Our intermediate computations get stored on the hard disk into m binary files. For convenience, the number of files m is chosen so that it divides N/B. Each file contains $N/(mB)$ coefficients of $F(x)$, $G(x)$ or $H(x)$, reduced modulo the primes $p_0, \ldots, p_{n-1}$, chosen so that they fit into a single *limb* (i.e. a machine word of size 32 or 64 bits, depending on the architecture).

The original algorithm has three phases, which we consider as three separate algorithms.

(a) *Algorithm 4.1, Initialization.* For the polynomial $f(x)$, compute its bundled polynomial $F(x)$ of degree $N_0 - 1$, where $N_0 = N/B$. Reduce its coefficients (in parallel) modulo each of the primes $p_0, \ldots, p_{n-1}$ in order to obtain polynomials $F_{p_i}(x)$ in $\mathbb{F}_{p_i}[x]$. Store their coefficients to files $F_0, \ldots, F_{m-1}$. Repeat the same procedure for the polynomial $g(x)$, storing the result to files $G_0, \ldots, G_{m-1}$.

(b) *Algorithm 4.2, Multiplication.* One by one, read the polynomials $F_{p_i}(x)$ and $G_{p_i}(x)$ from files into memory. Multiply them (in parallel) to degree $N_0 - 1$ over $\mathbb{F}_{p_i}$ for $0 \leq i < n$, and store the resulting polynomials $H_{p_i}(x) = F_{p_i}(x) \times G_{p_i}(x)$ to files $H_0, \ldots, H_{m-1}$.

(c) *Algorithm 4.3, Restoration.* For i from 0 to $N_0 - 1$ (in parallel), read i-th coefficient of $H_{p_0}(x), \ldots, H_{p_{n-1}}(x)$ from file $H_{\lfloor i/m \rfloor}$, and restore the i-th coefficient of $H(x)$ with the CRT algorithm. Extract B coefficients of $h(x) = f(x) \times g(x)$ from it using the partial data from the $(i-1)$-st coefficient of $H(x)$ if $i > 0$. Store the coefficients of $h(x)$ to files $R_0, \ldots, R_{m-1}$.

We utilize Algorithms 4.1, 4.2, 4.3 for the initialization of polynomials, their multiplication and restoration, respectively. Our algorithms match those described in [HTW10, Section 4.1]. We also present the following modifications:

(a) We introduce the bit size parameter s, which corresponds to the smallest number of bits which can fit every coefficient of $h(x)$. This is done in order to generalize the algorithm of Hart et al., who essentially used $s = 16$.

(b) Algorithm 4.3 reconstitutes the coefficients of $h(x)$ in parallel.

To simplify our computations, n is chosen to be a multiple of the number of threads T. Additionally, we use Algorithms 4.4 – 4.9 to initialize a block of coefficients of a specific polynomial (see Algorithm 4.1, steps 3 – 5).

Algorithm 4.1 Initialization [HTW10, Algorithm 1 : Phase 1]

Input:

1. Polynomial $f(x)$ of degree $N-1$;
2. Bundling parameter B, which divides N;
3. Number of files m, which divides $N_0 = N/B$;
4. Partition size S, which divides N/m;
5. Bit size parameter s;
6. Number of threads T;
7. Primes $p_0, \ldots, p_{n-1}$, such that $(2B-1)s < \log_2(p_0 \cdot \ldots \cdot p_{n-1})$, and T divides n.

Output:

Files $F_0, F_1, \ldots, F_{m-1}$, containing coefficients of a bundled polynomial $F(x)$, reduced modulo primes $p_0, \ldots, p_{n-1}$.

block $\leftarrow N/m$

for $i \leftarrow 0, \ldots, m-1$ **do**
// Compute the values $f_{i\cdot\text{block}+jS}, \ldots, f_{i\cdot\text{block}+jS+S-1}$ using Algorithms 4.4 – 4.9
  for $j \leftarrow 0, \ldots, \text{block}/S - 1$ (in parallel) **do**
    $\theta_{jS} \leftarrow f_{i\cdot\text{block}+jS}$, $\theta_{jS+1} \leftarrow f_{i\cdot\text{block}+jS+1}$, $\ldots$, $\theta_{jS+S-1} \leftarrow f_{i\cdot\text{block}+jS+S-1}$
  end for

  for $j \leftarrow 0, \ldots, \text{block}/B - 1$ (in parallel) **do**
  // bundle coefficients
    $c_j \leftarrow \theta_{jB} + \theta_{jB+1}2^s + \theta_{jB+2}2^{2s} + \ldots + \theta_{(j+1)B-1}2^{(B-1)s}$

  // reduce c_j modulo primes
    $r_{0,j} \leftarrow c_j \pmod{p_{n-1}}$, $r_{1,j} \leftarrow c_j \pmod{p_{n-2}}$, $\ldots$, $r_{n-1,j} \leftarrow c_j \pmod{p_0}$
  end for

  // write to file
  for $k \leftarrow 0, \ldots, n-1$ (in parallel) **do**
    write $r_{k,0}, \ldots, r_{k,\text{block}/B-1}$ to F_i
  end for
end for

Algorithm 4.2 Multiplication [HTW10, Algorithm 1 : Phase 2]

Input:

1. Number of threads T;
2. Primes $p_0, \ldots, p_{n-1}$, such that T divides n;
3. Two polynomials $F(x)$ and $G(x)$ of degree $N_0 - 1$, whose coefficients are reduced modulo primes $p_0, \ldots, p_{n-1}$ and saved to files $F_0, \ldots, F_{m-1}$ and $G_0, \ldots, G_{m-1}$, respectively, where m divides N_0;

Output:

Files $H_0, \ldots, H_{m-1}$, which contain coefficients of $H(x) = F(x) \times G(x)$, truncated to degree $N_0 - 1$ and reduced modulo primes $p_0, \ldots, p_{n-1}$.

block $\leftarrow N_0/m$

for $i \leftarrow 0, \ldots, n/T - 1$ **do**
// read coefficients of $F_{p_t}(x)$ for $iT \leq t < (i+1)T$
  for $j \leftarrow 0, \ldots, m-1$ (in parallel) **do**
    for $t \leftarrow 0, \ldots, T-1$ **do**
      initialize coefficients $j \cdot \text{block}, \ldots, (j+1) \cdot \text{block} - 1$ of $f_t(x)$
      by reading them from a file F_j starting from limb $T \cdot \text{block} \cdot (n/T - 1 - i) + t \cdot \text{block}$
      truncate F_j to $T \cdot \text{block} \cdot (n/T - 1 - i)$ limbs
    end for
  end for

// read coefficients of $G_{p_t}(x)$ for $iT \leq t < (i+1)T$
  for $j \leftarrow 0, \ldots, m-1$ (in parallel) **do**
    for $t \leftarrow 0, \ldots, T-1$ **do**
      initialize coefficients $j \cdot \text{block}, \ldots, (j+1) \cdot \text{block} - 1$ of $g_t(x)$
      by reading them from a file G_j starting from limb $T \cdot \text{block} \cdot (n/T - 1 - i) + t \cdot \text{block}$
      truncate G_j to $T \cdot \text{block} \cdot (n/T - 1 - i)$ limbs
    end for
  end for

// compute $H_{p_t}(x) = F_{p_t}(x) \times G_{p_t}(x)$
  for $t \leftarrow 0, \ldots, T-1$ (in parallel) **do**
    $h_t(x) \leftarrow f_t(x) \times g_t(x) \pmod{p_{t+iT}}$
  end for

// store coefficients of $H_{p_t}(x)$ to m files
  for $j \leftarrow 0, \ldots, m-1$ (in parallel) **do**
    for $t \leftarrow 0, \ldots, T-1$ **do**
      append coefficients $j \cdot \text{block}, \ldots, (j+1) \cdot \text{block} - 1$ of $h_t(x)$ to file H_j
    end for
  end for
end for

Algorithm 4.3 Restoration [HTW10, Algorithm 1 : Phase 3]

Input:

1. Number of threads T;
2. Primes $p_0, \ldots, p_{n-1}$, such that T divides n;
3. Residues of a polynomial $H(x)$ of degree $N_0 - 1$ modulo primes $p_0, \ldots, p_{n-1}$ that are contained in files $H_0, \ldots, H_{m-1}$, where m divides N_0;
4. Bundling parameter B, which divides N_0/m;
5. Bit size parameter s.

Output:

Files $R_0, \ldots R_{m-1}$, containing the coefficients of $h(x) = f(x) \times g(x)$ extracted from $H(x)$ using Kronecker substitution.

block $\leftarrow N_0/m$

for $i \leftarrow 0, \ldots, m-1$ **do**
// reconstruct coefficients of $H(x)$ with CRT
  for $j \leftarrow 0, \ldots,$ block $- 1$ (in parallel) **do**
    $t \leftarrow$ thread id
    for $k \leftarrow 0, \ldots, n-1$ **do**
      read $c_{t,k}$ from limb $k \cdot$ block $+ j$ of file H_i
    end for
    $d_j \leftarrow CRT(c_{t,0} \pmod{p_0}, \ldots, c_{t,n-1} \pmod{p_{n-1}})$
  end for
  delete H_i

// extract coefficients of $h(x)$ with Kronecker substitution
  for $t \leftarrow 0, \ldots, T-1$ (in parallel) **do**
    size $\leftarrow$ block$/T$
    if $t = T - 1$ **then**
      size $\leftarrow$ size + (block mod T)
    end if

    for $j \leftarrow 0, \ldots,$ size $- 1$ **do**
      if $t > 0$ or $j > 0$ **then**
        if $j = 0$ **then**
          $M \leftarrow d_{t\lfloor \text{block}/T \rfloor - 1}$
          Extract b_0 and b_1 from M, such that $M = b_0 + b_1 2^{Bs}$
        end if
      else
        $b_0 \leftarrow 0, b_1 \leftarrow 0$
      end if

Algorithm 4.3 Restoration (continued)

25: $M \leftarrow d_{t\lfloor \text{block}/T \rfloor + j}$
26: Extract a_0 and a_1 from M, such that $M = a_0 + a_1 2^{Bs}$
27: $T = a_0 + b_1$
28: Extract B coefficients from T, i.e. $r_{t,0} + r_{t,1}2^s + \ldots + r_{t,B-1}2^{(B-1)s}$
29: $b_0 \leftarrow a_0$
30: $b_1 \leftarrow a_1$
31: **end for**
32: **end for**

// write to file
33: **for** $t \leftarrow 0, \ldots, T-1$ (in parallel) **do**
34: write $r_{t,0}, \ldots, r_{t,B-1}$ to R_i
35: **end for**
36: **end for**

Algorithm 4.4 Initialization of a block of $\vartheta_3(q)$

Input:

1. Index of a smallest coefficient M;
2. Number of coefficients to initialize S.

Output:

Numbers $r_0, \ldots, r_{S-1}$ that are equal to coefficients from M to $(M + S - 1)$ of $\vartheta_3(q)$.

1: **if** $M \neq 0$ **then**
2: $i \leftarrow \lceil \sqrt{M} \rceil$
3: **else**
4: $r_0 \leftarrow 1$
5: $i \leftarrow 1$
6: **end if**
7: $t \leftarrow i^2 - M$
8: **while** $t < S$ **do**
9: $r_t \leftarrow 2$
10: $t \leftarrow t + 2i + 1$
11: $i \leftarrow i + 1$
12: **end while**

Algorithm 4.5 Initialization of a block of $\nabla(q)$

Input:

1. Index of a smallest coefficient M;
2. Number of coefficients to initialize S.

Output:

Numbers $r_0, \ldots, r_{S-1}$ that are equal to coefficients from M to $(M + S - 1)$ of $\nabla(q)$.

1: $i \leftarrow \lceil(\sqrt{1 + 8M} + 1)/2\rceil$
2: $t \leftarrow i(i-1)/2 - M$
3: **while** $t < S$ **do**
4: $\quad r_t \leftarrow 1$
5: $\quad t \leftarrow t + i$
6: $\quad i \leftarrow i + 1$
7: **end while**

Algorithm 4.6 Initialization of a block of $\nabla(q^2)$

Input:

1. Index of a smallest coefficient M;
2. Number of coefficients to initialize S.

Output:

Numbers $r_0, \ldots, r_{S-1}$ that are equal to coefficients from M to $(M + S - 1)$ of $\nabla(q^2)$.

1: $i \leftarrow \lceil(\sqrt{1 + 4M} + 1)/2\rceil$
2: $t \leftarrow i(i-1) - M$
3: **while** $t < S$ **do**
4: $\quad r_t \leftarrow 1$
5: $\quad t \leftarrow t + 2i$
6: $\quad i \leftarrow i + 1$
7: **end while**

Algorithm 4.7 Initialization of a block of $\vartheta_3^2(q)$

Input:

1. Index of a smallest coefficient M;
2. Number of coefficients to initialize S.

Output:

Numbers $r_0, \ldots, r_{S-1}$ that are equal to coefficients from M to $(M + S - 1)$ of $\vartheta_3^2(q)$.

$x \leftarrow 0$
$i \leftarrow 0$
while $x < M + S$ **do**
  $j \leftarrow (x < M)\ ?\ \lfloor\sqrt{M - x}\rfloor : 0$
  $y \leftarrow j^2$
  if $x + y < M$ **then**
    $y \leftarrow 2j + 1$
    $j \leftarrow j + 1$
  end if
  $t \leftarrow x + y - M$
  while $t < S$ **do**
    $r_t \leftarrow r_t + (x > 0\ ?\ 2 : 1) \cdot (y > 0\ ?\ 2 : 1)$
    $y \leftarrow y + 2j + 1$
    $t \leftarrow t + 2j + 1$
    $j \leftarrow j + 1$
  end while
  $x \leftarrow x + 2i + 1$
  $i \leftarrow i + 1$
end while

Algorithm 4.8 Initialization of a block of $\nabla^2(q)$

Input:

1. Index of a smallest coefficient M;
2. Number of coefficients to initialize S.

Output:

Numbers $r_0, \ldots, r_{S-1}$ that are equal to coefficients from M to $(M + S - 1)$ of $\nabla^2(q)$.

1: $x \leftarrow 0$
2: $i \leftarrow 1$
3: **while** $x < M + S$ **do**
4: $\quad j \leftarrow (M > x)\ ?\ \left\lfloor (\sqrt{1 + 8M} + 1)/2 \right\rfloor : 1$
5: $\quad y \leftarrow j(j-1)/2$
6: $\quad$ **if** $x + y < M$ **then**
7: $\quad\quad y \leftarrow y + j$
8: $\quad\quad j \leftarrow j + 1$
9: $\quad$ **end if**
10: $\quad t \leftarrow x + y - M$
11: $\quad$ **while** $t < S$ **do**
12: $\quad\quad r_t \leftarrow r_t + 1$
13: $\quad\quad y \leftarrow y + j$
14: $\quad\quad t \leftarrow t + j$
15: $\quad\quad j \leftarrow j + 1$
16: $\quad$ **end while**
17: $\quad x \leftarrow x + i$
18: $\quad i \leftarrow i + 1$
19: **end while**

Algorithm 4.9 Initialization of a block of $\nabla^2(q^2)$

Input:

1. Index of a smallest coefficient M;
2. Number of coefficients to initialize S.

Output:

Numbers $r_0, \ldots, r_{S-1}$ that are equal to coefficients from M to $(M + S - 1)$ of $\nabla^2(q^2)$.

$x \leftarrow 0$
$i \leftarrow 1$
while $x < M + S$ **do**
  $j \leftarrow (M > x)\ ?\ \left\lfloor (\sqrt{1 + 4M} + 1)/2 \right\rfloor : 1$
  $y \leftarrow j(j - 1)$
  if $x + y < M$ **then**
    $y \leftarrow y + 2j$
    $j \leftarrow j + 1$
  end if
  $t \leftarrow x + y - M$
  while $t < S$ **do**
    $r_t \leftarrow r_t + 1$
    $y \leftarrow y + 2j$
    $t \leftarrow t + 2j$
    $j \leftarrow j + 1$
  end while
  $x \leftarrow x + 2i$
  $i \leftarrow i + 1$
end while

4.3 Computational Parameters

As mentioned in the previous section, in order to perform an out-of-core multiplication we need an upper bound on coefficients of (3.3.1), (3.3.2) and (3.3.3). Recall the definition of $h_\omega(\Delta)$ from (2.1.4), and consider the analytic class number formula:

$$h_\omega(\Delta) = \frac{1}{\pi}\sqrt{|\Delta|}L(1, \chi_\Delta), \tag{4.3.1}$$

where $L(s, \chi_\Delta)$ denotes the Dirichlet L-function

$$L(s, \chi_\Delta) = \sum_{n=1}^{\infty} \frac{\chi_\Delta(n)}{n^s}$$

and $\chi_\Delta(n)$ is a Dirichlet character, corresponding to a Kronecker symbol $\left(\frac{\Delta}{n}\right)$. Over the past decades, many papers have been written on the problem[2] of finding explicit bounds on $|L(1,\chi)|$. For our computations, we utilize the unconditional bound established by Ramaré [Ram01]:

$$L(1,\chi_\Delta) \leq a \log|\Delta| + b, \tag{4.3.2}$$

where

$$a = \tfrac{1}{4}, b = \tfrac{5}{4} - \tfrac{\log 3}{2}, \quad \text{if } \Delta \equiv 0 \pmod 4;$$
$$a = \tfrac{1}{2}, b = \tfrac{5}{2} - \log 6, \quad \text{if } \Delta \equiv 1 \pmod 4.$$

We need to extend this result and determine an upper bound on the Hurwitz class number $H(\Delta)$, as it occurs in our class number tabulation formula (3.2.3).

Fix some discriminant $\Delta = g^2 e$, where e is a fundamental discriminant. Then by (4.3.1) and Definition 2.1.9,

$$H(\Delta) = \sum_{t \,|\, g} h_\omega\left(\frac{\Delta}{t^2}\right) \leq \frac{1}{\pi}\sum_{t \,|\, g}\sqrt{\frac{|\Delta|}{t^2}}\left(a\log\frac{|\Delta|}{t^2} + b\right) = \frac{\sqrt{|\Delta|}}{\pi}\left[(a\log|\Delta| + b)\sum_{t \,|\, g}\frac{1}{t} - 2a\sum_{t \,|\, g}\frac{\log t}{t}\right].$$

We obtain

$$H(\Delta) < \frac{1}{\pi}\sqrt{|\Delta|}(a\log|\Delta| + b)\sum_{t \,|\, g}\frac{1}{t}.$$

Now we need to estimate the sum $\sum_{t \,|\, g}\frac{1}{t}$. Recall that we would like to tabulate class numbers for all $|\Delta| < N$. For $N = 2^{40}$, we picked the largest possible $g = 605395$, and found the integer $n = 554400$ that does not exceed g, and has the largest value of $\sum_{t \,|\, n}\frac{1}{t} = \frac{1209}{275}$. If we define

$$C_N = \left\lfloor \frac{1209}{275}\cdot\frac{1}{\pi}\sqrt{N}\,(a\log N + b)\right\rfloor, \text{ where } N \leq 2^{40}, \tag{4.3.3}$$

and a and b are as in (4.3.2), then $H(\Delta) < C_N$ for any $|\Delta| < N$.

Now we can explain how to compute the bit size parameter s, which was introduced in Section 4.1. First, note that (3.3.2) requires $2C_N$ as the upper bound due the factor of 2, which appears on the left hand side. Second, observe that the formula (3.3.3), which allows

[2]See the survey in [JW09, Section 9.5].

to tabulate $h(-n)$ for $n \equiv 3 \pmod 8$, has $F(8k+3)$ on its left hand side. According to the relations (2.2.2) and (2.2.4), for such $n > 3$ we have $h(-n) = F_1(n) = F(n)/3$, which means that the required upper bound is $3C_N$. We conclude that our main class number tabulation formulas (3.3.1), (3.3.2) and (3.3.3) require C_N, $2C_N$ and $3C_N$ as their upper bounds, respectively. Considering this, the formula for s takes the form

$$s = \begin{cases} \lceil \log_2 C_N \rceil, & \text{for (3.3.1);} \\ \lceil \log_2(2C_N) \rceil, & \text{for (3.3.2);} \quad \text{where } N \leq 2^{40}. \\ \lceil \log_2(3C_N) \rceil, & \text{for (3.3.3),} \end{cases} \tag{4.3.4}$$

Finally, we need to determine how many primes to choose with respect to some bundling parameter B, in order to restore the coefficients of $h(x) = f(x) \times g(x)$, which should all fit into s bits. Recall from Section 4.1 that each coefficient of $H(x)$ has bit size $(2B-1)s$. In order to restore the coefficients of $H(x)$ with the CRT algorithm, we need to pick our primes $p_0, \ldots, p_{n-1}$ so that $(2B-1)s < \log_2(p_0 \cdot \ldots \cdot p_{n-1})$. We follow the implementation of Hart et al. by choosing the smallest prime p_0 exceeding some positive lower bound P, and $n-1$ primes $p_1, \ldots, p_{n-1}$, which consecutively follow after p_0. We choose n so that

$$(2B-1)s \leq n \log_2 p_0 < \sum_{i=0}^{n-1} \log_2 p_i,$$

i.e.

$$n = \left\lceil \frac{(2B-1)s}{\log_2 p_0} \right\rceil. \tag{4.3.5}$$

Note that for large p_0 and small n, the difference between $n \log_2 p_0$ and $\sum_{i=0}^{n-1} \log_2 p_i$ becomes negligible. Following Hart et al., we chose p_0 to be the smallest prime exceeding $P = 2^{62}$, which fits into a single machine word on a 64-bit system.

4.4 Complexity Analysis

In this section, we estimate the asymptotic running time of our algorithm. We would like not only to see how our program behaves relative to the length of the polynomial N, but

also understand how the bundling parameter B affects its scalability. We begin by studying the asymptotic complexity of Algorithms 4.4 – 4.9.

Theorem 4.4.1. *Consider the polynomial $f(q) = \vartheta_3(q)$, or $\nabla(q)$, or $\nabla(q^2)$. For M and/or S approaching infinity, Algorithms 4.4 – 4.6 compute coefficients $f_M, f_{M+1}, \ldots, f_{M+S-1}$ in $O(\sqrt{M+S})$ bit operations.*

Proof. Consider Algorithm 4.4, which initializes a block of coefficients of $\vartheta_3(q)$. It consists of a single loop, and its number of iterations is equal to the number of full squares on the interval from M to $M+S$, which is $\lfloor\sqrt{M+S} - \sqrt{S}\rfloor$. We conclude that Algorithm 4.4 requires $O(\sqrt{M+S})$ bit operations. Analogous results hold for Algorithms 4.5 and 4.6. □

Theorem 4.4.2. *Consider the polynomial $f(q) = \vartheta_3^2(q)$, or $\nabla^2(q)$, or $\nabla^2(q^2)$. For M and/or S approaching infinity, Algorithms 4.7 – 4.9 compute coefficients $f_M, f_{M+1}, \ldots, f_{M+S-1}$ in $O(M+S)$ bit operations.*

Proof. Consider Algorithm 4.7, which initializes a block of coefficients of $\vartheta_3^2(q)$. The problem of initialization reduces to counting all possible two squares representations of an integer n which lies in the interval from M to $M+S$. Our algorithm consists of two nested loops. The outer loop iterates over all possible full squares x which do not exceed $M+S$, and therefore has $\lfloor\sqrt{M+S}\rfloor$ iterations in total. The inner loop iterates over all integers n in our interval such that $n-x$ is a full square. We conclude that the inner loop requires at most $\lfloor\sqrt{M+S} - \sqrt{S}\rfloor$ iterations, which means that the overall complexity of Algorithm 4.7 is $O(\sqrt{M+S}(\sqrt{M+S} - \sqrt{S}))$, or simply $O(M+S)$. Analogous results hold for Algorithms 4.8 and 4.9. □

Note that each of our class number tabulation formulas (3.3.1), (3.3.2) and (3.3.3) involve *two* polynomial multiplications. For example, in order to determine (3.3.3), we first have to perform the multiplication $\nabla^2(q) = \nabla(q) \times \nabla(q)$, followed by the computation of $\nabla^3(q) = \nabla^2(q) \times \nabla(q)$. In practice, we use the different approach; that is, we utilize Algorithms 4.7 –

4.9 in order to initialize $\vartheta_3^2(q)$ (or $\nabla^2(q)$, or $\nabla^2(q^2)$, depending on the class number tabulation formula) *directly*, which allows us to perform one polynomial multiplication instead of two. We argue that for small N this approach is more beneficial in practice and does not result in a significant overhead. However, asymptotically, when used to initialize N/S blocks of coefficients of the partition size S, Algorithms 4.7 — 4.9 require $O(S)+O(2S)+\ldots+O(N) = O(N^2/S)$ bit operations, which is worse than the utilization of two sequential polynomial multiplications. For that reason, in Theorem 4.4.3 we limit our attention only to Algorithms 4.4 – 4.6.

