INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Application of Sieve Methods to Factorization and the Discrete Logarithm Problem

Antoine Khalil

A Thesis

in

The Department

of

Mathematics and Statistics

Presented in Partial Fulfilment of the Requirements for
the Degree of Master of Science at
Concordia University
Montréal, Québec, Canada

July 2002

Abstract

Application of Sieve Methods to factorization

and the Discrete Logarithm Problem

Antoine Khalil, M.Sc.

Concordia University, 2002

Factoring large numbers and computing discrete logarithms are presumed to be hard problems. No polynomial time solution for those problem has yet been found. Those problems have many significant applications, particularly in cryptography. Several cryptosystems base their security on their supposed difficulty. In this thesis we present some of the algorithms to solve these two problems. We mainly explore sieving as a tool for that purpose. Among other material we describe the Quadratic Sieve and the Number Field Sieve as they apply to factoring. We finaly sketch how the Number Field Sieve can be applied to compute discrete logarithms.

Table of Contents

Introduction

We begin with a comparison between the trial division and sieving methods to find primes. We demonstrate that sieving is much faster than trial division.

Chapter Two contains some mathematical preliminaries.

In Chapter Three we start exploring methods to factor large integers. First we present Kraitchik's method, the precursor of several subsequent algorithms (also see [Kraitchik, 1922, 1924]). In this section too, we present a detailed numerical example applying Kraitchik's method. The Maple program we used for that example is in Appendix B. Second we sketch the Quadratic Sieve. We show how sieving can bring a major speedup to Kraitchik's method. Another program implementing Quadratic Sieve is also found in Appendix B.

In Chapter Four we describe the Number Field Sieve. We present a simple way of generating the polynomial and talk about some possible alternatives. We also sketch how to find the rational and the algebraic square. In the later case we expose the complicating factors and present possible solutions. We mention the square root step and the running time of the algorithm though we do not present those in details. In the last section we present a detailed numerical example applying the Number Field Sieve.

In Chapter Five we focus on the Discrete Logarithm Problem. We mention some of its applications in cryptography. As for the Sieving approach we present it in section 5.2. We do not go into the details of this approach, we rather describe its main steps. For a detailed reference on this last section we refer the reader to [Weber, 1997].

Acknowledgements

First I would like to thank my supervisor Dr. David Ford. I consider myself very lucky to be working with him. His experience and valuable guidance made my Masters thesis a highly enjoyable journey.

Second, thank you to Dr. Nasser Saad who convinced me that I should continue my graduate studies and guided my first steps into my Masters. Without Nasser's help I would have never been at Concordia.

My third thank you is to Dr. Sebastian Pauli whose constant help and advice made my life so much easier. I also would like to thank him for his careful reading of my thesis.

Fourth I would like to thank all those people who were there for me past and present and without whom I could have never been where I am. Thank you to:

- Dr. Richard Hall
- Father Georges Khalil (M.M.O.),
- Father Francois Eid (M.M.O),
- Ms. Amale Khalil,
- Dr. Jean Fares,
- Mr. Antoine Saber,
- Mr. Youssef Semaan and Mr. Semaan Semaan,

• and many others who's names cannot be listed due to the lack of space.

Finally, my deepest thank you goes to my family: Charbel, Najat, Joseph, Ghassan, Chantalle, and Christine. There are not enough words to thank you. You are the best!

Chapter 1

The Power of Sieving

1.1 Introduction

Many of the most effective known algorithms for integer factorization and discrete logarithm computation are based on sieving methods. In a typical sieving application a (comparatively small) subset is to be extracted from a (comparatively large) interval of integers. A sieving approach is appropriate when the values to be eliminated (or retained) can be expressed as "incommensurable" linear subsequences of the original interval.

As an example, we will compare two methods of computing the set of primes up to the integer n. The first is a naive method based on trial division. The second uses sieving: as each successive prime p is discovered, the sequence of values $2p, 3p, \ldots$ is eliminated from further consideration.