Now let us turn our attention to Algorithms 4.1, 4.2 and 4.3, which we utilize to multiply two large polynomials.

Theorem 4.4.3. *Consider two polynomials, $f(x)$ and $g(x)$, both of degree $N-1$. The sequential algorithm which combines Algorithms 4.1, 4.2 and 4.3 requires*

$$O\left(\frac{N\sqrt{N}}{S} + Ns\left(\log(Bs)\right)^{2+\varepsilon} + Ns\left(\log\frac{N}{B}\right)^{1+\varepsilon}\right) \tag{4.4.1}$$

bit operations, where S is the partition size, B is the bundling parameter, s is the bit size parameter, and the positive number ε incorporates smaller $\log\log$ *factors.*

Proof. We begin by studying Algorithm 4.1, which essentially consists of four phases: a) initialization; b) bundling; c) multimodular reduction; and d) saving to disk. We assume that the disk I/O operations, i.e. saving to disk and reading from a file, require $O(N)$ operations and therefore do not affect the overall estimate of the complexity of our algorithm.

Consider phase a) of Algorithm 4.1. In order to initialize the coefficients of a polynomial $f(x)$ of degree $N-1$, we pick a partition size S which (for convenience) divides N, and apply one of Algorithms 4.4 – 4.6 to each of the N/S partitions. According to Theorem 4.4.1, it takes $O(\sqrt{S})$ to initialize a block $f_0,\ldots,f_{S-1}$, $O(\sqrt{2S})$ for a block $f_S,\ldots,f_{2S-1}$, and so on, until we reach the $(N/S-1)$-st block, which can be initialized in $O(\sqrt{N})$. The total is the

sum over all blocks, yielding

$$O(\sqrt{S}) + O(\sqrt{2S}) + \ldots + O(\sqrt{N}) = O\left(\sqrt{S}\left(1 + \sqrt{2} + \ldots + \sqrt{\frac{N}{S}}\right)\right) = O\left(\frac{N\sqrt{N}}{S}\right).$$

The bundling of coefficients in phase b) consists of sequential applications of logical shifts and ORs, and requires $O(N)$ bit operations. If we ignore the subdivision of our coefficients to files, the multimodular reduction phase c) gets performed within a single loop in steps 6 – 9 of Algorithm 4.1. This loop has N/B iterations. On each iteration, we utilize a *remainder tree* to reduce the bundled coefficient C modulo $p_0, \ldots, p_{n-1}$ [Ber04]. This technique allows us to deduce $C \pmod{p_0}$, $C \pmod{p_1}$, $\ldots$, $C \pmod{p_{n-1}}$ in time $O\left(t\left(\log t\right)^{2+\varepsilon}\right)$, where t is the total number of bits in $C, p_0, \ldots, p_{n-1}$. We conclude that the time required for the multimodular reduction phase is

$$O\left(\frac{N}{B} \cdot t\left(\log t\right)^{2+\varepsilon}\right) = O\left(\frac{N}{B} \cdot Bs\left(\log(Bs)\right)^{2+\varepsilon}\right) = O\left(Ns\left(\log(Bs)\right)^{2+\varepsilon}\right).$$

It is easy to estimate the running time of Algorithm 4.2, as it performs n multiplications of polynomials of degree $N/B - 1$. To multiply polynomials, we utilize the Schönhage-Strassen algorithm [GG03, Sections 8.2 – 8.4], which requires $O(N \log N \log\log N)$ bit operations to multiply two polynomials of degree N. This means that the asymptotic running time of Algorithm 4.2 is

$$O\left(n\frac{N}{B}\log\frac{N}{B}\log\log\frac{N}{B}\right) = O\left(Bs\frac{N}{B}\log\frac{N}{B}\log\log\frac{N}{B}\right) = O\left(Ns\left(\log\frac{N}{B}\right)^{1+\varepsilon}\right).$$

Finally, consider Algorithm 4.3, which consists of two phases: the CRT reconstitution phase and the extraction of coefficients phase. Though the latter involves certain sophisticated techniques discussed in Section 4.1, it simply iterates over all N coefficients of our resulting polynomial, and therefore requires $O(N)$ bit operations. Now, let us consider the CRT reconstitution phase, which occurs in steps 3 – 9 of Algorithm 4.3. For the CRT, we utilize the divide-and-conquer technique [HTW10, Section 4]. For n_1 integral coefficients of size n_2 bits, this approach allows us to complete the restoration of a coefficient in

$O\left(n_2 \left(\log(n_1 n_2)\right)^{2+\varepsilon}\right)$. In our case, n_2 is constant and $n_1 = n$, where n is the number of primes in use. In total, there are N/B coefficients to restore, which means that the running time is

$$O\left(\frac{N}{B}(\log n)^{2+\varepsilon}\right) = O\left(\frac{N}{B}\left(\log(Bs)\right)^{2+\varepsilon}\right).$$

Since $s > \log^{-1} B$, the asymptotic running time of the initialization phase absorbs the running time for the CRT reconstitution phase. If we now combine running times for initialization and multiplication phases, we shall get precisely the expression (4.4.1). □

Note that if we would use Algorithms 4.7 – 4.9 instead of Algorithms 4.4 – 4.6, the $O(N\sqrt{N}/S)$ term in (4.4.1) would be replaced by $O(N^2/S)$. Hence, in order to initialize N/S blocks of S coefficients in time $O(N \log N)$, one would have to choose $S = N/\log N$, which is unreasonable in practice.

Considering the observations made in Section 4.3, we estimate the asymptotic running time of the class number tabulation program in Corollary 4.4.4.

Corollary 4.4.4. *The class number tabulation algorithm which combines Algorithms 4.1, 4.2 and 4.3 requires*

$$O\left(N \log N (\log B)^{2+\varepsilon} + N \log N \left(\log \frac{N}{B}\right)^{1+\varepsilon}\right) \tag{4.4.2}$$

bit operations, where the positive number ε incorporates smaller $\log\log$ *factors.*

Proof. Consider Theorem 4.4.3. When $S = \sqrt{N}$, the $O(N\sqrt{N}/S) = O(N)$ term in (4.4.1) gets absorbed by two other terms. According to Section 4.3, we may substitute s in (4.4.1) with $\log(\sqrt{N} \log N)$, and obtain (4.4.2). □

Now we can draw certain conclusions. First of all, when B is considered to be a constant, the asymptotic running time of our program is $O(N \log N)$. The variation of B influences its performance. As B goes to 1, our program performs like a regular polynomial multiplication algorithm, spending little time on the initialization and CRT reconstitution phases, and

more time on the polynomial multiplication phase. On the other hand, when B increases, the number of primes grows proportionally, forcing the initialization and the CRT reconstitution phases to take more time.

In Corollary 4.4.5, we estimate the asymptotic running time of our algorithm with respect to the number of threads T.

Corollary 4.4.5. *When T threads are available, the parallel algorithm which combines Algorithms 4.1, 4.2 and 4.3 requires*

$$O\left(\frac{N\sqrt{N}}{TS} + \frac{N}{T}s\left(\log(Bs)\right)^{2+\varepsilon} + \frac{N}{T}s\left(\log\frac{N}{B}\right)^{1+\varepsilon}\right) \tag{4.4.3}$$

bit operations per thread, where the positive number ε incorporates smaller $\log\log$ *factors.*

Proof. This result follows immediately from Theorem 4.4.3 and the fact that Algorithms 4.1, 4.2, 4.3 are "embarrassingly parallel", i.e. every thread performs its computations independently from the others in the most straightforward manner possible. Such a parallelization allows us to achieve a theoretical asymptotic time complexity proportional to $1/T$ at every stage, including initialization, multimodular reduction, polynomial multiplication and CRT reconstitution. □

Though our asymptotic complexity is proportional to $1/T$, there are several factors that degrade the linear speedup. First of all, as the number of threads increases, the overhead of using them grows as well. Second, depending on the disk I/O capabilities of a computing node it might not be possible to read from or write to disks in parallel, as it is done in Algorithm 4.2. Also, in steps 3 – 9 of Algorithm 4.3, multiple threads read off coefficients from a single file, and these coefficients are spread uniformly across the file. When files get accessed with a memory map, such a positioning of coefficients inside the file increases the number of page loads, which also degrades the performance. Finally, in Algorithms 4.1 and 4.3, we write in parallel to a *single* file. In practice, these writing requests are *scheduled* at the machine level, and get performed sequentially.

Chapter 5

CLASS GROUP COMPUTATION

In this chapter, we turn our attention to the class group computation. Recall that equivalence classes of quadratic forms with discriminant Δ form a group, called the *class group*, which denoted by Cl_Δ. The class number $h(\Delta)$ is the cardinality of Cl_Δ, but it is also interesting to look at its *structure*. By the structure of the class group, we mean its decomposition as a direct product of cyclic subgroups, i.e.

$$Cl_\Delta \cong C(x_0) \times \ldots \times C(x_{r-1}) \tag{5.0.1}$$

for some integer $r > 0$, where $m_{j+1} \,|\, m_j$ for $0 \leq j < r-1$, and $C(x)$ denotes the cyclic group of order x [JW09, Section 7.1]. In order to compute Cl_Δ, we utilize a modified version of the Buchmann-Jacobson-Teske algorithm (BJT) [BJT97, Algorithm 4.1], developed by Ramachandran [Ram06, Algorithm 3.1]. This algorithm iteratively builds up a set of generators $\boldsymbol{\alpha}$ of Cl_Δ, and terminates whenever the size of the subgroup generated by $\boldsymbol{\alpha}$ exceeds a certain precomputed lower bound. We give a more detailed description of the BJT algorithm and introduce its pseudocode in Section 5.2.

In order for the BJT algorithm to work correctly, we require the size of the group $h(\Delta)$, or a good lower bound on $h(\Delta)$. Techniques of the previous chapter allow us to tabulate all $h(\Delta)$, except for $\Delta \equiv 1 \pmod 8$. In Section 5.1, we discuss Bach's *conditional* approach for the computation of the lower bound h^*, which satisfies the relation $h^* \leq h(\Delta) \leq 2h^*$. Due to the nature of BJT, knowing h^* is sufficient for us to be certain that the whole group structure was computed. However, the usage of Bach's method for $\Delta \equiv 1 \pmod 8$ makes our results correct under the ERH in that case. In order to make our results unconditional, we use the verification algorithm from [Ram06, Algorithm 4.1], based on the Eichler-Selberg trace formula [SvdV91]. We discuss this formula in the final section.

5.1 Bach's Bound on the Size of the Group

As it was pointed out before, the formula (3.2.4) along with its derivatives (3.3.1), (3.3.2) and (3.3.3), allows us to tabulate class numbers $h(\Delta)$ for every fundamental discriminant, except for $\Delta \equiv 1 \pmod 8$. Since the BJT algorithm requires a lower bound on $h(\Delta)$, we need to be able to generate it for every such Δ. To do this, we utilize Bach's averaging method, which allows to find a better approximation of $L(1, \chi_\Delta)$ by computing a weighted average of multiple (as opposed to one) truncated Euler products (6.3.2) [Bac95]. With Bach's method, we produce an approximation L_{approx} of $L(1, \chi_\Delta)$, such that $|\log L(1, \chi_\Delta) - \log L_{approx}| < \log \sqrt{2}$ under the assumption of ERH. The value

$$h^* = \frac{\sqrt{|\Delta|}}{\pi\sqrt{2}} L_{approx}$$

satisfies the relation $h^* \leq h(\Delta) \leq 2h^*$. We use h^* as the stopping condition for the BJT algorithm; that is, our algorithm terminates whenever the size of the currently generated subgroup exceeds h^*.

Algorithm 5.1 describes Bach's averaging method, and has $E = \log L_{approx}$ as its output. It is implemented in the Algebraic Number Theory Library (ANTL), maintained at the University of Calgary by Jacobson. Note that the algorithm asks us to provide the parameter Q. Jacobson's library contains the following table for the values of Q, which was computed using Bach's error estimates [Bac95]:

Table 5.1: Suggested values of Q for Bach's averaging method

$\log_{10} \lvert\Delta\rvert$	Q	$\log_{10} \lvert\Delta\rvert$	Q
< 5	947	< 25	8447
< 10	2269	< 30	11093
< 15	3929	< 35	11093
< 20	6011	< 40	14149

Algorithm 5.1 Bach's approximation of $\log L(1, \chi_\Delta)$ [Bac95]

Input:

1. Discriminant Δ;
2. Parameter Q;
3. The list of all primes up to $2Q$.

Output:

Number E such that $|\log L(1, \chi_\Delta) - E| < \log \sqrt{2}$.

$C \leftarrow \sum_{i=Q}^{2Q-1} i \log i$
$E \leftarrow 0, i \leftarrow 0, \omega \leftarrow 1$
while $p_i < Q$ **do**
  $E \leftarrow \log \left(\frac{p_i}{p_i - \left(\frac{\Delta}{p_i}\right)} \right)$
  $i \leftarrow i + 1$
end while

for $j \leftarrow Q, \ldots, p_i$ **do**
  $\omega \leftarrow \omega - \frac{j \log j}{C}$
end for

while $p_i < 2Q$ **do**
  $E \leftarrow E + \omega \log \left(\frac{p_i}{p_i - \left(\frac{\Delta}{p_i}\right)} \right)$
  $\omega \leftarrow \omega - \frac{((p_i - 1)\log(p_i - 1) + p_i \log p_i)}{C}$
  $i \leftarrow i + 1$
end while

return E

5.2 The Class Group Computation

Every group structure computation algorithm relies on the *order computation problem*, which can be formulated as follows: given a finite abelian group G and an element $\alpha \in G$, one has to find the smallest positive integer x such that $\alpha^x = 1_G$.

Given an upper bound L on x, Shanks's famous baby-step giant-step algorithm (BSGS) allows to compute x in $O(\sqrt{L})$ group operations [Sha71]. The idea behind the algorithm is simple: if we set $q = \lceil \sqrt{L} \rceil$, we precompute the table $\{1_G, \alpha^{-1}, \alpha^{-2}, \ldots, \alpha^{-(q-1)}\}$. We refer to elements of this table as *baby steps*. After that, we iteratively compute *giant steps*

$\alpha^q, \alpha^{2q}, \alpha^{3q}, \ldots$, and check whether the current giant step is contained in the table of baby steps. If it so happens that for some a and r the equality $\alpha^{aq} = \alpha^{-r}$ holds, then we know that $x = aq + r$. Because of the way we have chosen our q, x can be determined by computing at most q giant steps.

Since Shanks's publication of his algorithm in 1971, many scientists elaborated on its improvement (see, for example, the paper of Terr [Ter99]). The main advantage of the BJT algorithm is that its computational complexity is relative to an actual order x of an element α, not to its upper bound L. In order to achieve this, the BJT algorithm starts with a small table, which doubles in size with each iteration along with the giant step width [BJT97, Algorithm 3.1]. The underlying idea of this algorithm can be utilized to determine G.

In order to compute the structure of the generic group G (which in our case corresponds to Cl_Δ), we utilize a modified version of the BJT algorithm, developed by Ramachandran [Ram06, Algorithm 3.1], which is based on the original algorithm of Buchmann et al. [BJT97, Algorithm 4.1]. This algorithm takes as an input a function `next()`, which produces generators of G. The only requirement that we impose on `next()` is that eventually it has to produce a set of elements generating the whole group. For example, our implementation of `next()` simply iterates over all quadratic forms in Cl_Δ with the first coefficient being a prime. In order to ensure that the whole group is generated, Ramachandran's algorithm also requires the size of the group, $|G|$, or a lower bound h^* on the size of the group such that $h^* \leq |G| \leq 2h^*$.

In each iteration of our algorithm, we consider a subgroup $\langle \boldsymbol{\alpha} \rangle$ of G, generated by elements $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$, and a *potential generator* β. Our goal is to find whether β is outside of $\langle \boldsymbol{\alpha} \rangle$, the subgroup of G generated by the elements of $\boldsymbol{\alpha}$. If this is the case, then $\langle \boldsymbol{\alpha}, \beta \rangle$ is a larger subgroup than $\langle \boldsymbol{\alpha} \rangle$, so we expand $\boldsymbol{\alpha}$ by β:

$$\boldsymbol{\alpha} := \boldsymbol{\alpha} \circ \beta = (\alpha_1, \ldots, \alpha_n, \alpha_{n+1}), \text{ where } \alpha_{n+1} := \beta.$$

On the other hand, if β is an element of $\langle \boldsymbol{\alpha} \rangle$, we simply ignore it, and proceed to the next

generator. Each iteration which produces a non-trivial β allows us to write down an integer vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, satisfying $\boldsymbol{\alpha}^{\boldsymbol{x}} = \alpha_1^{x_1} \cdot \ldots \cdot \alpha_n^{x_n} = 1_G$, which we call a *relation*. All the relations found during our computations are stored in a *relation matrix* $M = \{m_{i,j}\}$. The values of the n-th relation $\boldsymbol{x}$ correspond to the n-th row $\boldsymbol{m}_n$ in M. The BJT algorithm ensures that M is in *Hermite Normal Form* (HNF); that is,

(a) $m_{i,i} > 0$ for all i;

(b) $m_{i,j} = 0$ if $i > j$;

(c) $0 \leq m_{i,j} < m_{i,i}$ if $i < j$.

Note that in order to terminate the algorithm, we have to keep track of the size of $\langle \boldsymbol{\alpha} \rangle$. Since M is in HNF, we can compute the size of the current subgroup by simply multiplying the diagonal entries of M. For $\Delta \not\equiv 1 \pmod 8$, our algorithm terminates when the product of the diagonal entries matches $h(\Delta)$, produced by our class number tabulation program. For $\Delta \equiv 1 \pmod 8$, our algorithm terminates when this product exceeds a conditional lower bound h^*, computed with Bach's averaging method. When the BJT algorithm terminates, we are left with the relation matrix M. By diagonalizing M over $\mathbb{Z}$, we can obtain the decomposition of G into the product of its cyclic subgroups. This process is referred to as the *Smith Normal Form* (SNF) computation. In order to compute the SNF of M, we utilize the algorithm of Hafner and McCurley [HMcC91] [Coh93, Algorithm 2.4.14].

If $|\boldsymbol{\alpha}|$ is the number of generators of G, and $|S|$ is the total number of generators considered, then the BJT group structure computation algorithm uses $O(|S| 2^{\frac{|\boldsymbol{\alpha}|}{2}} \sqrt{|G|})$ group operations in G [BJT97, Theorem 4.3]. It uses three tables: R', R and Q to store baby steps from the previous iteration, new baby steps, and some of the giant steps from the previous iteration, respectively. The usage of an indexed hash table allows to merge R and R' in a single table.

Our overview of the BJT group structure computation algorithm would not be complete if we did not mention its disadvantages. First of all, it is exponential relative to the bit

size of $|G|$. We could have used McCurley's index calculus algorithm instead, which has a subexponential complexity [McC89] [HMcC89] [Coh93, Section 5.5]. Hafner-McCurley's algorithm is much more effective for large $|\Delta|$, and the BJT algorithm proved its effectiveness for the tabulation purposes when we compute group structures for many small discriminants [JRW06]. One other disadvantage is that for groups of the form $G \cong (\mathbb{Z}/2\mathbb{Z})^l$ with $l > 0$ its complexity becomes $\Omega(|G|)$, which is even worse than the BSGS algorithm. The Buchmann-Schmidt (BS) algorithm was developed in order to eliminate this problem [BS05]. However, Ramachandran's estimates show that for the tabulation of groups with small discriminants, BJT outperforms BS [Ram06, Section 3.4]. In turn, BS is much more effective than BJT when it comes to groups with large $|\boldsymbol{\alpha}|$.

Finally, we would like to mention that the tabulation of class numbers for $\Delta \not\equiv 1 \pmod 8$ has another major advantage, aside from the fact that we were able to produce the size of each Cl_Δ unconditionally. If we consider the factorization of $h(\Delta) = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_k^{e_k}$, then we can ignore those primes p_i for $1 \leq i \leq k$ which have $e_i = 1$, as it means that the p_i-group of Cl_Δ is guaranteed to be non-cyclic. We can therefore resolve the structure of a smaller subgroup G of Cl_Δ, satisfying

$$|G| = \prod_{p_i^2 | h(\Delta)} p_i^{e_i}.$$

This approach is somewhat similar to Buell's, who tabulated the class numbers in order to be able to resolve the structure of each p-group individually [Bue99]. Instead, we consider all the potentially non-cyclic p-groups together. In this case, our generator producing function `next()` iterates over quadratic forms $(a, b, c)^{\frac{h(\Delta)}{|G|}} \in G$ with $(a, b, c) \in Cl_\Delta$ and a being prime.

The approach described above gives us a significant speedup in computations: those class groups which correspond to $\Delta \not\equiv 1 \pmod 8$ were computed over 4.72 times faster than those which correspond to $\Delta \equiv 1 \pmod 8$[1]. However, this approach does not come for free, as we are no longer able to test Bach's bound on the size of prime generators, required to produce

[1]We consider the time required for class number tabulation to be a part of computation of Cl_Δ with $\Delta \not\equiv 1 \pmod 8$, and the time required for the verification as a part of computation of Cl_Δ for $\Delta \equiv 1 \pmod 8$.

the whole group Cl_Δ [Bac90]. We call an element $[(a, b, c)]$ of Cl_Δ prime, if its reduced representative (a, b, c) has a prime first coefficient a. In [Bac90], Bach stated that under the ERH all prime elements of Cl_Δ with $a \leq 6 \log^2 |\Delta|$ have to generate the whole group. During our computations, we ignore the generators of those p-groups of Cl_Δ which have order 1, so we are no longer able to verify whether our generators match Bach's condition. For that reason, we did not study the Bach's bound in this thesis.

Algorithm 5.2 BJT algorithm [Ram06, Algorithm 3.1]

Input:

1. Function `next()`, which produces generators of the finite abelian group G;
2. Size of the group $|G|$ or a lower bound h^*, such that $h^* \leq |G| \leq 2h^*$.

Output:

Rank r and integers $x_0, \ldots, x_{r-1}$, such that $x_{r-1} > 1$, $x_{j+1} \,|\, x_j$ for $0 \leq j < r-1$, and $G \cong C(x_0) \times C(x_1) \times \ldots \times C(x_{r-1})$.

1: $R \leftarrow \{(1_G, \emptyset)\}$, $Q \leftarrow \{(1_G, \emptyset)\}$
2: $h \leftarrow 1$, $j \leftarrow 0$

// add generators until the entire group is generated
3: **while** $h < h^*$ **do**

// initializiations
4: $\beta \leftarrow$ `next()`
5: $s \leftarrow 1$, $y \leftarrow \omega$, $u \leftarrow \omega$
6: $\alpha \leftarrow 1_G$, $\gamma \leftarrow \beta^\omega$, $\sigma \leftarrow \gamma$

// check if current generator is contained in current subgroup
7: **for all** $(\delta, \boldsymbol{x}) \in Q$ **do**
8: **if** R' contains a pair $(\delta \cdot \beta, \boldsymbol{y})$ and $x_i + y_i < m_{i,i}, 1 \leq i < j$ **then**
9: **continue while**
10: **end if**
11: **end for**

12: **while** $m_{j,j} = 0$ **do**
13: **if** $j > 0$ **then**
14: expand all vectors in R' and Q to length $j-1$ by padding them with zeros
15: **end if**

// compute new baby steps
16: **for** $i \leftarrow s, \ldots, u$ **do**
17: $\alpha \leftarrow \alpha \cdot \beta^{-1}$
18: **if** $s = 1$ and $i > 1$ **then**
19: **for all** $(\delta, \boldsymbol{x}) \in R'$ **do**
20: **if** Q constains a pair $(\delta \cdot \alpha, \boldsymbol{y})$ and $x_i + y_i < m_{i,i}, 1 \leq i < j$ **then**
21: $\boldsymbol{m}_j \leftarrow (\boldsymbol{x} + \boldsymbol{y}) \circ i$
22: **break while**

BJT Algorithm [Ram06, Algorithm 3.1] (continued)

```
23:              else
24:                 R ← R ∪ {δ · α, x ∘ i}
25:              end if
26:            end for
27:          else
28:            R ← R ∪ {(δ · α, x ∘ i) : (δ, x) ∈ R'}
29:          end if
30:        end for

   // compute giant steps
31:        while m_{j,j} = 0 and y < u^2 do
32:          for all (δ, x) ∈ Q do
33:            if R contains a pair (δ · γ, y) and x_i + y_i < m_{i,i}, 1 ≤ i < j then
34:              m_j ← (x + y) ∘ y
35:              break while
36:            end if
37:          end for
38:          y ← y + u
39:          γ ← γ · σ
40:        end while

   // double step width
41:        s ← u + 1, u ← 2u
42:        σ ← σ^2
43:      end while

   // update R', R and Q
44:      q ← ⌈√m_{j,j}⌉
45:      R' ← R \ {(δ, x) : (δ, x) ∈ R, x_j ≥ q}
46:      R ← {(1_G, ∅)}
47:      Q ← Q ∪ {(δ · β^{aq}, x ∘ aq) : (δ, x) ∈ Q, 1 ≤ a < q, aq < m_{j,j}}
48:      h ← h · m_{j,j}
49:      j ← j + 1
50:    end while

   // compute the Smith Normal Form of a relation matrix M
51:    M ← SNF(M)
52:    r ← 0
53:    while r < j and m_{r,r} > 1 do
54:      x_r ← m_{r,r}
55:      r ← r + 1
56:    end while

57:    return r and x_0, ..., x_{r-1}
```

5.3 Unconditional Verification

In Section 5.1, we discussed how to compute a lower bound h^* on the size of the class group using Bach's averaging method. We utilize h^* in order to tabulate Cl_Δ for $\Delta \equiv 1 \pmod 8$. However, the correctness of the output from this approach is ERH dependent, and we would like to eliminate this dependency. We chose to use the unconditional verification technique, based on the Eichler-Selberg trace formula, which was previously used by Jacobson et al. for similar purposes [JRW06]. This formula arises in the theory of modular forms, and it relates the trace of a certain Hecke operator to a sum of Hurwitz class numbers [SvdV91]:

$$H(-4n) + 2\sum_{t=1}^{\lfloor\sqrt{4n}\rfloor} H(t^2 - 4n) = 2\left(\sum_{\substack{d \mid n \\ d \geq \sqrt{n}}} d\right) - \chi(n)\sqrt{n} + \frac{1}{6}\chi(n), \tag{5.3.1}$$

where $\chi(n)$ is the indicator function, which is 1 whenever n is a perfect square, and 0 otherwise [JRW06, Formula 2.2]. Due to the nature of the BJT algorithm, the size of the class group produced by this algorithm will always divide $h(\Delta)$. Therefore, in case if one or more $h(\Delta)$ are larger than what we expect and $H(\Delta)$ occurs in the formula above, the left hand side would have to be smaller than the right hand side.

We point out that there exist other formulas discovered by Kronecker that can be potentially used for the verification procedure, i.e. [Kro60, Formula I] and [Kro60, Formula IX]:

$$F(4n) + 2F(4n - 1^2) + 2F(4n - 2^2) + 2F(4n - 3^2) + \ldots = 2X(n) + \Phi(n) + \Psi(n);$$

$$F(n) + F(n-2) + F(n-6) + F(n-12) + F(n-20) + \ldots = \frac{1}{8}\Psi(4n+1),$$

where $X(n)$ denotes the sum of odd divisors of n, $\Phi(n)$ denotes the sum of all divisors of n, and $\Psi(n)$ denotes the difference between the sum of divisors $> \sqrt{n}$ and the sum of divisors $\leq \sqrt{n}$. Though potentially these formulas can be used for the verification, we were unable to apply them in our computations. The reason is that we use the BJT algorithm to compute

$F(8n+7)$ for all $8n+7 < 2^{40}$. However, the values $F(4n+1), F(4n+2)$ are known only up to 2^{38}, which means that we are missing all $F(4n+1)$ and $F(4n+2)$ on the interval from 2^{38} to 2^{40}.

We can now turn our attention to the computation of (5.3.1). First of all, recall that we would like to verify our results for $\Delta \equiv 1 \pmod 8$. It is therefore sufficient for us to consider the formula (5.3.1) only for *even* values of n. Second, we compute the left and right hand sides separately. If we write $\Delta = g^2 e$, where e is the fundamental discriminant, then the following formula allows us to compute $H(\Delta)$ [JRW06, Section 2.2]:

$$H(\Delta) = h_\omega(e)K(\Delta), \text{ where } K(\Delta) = \sum_{t \mid g} t \prod_{\substack{q \mid t \\ \text{q is prime}}} \left(1 - \frac{\left(\frac{e}{q}\right)}{q}\right). \tag{5.3.2}$$

Now we have to determine how many times to use (5.3.1) in order to cover all the fundamental discriminants $\Delta \equiv 1 \pmod 8$ for $|\Delta| < N$. One approach would be to figure out the smallest set of n. The brute force approach to determining such n was applied by Jacobson et al. [JRW06, Section 2]. However, later Ramachandran suggested a more straight forward and much more efficient approach by simply summing left and right hand sides X times. For our purposes, we use $X = \lfloor N/8 \rfloor$, where $N = 2^{40}$. Following Ramachandran [Ram06, Formulas 4.10, 4.11], we define two integers, LHS and RHS, as follows:

$$LHS = \left(\sum_{\substack{\Delta \equiv 0 \pmod 8 \\ |\Delta| \leq 8X}} H(\Delta) \right) + 2 \left(\sum_{\substack{\Delta \equiv 0,1 \pmod 4 \\ |\Delta| \leq 8X}} r(\Delta, X) H(\Delta) \right); \tag{5.3.3}$$

$$RHS = \sum_{n=1}^{X} \left(2 \left(\sum_{\substack{d \mid 2n \\ d \geq \sqrt{2n}}} d \right) - \chi(2n)\sqrt{2n} + \frac{1}{6}\chi(2n) \right). \tag{5.3.4}$$

Here, $r(\Delta, X)$ denotes the function which indicates the number of solutions to the equation

$\Delta = t^2 - 8n$ for $1 \leq n \leq X$:

$$r(\Delta, X) = \begin{cases} 0, & \text{if } \Delta \equiv 5 \pmod 8; \\ \lfloor \frac{Y+1}{2} \rfloor, & \text{if } \Delta \equiv 1 \pmod 8; \\ \lfloor \frac{Y+2}{4} \rfloor, & \text{if } \Delta \equiv 4 \pmod 8; \\ \lfloor \frac{Y}{4} \rfloor, & \text{if } \Delta \equiv 0 \pmod 8, \end{cases} \tag{5.3.5}$$

where $Y = \lfloor \sqrt{8X + \Delta} \rfloor$.

Though computationally more intensive, the calculation of the RHS is more straightforward and easily parallelizable. In order to compute the divisors for each $n \leq X$, we use the formula (5.3.4) in conjunction with a segmented sieve [CM05]. For large X, a segmented sieve allows to determine prime factors of each $2 \leq n \leq X$ by breaking X integers into X/S blocks of the partition size S, chosen so that S integers can comfortably fit into memory. The factorization then happens block-by-block, and primes found while processing previous blocks are used to factor integers, occurring in the next block.

Algorithm 5.3 Verification algorithm [Ram06, Algorithm 4.1]

Input:

1. Integer X;
2. Class numbers $h_\omega(\Delta)$ for every fundamental discriminant $\Delta < 0$ such that $|\Delta| \leq 8X$.

Output:

LHS, i.e. the left hand side of (5.3.3)

$LHS \leftarrow 0$
for each Δ **do**
  $r \leftarrow r(\Delta, X)$, see formula (5.3.5)
  $h \leftarrow h_\omega(\Delta)$
  $LHS \leftarrow LHS + 2rh$

  // account for $H(-8n)$
  if $\Delta \equiv 0 \pmod 8$ **then**
    $LHS \leftarrow LHS + h$
  end if

  for $i \leftarrow 2, \ldots, \lfloor 8X/\Delta \rfloor$ **do**

  // compute new discriminant
    $\Delta_{next} \leftarrow i^2\Delta$
    $r \leftarrow r(\Delta_{next}, X)$, see formula (5.3.5)

  // compute K with formula (5.3.2)
    $K \leftarrow \sum_{t \mid i} t \prod_{\substack{q \mid t \\ q \text{ is prime}}} \left(1 - \frac{\left(\frac{\Delta}{q}\right)}{q}\right)$
    $LHS \leftarrow LHS + 2rhK$

  // account for $H(-8n)$
    if $\Delta_{next} \equiv 0 \pmod 8$ **then**
      $LHS \leftarrow LHS + hK$
    end if
  end for
end for

return LHS

Chapter 6

IMPLEMENTATION AND NUMERICAL RESULTS

In this chapter, we present numerical data. In Sections 6.1 and 6.2, we discuss the implementation of our program and its performance [Mos14]. In Section 6.3, we study the bounds of Littlewood and Shanks [Lit28, Sha73], as well as the conjecture of Granville and Soundararajan [GS03], and list successive maxima and minima of functions associated to $L(1, \chi_\Delta)$ for various congruence classes of Δ modulo 8. In Section 6.4, we study the Cohen-Lenstra heuristics [CL83, CL84]. Section 6.5 concludes the chapter with the study of Euler's conjecture on idoneal numbers [Kan11], and refined tables of exotic groups [Bue99].

6.1 Implementation

For the class number tabulation, we used the FLINT library for number theory, maintained by Hart [Har14]. In particular, we used `nmod_poly_mullow` routine for the polynomial multiplication in $\mathbb{F}_{p_i}[x]$. This routine utilizes the Schönhage-Strassen FFT algorithm [GG03, Sections 8.2 – 8.4].

The FLINT library also contains subroutines `fmpz_multi_mod_ui` and `fmpz_multi_CRT_ui` for the fast reduction modulo primes $p_0, \ldots, p_{n-1}$ and CRT reconstitution, respectively. We utilize OpenMP for parallelization.

For the class group tabulation, we utilize Sayles libraries `optarith` and `qform`, which contain fast implementations of binary quadratic form arithmetic [Say13a, Say13b]. There exist other libraries which implement algorithms on quadratic forms, such as Algebraic Number Theory Library (ANTL) maintained at the University of Calgary by Jacobson, and PARI/GP [Par14]. We compare their performance for tabulation purposes in the following section. We also use Message Passing Interface (MPI) for parallelization.

6.2 Performance

Our computations were performed on WestGrid's supercomputer Hungabee, located at the University of Alberta, Canada [Wes14]. Hungabee is a 16 Tb shared memory system with 2048 Intel Xeon cores, 2.67 GHz each. Each user of Hungabee may request at most 8 Gb of memory per core. Also, Hungabee provides a high performance storage space with two SGI IS5500 disk arrays, which constitute a 53 Tb file system, allowing to write to hard disks in parallel. Note that the last disk I/O requirement is essential for the high performance of our program.

We first discuss our class number tabulation program. We performed three polynomial multiplications, described in formulas (3.3.1), (3.3.2) and (3.3.3). After running several tests, we determined that Hungabee can comfortably multiply polynomials of 2^{25} coefficients, without the request for additional memory. This observation allowed us to make the proper choice of a bundling parameter B. The following table contains the list of parameters which we used for our computations, and the amount of disk space required for the multiplication to store our intermediate computations.

Table 6.1: Computational parameters

Formula	Δ	N	B	C	s	n	Disk space required
$\nabla^2(q^2)\cdot\vartheta_3(q)$	8 (mod 16)	2^{36}	2^{11}	11199314	24	1586	859 Gb
$\vartheta_3^2(q)\cdot\nabla(q^2)$	12 (mod 16)	2^{36}	2^{11}	11199314	25	1652	893.4 Gb
$\nabla^2(q)\cdot\nabla(q)$	5 (mod 8)	2^{37}	2^{12}	21381515	26	3435	1855 Gb

Here, C is the bound on $H(\Delta)$, defined in (4.3.3), s is the bit size parameter (4.3.4), and n is the number of 63 bit primes, required for correct CRT reconstitution (4.3.5).

For each multiplication, we requested 64 processors and 8Gb of memory per core. The number of files was chosen to be $m = 2^{12}$. The following table lists timings for each of the three polynomial multiplications. We also outline timings for each of the phases, i.e. initialization, multiplication and CRT reconstitution.

By multiplying $\nabla^2(q^2)$ and $\vartheta_3(q)$, we also tested the performance and scalability of our

Table 6.2: Timings for the class number tabulation program

$f(x)$	$g(x)$	CPU time	Real time	Init. $f(x)$	Init. $g(x)$	Multiplication	Reconstitution
$\nabla^2(q^2)$	$\vartheta_3(q)$	23d 11h 10m 56s	8h 47m 59s	4740s	1279s	7723s	17937s
$\vartheta_3^2(q)$	$\nabla(q^2)$	29d 21h 2m 56s	11h 12m 14s	5529s	1394s	7782s	25629s
$\nabla^2(q)$	$\nabla(q)$	68d 5h 8m 16s	25h 34m 49s	24675s	3983s	16429s	47002s

program for various degrees N, bundling parameters B, and number of threads T. The following three figures demonstrate our results.

Figure 6.1: Program performance when N varies, $B = 2^{11}$, $T = 2^6$, $m = 2^{12}$

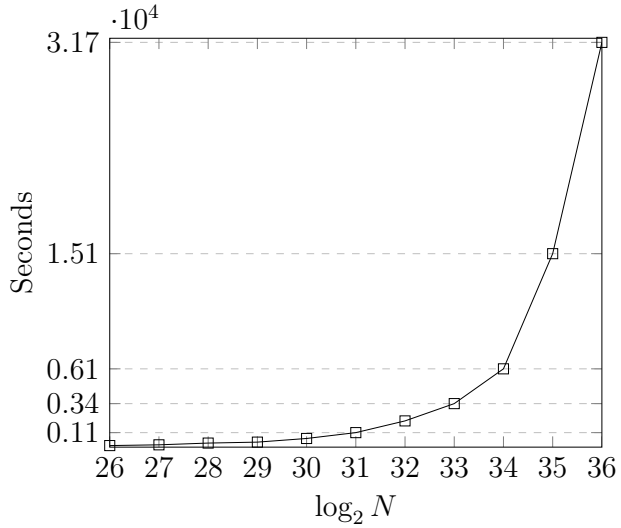


Figure 6.2: Program performance when B varies, $N = 2^{32}$, $T = 2^6$, $m = 2^{12}$

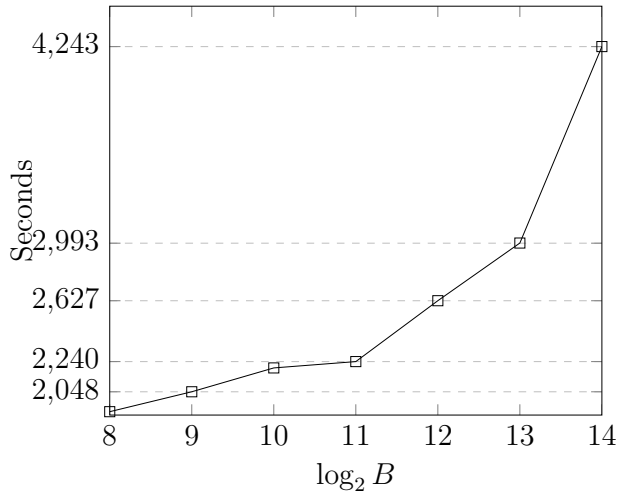
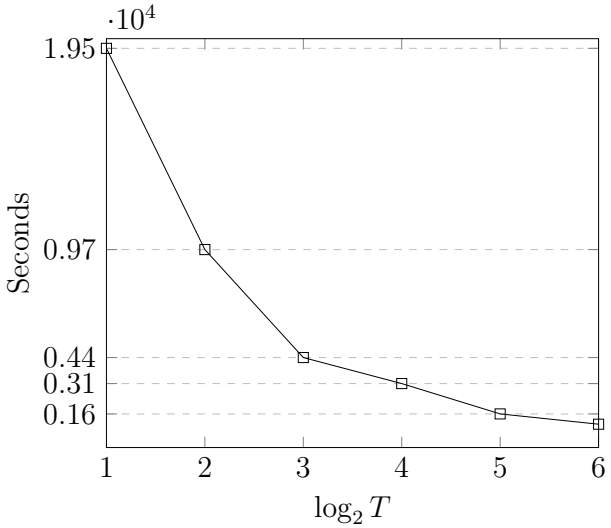


Figure 6.3: Program performance when T varies, $N = 2^{32}$, $B = 2^8$, $m = 2^8$



In Figure 6.1, we plot the scalability of our program with respect to parameter N. The data supports our observations made in Section 4.4 that the runtime scales somewhat like $O(N \log N)$.

Figure 6.2 plots the scalability of our program with respect to parameter B. According to Section 4.4, when N and T are fixed, the program scales as $O\left((\log B)^{2+\varepsilon}\right)$. When B grows exponentially, we expect our graph to take the form of a parabola. We can observe that its shape is somewhere in between linear and quadratic.

Finally, in Figure 6.3 we can see that as the number of threads increases, the class numbers get computed faster. We argue that the speedup decreases due to the series of factors, discussed in Section 4.4.

Now we turn our attention to the performance of the class group tabulation program. We outline timings for processing each congruence class of Δ in Table 6.3.

As expected, for $\Delta \equiv 1 \pmod 8$ our program takes significantly more time, since Ramachandran's approach requires the resolution of the whole group. If we assume that all Δ were handled using solely Ramachandran's technique, then 64 processors would complete the

Table 6.3: Timings for the class group tabulation program

Δ	CPU time	Real time	Number of processors
8 (mod 16)	83d 5h 4m 22s	1d 7h 12m 15s	64
12 (mod 16)	76d 3h 29m 5s	1d 4h 33m 16s	64
5 (mod 8)	105d 19h 58m 13s	1d 15h 41m 13s	64
1 (mod 8)	1657d 22h 12m 6s	39h 28m 27s	1008

(conditional) tabulation to 2^{40} in 80d 11h 9m 48s[1], as opposed to 31d 22h 45m 8s (counting the class number tabulation and the verification). Note that 81.13% of time in our computations got spent on the resolution of Cl_Δ for $\Delta \equiv 1 \pmod 8$ and the verification of the result.

We observed that the structures of Cl_Δ for all congruence classes with $\Delta \not\equiv 1 \pmod 8$ got computed over 6.25 times faster than those with $\Delta \equiv 1 \pmod 8$[2]. Such a significant speedup occurs due to the fact that class numbers $h(\Delta)$ often have non-square prime factors. In Table 6.4, for $|\Delta| < 2^{40}$ we list a) the total number of $h(\Delta)$ with non-square factors; b) the total number of $h(\Delta) = g^2 e$ with a large squarefree factor $e > \sqrt{h(\Delta)}$; and c) the total number of squarefree $h(\Delta)$.

Table 6.4: Counts of $h(\Delta)$ with various divisibility properties

Δ	Total Δ	$p \mid h$, $p^2 \nmid h$	$h = g^2 e$, $e > \sqrt{h}$	squarefree h
8 (mod 16)	55701909754	47077629143	32012088117	941347842
12 (mod 16)	55701909855	47091713960	31927265003	915383075
5 (mod 8)	111403819688	95517292502	63828635213	8828052571
1 (mod 8)	111403819373	94502061670	55851403024	7295483368

As was mentioned in Chapter 5, class numbers for $\Delta \equiv 1 \pmod 8$ got computed along with the class group structure. We present counts for this congruence class simply for completeness of the table. A close examination of other congruence classes allowed us to conclude that 85.13% of class numbers contained non-square factors, so Cl_Δ got resolved

[1]This estimate is obtained by multiplying the CPU time for $\Delta \equiv 1 \pmod 8$ by 3, dividing the result by 64, and adding it to the three times the verification program's real time, listed in Table 6.6.

[2]Not counting the verification as a part of computation of Cl_Δ with $\Delta \equiv 1 \pmod 8$, and class number tabulation as a part of computation of Cl_Δ with $\Delta \not\equiv 1 \pmod 8$. Otherwise the former Cl_Δ got computed around 4.72 times faster.

faster in this case. In 57.34% of the cases $h(\Delta)$ had a squarefree factor exceeding $\sqrt{h(\Delta)}$, which means that the size of the subgroup that we had to resolve was small relative to the size of the group itself. Finally, in 1.67% of the cases $h(\Delta)$ were squarefree; for these Δ, no class group resolution was needed at all.

We also compared the performance of our program to the ANTL based implementation of Ramanchandran [Ram06], and the `quadclassunit0` routine of the PARI/GP library [Par14], which utilizes McCurley's index calculus algorithm [McC89] [HMcC89]. For each implementation, we used a single Intel Xeon 2.27 GHz processor to compute Cl_Δ for every fundamental $\Delta < 0$ such that $|\Delta|$ lies in the interval from 2^{39} to $2^{39} + 2^{20}$. For this comparison, we did not utilize precomputed class numbers, and determined the lower bond h^* on $h(\Delta)$ instead using the Bach's averaging method, described in Section 5.1. Of course, in this case, our implementation and the program of Ramachandran utilize exactly the same algorithm. The following table demonstrates our timings:

Table 6.5: Timings for various class group tabulation implementations

$\lvert\Delta_{min}\rvert$	$\lvert\Delta_{max}\rvert$	Total Δ	Our program	ANTL	PARI/GP
2^{39}	$2^{39} + 2^{20}$	318729	438s	736s	1181s

All three implementations provide correct results under the assumption of the ERH. Though our program and Ramachandran's ANTL based implementation utilize the same algorithm, including the hash table in use, our program outperforms ANTL by over 40%. We argue that the Sayles libraries `optarith` and `qform` which we use in our program play a crucial role here [Say13a, Say13b]. These libraries implement an optimized version of Shanks's NUCOMP algorithm [JW09, Section 5.4], and utilize assembly language in their implementations. We conclude that even without the precomputed class numbers our program is faster than Ramachandran's. The tabulation of $h(\Delta)$ allowed us to speedup the computation of class groups even further.

Finally, let us also discuss the performance of our class number verification algorithm.

For the computation of (5.3.3) and (5.3.4), we requested 64 cores on Hungabee. Table 6.6 demonstrates our program's timings:

Table 6.6: Timings for the verification program

	CPU Time	Real time
Total	58d 15h 56m 48s	21h 59m 57s
LHS	19d 4h 5m 20s	7h 11m 20s
RHS	39d 11h 51m 28s	14h 48m 37s

Figures 6.4 and 6.5 plot timings for the verification program with respect to various values of N.

Figure 6.4: Scalability of the verification algorithm, LHS

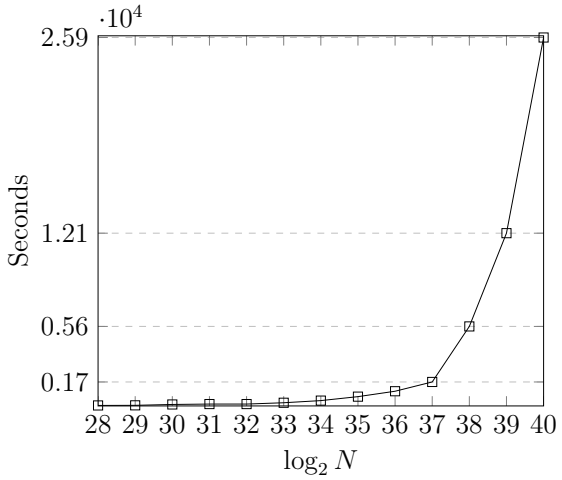
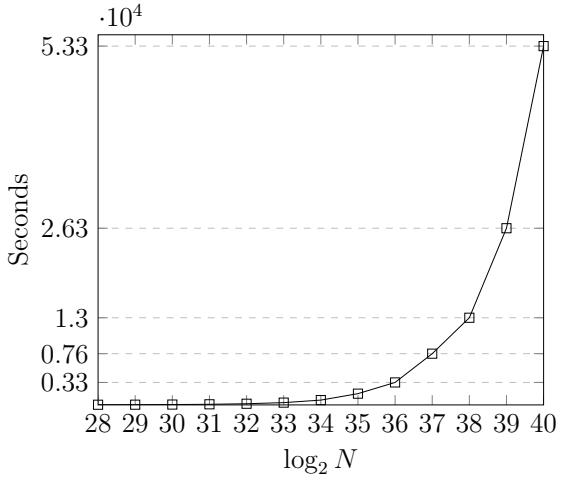


Figure 6.5: Scalability of the verification algorithm, RHS



In the following two sections, we present our numerical results, which address the study of Littlewood's Bounds and the Cohen-Lenstra heuristics.