1.2 Generating Primes by Trial Division

A simple way to generate the prime numbers between 1 and n is by *trial division.*

Algorithm 1.2.1 (Trial Division).

Input: array of "unmarked" numbers 1 up to n

Output: prime numbers up to n

- Mark 1
- **for** k **from** 2 **to** n **do**
  - $j \leftarrow 2$
  - **while** $j \leqslant \sqrt{k}$ **and** k unmarked **do**
    - **if** $j \mid k$ **then** mark k
    - $j \leftarrow j + 1$
  - **od**:
- **od**:
- Unmarked numbers are the primes.

For purposes of comparison we will need a lower bound on the running time of the Trial Division algorithm.

Let T_D be number of steps required for the Trial Division algorithm to terminate. Each prime value of k requires at least $C_1\sqrt{k}$ steps, for some positive constant C_1. Applying Lemma A.0.3 of Appendix A, it follows that

$$T_D \geqslant \sum_{\substack{k \text{ prime} \\ k \leqslant n}} C_1\sqrt{k} \geqslant C_2 \frac{n\sqrt{n}}{\ln n}$$

for some positive constant C_2.

1.3 Generating Primes with the Sieve of Eratosthenes

A far more efficient method to find the prime numbers in the interval $1 \leqslant p \leqslant n$ is the Sieve of Eratosthenes.

Algorithm 1.3.1 (Sieve of Eratosthenes).

Input: array of "unmarked" numbers 1 up to n

Output: prime numbers up to n

- Mark 1

- Initialise $k \leftarrow 2$
- **while** $k \leqslant \sqrt{n}$ **do**
  - Mark every k^{th} number $(2k, 3k, 4k, \ldots, \lfloor n/k \rfloor k)$.
  - $k \leftarrow$ next unmarked number.
  - **od**:
- Unmarked numbers are the primes.

We will determine an upper bound for T_S, the number of steps required for Sieve to terminate.

The number of initialisation steps is bounded, say by C_0. In the outer loop, each prime value of k requires at most $C_3\lfloor n/k \rfloor$ steps and each composite value of k requires at most C_4 steps, for some positive constants C_3, C_4.

Remark 1.3.2. This is already enough to show that the Sieve is superior. In the outer loop, the Sieve ignores composite values of k, whereas trial division performs at least one, and possibly several, tests. For the prime values of k, the Sieve will be faster when $C_3\lfloor n/k \rfloor \leqslant C_3 n/k \leqslant C_1\sqrt{k}$, and this is the case for all $k \geqslant (C_3/C_1)^{2/3}\, n^{2/3}$.

To estimate the total running time we apply [Hardy & Wright, 1960], Theorem 427 (see Appendix A, Theorem A.0.4), which yields

$$\begin{aligned} T_S &\leqslant C_0 + \sum_{\substack{k \text{ prime} \\ k \leqslant n}} C_3 \left\lfloor \frac{n}{k} \right\rfloor + \sum_{\substack{k \text{ composite} \\ k \leqslant n}} C_4 \\ &\leqslant C_0 + \sum_{\substack{k \text{ prime} \\ k \leqslant n}} C_3 \frac{n}{k} + C_4 n \\ &= C_0 + C_3 n \sum_{\substack{k \text{ prime} \\ k \leqslant n}} \frac{1}{k} + C_4 n \\ &\leqslant C_0 + C_5 n \ln\ln n + C_4 n \\ &\leqslant C_6 n \ln\ln n \end{aligned}$$

for some positive constants C_5, C_6.

The superiority of sieving is apparent when we compute the ratio

$$\frac{T_S}{T_D} \leqslant C \frac{\ln n \, \ln \ln n}{\sqrt{n}}$$

with $C = C_6/C_2 > 0$.

Chapter 2

Mathematical Preliminaries

2.1 Algebraic Number Fields

Definition 2.1.1. A complex number α is called an *algebraic number* if it satisfies some polynomial equation $f(x) = 0$ where $f(x)$ is a polynomial over $\mathbb{Q}$.