6.3 Littlewood's Bounds

In 1928, Littlewood proved that under the assumption of the ERH the following inequalities hold:

$$(1+o(1))(c_1 \log\log|\Delta|)^{-1} < L(1,\chi_\Delta) < (1+o(1))c_2 \log\log|\Delta|, \tag{6.3.1}$$

where

$$c_1 = \frac{12e^\gamma}{\pi^2} \text{ and } c_2 = 2e^\gamma \text{ when } 2 \nmid \Delta;$$
$$c_1 = \frac{8e^\gamma}{\pi^2} \text{ and } c_2 = e^\gamma \text{ when } 2 \mid \Delta,$$

and $\gamma \approx 0.57722$ is the Euler-Mascheroni constant [Lit28]. Later, Shanks studied these bounds more carefully by defining two quantities,

$$ULI = \frac{L(1,\chi_\Delta)}{c_2 \log\log|\Delta|} \text{ and } LLI = L(1,\chi_\Delta)c_1 \log\log|\Delta|,$$

and ignoring the $o(1)$ terms in Littlewood's estimates [Sha73]. These quantities, which Shanks called the *upper* and *lower Littlewood indices*, allow us to test whether Littlewood's bounds are violated, for if the ERH does not hold, then for large $|\Delta|$ we might find $ULI > 1$ or $LLI < 1$. Note that there *are* small Δ such that $ULI > 1$ or $LLI < 1$, namely $\Delta = -3, -4, -163$. We assume that values of ULI and LLI for these discriminants are largely influenced by the $o(1)$ terms. Aside from $\Delta = -3, -4, -163$, we did not find any occurrences of discriminants which violate Littlewood's bounds.

In [Sha73], Shanks observed that the bounds given in (6.3.1) can be improved for odd Δ. Consider the Euler product representation of $L(1,\chi_\Delta)$:

$$L(1,\chi_\Delta) = \prod_{p \text{ is prime}} \left(\frac{p}{p - \left(\frac{\Delta}{p}\right)} \right) \tag{6.3.2}$$

As one may notice, when Δ is odd the factor corresponding to $p = 2$ significantly influences the value of $L(1, \chi_\Delta)$. This results in a notable difference in magnitude from those $L(1, \chi_\Delta)$ which have even Δ and $\left(\frac{\Delta}{2}\right) = 0$. Following Shanks, we also introduce the function $L_\Delta(1)$, which divides out the most significant factor of $L(1, \chi_\Delta)$:

$$L_\Delta(1) = \begin{cases} L(1, \chi_\Delta), & \text{if } \Delta \equiv 0 \pmod 4; \\ \frac{1}{2}L(1, \chi_\Delta), & \text{if } \Delta \equiv 1 \pmod 8; \\ \frac{3}{2}L(1, \chi_\Delta), & \text{if } \Delta \equiv 5 \pmod 8. \end{cases}$$

We can now write down a stronger pair of inequalities for odd discriminants that will further allow us to compare extreme values of $L(1, \chi_\Delta)$ across all Δ:

$$(1 + o(1)) \left(\frac{8e^\gamma}{\pi^2} \log\log|4\Delta|\right)^{-1} < L_\Delta(1) < (1 + o(1))e^\gamma \log\log|4\Delta|. \tag{6.3.3}$$

We tested these tighter bounds developed by Shanks by looking at the following quantities:

$$ULI_\Delta = \frac{L_\Delta(1)}{e^\gamma \log\log|4\Delta|} \text{ and } LLI_\Delta = L_\Delta(1)\frac{8e^\gamma}{\pi^2} \log\log|4\Delta|.$$

In Tables 6.9, 6.13, 6.17 we list successive ULI_Δ maxima for various congruence classes of Δ. We also list successive LLI_Δ minima in Tables 6.11, 6.15 and 6.19. Note that these tables ignore values of $\Delta = -3, -4, -163$, as we assume that they are largely influenced by the $o(1)$ terms. Of course, other values of ULI_Δ and LLI_Δ for small Δ are also very much influenced by these terms. As a result, some of our tables consist of one or two entries, like Tables 6.11, 6.17 and 6.19. However, except for $\Delta = -3, -4, -163$, the tighter bounds of Shanks were not violated, so we decided to keep these small tables as they are. We did not make any new discoveries regarding the maximal value of ULI_Δ and minimal value of LLI_Δ found. As it was previously observed by Jacobson et al. [JRW06, Section 3.1], the largest $ULI_\Delta \approx 0.85183$ corresponds to $\Delta = -8$, and the smallest[3] $LLI_\Delta \approx 1.00944$ corresponds to $\Delta = -232$.

[3]Note that there is an error in [JRW06] and [Ram06], which state that the smallest $LLI_\Delta = LLI$ corresponds to $\Delta = -1012$.

Aside from Littlewood's bounds, we also studied the growth of $L(1, \chi_\Delta)$ independently of the discriminant growth. In Tables 6.8, 6.12, 6.16 we list successive maxima of $L(1, \chi_\Delta)$ for various congruence classes of Δ. The reader can find successive minima of $L(1, \chi_\Delta)$ in Tables 6.10, 6.14 and 6.18. In Table 6.8, we indicate in bold new discriminants with largest $L(1, \chi_\Delta)$ that did not occur in Ramachandran's list [Ram06, Table 5.3]. The last discriminant found was $\Delta = -685122125399$, which corresponds to the largest $L(1, \chi_\Delta) \approx 8.47178$ with $|\Delta| < 2^{40}$. As for the smallest $L(1, \chi_\Delta)$, no new discoveries were made. We would also like to mention that the largest $L_\Delta(1) \approx 4.34082$ found corresponds to $\Delta = -921824947931$, and the smallest $L_\Delta(1) \approx 0.25346$ found corresponds to $\Delta = -569078186623$.

One other important result was developed by Chowla, who proved unconditionally that inequalities

$$(1 + o(1))e^\gamma \log\log|\Delta| < L(1, \chi_\Delta) < (1 + o(1))\frac{\pi^2}{6e^\gamma \log\log|\Delta|}$$

hold for infinitely many Δ, which implies that

$$ULI_\Delta > \frac{1}{2}(1 - \varepsilon) \text{ and } LLI_\Delta < 2(1 - \varepsilon)$$

hold for infinitely many Δ as well, where $0 < \varepsilon < 1$. Later, Granville and Soundararajan conjectured that extreme values of $L(1, \chi_\Delta)$ tend to Chowla's bounds; in other words, extreme values of ULI_Δ should approach $1/2$, and extreme values of LLI_Δ should approach 2 as $|\Delta|$ grows [GS03]. Figures 6.6 and 6.7 plot local ULI_Δ maxima and local LLI_Δ minima on each interval of size 2^{28}, respectively. The behaviour of their extreme values fully coincides with the observations of Littlewood and Shanks. The local maxima of ULI_Δ and local minima of LLI_Δ seem to be approaching some bounds that lie below 0.69 and above 1.37, respectively. We conclude that the bound 2^{40} on $|\Delta|$ is still too low to test whether $1/2$ and 2 are lower and upper limits of ULI_Δ and LLI_Δ, respectively. In Figures 6.6 and 6.7, we may also observe that both local ULI_Δ maxima and local LLI_Δ minima approach mentioned bounds slowly, presumably *logarithmically*. In Figure 6.6, we observe that for 2^{39} the local

ULI_Δ maxima approaches 0.693, and for 2^{40} it reaches 0.690. By interpolation, we obtain an estimate that the local ULI_Δ maxima should have values around 1/2 for $|\Delta| \geq 2^{100}$. In Figure 6.7, we observe that for $x = 2^{39}$ and $x = 2^{40}$ the values of local LLI_Δ minima approach 1.36 and 1.37, respectively. The interpolation yields us a similar estimate that the values of local LLI_Δ minima should approach 2 for $|\Delta| \geq 2^{100}$.

Following Buell [Bue99], we also calculated the arithmetic mean values of $L(1, \chi_\Delta)$ for various congruence classes of Δ. We list these values in Table 6.7. Note that our results fully coincide with Buell's findings. Also, observe that the mean values of $L(1, \chi_\Delta)$ are small for $\Delta \equiv 5 \pmod 8$, large for $\Delta \equiv 1 \pmod 8$, and in between the two for $\Delta \equiv 0 \pmod 4$. This result is expected according to the observation of Shanks that the term corresponding to $p = 2$ in the Euler representation (6.3.2) significantly influences the value of $L(1, \chi_\Delta)$.

Table 6.7: Mean values of $L(1, \chi_\Delta)$ for various congruence classes of Δ

Δ	Mean of $L(1, \chi_\Delta)$
8 (mod 16)	1.1863899379
12 (mod 16)	1.1863899387
5 (mod 8)	0.7909266258
1 (mod 8)	2.3727798697
Even Δ	1.1863899383
Odd Δ	1.5818532467
All Δ	1.4500321438

Table 6.8: Successive $L(1, \chi_\Delta)$ maxima, $\Delta \equiv 1 \pmod 8$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
7	1.18741	0.50072	0.59371	0.27695
15	1.62231	0.45716	0.81116	0.32309
23	1.96520	0.48276	0.98260	0.36562
39	2.01223	0.43506	1.00611	0.34884
47	2.29124	0.47713	1.14562	0.38850
71	2.60987	0.50532	1.30493	0.42315
119	2.87989	0.51684	1.43995	0.44447
191	2.95513	0.50016	1.47756	0.43827
215	2.99957	0.50095	1.49978	0.44074
239	3.04819	0.50323	1.52410	0.44427
311	3.38472	0.54377	1.69236	0.48386
479	3.58858	0.55354	1.79429	0.49808
671	3.63840	0.54529	1.81920	0.49433
959	3.65210	0.53217	1.82605	0.48579
1151	3.79661	0.54579	1.89831	0.49983
1319	3.89260	0.55416	1.94630	0.50865
1511	3.96017	0.55847	1.98008	0.51372
1559	4.05786	0.57102	2.02893	0.52552
2351	4.08192	0.55917	2.04096	0.51766
2999	4.18779	0.56516	2.09389	0.52486
3071	4.30847	0.58062	2.15424	0.53938
5711	4.53127	0.58958	2.26564	0.55158
6551	4.54132	0.58661	2.27066	0.54957
8399	4.59347	0.58583	2.29673	0.55015
10391	4.71534	0.59508	2.35767	0.55992
13439	4.71537	0.58783	2.35768	0.55432
13991	4.72765	0.58826	2.36383	0.55491
14951	4.75320	0.58963	2.37660	0.55650
15791	4.87506	0.60323	2.43753	0.56959
18191	4.96137	0.61000	2.48068	0.57662

Continued on next page

Table 6.8 — continued from previous page

$\lvert\Delta\rvert$	$L(1,\chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
31391	5.12442	0.61546	2.56221	0.58406
38639	5.19422	0.61859	2.59711	0.58783
45239	5.19919	0.61531	2.59960	0.58530
63839	5.25953	0.61426	2.62977	0.58551
88919	5.32040	0.61383	2.66020	0.58618
95471	5.41928	0.62365	2.70964	0.59578
118271	5.53584	0.63227	2.76792	0.60468
201431	5.65585	0.63447	2.82793	0.60833
331679	5.71679	0.63121	2.85839	0.60652
366791	5.81495	0.64006	2.90748	0.61528
514751	5.89819	0.64265	2.94910	0.61859
628319	5.92517	0.64184	2.96259	0.61827
701399	5.93061	0.64040	2.96530	0.61713
819719	5.98558	0.64348	2.99279	0.62045
890951	6.05086	0.64899	3.02543	0.62593
1238639	6.06898	0.64507	3.03449	0.62285
1339439	6.14018	0.65127	3.07009	0.62899
2155919	6.35035	0.66522	3.17517	0.64342
4305479	6.39987	0.65901	3.19994	0.63866
6077111	6.48918	0.66279	3.24459	0.64290
11915279	6.55831	0.65972	3.27916	0.64096
12537719	6.61614	0.66479	3.30807	0.64597
16368959	6.65069	0.66442	3.32535	0.64599
20357039	6.70670	0.66692	3.35335	0.64873
29858111	6.83310	0.67411	3.41655	0.65625
48796439	6.83371	0.66757	3.41685	0.65050
54694631	6.88505	0.67108	3.44252	0.65407
63434159	6.92096	0.67265	3.46048	0.65577
69671159	6.94604	0.67387	3.47302	0.65708
82636319	7.01518	0.67838	3.50759	0.66167

Continued on next page

Table 6.8 — continued from previous page

$\lvert\Delta\rvert$	$L(1,\chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
106105271	7.03819	0.67742	3.51909	0.66103
131486759	7.07318	0.67811	3.53659	0.66194
173540351	7.07662	0.67504	3.53831	0.65924
197317559	7.17063	0.68244	3.58532	0.66661
242230271	7.19930	0.68270	3.59965	0.66707
258363551	7.20524	0.68250	3.60262	0.66694
354544511	7.20973	0.67921	3.60486	0.66404
469058399	7.24019	0.67886	3.62010	0.66397
499769591	7.26997	0.68093	3.63499	0.66606
507138551	7.27541	0.68127	3.63771	0.66641
625378991	7.28980	0.68026	3.64490	0.66561
693126671	7.33941	0.68374	3.66970	0.66911
826771391	7.35663	0.68339	3.67831	0.66893
967603199	7.37448	0.68332	3.68724	0.66900
1001354639	7.42287	0.68743	3.71143	0.67305
1017054191	7.43727	0.68859	3.71863	0.67421
1514970551	7.49759	0.68985	3.74879	0.67578
2438526191	7.52739	0.68757	3.76369	0.67394
2570169839	7.56669	0.69062	3.78335	0.67697
2772244991	7.58892	0.69186	3.79446	0.67825
3555265271	7.59038	0.68945	3.79519	0.67607
5111994359	7.64749	0.69097	3.82374	0.67784
6194583071	7.69307	0.69318	3.84654	0.68016
7462642151	7.70257	0.69221	3.85129	0.67934
7979490791	7.70933	0.69217	3.85467	0.67934
8462822759	7.77325	0.69733	3.88663	0.68445
12123145319	7.80183	0.69642	3.90091	0.68381
13005495359	7.82594	0.69790	3.91297	0.68531
17833071959	7.89105	0.70071	3.94553	0.68829
29414861999	7.89941	0.69683	3.94970	0.68480

Continued on next page

Table 6.8 — continued from previous page

$\lvert\Delta\rvert$	$L(1,\chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
35535649679	7.94608	0.69923	3.97304	0.68728
42775233959	7.99504	0.70187	3.99752	0.68998
45716419031	8.09414	0.70996	4.04707	0.69798
119634129599	8.11062	0.70297	4.05531	0.69166
132373657391	8.11864	0.70280	4.05932	0.69156
133614550319	8.12420	0.70320	4.06210	0.69196
150451616279	8.15474	0.70484	4.07737	0.69364
210015218111	8.26604	0.71164	4.13302	0.70051
332323080311	8.30989	0.71161	4.15495	0.70072
503494619759	8.31253	0.70848	4.15627	0.69785
603231310919	8.32466	0.70807	4.16233	0.69754
685122125399	8.47178	0.71957	4.23589	0.70892

Table 6.9: Successive ULI_Δ maxima, $\Delta \equiv 1 \pmod 8$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
7	1.18741	0.50072	0.59371	0.27695
15	1.62231	0.45716	0.81116	0.32309
23	1.96520	0.48276	0.98260	0.36562
47	2.29124	0.47713	1.14562	0.38850
71	2.60987	0.50532	1.30493	0.42315
119	2.87989	0.51684	1.43995	0.44447
311	3.38472	0.54377	1.69236	0.48386
479	3.58858	0.55354	1.79429	0.49808
1319	3.89260	0.55416	1.94630	0.50865
1511	3.96017	0.55847	1.98008	0.51372
1559	4.05786	0.57102	2.02893	0.52552
3071	4.30847	0.58062	2.15424	0.53938
5711	4.53127	0.58958	2.26564	0.55158
10391	4.71534	0.59508	2.35767	0.55992
15791	4.87506	0.60323	2.43753	0.56959
18191	4.96137	0.61000	2.48068	0.57662
31391	5.12442	0.61546	2.56221	0.58406
38639	5.19422	0.61859	2.59711	0.58783
95471	5.41928	0.62365	2.70964	0.59578
118271	5.53584	0.63227	2.76792	0.60468
201431	5.65585	0.63447	2.82793	0.60833
366791	5.81495	0.64006	2.90748	0.61528
514751	5.89819	0.64265	2.94910	0.61859
819719	5.98558	0.64348	2.99279	0.62045
890951	6.05086	0.64899	3.02543	0.62593
1339439	6.14018	0.65127	3.07009	0.62899
2155919	6.35035	0.66522	3.17517	0.64342
20357039	6.70670	0.66692	3.35335	0.64873
29858111	6.83310	0.67411	3.41655	0.65625
82636319	7.01518	0.67838	3.50759	0.66167

Continued on next page

Table 6.9 — continued from previous page

$\|\Delta\|$	$L(1,\chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
197317559	7.17063	0.68244	3.58532	0.66661
242230271	7.19930	0.68270	3.59965	0.66707
693126671	7.33941	0.68374	3.66970	0.66911
1001354639	7.42287	0.68743	3.71143	0.67305
1017054191	7.43727	0.68859	3.71863	0.67421
1514970551	7.49759	0.68985	3.74879	0.67578
2570169839	7.56669	0.69062	3.78335	0.67697
2772244991	7.58892	0.69186	3.79446	0.67825
6194583071	7.69307	0.69318	3.84654	0.68016
8462822759	7.77325	0.69733	3.88663	0.68445
13005495359	7.82594	0.69790	3.91297	0.68531
17833071959	7.89105	0.70071	3.94553	0.68829
42775233959	7.99504	0.70187	3.99752	0.68998
45716419031	8.09414	0.70996	4.04707	0.69798
210015218111	8.26604	0.71164	4.13302	0.70051
685122125399	8.47178	0.71957	4.23589	0.70892

Table 6.10: Successive $L(1, \chi_\Delta)$ minima, $\Delta \equiv 1 \pmod 8$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	LLI	$L_\Delta(1)$	LLI_Δ
7	1.18741	1.71184	0.59371	1.03166
463	1.02202	4.01575	0.51101	1.48882
487	0.99651	3.93324	0.49826	1.45649
823	0.98558	4.06379	0.49279	1.48816
1087	0.85759	3.61145	0.42879	1.31580
1423	0.74953	3.21777	0.37477	1.16713
4687	0.73421	3.39371	0.36711	1.21173
176863	0.71714	3.86973	0.35857	1.34613
577663	0.71095	3.98023	0.35548	1.37775
669463	0.69497	3.90738	0.34748	1.35180
678463	0.69034	3.88287	0.34517	1.34325
773767	0.66072	3.73018	0.33036	1.28982
1543063	0.64491	3.71026	0.32245	1.27997
2185423	0.63753	3.70115	0.31877	1.27547
2430943	0.62866	3.65954	0.31433	1.26073
3668527	0.62329	3.66550	0.31164	1.26131
3855223	0.60961	3.58939	0.30480	1.23495
4796863	0.60819	3.59987	0.30409	1.23784
7761007	0.59655	3.57077	0.29827	1.22633
10886023	0.58844	3.54914	0.29422	1.21792
31375807	0.57544	3.54957	0.28772	1.21528
661294327	0.57247	3.73287	0.28624	1.27158
5592560527	0.57200	3.85358	0.28600	1.30927
5607474007	0.56939	3.83617	0.28470	1.30335
9194122543	0.56842	3.85645	0.28421	1.30956
9557534023	0.56680	3.84750	0.28340	1.30646
10703135983	0.56315	3.82872	0.28157	1.29994
11116732687	0.56065	3.81371	0.28032	1.29479
15961519303	0.55099	3.76654	0.27549	1.27832
18193030087	0.54980	3.76499	0.27490	1.27763

Continued on next page

Table 6.10 — continued from previous page

$\lvert\Delta\rvert$	$L(1,\chi_\Delta)$	LLI	$L_\Delta(1)$	LLI_Δ
23397269527	0.54279	3.72947	0.27139	1.26527
45130935103	0.53823	3.72976	0.26911	1.26461
65365858903	0.53754	3.74246	0.26877	1.26851
83001193927	0.53699	3.74974	0.26850	1.27072
84003648367	0.53251	3.71900	0.26626	1.26029
85031215327	0.52224	3.64779	0.26112	1.23615
139564760743	0.52200	3.66817	0.26100	1.24255
265635157327	0.52188	3.69528	0.26094	1.25111
291542071087	0.52070	3.69091	0.26035	1.24954
311588841943	0.51269	3.63695	0.25634	1.23121
569078186623	0.50692	3.62076	0.25346	1.22520

Table 6.11: Successive LLI_Δ minima, $\Delta \equiv 1 \pmod 8$

$\lvert\Delta\rvert$	$L(1,\chi_\Delta)$	LLI	$L_\Delta(1)$	LLI_Δ
7	1.18741	1.71184	0.59371	1.03166

Table 6.12: Successive $L(1, \chi_\Delta)$ maxima, $\Delta \equiv 5 \pmod 8$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
3	0.60460	1.80471	0.90690	0.55940
11	0.94723	0.30404	1.42084	0.59943
35	1.06205	0.23505	1.59308	0.55984
59	1.22700	0.24508	1.84050	0.60853
131	1.37241	0.24321	2.05862	0.63008
251	1.38807	0.22796	2.08210	0.60470
299	1.45346	0.23443	2.18020	0.62510
899	1.46689	0.21480	2.20034	0.58755
971	1.51228	0.22015	2.26842	0.60305
1091	1.61691	0.23335	2.42537	0.64052
1811	1.69792	0.23654	2.54689	0.65454
3251	1.70806	0.22940	2.56209	0.63978
5099	1.71582	0.22462	2.57373	0.62969
5771	1.81960	0.23662	2.72941	0.66419
11051	1.85285	0.23313	2.77928	0.65844
12011	1.86326	0.23351	2.79489	0.65998
26771	1.88167	0.22750	2.82251	0.64698
39731	1.98589	0.23624	2.97884	0.67360
42059	1.99142	0.23636	2.98714	0.67419
66491	2.01026	0.23442	3.01539	0.67050
74051	2.02033	0.23465	3.03050	0.67157
112811	2.02036	0.23113	3.03053	0.66298
143051	2.05995	0.23374	3.08992	0.67125
246419	2.06315	0.22994	3.09472	0.66199
262499	2.08480	0.23188	3.12720	0.66778
275651	2.10028	0.23324	3.15042	0.67184
412331	2.11354	0.23181	3.17031	0.66883
437051	2.16695	0.23726	3.25042	0.68469
954971	2.21500	0.23711	3.32250	0.68624
1692851	2.26970	0.23925	3.40455	0.69372

Continued on next page

Table 6.12 — continued from previous page

$\lvert\Delta\rvert$	$L(1,\chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
3230939	2.27211	0.23561	3.40816	0.68445
3724811	2.27890	0.23549	3.41835	0.68438
5162219	2.29668	0.23548	3.44503	0.68494
7337651	2.33925	0.23789	3.50888	0.69257
8868851	2.34401	0.23735	3.51602	0.69132
10538219	2.34875	0.23691	3.52312	0.69033
12727139	2.35388	0.23644	3.53081	0.68926
14549531	2.39014	0.23939	3.58521	0.69806
21706571	2.39445	0.23778	3.59167	0.69399
23662019	2.40446	0.23835	3.60669	0.69576
38567051	2.41251	0.23678	3.61876	0.69185
42143219	2.49178	0.24412	3.73766	0.71344
142044971	2.51180	0.24047	3.76769	0.70429
174795851	2.52662	0.24098	3.78993	0.70604
272941451	2.54204	0.24056	3.81305	0.70528
314928851	2.54213	0.23997	3.81320	0.70372
426521651	2.54462	0.23897	3.81693	0.70110
427514699	2.56142	0.24054	3.84213	0.70570
464951891	2.57109	0.24111	3.85663	0.70745
544237979	2.58934	0.24218	3.88401	0.71077
1128586499	2.60375	0.24068	3.90562	0.70704
1803355571	2.62426	0.24081	3.93639	0.70784
1863357011	2.63356	0.24154	3.95033	0.71002
2552798531	2.63762	0.24076	3.95643	0.70800
3015365051	2.66255	0.24243	3.99382	0.71306
4438732451	2.67581	0.24226	4.01372	0.71287
6794527379	2.68870	0.24195	4.03305	0.71227
7190337899	2.72698	0.24520	4.09048	0.72188
11423307299	2.75974	0.24655	4.13961	0.72620
36085593491	2.76694	0.24343	4.15041	0.71783

Continued on next page

Table 6.12 — continued from previous page

$\lvert\Delta\rvert$	$L(1,\chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
51921829571	2.77447	0.24297	4.16171	0.71668
70487240339	2.79977	0.24424	4.19965	0.72063
110055989819	2.80027	0.24295	4.20041	0.71709
132955516139	2.81989	0.24409	4.22983	0.72057
205835168171	2.82388	0.24317	4.23582	0.71809
224571061139	2.82907	0.24337	4.24360	0.71873
293546318771	2.88332	0.24726	4.32498	0.73038
846154222619	2.88480	0.24446	4.32721	0.72262
921824947931	2.89388	0.24499	4.34082	0.72426

Table 6.13: Successive ULI_Δ maxima, $\Delta \equiv 5 \pmod 8$

$\lvert\Delta\rvert$	$L(1,\chi_\Delta)$	ULI	$L_\Delta(1)$	ULI_Δ
11	0.94723	0.30404	1.42084	0.59943
59	1.22700	0.24508	1.84050	0.60853
131	1.37241	0.24321	2.05862	0.63008
1811	1.69792	0.23654	2.54689	0.65454
5771	1.81960	0.23662	2.72941	0.66419
437051	2.16695	0.23726	3.25042	0.68469
1692851	2.26970	0.23925	3.40455	0.69372
14549531	2.39014	0.23939	3.58521	0.69806
42143219	2.49178	0.24412	3.73766	0.71344
7190337899	2.72698	0.24520	4.09048	0.72188
11423307299	2.75974	0.24655	4.13961	0.72620
293546318771	2.88332	0.24726	4.32498	0.73038

Table 6.14: Successive $L(1, \chi_\Delta)$ minima, $\Delta \equiv 5 \pmod 8$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	LLI	$L_\Delta(1)$	LLI_Δ
3	0.60460	0.12313	0.90690	1.19175
43	0.47909	1.37439	0.71863	1.69992
67	0.38381	1.19369	0.57571	1.43053
163	0.24607	0.86752	0.36910	0.99578
85507	0.23636	1.24366	0.35454	1.30261
111763	0.22553	1.19808	0.33830	1.25311
166147	0.22351	1.20358	0.33527	1.25641
222643	0.21971	1.19458	0.32957	1.24534
462883	0.21241	1.18141	0.31861	1.22787
958483	0.21179	1.20286	0.31768	1.24685
991027	0.19881	1.13022	0.29822	1.17141
351036667	0.19752	1.27442	0.29629	1.30355
376058587	0.19554	1.26309	0.29331	1.29182
553348867	0.19552	1.27118	0.29328	1.29937
643338763	0.19223	1.25289	0.28835	1.28042
986305483	0.19056	1.25063	0.28584	1.27737
1560135427	0.18834	1.24500	0.28252	1.27087
1595514187	0.18829	1.24506	0.28243	1.27090
1930143763	0.18764	1.24439	0.28146	1.26992
2426489587	0.18655	1.24146	0.27982	1.26658
2562211723	0.18470	1.23020	0.27706	1.25501
3030266803	0.18445	1.23160	0.27668	1.25619
3416131987	0.18152	1.21415	0.27227	1.23822
6465681643	0.18082	1.22069	0.27122	1.24401
6623767483	0.17973	1.21375	0.26959	1.23690
15442196323	0.17843	1.21922	0.26765	1.24140
21538327507	0.17609	1.20857	0.26413	1.23017
45640185427	0.17604	1.22007	0.26406	1.24101
84291143203	0.17599	1.22914	0.26398	1.24958
85702502803	0.17448	1.21885	0.26172	1.23910
107415709003	0.17070	1.19573	0.25605	1.21538

Table 6.15: Successive LLI_Δ minima, $\Delta \equiv 5 \pmod 8$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	LLI	$L_\Delta(1)$	LLI_Δ
11	0.94723	1.79400	1.42084	2.72986
19	0.72073	1.68549	1.08110	2.28766
43	0.47909	1.37439	0.71863	1.69992
67	0.38381	1.19369	0.57571	1.43053
187	0.45947	1.64635	0.68921	1.88025
235	0.40987	1.50656	0.61480	1.70740
267	0.38452	1.43265	0.57679	1.61720
403	0.31299	1.21430	0.46948	1.35521
427	0.30406	1.18600	0.45610	1.32172
462883	0.21241	1.18141	0.31861	1.22787
991027	0.19881	1.13022	0.29822	1.17141

Table 6.16: Successive $L(1, \chi_\Delta)$ maxima, $\Delta \equiv 0 \pmod 4$

Δ	$L(1, \chi_\Delta)$	ULI
4	0.78540	1.35004
8	1.11072	0.85183
24	1.28255	0.62278
56	1.67925	0.67702
104	1.84835	0.67578
164	1.96254	0.67633
356	1.99805	0.63355
404	2.18820	0.68560
776	2.25553	0.66819
1256	2.30477	0.65851
1364	2.38177	0.67655
2756	2.39370	0.64939
4616	2.58943	0.68171
7556	2.60217	0.66730
8564	2.64793	0.67475
13796	2.67469	0.66605
14024	2.75897	0.68652
22244	2.78046	0.67770
25016	2.78080	0.67436
32504	2.78806	0.66875
35096	2.81728	0.67364
42836	2.88402	0.68411
52664	2.90221	0.68288
61844	2.95609	0.69131
92804	3.01127	0.69378
98276	3.04649	0.70045
120056	3.08274	0.70381
324596	3.08792	0.68235
326504	3.17785	0.70209
650744	3.24017	0.70127
973496	3.32417	0.71132
2577896	3.34590	0.69782
3357416	3.37421	0.69912
3519764	3.39260	0.70211
4534724	3.41675	0.70279
5610596	3.42188	0.70031
6799316	3.45779	0.70450
8736404	3.49262	0.70754
11307656	3.53147	0.71130
22633316	3.56326	0.70709
32826356	3.58276	0.70554
33968024	3.63200	0.71475
35478596	3.67093	0.72177
82896824	3.68374	0.71240
105019736	3.70569	0.71348
145546964	3.72482	0.71288
190346516	3.74807	0.71388
257889176	3.80342	0.72056
430932596	3.81490	0.71641
563886116	3.86523	0.72261
973849796	3.87059	0.71723
1214887736	3.89084	0.71846
1769558216	3.90469	0.71681
1898042324	3.92092	0.71902
2427116936	3.95618	0.72279
4975524644	3.97884	0.71928
5003320724	4.01388	0.72555
6461737064	4.01566	0.72322
10443781256	4.05538	0.72548
13441926104	4.05749	0.72335
13832708984	4.10757	0.73200
27867502724	4.14624	0.73202
56133619736	4.17271	0.73011
135476600936	4.17408	0.72247
143302472024	4.17656	0.72241
143843317796	4.19806	0.72610

Table 6.17: Successive ULI maxima, $\Delta \equiv 0 \pmod 4$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	ULI
8	1.11072	0.85183

Table 6.18: Successive $L(1, \chi_\Delta)$ minima, $\Delta \equiv 0 \pmod 4$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	LLI
4	0.78540	0.37036
88	0.66979	1.44951
148	0.51647	1.19962
232	0.41251	1.00944
1012	0.39502	1.10314
21652	0.38430	1.27655
45208	0.38416	1.31555
53188	0.38142	1.31443
105172	0.36811	1.30088
120712	0.36169	1.28436
121972	0.34182	1.21426

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	LLI
189352	0.33210	1.19740
332872	0.32671	1.19936
424708	0.30852	1.14104
539092	0.30807	1.14749
1154008	0.30414	1.15748
89593288	0.29208	1.22598
570297652	0.28994	1.25734
1633938772	0.28103	1.23936
2107116328	0.28101	1.24409
2772969028	0.28064	1.24758
3057078952	0.27887	1.24153

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	LLI
3645619972	0.27722	1.23741
5242483108	0.27578	1.23749
6206516152	0.27085	1.21829
12988838692	0.26849	1.22016
14300408068	0.26786	1.21890
25721176072	0.26413	1.21140
42230741512	0.26258	1.21203
111890044792	0.25888	1.20955
140113131412	0.25800	1.20871
382853517832	0.25689	1.21778

Table 6.19: Successive LLI minima, $\Delta \equiv 0 \pmod 4$

$\lvert\Delta\rvert$	$L(1, \chi_\Delta)$	LLI
8	1.11072	1.17394
232	0.41251	1.00944

Figure 6.6: Local ULI_Δ maxima for $x \leq |\Delta| < x + 2^{28}, x \in 2^{28}\mathbb{Z}$

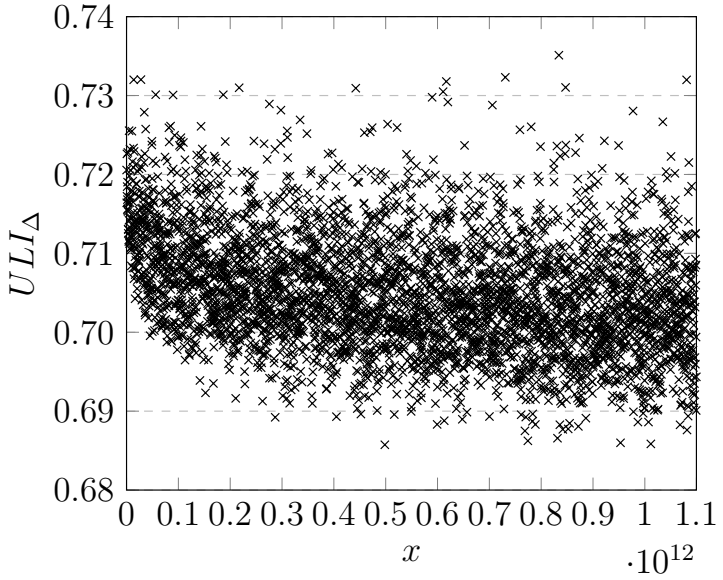
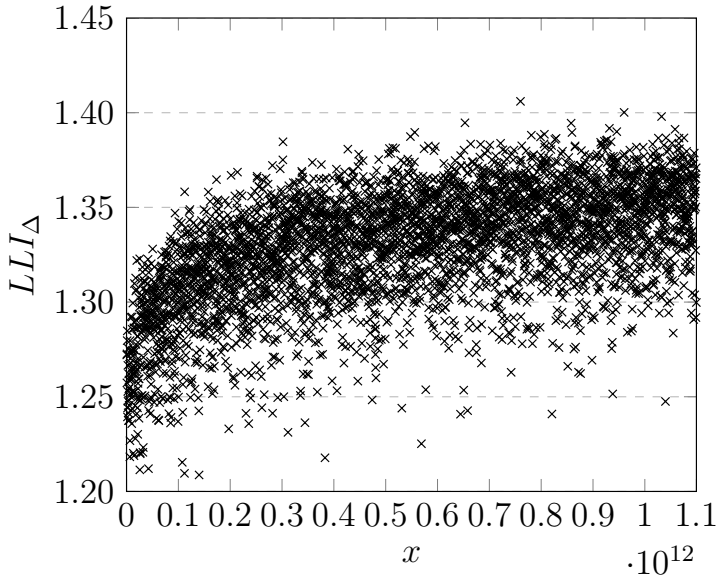


Figure 6.7: Local LLI_Δ minima for $x \leq |\Delta| < x + 2^{28}, x \in 2^{28}\mathbb{Z}$



6.4 The Cohen-Lenstra Heuristics

In the middle of 1980s, Cohen and Lenstra published two papers with a series of powerful heuristics on the structure of the odd part of the class group Cl_Δ [CL83, CL84]. The odd part Cl^*_Δ is simply the largest subgroup of Cl_Δ with an *odd* cardinality. Thus, for example, for $\Delta = -3896$ we have $Cl_\Delta \cong C(12) \times C(3)$, and therefore its odd part $Cl^*_\Delta \cong C(3) \times C(3)$.

One of the major observations that Cohen and Lenstra made was that Cl^*_Δ is much more often cyclic than non-cyclic. Their result was stated in the form of a conjecture.

Conjecture 6.4.1. [CL84, C1] *Define*

$$\eta_k(l) = \prod_{i=1}^{k} \left(1 - \frac{1}{l^i}\right) \text{ and } C_\infty = \prod_{i=2}^{\infty} \zeta(i) \approx 2.294856589,$$

where $\zeta(s) = \sum_{n=1}^{\infty} n^{-s}$ *denotes the Riemann zeta function. For* $\Delta < 0$*, the probability for the odd part of the class group* Cl_Δ *to be cyclic is*

$$\Pr(Cl^*_\Delta \text{ is cyclic}) = \frac{315\zeta(3)}{6\pi^4 \eta_\infty(2) C_\infty} \approx 0.977575. \tag{6.4.1}$$

There are two other heuristics that Cohen and Lenstra stated in their paper, which we studied during our computations.

Conjecture 6.4.2. [CL84, C2] [Bue99] *Let l be an odd prime. For $\Delta < 0$, the probability that l divides $h(\Delta)$ is*

$$\Pr(l \mid h(\Delta)) = 1 - \eta_\infty(l); \tag{6.4.2}$$

the probability that l^2 divides $h(\Delta)$ is

$$\Pr(l^2 \mid h(\Delta)) = 1 - \frac{l\eta_\infty(l)}{l-1}; \tag{6.4.3}$$

the probability that l^3 divides $h(\Delta)$ is

$$\Pr(l^3 \mid h(\Delta)) = 1 - \frac{l^3\eta_\infty(l)}{(l-1)^2(l+1)}. \tag{6.4.4}$$

Conjecture 6.4.3. [CL84, C5] *Let l be an odd prime. For $\Delta < 0$, the probability that the l-rank of Cl_Δ is equal to r is*

$$\Pr(l\text{-rank} = r) = \frac{\eta_\infty(l)}{l^{r^2}\eta_r(l)^2}. \tag{6.4.5}$$

In order to study these conjectures, we follow the approach of Ramachandran and introduce three functions: $c(x)$, $p_l(x)$ and $p_{l,r}(x)$ [Ram06]:

$$c(x) = \frac{\#\text{ of } Cl^*_\Delta \text{ cyclic with } |\Delta| < x}{\#\text{ of } \Delta \text{ with } |\Delta| < x} \ / \ \Pr(Cl^*_\Delta \text{ is cyclic});$$

$$p_l(x) = \frac{\#\text{ of } h(\Delta) \text{ divisible by } l \text{ with } |\Delta| < x}{\#\text{ of } \Delta \text{ with } |\Delta| < x} \ / \ \Pr(l \mid h(\Delta));$$

$$p_{l,r}(x) = \frac{\#\text{ of } Cl_\Delta \text{ with } l\text{-rank} = r \text{ and } |\Delta| < x}{\#\text{ of } \Delta \text{ with } |\Delta| < x} \ / \ \Pr(l\text{-rank} = r).$$

If the Cohen-Lenstra heuristics hold, we would expect each of those functions to approach 1 as x grows. We can observe this behaviour on Figures 6.8 – 6.13, which plot $c(x)$, $p_l(x)$, $p_{l,2}(x)$ and $p_{l,3}(x)$ for various values of l. There are three observations that we would like to mention when interpreting this data:

(a) Note that the curves on Figure 6.13, which correspond to $p_{5,3}(x)$ and $p_{7,3}(x)$, are jagged. We argue that this is the case due to the fact that our limit $N = 2^{40}$ is too low. As N increases, we expect these curves to take a smoother form. In support of this observation, compare our Figure 6.13 to Figures 5.12 and 5.14 of Ramachandran [Ram06], which plot $p_{5,3}(x)$ and $p_{7,3}(x)$ for $x < 2 \cdot 10^{11}$, respectively. Our curves look notably smoother.

(b) Note that in Figure 6.9 the curve $p_7(x)$ overlaps $p_5(x)$. Analogously, in Figure 6.10 the curve $p_{7^2}(x)$ approaches 1 faster than $p_{5^2}(x)$. However, this is not the case for the curves in Figure 6.11, as $p_{5^3}(x)$ is above $p_{7^3}(x)$. A similar observation can be made regarding the function $p_{l,r}(x)$. In Figure 6.12, the curve $p_{7,2}(x)$ is clearly above $p_{5,2}(x)$. In contract, the curve $p_{5,3}(x)$ overlaps $p_{7,3}(x)$ in Figure 6.13. We argue that this also happens for the reason that the limit $N = 2^{40}$ is low. Figures 6.11 and 6.12 suggest that the curves $p_{5^3}(x)$ and $p_{7^3}(x)$, $p_{5,3}(x)$ and $p_{7,3}(x)$, will eventually intersect, and those curves which correspond to a larger prime will start to approach 1 faster. Most probably, a further increase of the class group tabulation limit to 2^{41} would allow us to encounter the points of their intersection. Based on the data in Figures 6.9 — 6.13, we conjecture that as the value of l grows, the corresponding functions $p_l(x)$ and $p_{l,r}(x)$ for a fixed r approach 1 faster as x goes to infinity.

(c) Clearly, all of the curves on Figures 6.8 — 6.13 approach the expected value of 1. Moreover, one may observe that the curve $p_{l^k}(x)$ for a fixed $k = 1, 2$ (and conjecturably $k = 3$) tends to 1 faster as the prime l grows. A similar observation can be made regarding the function $p_{l,r}(x)$ for a fixed $r = 2$ (and conjecturably $r = 3$). Why is that happening? What happens when the value of a prime l remains fixed, and parameters k and r vary? These questions lie outside of the scope of this thesis, and are left for the future work.

In addition to Figures 6.8 – 6.13, we also present several tables. In Table 6.8, we list the total number of class groups with non-cyclic odd parts. Tables 6.21, 6.22, 6.23 count the

total number of $h(\Delta)$ divisible by a prime l and its powers l^2, l^3, respectively. Finally, Table 6.24 counts class groups with a certain l-rank.

Table 6.20: Number of noncyclic odd parts of class groups

x	Total	Non-cyclic	Percent	$c(x)$
10^7	3039632	47218	1.55341	1.00705
10^8	30396385	536238	1.76415	1.00489
10^9	303963510	5833414	1.91912	1.00331
10^{10}	3039635443	61494437	2.02309	1.00224
10^{11}	30396355052	636501087	2.09400	1.00152
$2 \cdot 10^{11}$	60792710179	1283029629	2.11050	1.00135
$3 \cdot 10^{11}$	91189065248	1932535723	2.11926	1.00126
$4 \cdot 10^{11}$	121585420327	2583844783	2.12513	1.00120
$5 \cdot 10^{11}$	151981775550	3236429002	2.12948	1.00116
$6 \cdot 10^{11}$	182378130683	3889995513	2.13293	1.00112
$7 \cdot 10^{11}$	212774486110	4544337515	2.13575	1.00109
$8 \cdot 10^{11}$	243170840635	5199342505	2.13814	1.00107
$9 \cdot 10^{11}$	273567195607	5854902775	2.14021	1.00105
10^{12}	303963550712	6510933430	2.14201	1.00103
2^{40}	334211458670	7164219493	2.14362	1.00101

Figure 6.8: Ratios of non-cyclic odd parts of class groups

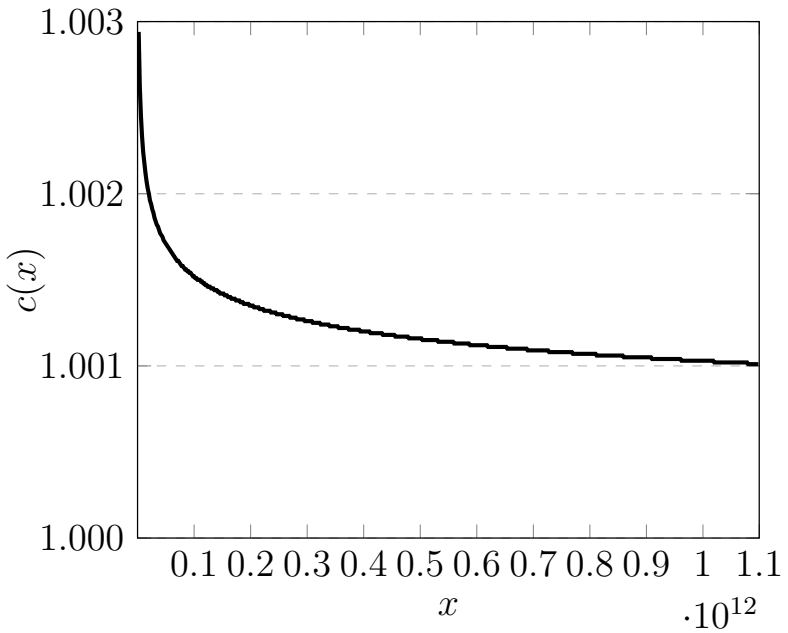


Table 6.21: Counts of class numbers divisible by l

x	$3 \mid h$	$5 \mid h$	$7 \mid h$	$11 \mid h$	$13 \mid h$	$17 \mid h$	$19 \mid h$	$23 \mid h$
10^7	1257631	713387	487838	293520	245239	182689	161630	131271
10^8	12840789	7197319	4923629	2979924	2486384	1865874	1655713	1355556
10^9	130131918	72375150	49414201	30015516	25050836	18832174	16729980	13691451
10^{10}	1312874060	725975201	495177221	300860836	251323163	188884363	168012939	137517024
10^{11}	13206088529	7271547905	4956628127	3011896994	2516050182	1891543105	1682478893	1377653213
$2 \cdot 10^{11}$	26447989308	14547903930	9914941601	6025009729	5032948550	3783959331	3365616747	2756018300
$3 \cdot 10^{11}$	39700741936	21825546084	14873726078	9038458883	7550137579	5676465937	5048888332	4134562522
$4 \cdot 10^{11}$	52959934649	29103662856	19832681021	12052003780	10067454468	7569126724	6732338269	5513203250
$5 \cdot 10^{11}$	66223739128	36382211005	24791661364	15065606774	12584840703	9461748773	8415821817	6891832785
$6 \cdot 10^{11}$	79491008890	43661126382	29750874514	18079320114	15102137499	11354490190	10099409592	8270489579
$7 \cdot 10^{11}$	92761033879	50940277442	34710302571	21092999797	17619561852	13247335884	11783040709	9649175946
$8 \cdot 10^{11}$	106033521908	58219944093	39669843978	24106720004	20137035912	15140137396	13466635439	11027858679
$9 \cdot 10^{11}$	119308020675	65499671827	44629193028	27120432707	22654498276	17032878861	15150299084	12406578239
10^{12}	132584350621	72779583545	49588756987	30134192653	25171929972	18925544408	16833977643	13785279159
2^{40}	145797270882	80023955398	54524158518	33133247297	27677104824	20809106232	18509507749	15157283235

Figure 6.9: Values of $p_l(x)$

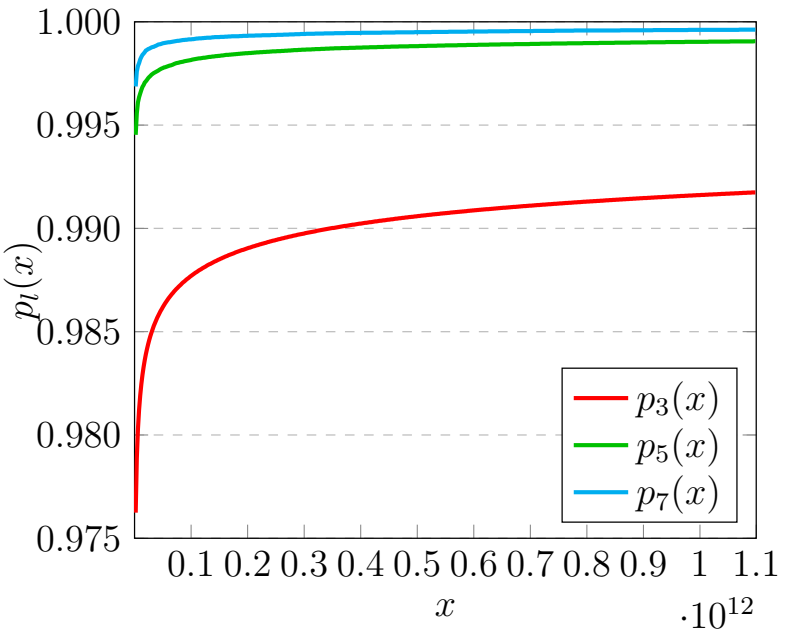


Table 6.22: Counts of class numbers divisible by l^2

x	$3^2 \mid h$	$5^2 \mid h$	$7^2 \mid h$	$11^2 \mid h$	$13^2 \mid h$	$17^2 \mid h$	$19^2 \mid h$	$23^2 \mid h$
10^7	440961	141641	64815	20586	13213	5771	4001	1931
10^8	4565681	1467965	691973	247785	168238	86067	64024	37589
10^9	46669516	14880843	7096360	2656557	1841506	1015601	785732	503084
10^{10}	473059821	149794492	71684347	27213051	19087911	10793671	8506718	5633927
10^{11}	4771798243	1502766412	719737046	274638773	193328646	110343171	87474411	58698021
$2 \cdot 10^{11}$	9562686509	3007297283	1440383575	550036450	387401212	221433629	175703825	118121646
$3 \cdot 10^{11}$	14359328108	4512243583	2161169915	825645733	581617349	332641649	264052308	177666170
$4 \cdot 10^{11}$	19159171908	6017374304	2882055007	1101293604	775928441	443933802	352495173	237291697
$5 \cdot 10^{11}$	23961605597	7522623520	3602985109	1376986032	970290362	555240837	440955802	296987824
$6 \cdot 10^{11}$	28765745620	9028024072	4323998999	1652744904	1164669568	666614549	529468551	356691806
$7 \cdot 10^{11}$	33571356157	10533455763	5045033603	1928501566	1359089068	778018177	618002388	416439786
$8 \cdot 10^{11}$	38378418685	12039064282	5766148996	2204255173	1553537388	889437964	706555788	476205994
$9 \cdot 10^{11}$	43186508348	13544767263	6487255702	2480047774	1747985673	1000871807	795136092	535980144
10^{12}	47995382529	15050453769	7208361986	2755865977	1942471388	1112317212	883725368	595776680
2^{40}	52781733309	16548745048	7925996557	3030352820	2136010611	1223236728	971898457	655284622

Figure 6.10: Values of $p_{l^2}(x)$

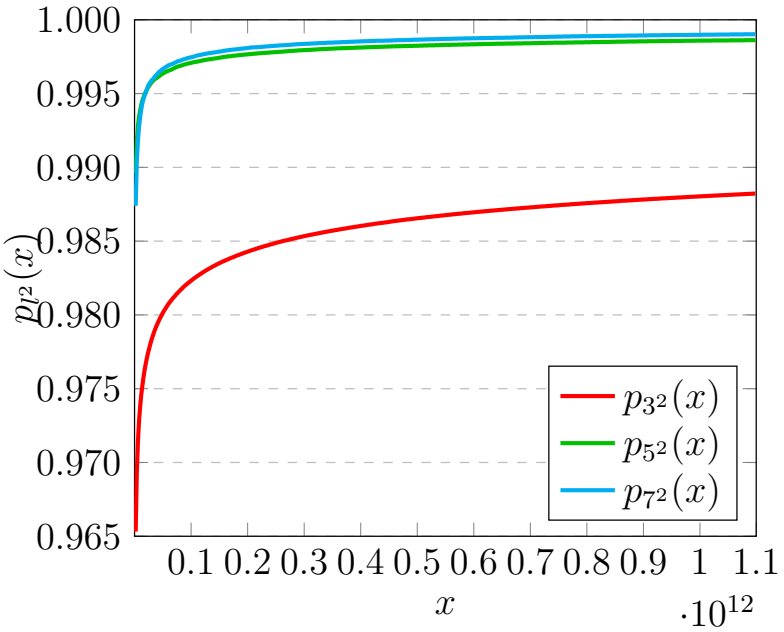


Table 6.23: Counts of class numbers divisible by l^3

x	$3^3 \mid h$	$5^3 \mid h$	$7^3 \mid h$	$11^3 \mid h$	$13^3 \mid h$	$17^3 \mid h$	$19^3 \mid h$	$23^3 \mid h$
10^7	145653	22720	4862	301	71	3	0	0
10^8	1538588	273633	76485	8963	3470	497	202	28
10^9	15866712	2912369	929064	164528	77825	18923	9686	2827
10^{10}	161467472	29856223	9946520	2147462	1158118	384459	230341	86788
10^{11}	1631887079	301457152	101922382	23754726	13644773	5335121	3523233	1630450
$2 \cdot 10^{11}$	3271518412	603855802	204592839	48226576	27980961	11278511	7574184	3660907
$3 \cdot 10^{11}$	4913394152	906443589	307393826	72836539	42456879	17341570	11755606	5796508
$4 \cdot 10^{11}$	6556565122	1209120619	410268962	97521425	57009680	23486316	16004100	7989844
$5 \cdot 10^{11}$	8200684872	1511876078	513188083	122252427	71609984	29672355	20298935	10228748
$6 \cdot 10^{11}$	9845465590	1814685428	616142307	147024518	86236376	35892244	24624277	12499490
$7 \cdot 10^{11}$	11490876501	2117506075	719141703	171817968	100895539	42138068	28977442	14796853
$8 \cdot 10^{11}$	13136822282	2420386531	822166272	196626796	115573846	48403009	33348215	17115844
$9 \cdot 10^{11}$	14783209563	2723347174	925182556	221449208	130273120	54688838	37736516	19447524
10^{12}	16429907430	3026260612	1028226668	246296110	144991953	60988083	42141953	21789632
2^{40}	18068926477	3327728333	1130793659	271032590	159653032	67269462	46534473	24134461

Figure 6.11: Values of $p_{l^3}(x)$

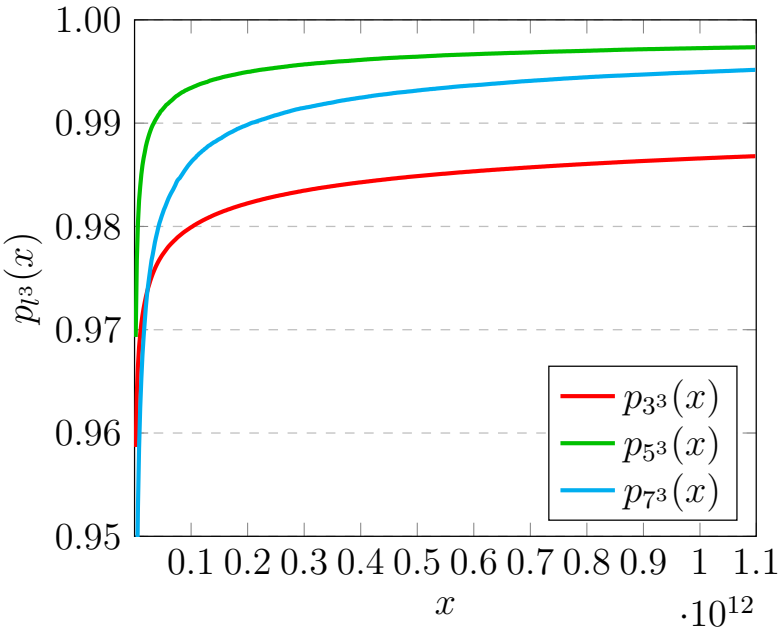


Table 6.24: Counts of class groups with l-rank $= r$

x	3-rank = 2	5-rank = 2	7-rank = 2	11-rank = 2	3-rank = 3	5-rank = 3	7-rank = 3	3-rank = 4
10^7	40285	5469	1211	148	25	0	0	0
10^8	461925	57336	13619	2036	725	14	0	0
10^9	5049471	600025	144273	21685	11769	189	3	1
10^{10}	53457200	6121548	1475944	225293	157738	1904	66	26
10^{11}	554992183	61905528	14909598	2278429	1891327	19701	824	396
$2 \cdot 10^{11}$	1119549000	124086783	29864434	4566277	3941440	39455	1699	903
$3 \cdot 10^{11}$	1686937952	186346310	44837690	6858856	6041677	59455	2555	1442
$4 \cdot 10^{11}$	2256067209	248638170	59813385	9153290	8170672	79056	3392	2052
$5 \cdot 10^{11}$	2826419025	310963856	74791724	11448254	10319592	99020	4302	2667
$6 \cdot 10^{11}$	3397716149	373303706	89772515	13746350	12486498	119058	5158	3317
$7 \cdot 10^{11}$	3969781768	435637308	104762170	16040965	14667860	138969	6050	3974
$8 \cdot 10^{11}$	4542454057	498010970	119754407	18334765	16861780	158992	6949	4634
$9 \cdot 10^{11}$	5115675246	560398913	134735076	20629307	19064061	179000	7804	5303
10^{12}	5689326792	622806579	149727575	22924492	21274374	199005	8677	5987
2^{40}	6260628955	684906543	164647966	25208614	23481723	218977	9586	6627

Figure 6.12: Values of $p_{l,2}(x)$

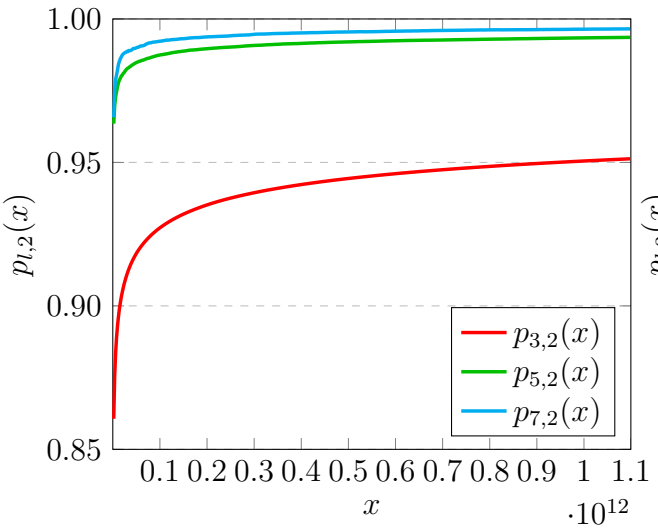
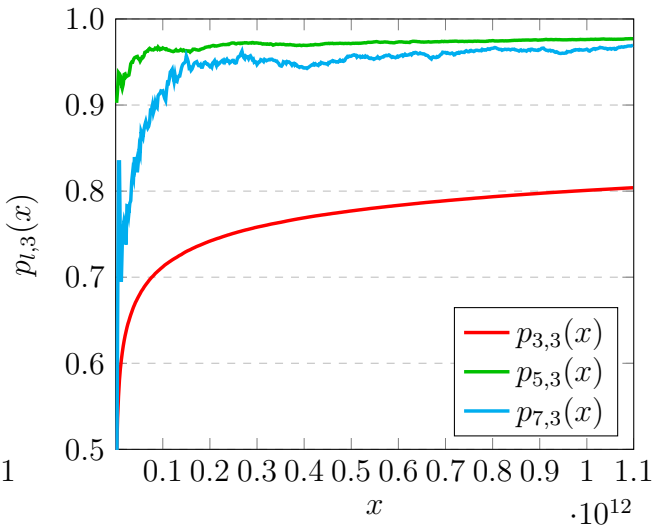


Figure 6.13: Values of $p_{l,3}(x)$



6.5 Exotic Groups

As we observed previously, some class groups possess curious structures, such as a high p-rank, or a high order of a particular p-group. Buell referred to these groups as "exotic" [Bue99], and one exotic structure that we were interested in was $C(2) \times \ldots \times C(2)$. All Cl_Δ of such a form are related to so-called *idoneal numbers*, which were studied by Euler and

Gauß[4]. A positive number D is idoneal if every integer n, which is uniquely representable in the form $n = x^2 \pm Dy^2$ with $\gcd(x^2, Dy^2) = 1$, is either a prime, or a prime power, or twice one of these. Both Euler and Gauß conjectured that the largest idoneal number does not exceed 1848 [Gau98, §303]. From the class group perspective it means that $\Delta = -5460$ is the largest fundamental discriminant such that $Cl_\Delta \cong C(2) \times \ldots \times C(2)$. In 1918, the hypothesis of Euler and Gauß was confirmed by Hecke and Landau under the assumption of the ERH [Lan18]. However, unconditionally this problem still remains open, though Weinberger was able to prove that there exists *at most* one idoneal number exceeding 1848 [Wei73]. In our computations, we were able to confirm that up to 2^{40} the largest in its absolute value fundamental discriminant Δ which has $Cl_\Delta \cong C(2) \times \ldots \times C(2)$ is $\Delta = -5460$. This result agrees with findings of Euler and Gauß.

During our computations, we also looked at the problem of finding Cl_Δ with the smallest $|\Delta|$ which corresponds to a certain p-group structure. This question was first explored by Buell [Bue99], where he tabulated the first occurrences of exotic groups. He gave a list of first even and odd Δ, as well as the total number of them up to $2.2 \cdot 10^9$. Later, this list was extended by Jacobson et al. and Ramachandran [JRW06, Ram06]. In her Master's thesis, Ramachandran discovered first $\Delta = -104862921055$ with the 4-rank equal to 5. In Tables 6.25 – 6.31, we further refine this list. Though we are still quite far away from Cl_Δ with the smallest 3-rank equal to 5, which corresponds to $\Delta = -90338558707159$ [Bel04], we were able to extend the table of exotic groups which have 4-rank 5. For example, $\Delta = -940830993327$ is the smallest discriminant in absolute value with 2-Sylow subgroup of Cl_Δ being isomorphic to $C(2^4) \times C(2^4) \times C(2^2) \times C(2^2) \times C(2^2)$. Note that we are not interested in the 2-rank of a class group, as it is known that when Δ has k distinct prime factors, the 2-rank will always be $k - 1$ [Coh62, p. 225, Table 1]. That is why instead of looking at a 2-Sylow subgroup, we consider its subgroup which consists of squares of its elements. In Tables 6.25 – 6.31, the columns e_i for $1 \leq i \leq 5$ indicate that the order of the i-th cyclic component in the canonical

[4]See the extensive survey on idoneal numbers by Kani [Kan11].

decomposition (5.0.1) is equal to p^{e_i} for a prime $p > 2$, and to p^{e_i+1} for $p = 2$.

Apart of that, we also looked at the first occurrences of doubly, trebly and quadruply non-cyclic class groups, which are listed in Tables 6.32, 6.33 and 6.34, respectively. Perhaps the most interesting discovery were four class groups of the form $C(2^2 \cdot 3 \cdot 5 \cdot 13) \times C(2^2 \cdot 3 \cdot 5 \cdot 13)$, the smallest absolute discriminant with that class group is $\Delta = -616825494863$. In the following tables, we write in bold first occurrences of non-cyclic p-groups that are not present in Tables 5.12 – 5.19, provided by Ramachandran [Ram06].

In Tables 6.25 — 6.34, we use * in order to indicate that class groups corresponding to a particular structure were not yet discovered. In columns entitled "# even Δ" and "# odd Δ" we present the total number of even and odd negative discriminants Δ, respectively, such that $|\Delta| < 2^{40}$ and Cl_Δ has a specificied structure.

Table 6.25: Non-cyclic rank 2 2-Sylow squared subgroups

e_1	e_2	First even $\|\Delta\|$	# even Δ	First odd $\|\Delta\|$	# odd Δ
1	1	6328	2620216706	2379	4318069327
2	1	6392	1975722271	5795	3256294593
2	2	25988	162222848	32331	267679917
3	1	7544	987906188	4895	1628035009
3	2	118040	121693492	22127	200707888
3	3	636664	10141269	618947	16732631
4	1	39236	493980133	10295	813871838
4	2	264452	60835616	103727	100334740
4	3	1353316	7611891	804639	12545350
4	4	4126328	634423	2365599	1042805
5	1	145604	246975869	60695	406962692
5	2	605816	30422164	310295	50174674
5	3	3118916	3809147	1008095	6267227
5	4	14060036	476852	13263095	784673
5	5	53231864	39533	22128095	65357
6	1	312584	123500326	187239	203488548
6	2	2472824	15210435	968495	25079592
6	3	8049284	1901712	3194495	3135441
6	4	49563236	238519	12993671	392024
6	5	115629944	29829	122764631	49191
6	6	979202552	2553	1059634567	4214
7	1	1297544	61744136	535871	101760149
7	2	5407736	7605776	3177095	12544525
7	3	43637624	950209	6342959	1565866
7	4	217291076	118846	24475919	195648
7	5	585612296	14841	353879327	24331
7	6	3777569528	1907	1331102631	3128
7	7	49964393960	141	4487508695	263
8	1	4765316	30871409	2009111	50884698
8	2	23989796	3798734	6851831	6274647

Continued on next page

Table 6.25 — continued from previous page

e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
8	3	112475684	475270	27310895	784371
8	4	415337096	59715	155791391	97523
8	5	1872668936	7454	700226279	12235
8	6	19379520584	891	2500463471	1484
8	7	43627697252	79	20564153183	185
8	8	**471197076872**	7	28148188439	7
9	1	16899704	15434855	5266439	25427916
9	2	106946936	1900223	27704351	3136238
9	3	348733304	237565	153404279	392213
9	4	1525528196	29748	434237639	49223
9	5	8793326372	3483	1697415695	6123
9	6	40109627876	319	9658267583	722
9	7	108887003384	22	36156111551	65
9	8	*	*	**877206528943**	3
10	1	74198264	7716817	22858871	12713942
10	2	200896484	952191	105684095	1566438
10	3	1186927304	118637	342897959	196135
10	4	5778168824	13979	1544290079	23972
10	5	27799085816	1332	10843705871	2783
10	6	120308887544	105	32772714719	238
10	7	**681281598344**	6	**307122265871**	21
11	1	236054264	3862283	75612599	6355721
11	2	827711876	472520	304561631	783881
11	3	3722450696	55519	1324096199	96087
11	4	14034192644	5525	4543687511	10985
11	5	92221912184	395	29925386231	995
11	6	**531104802884**	9	135586529279	62
11	7	*	*	**537868128719**	3
12	1	929170436	1923726	280073351	3175683
12	2	3562207736	223979	1120758911	383847

Continued on next page

Table 6.25 — continued from previous page

e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
12	3	17874504584	21724	5855914895	44418
12	4	82876399304	1555	20975257511	4007
12	5	**338850531704**	61	90998785895	265
12	6	*	*	**561829505111**	10
13	1	3989574536	910342	966467519	1555015
13	2	15271434884	87539	4429883519	177017
13	3	73338178436	6257	15406567679	16059
13	4	**288168873464**	251	94172111879	1052
13	5	*	*	**337146930095**	51
14	1	13471444964	357621	3899095199	718773
14	2	53549964536	24833	12633802271	63764
14	3	187900973816	953	70219409399	4359
14	4	**1007353992776**	1	**338019115391**	172
15	1	49186240484	102603	15649176791	259285
15	2	**206869987364**	3612	63808583879	16865
15	3	**782288913656**	11	**248068193831**	709
16	1	178998749816	15305	54229304951	68787
16	2	**854051236484**	31	**230818363751**	2802
16	3	*	*	**782114375831**	7
17	1	**803357850104**	104	**233831956511**	11383
17	2	*	*	**890526156911**	11
18	1	*	*	**839658626351**	48

Table 6.26: Non-cyclic rank 2 p-Sylow subgroups

p	e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
3	1	1	3896	1232952143	4027	2478116574
3	2	1	27656	547833376	3299	1100928519
3	2	2	208084	15211775	134059	30550078
3	3	1	55316	182621232	17399	366958093
3	3	2	998708	6757809	351751	13568629
3	3	3	39337384	187578	6207263	376883
3	4	1	462356	60876705	29399	122313724
3	4	2	4279448	2254552	1332167	4524572
3	4	3	88848836	83448	41361815	167519
3	4	4	1271559208	2282	136071631	4550
3	5	1	3935384	20300215	508847	40762401
3	5	2	36356456	751321	15042011	1510971
3	5	3	576873236	27523	152637311	55659
3	5	4	16887409796	934	4301015239	1931
3	5	5	100684202984	15	6743415071	35
3	6	1	28026164	6760157	3582743	13585248
3	6	2	263591156	248716	19180391	501225
3	6	3	2671485416	8238	636617543	17985
3	6	4	49547047976	165	7274282423	517
3	6	5	*	*	133786229531	22
3	7	1	232838744	2235165	32681951	4515275
3	7	2	2175729716	73809	167885231	161074
3	7	3	17668343384	1657	3541241903	4909
3	7	4	175380473204	18	47649110911	109
3	7	5	*	*	**253237383431**	2
3	8	1	1723181864	664412	98311919	1451967
3	8	2	17082145064	14852	1173834359	44211
3	8	3	137781248216	118	37703425007	922
3	8	4	*	*	**225796561799**	10
3	9	1	11132690456	133772	1106108639	397069

Continued on next page

Table 6.26 — continued from previous page

p	e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
3	9	2	93287426216	1110	11901791639	8640
3	9	3	*	*	60543925679	100
3	10	1	98284577816	9856	8795475911	77561
3	10	2	**1018482429656**	2	65798421911	908
3	10	3	*	*	**766483839959**	2
3	11	1	**786365476244**	16	52623967679	7879
3	11	2	*	*	**677250946319**	24
3	12	1	*	*	**512068796879**	177
5	1	1	17944	175301206	11199	350916611
5	2	1	178004	42073554	50783	84205799
5	2	2	9623444	280597	1390367	562093
5	3	1	2189204	8413880	621599	16838347
5	3	2	273928024	66734	52456111	134425
5	3	3	13603495364	380	1068156239	873
5	4	1	56245556	1673322	5820119	3359072
5	4	2	2194276244	11442	290810159	25827
5	4	3	188682132248	35	10036313687	146
5	5	1	1163891636	288437	88527911	637167
5	5	2	26611903016	922	5180829911	3444
5	5	3	*	*	**213265691687**	15
5	6	1	25411429364	21557	1614153239	85742
5	6	2	**775319038196**	5	75913193999	175
5	7	1	**573881434136**	107	48662190359	4626
5	8	1	*	*	**941197327199**	3
7	1	1	159592	46118676	63499	92233234
7	2	1	890984	7531563	480059	15055109
7	2	2	288854504	19090	59288543	38396
7	3	1	50642024	1068869	4603007	2143985
7	3	2	5468598824	2539	528784319	5572
7	3	3	**798957687128**	2	40111506371	10

Continued on next page

Table 6.26 — continued from previous page

p	e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
7	4	1	1157188724	123174	172820591	282052
7	4	2	75003362216	80	16336216607	453
7	5	1	64461971636	3718	5800676279	21149
7	5	2	*	*	**336699684383**	5
7	6	1	*	*	174018745031	290
11	1	1	580424	7591533	65591	15176167
11	2	1	24557096	750646	7948999	1504473
11	2	2	8124316712	423	4536377039	951
11	3	1	712328756	54237	218130623	125116
11	3	2	**344379903284**	5	91355041631	29
11	4	1	89983172564	574	7219509359	4458
11	5	1	*	*	**935094698711**	2
13	1	1	703636	3893604	228679	7777257
13	2	1	86189912	319460	14127343	641144
13	2	2	15290030216	93	10692322055	234
13	3	1	5247449576	14867	781846103	40172
13	3	2	*	*	**366445322799**	2
13	4	1	**604812537944**	15	55385334839	522
17	1	1	4034356	1333908	1997799	2660006
17	2	1	558578648	78066	43780679	162839
17	2	2	**522715590248**	3	94733724779	12
17	3	1	28205334296	1181	5767994839	5201
17	4	1	*	*	**607531396391**	7
19	1	1	3419828	853097	373391	1705545
19	2	1	921151124	42752	17803439	90573
19	2	2	*	*	106460379983	15
19	3	1	131937350936	278	5862529559	1981
23	1	1	11137012	398194	7472983	794202
23	2	1	1188873236	14433	510491431	32338
23	3	1	**428918887976**	12	74447537447	296

Continued on next page

Table 6.26 — continued from previous page

p	e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
29	1	1	16706324	156769	20113607	314441
29	2	1	5614832984	3478	296873471	9004
29	3	1	*	*	**323459074199**	19
31	1	1	96468488	119918	11597903	241058
31	2	1	14560212776	2195	362103671	6146
31	3	1	*	*	**503905534439**	14
37	1	1	25162772	58584	51461727	118698
37	2	1	33184320308	607	2793641999	2142
41	1	1	99141272	38712	6112511	77973
41	2	1	29030848244	282	12558317543	1155
43	1	1	66614312	31888	39405967	64620
43	2	1	120525154964	194	28602441479	827
47	1	1	1054312388	22060	57403799	45099
47	2	1	65816894324	83	20751947191	428
53	1	1	777263864	13434	26675327	27524
53	2	1	**313806056276**	24	34862413351	200
59	1	1	244858136	8685	133943879	17770
59	2	1	**278155567784**	6	65887828631	81
61	1	1	1264482536	7413	137323663	15566
61	2	1	**388888967156**	6	148712371111	62
67	1	1	1516640872	5071	448220959	10779
67	2	1	**323124297044**	3	131240605511	28
71	1	1	839514836	3919	198786779	8405
71	2	1	*	*	183648001319	16
73	1	1	420255476	3501	483264167	7474
73	2	1	*	*	**350771311831**	17
79	1	1	5114393428	2554	888934163	5560
79	2	1	*	*	179492976431	4
83	1	1	2390420804	2036	884989055	4455
83	2	1	*	*	**589364144599**	3

Continued on next page

Table 6.26 — continued from previous page

p	e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
89	1	1	2339707096	1521	1941485183	3294
89	2	1	*	*	**619130566127**	2
97	1	1	9388308724	1013	2179032511	2246
97	2	1	*	*	**438994809599**	2
101	1	1	4293806984	868	758562611	1972
101	2	1	*	*	**981198752759**	1
103	1	1	19084053944	830	787024943	1770
107	1	1	4576627816	666	4041299887	1493
109	1	1	2202664232	579	4903396807	1461
113	1	1	3422486836	514	1047199379	1184
127	1	1	7127111912	322	3482629127	763
131	1	1	16018714472	269	2884161823	589
137	1	1	37914915092	194	4549823483	503
139	1	1	50553654520	168	8396560295	472
149	1	1	56336668888	132	15233330011	335
151	1	1	42941394424	126	13310472899	346
157	1	1	19416052676	113	15661511531	279
163	1	1	46586000024	94	10302820679	242
167	1	1	20926233044	97	22669688623	215
173	1	1	84419230376	76	14602373903	190
179	1	1	89298106628	50	28362963611	177
181	1	1	89809227124	49	8991716639	156
191	1	1	108225487880	37	48122759783	143
193	1	1	28354858472	46	84647431783	99
197	1	1	169507937464	26	66490566011	108
199	1	1	159624206436	37	6561724871	99
211	1	1	97297226504	21	18146008687	67
223	1	1	**229260698804**	17	36799898071	49
227	1	1	**248738329160**	17	129251563279	43
229	1	1	103810044568	17	89550601631	50

Continued on next page

Table 6.26 — continued from previous page

p	e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
233	1	1	172430289704	12	29922371399	50
239	1	1	84169153736	14	55757811107	47
241	1	1	**275897077784**	13	74882513855	33
251	1	1	**274131019432**	7	78181110431	24
257	1	1	123259359896	3	23738884679	38
263	1	1	**482147329592**	7	37893813311	31
269	1	1	**241103392196**	4	11129396567	22
271	1	1	**291445797352**	5	171753801031	18
277	1	1	**266610558308**	6	128621435167	18
281	1	1	**644634989492**	2	266379885935	13
283	1	1	**400355665764**	3	94175615183	11
293	1	1	**874615243688**	3	158602460567	17
307	1	1	**749662659128**	3	149654057447	13
311	1	1	**666221368184**	4	111301462879	8
313	1	1	**416363928728**	3	**303265490831**	14
317	1	1	461612707316	3	190338535131	6
331	1	1	158739065384	3	**388995885319**	7
337	1	1	**506841655124**	2	**283026340679**	6
347	1	1	427842776408	1	425834455071	7
349	1	1	**804641768168**	1	32819826815	5
353	1	1	**839537284648**	2	**305328598259**	9
359	1	1	*	*	**627072510479**	4
367	1	1	*	*	176235988399	2
373	1	1	*	*	**215425181891**	5
379	1	1	*	*	**356510006687**	4
383	1	1	**878382375224**	1	137740312007	6
389	1	1	*	*	85401404639	7
397	1	1	*	*	**434530437127**	2
409	1	1	*	*	**594857692087**	1
419	1	1	*	*	116218001623	4

Continued on next page

Table 6.26 — continued from previous page

p	e_1	e_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
421	1	1	*	*	**422660888879**	4
431	1	1	*	*	**567134500223**	3
433	1	1	*	*	**791181108079**	2
439	1	1	*	*	**782761871063**	2
443	1	1	**462953812184**	2	146805555551	2
449	1	1	*	*	**347760731679**	3
457	1	1	**212262246356**	1	**413877350951**	2
461	1	1	*	*	**353455619411**	4
463	1	1	**1047876328724**	1	**679010903567**	1
467	1	1	**683325795752**	1	**817093587359**	1
503	1	1	*	*	**544027580079**	1
521	1	1	**969683875304**	1	**363850136623**	1
577	1	1	*	*	**733117084823**	1
683	1	1	*	*	192932176559	1
719	1	1	*	*	**737463696271**	1

Table 6.27: Non-cyclic rank 3 2-Sylow squared subgroups

e_1	e_2	e_3	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
1	1	1	1148984	49739875	1295823	85202404
2	1	1	568888	43837793	503659	74964095
2	2	1	3040888	5414685	2209467	9297616
2	2	2	29418872	84417	31078723	150970
3	1	1	550712	21922028	1696071	37472470
3	2	1	5235592	4063709	1456131	6973956
3	2	2	58984568	74117	38432395	132277
3	3	1	42747512	338068	15254499	581312
3	3	2	180245764	9288	306703595	16533
3	3	3	26037089032	182	3072761723	329
4	1	1	2256376	10955149	863455	18721891
4	2	1	4605572	2032386	4312495	3484880
4	2	2	84855928	36845	113368287	65863
4	3	1	32985032	253528	12315783	434659
4	3	2	709356440	6854	262912611	12392
4	3	3	12536161012	168	968674895	229
4	4	1	194468984	21227	63983699	36236
4	4	2	599039224	594	1348092695	1011
4	4	3	48635750392	18	15453558059	36
5	1	1	3600632	5474880	2600247	9364380
5	2	1	6030584	1015849	11888359	1741436
5	2	2	656353752	18378	176472095	33124
5	3	1	110604856	126550	89794955	218170
5	3	2	942374776	3458	530051079	6212
5	3	3	36300169992	68	29812383539	135
5	4	1	306870392	15764	52959695	26996
5	4	2	1950599672	439	745029527	785
5	4	3	19285756744	16	26840480251	24
5	5	1	632499896	1284	3153932195	2305
5	5	2	20634196984	36	30024717407	87

Continued on next page

Table 6.27 — continued from previous page

e_1	e_2	e_3	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
5	5	3	**654525620632**	1	**484182251499**	1
6	1	1	9601544	2743003	3285399	4681028
6	2	1	40329464	508550	12870695	870817
6	2	2	475618808	9252	116917743	16627
6	3	1	110476484	63682	80285439	108877
6	3	2	1685435012	1722	245004591	3154
6	3	3	19231492484	46	5154637111	74
6	4	1	1028005256	7976	709215951	13535
6	4	2	18928358948	212	1745876431	402
6	4	3	**207143057928**	8	**288125782387**	13
6	4	4	*	*	**305719802099**	1
6	5	1	11230043192	974	2025881495	1729
6	5	2	103931852152	23	12135120919	48
6	5	3	*	*	**877143467991**	1
6	6	1	41596381252	83	13864237495	138
6	6	2	**337781772292**	2	**428890037995**	4
7	1	1	29971256	1369650	11621255	2340407
7	2	1	114856964	253703	39546239	435186
7	2	2	1598656804	4612	1153171407	8326
7	3	1	518626616	31705	104663495	54332
7	3	2	3452129864	803	3937488031	1549
7	3	3	17129098616	16	43500603367	34
7	4	1	1985708324	4089	805283687	6786
7	4	2	30957938552	107	37875548711	218
7	4	3	**209679734360**	3	116761650559	7
7	5	1	20184931576	480	2561173655	857
7	5	2	**538462706212**	6	71448310047	22
7	6	1	71493317624	64	11605902591	115
7	6	2	**260475668612**	1	**999087514595**	1
7	7	1	**259572201464**	8	42732217895	17

Continued on next page

Table 6.27 — continued from previous page

e_1	e_2	e_3	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
8	1	1	65195576	684418	41667695	1170245
8	2	1	322681316	127019	162267599	217666
8	2	2	1927781624	2262	1195516295	3987
8	3	1	1264546436	15860	698196239	27365
8	3	2	18368038136	440	7762724495	769
8	3	3	40374695204	15	19206387503	17
8	4	1	6533780984	2041	3423173871	3264
8	4	2	97911313784	44	36780288287	77
8	4	3	**950476486040**	2	**316447950295**	2
8	5	1	21419706104	242	8359014839	442
8	5	2	181708658936	4	107006725631	10
8	6	1	**213994732808**	18	79607470655	37
8	6	2	**1017884605304**	1	183508431839	3
8	7	1	*	*	**280421815031**	3
8	7	2	*	*	**827148073295**	1
9	1	1	317455544	341763	118575119	585508
9	2	1	1697181944	63736	417704759	109340
9	2	2	15725508164	1114	9044868391	1944
9	3	1	5874224996	7885	2678195351	13460
9	3	2	38767885124	190	9656429351	369
9	3	3	**648263005688**	3	180341937335	4
9	4	1	35512754756	938	8510136095	1684
9	4	2	91670247044	16	188923218895	45
9	5	1	**273752132132**	90	20210353631	167
9	5	2	*	*	1006674452791	1
9	6	1	**456982685816**	3	160385785919	8
10	1	1	1387470776	171237	408615119	291519
10	2	1	4688751044	31814	2482130199	54323
10	2	2	25496650616	489	12194979695	1043
10	3	1	20948614904	3772	5655701831	6840

Continued on next page

Table 6.27 — continued from previous page

e_1	e_2	e_3	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
10	3	2	173930539256	89	38525569031	149
10	4	1	89449248164	322	25800653711	675
10	4	2	**740275424888**	3	**229249550999**	14
10	5	1	**285133396616**	12	154588056719	58
10	5	2	*	*	**902255215895**	1
10	6	1	*	*	**966988154495**	1
11	1	1	4022802824	85668	1345931495	145122
11	2	1	19408564964	14891	7545348695	27738
11	2	2	102796104824	212	27487687295	443
11	3	1	59346834104	1324	25662174551	2724
11	3	2	**357522684536**	13	168077989551	45
11	4	1	**402435170276**	47	115480891655	200
11	5	1	*	*	**496036448255**	12
12	1	1	16063397444	40477	5744234831	74207
12	2	1	65603208824	5277	16360877231	10685
12	2	2	**355824525944**	47	118699623191	135
12	3	1	**208232323064**	253	91844298959	759
12	3	2	*	*	**552618912095**	8
12	4	1	*	*	**266702660111**	39
13	1	1	57410101124	14766	22371341879	28958
13	2	1	**244875549896**	1104	59760022871	3059
13	2	2	*	*	**274952301599**	20
13	3	1	**784746326264**	5	**440711621495**	129
14	1	1	**257270664824**	2798	58698960239	8421
14	2	1	**908401250756**	4	**253494621071**	554
15	1	1	**944946576824**	17	**273146579471**	1565
15	2	1	*	*	**950040365951**	2
16	1	1	*	*	**988815743159**	3

Table 6.28: Non-cyclic rank 3 p-Sylow subgroups

p	e_1	e_2	e_3	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
3	1	1	1	4447704	4402943	4897363	9006825
3	2	1	1	5288968	2120479	3321607	4330730
3	2	2	1	145519608	78893	101375499	160673
3	2	2	2	3457439416	224	364435991	422
3	3	1	1	12755172	705714	5153431	1443249
3	3	2	1	57236692	34561	79378899	71137
3	3	2	2	18741973496	119	11037391871	209
3	3	3	1	1940867992	964	559587163	1891
3	3	3	2	**341946799364**	2	20687610651	11
3	4	1	1	35180884	235853	13275687	481883
3	4	2	1	192757064	11587	53209523	23574
3	4	2	2	12251300788	34	9766538987	78
3	4	3	1	2245873412	426	522302531	841
3	4	3	2	**295863285976**	3	**744853350587**	1
3	4	4	1	190975971556	10	26320580987	28
3	5	1	1	111442868	78530	32852423	160883
3	5	2	1	3130903236	3876	413771887	7854
3	5	2	2	**412703787940**	9	45248632247	24
3	5	3	1	43721231572	132	2232519167	240
3	5	3	2	*	*	788521058151	1
3	5	4	1	186447381556	3	**376544421947**	5
3	6	1	1	509049176	26212	124438679	53143
3	6	2	1	19996254456	1150	376424303	2519
3	6	2	2	**582608055992**	1	9483757583	9
3	6	3	1	27291040424	22	53192765699	66
3	6	4	1	**900453600692**	1	**276331426207**	2
3	7	1	1	6382094504	7722	461309711	17096
3	7	2	1	33828950744	254	4163792239	742
3	7	2	2	*	*	**484468933679**	1
3	7	3	1	**1076743681124**	1	**338926563823**	10

Continued on next page

Table 6.28 — continued from previous page

p	e_1	e_2	e_3	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
3	8	1	1	20594835764	1660	5347129751	4812
3	8	2	1	**276573602516**	12	59714529551	139
3	8	3	1	*	*	124071345551	3
3	9	1	1	**182514096404**	127	12792023879	978
3	9	2	1	*	*	**581116399159**	14
3	10	1	1	**989021051864**	1	146114436719	104
3	11	1	1	*	*	**797107037711**	1
5	1	1	1	11203620	55523	18397407	111320
5	2	1	1	272394484	13809	51213139	27704
5	2	2	1	7095550408	102	6896149079	228
5	2	2	2	*	*	287442559199	1
5	3	1	1	300240404	2855	145367147	5571
5	3	2	1	49468612564	22	29867315295	53
5	4	1	1	5871738932	495	3511272455	1093
5	4	2	1	**204195796664**	3	116279191211	7
5	5	1	1	161307848024	43	25384593659	143
5	6	1	1	*	*	**349008665407**	5
7	1	1	1	1898879592	2654	501510767	5446
7	2	1	1	2760876184	456	648153647	857
7	2	2	1	**439240920004**	1	**868770849819**	3
7	3	1	1	32727392168	48	19379510159	116
7	4	1	1	**356820088964**	1	**451900165735**	4
11	1	1	1	3035884424	45	23235125867	102
11	2	1	1	**889484965924**	2	145931588651	9
13	1	1	1	**218639119912**	11	38630907167	20
13	2	1	1	*	*	105479207735	1
17	1	1	1	*	*	**824746962451**	2
19	1	1	1	*	*	136073793499	1

Table 6.29: Non-cyclic rank 4 2-Sylow squared subgroups

e_1	e_2	e_3	e_4	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
1	1	1	1	471960184	156923	197731195	274277
2	1	1	1	498994552	148187	511858407	258905
2	2	1	1	942647416	21323	914157695	37438
2	2	2	1	4647849848	488	4924087483	961
2	2	2	2	*	*	**1082104673395**	1
3	1	1	1	194401336	73725	349519339	129040
3	2	1	1	1649835652	16210	872841047	28077
3	2	2	1	11726216888	461	2659965695	834
3	2	2	2	**679836481816**	2	**223284424495**	1
3	3	1	1	7736922872	1272	4199412607	2399
3	3	2	1	52035316984	49	17184321295	89
3	3	3	1	**263297633828**	2	**843441277895**	1
4	1	1	1	934558264	36965	141244707	64803
4	2	1	1	437474872	7957	440663695	14130
4	2	2	1	5648488952	227	1992071683	420
4	2	2	2	*	*	**246384569947**	1
4	3	1	1	7171718852	1027	530870223	1799
4	3	2	1	68445834520	32	18394768247	71
4	3	3	1	**409765017848**	1	71657045499	2
4	4	1	1	70040577764	76	1737119891	165
4	4	2	1	**244654266616**	3	35723521195	7
5	1	1	1	1110786104	18667	385521331	32302
5	2	1	1	4943558040	3989	2034087987	7072
5	2	2	1	33861890488	121	35738073111	211
5	3	1	1	16381065848	456	2547515451	852
5	3	2	1	35219141368	23	41645702735	38
5	3	3	1	**1046969085944**	1	**560991763371**	1
5	4	1	1	18891451144	71	26517688015	117
5	4	2	1	**371944696760**	4	171996454555	6
5	5	1	1	**224848338424**	7	143225068051	7

Continued on next page

Table 6.29 — continued from previous page

e_1	e_2	e_3	e_4	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
6	1	1	1	1036693796	9046	905764295	16149
6	2	1	1	4350311096	2091	4942746003	3510
6	2	2	1	43143969656	63	8628191135	110
6	3	1	1	24951539576	229	29583016707	420
6	3	2	1	**245982210340**	9	53490173795	22
6	4	1	1	76332750328	30	68390991723	65
6	4	2	1	**450318048248**	1	**331486531779**	2
6	5	1	1	31903643768	6	42458106639	12
6	6	1	1	**687670228984**	1	*	*
7	1	1	1	726515384	4679	816714055	8177
7	2	1	1	4610055416	991	672821903	1809
7	2	2	1	40338576376	28	75905537331	54
7	3	1	1	19570359032	111	16839000895	210
7	3	2	1	153487259768	8	167130403891	6
7	3	3	1	*	*	**1054957819763**	1
7	4	1	1	75305261828	15	167511791767	23
7	4	2	1	**1071201322168**	1	**267088472339**	2
7	5	1	1	**940052161252**	2	153791430767	6
7	6	1	1	*	*	**469072753055**	1
8	1	1	1	12243038648	2293	2436431439	4104
8	2	1	1	12815163704	517	8573244695	917
8	2	2	1	**92790235832**	15	**91630708055**	37
8	3	1	1	61265888312	58	51413000223	107
8	3	2	1	**440675634488**	4	187748187151	8
8	4	1	1	95938795256	10	**289841742903**	16
8	5	1	1	**879914797304**	1	**857945640295**	2
9	1	1	1	14577769412	1133	16567132647	2011
9	2	1	1	44331931256	235	15026935655	470
9	2	2	1	**294944781304**	7	138574355567	9
9	3	1	1	140992756856	26	47502531911	42

Continued on next page

Table 6.29 — continued from previous page

e_1	e_2	e_3	e_4	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
9	3	2	1	*	*	**969121445095**	1
9	4	1	1	**876796541924**	1	*	*
10	1	1	1	34219906616	582	13189888895	1104
10	2	1	1	108802824584	108	73131029751	178
10	2	2	1	**802440845048**	2	**486979070799**	4
10	3	1	1	**494541501176**	7	**203712802655**	12
10	4	1	1	*	*	**857496953039**	1
11	1	1	1	83893756964	305	31482746399	438
11	2	1	1	**410763625604**	33	160670086055	64
11	3	1	1	*	*	**752090449679**	2
12	1	1	1	**352087167416**	65	171460689231	131
12	2	1	1	*	*	**630764329719**	9
13	1	1	1	*	*	**356012217431**	36

Table 6.30: Non-cyclic rank 4 p-Sylow subgroups

p	e_1	e_2	e_3	e_4	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
3	1	1	1	1	2520963512	1140	653329427	2659
3	2	1	1	1	11451958228	564	3972542271	1263
3	2	2	1	1	64435895236	26	32543535351	55
3	3	1	1	1	26041127732	174	5288116947	405
3	3	2	1	1	34245189208	11	170168896891	20
3	3	3	1	1	*	*	**1074734433547**	1
3	4	1	1	1	3146813128	69	35684560479	143
3	4	2	1	1	**426126877012**	3	128589208863	8
3	5	1	1	1	17346090376	19	7993105123	37
3	5	2	1	1	*	*	**473827747963**	2
3	6	1	1	1	**460093393912**	8	76951070303	15
3	7	1	1	1	**1047320556596**	1	**513092626699**	2
3	8	1	1	1	*	*	**226138531999**	2

Table 6.31: Non-cyclic rank 5 2-Sylow squared subgroups

e_1	e_2	e_3	e_4	e_5	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
1	1	1	1	1	**517740321848**	3	**252686780823**	20
2	1	1	1	1	**224048553848**	13	**237475165195**	16
2	2	1	1	1	**1027358361464**	1	**578330063139**	3
2	2	2	1	1	*	*	**854250929495**	1
3	1	1	1	1	**322218299192**	2	**395060936767**	9
3	2	1	1	1	**603550491044**	1	**319053004023**	3
3	3	1	1	1	*	*	**940830993327**	1
4	1	1	1	1	**213258265592**	6	**769181116495**	6
4	2	1	1	1	*	*	104862921055	2
5	1	1	1	1	*	*	**565717844839**	4
5	2	1	1	1	**766857327928**	1	*	*
6	1	1	1	1	**533449412536**	1	**1089673039299**	1
6	2	1	1	1	*	*	**1025865373595**	1
7	1	1	1	1	*	*	**340295386595**	3
10	1	1	1	1	*	*	**798456328671**	1

Table 6.32: Doubly non-cyclic class groups

p_1	p_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
2	3	64952	133365397	104255	220405887
2	5	472196	14356325	280847	23675520
2	7	858296	3439749	465719	5675285
2	11	11221832	524314	10724727	865381
2	13	34388612	262760	13725759	434957
2	17	62975684	87626	50684339	145545
2	19	287484728	55329	41698223	91768
2	23	427072292	25214	206527919	41967
2	29	1189666616	9489	621746551	16070
2	31	1893349604	7070	410359895	12115
2	37	3833690756	3287	1839127055	5998
2	41	5503733780	2188	2620148255	3816
2	43	6036095204	1773	3155959391	3100
2	47	5201404616	1170	4812997295	2165
2	53	17170629304	689	2436482551	1208
2	59	12050201444	406	9075455255	736
2	61	19120092536	344	19914584807	648
2	67	22395736184	211	2962630655	473
2	71	15544112516	155	17401096571	313
2	73	18251641796	146	34627398895	280
2	79	63115620356	100	33743280671	196
2	83	52717441208	87	44282533439	145
2	89	57218613128	62	24120821559	103
2	97	194876101124	34	55968627055	59
2	101	136696766216	29	69000110911	61
2	103	**249867224036**	20	103632639095	44
2	107	202497107144	21	256026173359	41
2	109	175445318264	19	167437181855	49
2	113	116375297096	21	168389069839	28
2	127	**600915430712**	4	**447549178255**	11

Continued on next page

Table 6.32 — continued from previous page

p_1	p_2	First even $\|\Delta\|$	# even Δ	First odd $\|\Delta\|$	# odd Δ
2	131	**219974481416**	3	**330144882719**	9
2	137	**656783816696**	5	**422293096959**	10
2	139	**874776631544**	2	102213969911	12
2	149	**865959626372**	2	**474523426451**	7
2	151	**951421789028**	3	**255493436207**	8
2	157	**930502831736**	1	**289807552047**	9
2	163	*	*	160332909911	5
2	167	*	*	**459875288639**	2
2	173	**572877537272**	2	**997627201055**	1
2	179	*	*	**352635876623**	1
2	181	*	*	**998404753191**	1
2	191	*	*	**797578007543**	2
2	193	**961848140804**	1	*	*
2	197	*	*	**685792326599**	1
2	223	*	*	**614891960927**	1
2	227	*	*	**880087218911**	1
3	5	2766392	3991683	119191	8132562
3	7	16053944	957145	3561799	1947751
3	11	22297448	144979	14898623	297324
3	13	70887272	72963	39186347	149370
3	17	142736408	24106	188315447	49873
3	19	372243764	15315	234113631	31048
3	23	1675854452	6745	351756527	14382
3	29	3395393108	2579	557577743	5549
3	31	3792995864	1876	386659943	4057
3	37	6112785556	895	1455428855	1992
3	41	17658330596	543	1166119039	1319
3	43	3286197848	429	4075192859	1068
3	47	16964359736	267	485163311	691
3	53	41696300984	167	457096511	388

Continued on next page

Table 6.32 — continued from previous page

p_1	p_2	First even $\|\Delta\|$	# even Δ	First odd $\|\Delta\|$	# odd Δ
3	59	49943038232	89	10227491279	290
3	61	32515774996	83	8522929927	240
3	67	84253538216	66	26792580191	144
3	71	113923334836	35	17614533947	107
3	73	159201388244	23	11752995103	123
3	79	85480238756	23	51762875627	49
3	83	**411040250696**	8	50476998239	69
3	89	**271776528392**	14	146604777199	29
3	97	**373716927704**	5	43344787079	22
3	101	**204919229864**	3	**270845549231**	12
3	103	**374301791476**	8	93069031703	14
3	107	**747657517988**	2	193384461719	15
3	109	**379370724596**	5	35029686023	17
3	113	*	*	56428950647	21
3	127	**761263140536**	1	127466536019	10
3	131	*	*	**248486020319**	2
3	137	*	*	**373309196719**	4
3	139	*	*	**261265037799**	3
3	149	*	*	**555574557467**	4
3	157	*	*	**258504106919**	2
3	163	*	*	**288700332223**	5
3	181	*	*	195280405559	1
3	191	*	*	**778133573263**	1
3	193	*	*	4 **15837893871**	1
3	197	*	*	**675588676571**	1
3	223	*	*	**1044678632711**	1
5	7	45324248	101421	19399067	207894
5	11	195367988	15362	112179371	31521
5	13	644440376	7435	94672727	15749
5	17	714706004	2390	135145159	5090

Continued on next page

Table 6.32 — continued from previous page

p_1	p_2	First even $\|\Delta\|$	# even Δ	First odd $\|\Delta\|$	# odd Δ
5	19	3925533652	1447	965381231	3309
5	23	14260068616	642	336603767	1447
5	29	15541379720	212	10138338695	536
5	31	11788579624	143	17205833747	401
5	37	10719968216	55	16249120831	191
5	41	158975194472	37	26948199679	120
5	43	51986729896	37	71114945339	87
5	47	**337410526616**	16	8182208159	78
5	53	**375201391636**	8	22759605719	28
5	59	**842452697976**	2	166413410411	20
5	61	**621148062232**	2	198540663599	14
5	67	**952877473160**	1	**202658297511**	13
5	71	*	*	14917874303	4
5	73	*	*	63515115611	7
5	79	*	*	**695299489415**	4
5	83	*	*	**255558978287**	5
5	89	*	*	112383267127	7
5	97	*	*	**957408127639**	1
5	107	*	*	**895542638663**	1
7	11	629704808	3467	235436591	7256
7	13	1419402728	1686	1580010631	3596
7	17	6198957812	508	1851928807	1134
7	19	24082268968	281	5166049215	723
7	23	22198579640	112	2591136407	308
7	29	116113422452	29	21164450935	112
7	31	18704562356	26	68200813691	89
7	37	**220308406520**	5	49918973471	36
7	41	161694761896	8	103077938087	16
7	43	**395768104936**	1	57006644887	18
7	47	**611628524996**	2	98533572251	12

Continued on next page

Table 6.32 — continued from previous page

p_1	p_2	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
7	53	**819974042456**	1	**532593252151**	6
7	59	*	*	**746029216663**	3
7	61	*	*	**530458082031**	2
7	79	*	*	**1010896284767**	1
7	101	*	*	**613532171711**	1
11	13	31664474564	203	13609279311	490
11	17	50159859416	54	41219120419	142
11	19	**293745669956**	33	19439678123	86
11	23	**440245788692**	7	94266055451	45
11	29	**258828614756**	2	**246806029679**	13
11	31	**752299766228**	1	167546860535	6
11	37	*	*	**507297592171**	1
11	41	*	*	199926563159	1
13	17	72831993636	17	41507696303	76
13	19	138488491556	14	75779342435	41
13	23	**886308340568**	1	**303087341987**	15
13	29	*	*	140248449319	9
13	31	**1042065325544**	1	**309693265351**	5
13	37	*	*	**583833769207**	1
13	41	**969016080404**	1	**407911409771**	2
17	19	150334566104	2	**473841789911**	9
17	23	**432363302164**	2	54134972891	3
17	29	*	*	**892052200651**	2
17	31	*	*	**1035367542059**	1
19	23	*	*	**659380117199**	1
19	29	*	*	**915336787039**	1

Table 6.33: Trebly non-cyclic class groups

p_1	p_2	p_3	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
2	3	5	37892516	268946	23677127	446684
2	3	7	108485576	63127	82966895	106275
2	3	11	1872785336	8996	264640055	15595
2	3	13	7213488644	4412	1047903271	7733
2	3	17	4928392004	1323	1882856559	2436
2	3	19	17805303416	824	6136370399	1445
2	3	23	23420913668	320	15603393095	591
2	3	29	80314582984	89	15426649911	204
2	3	31	66961530788	71	22141447799	145
2	3	37	178914112264	16	83842784855	62
2	3	41	**336418071608**	7	136497900751	29
2	3	43	137593842104	10	88786054727	20
2	3	47	**601202450744**	5	276385799983	15
2	3	53	*	*	96869926295	1
2	5	7	1282481528	6323	2003704023	10832
2	5	11	13208419364	813	7602070071	1441
2	5	13	20738568548	375	17041306931	718
2	5	17	159910639736	74	61842180179	187
2	5	19	104353792952	44	75528143095	125
2	5	23	**260804694404**	12	164960967583	38
2	5	29	**377368163384**	5	**279860256751**	9
2	5	31	*	*	162260974199	10
2	5	37	**723072613064**	1	**568229492159**	3
2	5	41	*	*	**1088620718783**	1
2	5	47	*	*	**959710657655**	1
2	7	11	18638540036	144	19274727711	303
2	7	13	32415492836	43	16664272895	120
2	7	17	**432717685604**	12	61444342919	26
2	7	19	**612058391864**	3	67611298199	18
2	7	23	**249749066276**	1	**389853175871**	5

Continued on next page

Table 6.33 — continued from previous page

p_1	p_2	p_3	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
2	7	29	*	*	**695251729839**	1
2	11	13	**566959475396**	2	**263645533047**	7
2	11	17	*	*	**615708693399**	2
2	13	17	*	*	**801714897111**	1
2	13	19	*	*	**481734766679**	1
3	5	7	6890424056	1665	1475373743	3602
3	5	11	49957566964	183	4643885759	496
3	5	13	84831842696	82	13308756863	240
3	5	17	**278849168408**	19	60235736039	63
3	5	19	190022164916	10	108986280031	34
3	5	23	**703386940456**	3	148439200263	14
3	5	29	*	*	**300193517399**	5
3	5	31	*	*	**323714678543**	5
3	5	37	*	*	**999098015071**	1
3	7	11	42843308072	32	108652760399	90
3	7	13	127342860404	19	38986878143	40
3	7	17	*	*	107246624663	13
3	7	19	*	*	61164913211	5
3	7	23	*	*	**805192394183**	1
3	11	13	*	*	165908142839	4
5	7	11	*	*	**656450533751**	6
5	7	13	**786460186856**	1	110671542299	3
5	7	17	*	*	**658234953151**	1

Table 6.34: Quadruply non-cyclic class groups

p_1	p_2	p_3	p_4	First even $\lvert\Delta\rvert$	# even Δ	First odd $\lvert\Delta\rvert$	# odd Δ
2	3	5	7	**118507470392**	39	20777253551	127
2	3	5	11	**440057905316**	1	**532681210111**	4
2	3	5	13	*	*	**616825494863**	4

Chapter 7

CONCLUSION

In our work, we tabulated class groups of binary quadratic forms for negative discriminants Δ satisfying $|\Delta| < 2^{40}$. For $\Delta \not\equiv 1 \pmod 8$, we presented a novel unconditional class group tabulation approach which consists of two stages. In the first stage, we applied the technique of Hart et al. for multiplication of large polynomials to compute all class numbers [HTW10]. This allowed us to achieve a significant speedup of the second stage, which is the resolution of class groups with the Buchmann-Jacobson-Teske algorithm [BJT97]. We also applied the previous class group tabulation technique due to Jacobson et al. to compute all Cl_Δ for $\Delta \equiv 1 \pmod 8$ [JRW06]. Our computations showed that class groups with $\Delta \not\equiv 1 \pmod 8$ got computed over 4.72 times faster than class groups with $\Delta \equiv 1 \pmod 8$.

We used the data on class groups to provide extensive computational evidence in support of certain hypotheses, namely the Littlewood bounds and the heuristics of Cohen and Lenstra [Lit28, CL83, CL84]. We also partially confirmed the hypothesis of Euler and Gauß on idoneal numbers [Kan11]. That is, up to 2^{40} we did not find any occurrences of fundamental discriminants with $|\Delta| > 5460$ and the class group $Cl_\Delta \cong C(2) \times \ldots \times C(2)$. Finally, we further extended Buell's tables of exotic groups [Bue99], the last exhaustive computation of which was previously done to $2 \cdot 10^{11}$ by Ramachandran [Ram06].

The data produced by our program is presently stored on the penguin1 server, located at the University of Calgary, Alberta, Canada. In total, there is 1 Tb of data on class numbers $h(\Delta)$ with $\Delta \not\equiv 1 \pmod 8$, and 2.24 Tb of data on class groups Cl_Δ. To get access to this information, or inquire about certain statistics that you are interested in, please contact the author or his supervisor, Dr. Michael J. Jacobson, Jr., at `asmosuno@ucalgary.ca` or `jacobs@ucalgary.ca`, respectively.

7.1 Future Work

The discovery of a novel class number tabulation approach which we presented in this thesis allows us to state that the feasibility limit of our computations has not yet been reached, and it is at least possible to increase it by a factor of 2. However, the further speedup would most probably require a class number tabulation mechanism for $\Delta \equiv 1 \pmod 8$. Presently, no efficient class number tabulation formulas are known for this congruence class. One formula that might be of interest for future exploration is due to Humbert [Hum07, Wat35], who discovered that

$$\sum_{n=0}^{\infty} F(8n+7)q^n = S^{-1}(q) \sum_{n=1}^{\infty} (-1)^{n+1} n^2 \frac{q^{\frac{n(n+1)}{2}} - 1}{1+q^n}, \tag{7.1.1}$$

where

$$S(q) = \sum_{n=0}^{\infty} (-1)^n (2n+1) q^{\frac{n(n+1)}{2}} = 1 - 3q + 5q^3 - 7q^6 + 9q^{10} - \dots \ .$$

Despite its look, the large series on the right hand side of (7.1.1) does not take long to initialize, as it is easy to derive the formula for its n-th coefficient. The main problem lies in the computation of $S^{-1}(q)$, which can take significant amount of time, especially if $S(q)$ is of a high degree. Also, the coefficients of $S(q)$ grow very fast. For example, its 16-th coefficient has bit size 10, 64-th bit size 59, and 65536-th fits into 1135 bits. These coefficients have to be somehow truncated, for example, by reducing them modulo some prime p, such that $F(8n+7) < p$ for any $n < (N-7)/8$. However, this approach also brings certain difficulties, as it significantly increases the bit size parameter s. Despite all the obstacles, the usage of the formula (7.1.1) might still be faster than the conditional computation of $\Delta \equiv 1 \pmod 8$, followed by the verification procedure. In our research, we tried to use this approach, and in fact our implementation includes a subroutine `invert` for out-of-core polynomial inversion [Mos14]. This subroutine utilizes a Newton iteration algorithm [GG03, Algorithm 9.3], which performs the inversion of an arbitrary polynomial to degree $2^n - 1$ by sequentially computing its inverse to degrees $3, 7, \dots, 2^k - 1, \dots, 2^n - 1$ for $2 \le k \le n$. Each iteration in this algorithm requires one squaring of a polynomial and one polynomial multiplication. Unfortunately, we

were unable to produce class numbers using this method due to the number of difficulties previously mentioned.

Another potentially interesting approach that we would like to study in the near future is related to the solvability of the equation

$$n = x^2 + y^2 + z^2 \tag{7.1.2}$$

in rings different from $\mathbb{Z}$. Recall from Chapter 3 that we cannot tabulate class numbers $h(-n)$ for $n \equiv 7 \pmod 8$ because (7.1.2) has no solutions in $\mathbb{Z}$ for such n. However, it *does* have solutions, for example, in the ring $\mathbb{Z}\left[\sqrt{2}\right]$, as the following example demonstrates:

$$7 = \left(1 + \sqrt{2}\right)^2 + \left(1 - \sqrt{2}\right)^2 + 1^2.$$

We conjecture that the number of solutions $R_3(n)$ to (7.1.2) in $\mathbb{Z}\left[\sqrt{2}\right]$ is also somehow related to class numbers $h(-n)$ or $h(-4n)$, depending on the congruence class of n modulo 4. The following table demonstrates that such a connection seem to exist at least for squarefree $n \equiv 7 \pmod 8$:

Table 7.1: Values of $R_3(n)$ for squarefree $n \equiv 7 \pmod 8$

n	$R_3(n)$	$h(-n)$	Relation
7	96	1	$R_3 = 96 \cdot 1 \cdot h$
15	192	2	$R_3 = 96 \cdot 1 \cdot h$
23	288	3	$R_3 = 96 \cdot 1 \cdot h$
31	576	3	$R_3 = 96 \cdot 2 \cdot h$
39	384	4	$R_3 = 96 \cdot 1 \cdot h$
47	960	5	$R_3 = 96 \cdot 2 \cdot h$
55	1152	4	$R_3 = 96 \cdot 3 \cdot h$

n	$R_3(n)$	$h(-n)$	Relation
71	672	7	$R_3 = 96 \cdot 1 \cdot h$
79	960	5	$R_3 = 96 \cdot 2 \cdot h$
87	1728	6	$R_3 = 96 \cdot 3 \cdot h$
95	768	8	$R_3 = 96 \cdot 1 \cdot h$
103	2400	5	$R_3 = 96 \cdot 5 \cdot h$
111	2304	8	$R_3 = 96 \cdot 3 \cdot h$
119	1920	10	$R_3 = 96 \cdot 2 \cdot h$

If our conjecture is true, then coefficients of the series

$$\sum_{x,y,z \in \mathbb{Z}\left[\sqrt{2}\right]} q^{x^2+y^2+z^2} = \left(\sum_{x \in \mathbb{Z}\left[\sqrt{2}\right]} q^{x^2} \right)^3, \tag{7.1.3}$$

which correspond to integer powers of q, will allow us to derive a multiple of $h(-4n)$ or $h(-n)$. Of course, nothing restricts us from looking at binary quadratic forms with coefficients in

$\mathbb{Z}[\sqrt{k}]$ for an arbitrary positive integer k that is not a perfect square. Equations analogous to (7.1.3) can be deduced as well, and we suspect that these series would also count the number of equivalence classes $[(a, b, c)]$ of a certain determinant with $a, b, c \in \mathbb{Z}[\sqrt{k}]$. Aside from the fact that this relationship needs further exploration in order to be established or disproved, there exists one more difficulty, which is to multiply large polynomials in non-integer powers of q.

We also believe that the class group tabulation can be accelerated by using Sutherland's p-group discrete logarithm algorithms [Sut11]. The idea is simple: when the class number $h(\Delta)$ is known, instead of computing the whole group Cl_Δ we resolve the structure of each p-group separately. Sutherland's algorithms may be especially useful when resolving the structure of a 2-group, as we can precompute its 2-rank (by factoring Δ) and its 4-rank [Ste91, Proposition 2.2]. In some cases, the precomputation of a p-rank allows us to terminate the p-group resolution earlier by ignoring some of its generators of order p.

Finally, we would like to mention that the question of unconditional tabulation of class groups with positive Δ is still left open, and hopefully will soon to be explored by the new generation of algebraic number theory enthusiasts.

Bibliography

[AE06] J. V. Armitage, W. F. Eberlein, *Elliptic Functions*, Cambridge University Press, 2006.

[Bac90] E. Bach, *Explicit bounds for primality testing and related problems*, Mathematics of Computation 55, pp. 355 – 380, 1990.

[Bac95] E. Bach, *Improved approximations for Euler products*, Number Theory: CMS Proceedings 15, pp. 13 – 28. American Mathematical Society Providence, RI, 1995.

[Bel04] K. Belabas, *On quadratic fields with large 3-rank*, Mathematics of Computation 73, pp. 2061 – 2074, 2004.

[Bel10] D. H. Bellard, *Computation of 2700 billion decimal digits of pi using a desktop computer*, `http://bellard.org/pi/pi2700e9/pipcrecord.pdf`, 2010.

[Bel24] E. T. Bell, *The class number relations implicit in the Disquisitiones arithmeticae*, Bulletin of the American Mathematical Society 30 (5-6), pp. 236 – 238, 1924.

[Ber04] D. J. Bernstein, *Scaled remainder trees*, `http://cr.yp.to/arith/scaledmod-20040820.pdf`, 2004.

[BJT97] J. Buchmann, M. J. Jacobson, Jr., E. Teske, *On some computational problems in finite abelian groups*, Mathematics of Computation 66 (220), pp. 1663 – 1687, 1997.

[BS05] J. Buchmann, A. Schmidt, *Computing the structure of a finite abelian group*, Mathematics of Computation 74 (252), pp. 2017 – 2026, 2005.

[Bue76] D. A. Buell, *Class groups of quadratic fields*, Mathematics of Computation 30 (135), pp. 610 – 623, 1976.

[Bue87] D. A. Buell, *Class groups of quadratic fields. II*, Mathematics of Computation 48 (177), pp. 85 – 93, 1987.

[Bue99] D. A. Buell, *The last exhaustive computation of class groups of complex quadratic number fields*, Number theory, CRM Proceedings and Lecture Notes 19, American Mathematical Society, Providence, RI, pp. 35 – 53, 1999.

[BV07] J. Buchmann, U. Vollmer, *Binary Quadratic Forms. An Algorithmic Approach*, Springer-Verlag Berlin Heidelberg, 2007.

[Cha85] K. Chandrasekharan, *Elliptic Functions*, Springer-Verlag Berlin Heidelberg, 1985.

[Cho49] S. Chowla, *Improvement of a theorem of Linnik and Walfisz*, Proceedings of London Mathematics Society 50, pp. 423 – 429, 1949.

[CL83] H. Cohen, H. W. Lenstra, Jr., *Heurstics on class groups of number fields*, Number Theory, Lecture Notes in Mathematics 1068, Springer-Verlag, New York, pp. 33 – 62, 1983.

[CL84] H. Cohen, H. W. Lenstra, Jr., *Heuristics on class groups*, Number Theory, Noordwijkerhout, Lecture Notes in Mathematics 1052, Springer-Verlag, New York, pp. 26 – 36, 1984.

[CM05] A. C. Cojocaru, M. R. Murty, *An Introduction to Sieve Methods and Their Applications*, Cambridge University Press, 2005.

[Coh62] H. Cohn, *A Second Course in Number Theory*, John Wiley and Sons, New York, 1962.

[Coh93] H. Cohen, *A Course in Computational Algebraic Number Theory*, Springer-Verlag, Berlin, 1993.

[CP05] R. Crandall, C. Pomerance, *Prime Numbers. A Computational Perspective*, Springer Science + Business Media, Inc., 2nd edition, 2005.

[Dir63] P. G. L. Dirichlet, *Vorlesungen über Zahlentheorie*, 1863. Translated into English by J. Stillwell and reprinted, American Mathematical Society, 1999.

[DS05] F. Diamond, J. Shurman, *A First Course in Modular Forms*, Springer Science + Business Media, Inc., 2005.

[DyDSW79] F. Diaz y Diaz, D. Shanks, H. C. Williams, *Quadratic fields with 3-rank equal to 4*, Mathematics of Computation 33 (146), pp. 836 – 840, 1979.

[Gau98] C. F. Gauß, *Disquisitiones Arithmeticae*, 1798. Translated into English by A. A. Clarke and reprinted, Springer-Verlag, 1986.

[GG03] J. von zur Gathen, J. Gerhard, *Modern Computer Algebra*, Cambridge University Press, 2nd edition, 2003.

[Gro85] E. Grosswald, *Representation of Integers as Sums of Squares*, Springer-Verlag, New York, 1985.

[GS03] A. Granville, K. Soundararajan, *The distribution of values of* $L(1, \chi_d)$, Geometric and Functional Analysis 13 (5), pp. 992 – 1028, 2003.

[Har08] D. Harvey, *zn_poly library*, `http://web.maths.unsw.edu.au/~davidharvey/code/zn_poly/`, 2008.

[Har14] W. B. Hart, *FLINT: fast library for number theory, version 2.4.1*, `http://www.flintlib.org/`, 2014.

[HMcC89] J. Hafner, K. McCurley, *A rigorous subexponential algorithm for computation of class groups*, Journal of American Mathematical Society 2, pp. 837 – 850, 1989.

[HMcC91] J. Hafner, K. McCurley, *Asymptotically fast triangularization of matrices over rings*, SIAM Journal of Computation 20, pp. 1068 – 1083, 1991.

[HTW10] W. B. Hart, G. Tornaría, M. Watkins, *Congruent number theta coefficients to* 10^{12}, Algorithmic Number Theory — ANTS-IX (Nancy, France), Lecture Notes in Computer Science 6197, Springer-Verlag, Berlin, pp. 186 – 200, 2010.

[Hum07] P. Humbert, *Formules relatives aux nombres de classes des formes quadratiques binaries et positives*, Journal de mathématiques pures et appliquées 6 (3), pp. 337 – 449, 1907.

[Jac29] C. G. J. Jacobi, *Fundamenta Nova Theoriae Functionum Ellipticarum*, 1829.

[JRW06] M. J. Jacobson, Jr., S. Ramachandran, H. C. Williams, *Numerical results on class groups of imaginary quadratic fields*, Algorithmic Number Theory — ANTS-VII (Berlin, Germany), Lecture Notes in Computer Science 4076, Springer-Verlag, Berlin, pp. 87 – 101, 2006.

[JW09] M. J. Jacobson, Jr., H. C. Williams, *Solving the Pell Equation*, Springer Science + Business Media, LLC, 2009.

[Kan11] E. Kani, *Idoneal numbers and some generalizations*, Annales des Sciences Mathématiques du Québec 35, pp. 197 – 227, 2011.

[Kob84] N. Koblitz, *Introduction to Elliptic Curves and Modular Forms*, Springer-Verlag New York Inc., 1984.

[Kro60] L. Kronecker, *Über die Anzahl der verschiedenen classen quadratischer Formen von negativer Determinante*, Journal für die reine und angewandte Mathematik 57 (4), pp. 248 – 255, 1860.

[Kro62] L. Kronecker, *Über eine neue Eigenschaft der quadratischen Formen von negativer Determinante*, Monatsbericht der Akademie der Wissenschaften zu Berlin, May 26th, 1862.

[Lan18] E. Landau, *Über die Klassenzahl imaginär-quadratischer Zahlkörper*, Ges. Wiss. Göttingen, Math.-Phys., pp. 285 – 295, 1918.

[Lit28] J. E. Littlewood, *On the class number of the corpus* $P(\sqrt{-k})$, Proceedings of the London Mathematical Society 27, pp. 358 – 372, 1928.

[McC89] K. McCurley, *Cryptographic key distribution and computation in class groups*, Proceedings of NATO ASI Number Theory and Applications, Kluwer Academic Publishers, pp. 459 – 479, 1989.

[Mos14] A. S. Mosunov, *Class group tabulation program*, `https://github.com/amosunov`, 2014.

[Par14] The PARI Group, *PARI/GP, version 2.7.1*, `http://pari.math.u-bordeaux.fr/`, Bordeaux, 2014.

[Ram01] O. Ramaré, *Approximate formulae for* $L(1, \chi)$, Acta Arithmetica 100, pp. 245 – 266, 2001.

[Ram06] S. Ramachandran, *Class groups of quadratic fields*, Master's thesis, University of Calgary, Calgary, Alberta, 2006.

[Say13a] M. Sayles, *Improved arithmetic in the ideal class group of imaginary quadratic fields with an application to integer factoring*, Master's thesis, University of Calgary, Calgary, Alberta, 2013.

[Say13b] M. Sayles, ***optarith*** *and* ***qform*** *libraries for fast binary quadratic form arithmetic*, `https://github.com/maxwellsayles`, 2013.

[SBW94] R. Scheidler, J. Buchmann, H. C. Williams, *A key-exchange protocol using real quadratic fields*, Journal of Cryptography 7, pp. 171 – 199, 1994.

[Sch83] R. J. Schoof, *Class groups of complex quadratic fields*, Mathematics of Computation 41 (163), pp. 295 – 302, 1983.

[Sha71] D. Shanks, *Class number, a theory of factorization, and genera*, Proceedings of Symposia in Pure Mathematics 20, pp. 415 – 440, AMS, Providence, R.I., 1971.

[Sha73] D. Shanks, *Systematic examination of Littlewood's bounds on* $L(1, \chi)$, Proceedings of Symposia in Pure Mathematics, pp. 267 – 283, AMS, Providence, R.I., 1973.

[Sha89] D. Shanks, *On Gauss and composition I, II*, Number Theory and Applications, NATO ASI Series C 265, Kluwer, Dordrecht, 1989.

[Smi94] H. J. S. Smith, *Report on the theory of numbers*, Collected Mathematical Papers, Oxford University Press, 1894.

[Ste91] P. Stevenhagen, *The number of real quadratic fields having units of negative form*, Experimental Mathematics 2 (2), pp. 121 – 136, 1993.

[Sut11] A. V. Sutherland, *Structure computation and discrete logarithms in finite abelian p-groups*, Mathematics of Computation 80 (273), pp. 477 – 500, 2011.

[SvdV91] R. Schoof, M. van der Vlugt, *Hecke operators and the weight distributions of certain codes*, Journal of Combinatorial Theory 57, pp. 163 – 186, 1991.

[Ter99] D. C. Terr, *A modification of Shanks' baby-step giant-step algorithm*, Mathematics of Computation 69 (230), pp. 767 – 773, 1999.

[Wat03] M. Watkins, *Class numbers of imaginary quadratic fields*, Mathematics of Computation 73 (246), pp. 907 – 938, 2003.

[Wat35] G. N. Watson, *Generating functions of class numbers*, Compositio Mathematica 1, pp. 39 – 68, 1935.

[Wei73] P. Weinberger, *Exponents of the class groups of complex quadratic fields*, Acta Arithmetica 22, pp. 117 – 124, 1973.

[Wes14] Hungabee specification, `https://www.westgrid.ca/support/systems/Hungabee`, 2014.

[WW02] E. T. Whittaker, G. N. Watson, *A Course of Modern Analysis*, Cambridge University Press, Fourth Edition, 1902.