Theorem 2.1.2. *Every algebraic number α satisfies a unique irreducible monic polynomial equation $g(x) = 0$ over $\mathbb{Q}$. Moreover, if $f(x)$ is any polynomial over $\mathbb{Q}$ such that $f(\alpha) = 0$ then $f(x)$ is divisible by $g(x)$.*

Definition 2.1.3. The *minimal equation* of the algebraic number α is the equation $g(x) = 0$ described in theorem 2.1.2. The polynomial $g(x)$ is the *minimal polynomial* of α. The *degree* of an algebraic number is the degree of its minimal polynomial.

Definition 2.1.4. An *algebraic number field* is an extension of $\mathbb{Q}$ by an algebraic number. If α is an algebraic number then the *degree* of the algebraic number field $\mathbb{Q}(\alpha)$ is the degree of the minimal polynomial of α (over $\mathbb{Q}$).

Theorem 2.1.5. *If K is an algebraic number field of degree n then K is a vector space of dimension n over $\mathbb{Q}$.*

Definition 2.1.6. An element of an algebraic number field is *integral* if the coefficients of its minimal polynomial all lie in $\mathbb{Z}$. The set of all integral elements of the algebraic number field K is denoted by $\mathcal{O}_K$.

Theorem 2.1.7. *If K is an algebraic number field of degree n then $\mathcal{O}_K$ is an order, i.e., a commutative ring with 1 that is also a $\mathbb{Z}$-module of rank n.*

2.1.1 The Conjugates of an Algebraic Number

Definition 2.1.8. Let α be an algebraic number of degree n and let $f(x)$ be the minimal polynomial of α. Then the (necessarily distinct) roots $\alpha_1, \alpha_2, \ldots, \alpha_n$ of $f(x)$ are called the *conjugates* of α.

2.1.2 The Relative Norm of an Algebraic Number

Definition 2.1.9. Let α be an algebraic number of degree n, let $K = \mathbb{Q}(\alpha)$ be the algebraic number field generated by α, and let $\alpha_1, \alpha_2, \ldots, \alpha_n$ be the conjugates of α. If $\beta \in K$ then the *norm of β (with respect to the extension $K/\mathbb{Q}$)* is defined to be

$$N(\beta) = N_{K/\mathbb{Q}}(\beta) = h(\alpha_1)h(\alpha_2)\cdots h(\alpha_n)$$

where $h(x) \in \mathbb{Q}[x]$ and $\beta = h(\alpha)$.

2.1.3 Ideals of $\mathbb{Z}[\alpha]$

We assume the following.

- $f(x)$ is an irreducible monic polynomial in $\mathbb{Z}[x]$.
- α is a root of $f(x)$.
- $K = \mathbb{Q}(\alpha)$, the extension of $\mathbb{Q}$ generated by α.

Remark 2.1.10. It follows from these assumptions that $\mathbb{Z}[\alpha] \subseteq \mathcal{O}_K$.

Definition 2.1.11. A subset $\mathfrak{a}$ of $\mathbb{Z}[\alpha]$ is an *ideal* of $\mathbb{Z}[\alpha]$ if:

- $\mathfrak{a}$ is closed under addition, and

- $ax \in \mathfrak{a}$ for all $x \in \mathbb{Z}[\alpha]$ and $a \in \mathfrak{a}$.

Definition 2.1.12. An ideal $\mathfrak{a}$ of $\mathbb{Z}[\alpha]$ is a *principal ideal* if $\mathfrak{a}$ is of the form

$$a\mathbb{Z}[\alpha] = \{\, xa : x \in \mathbb{Z}[\alpha] \,\}.$$

Definition 2.1.13. An ideal $\mathfrak{a} \subsetneq \mathbb{Z}[\alpha]$ is a *maximal ideal* if

$$\mathfrak{a} \subseteq \mathfrak{b} \subseteq \mathbb{Z}[\alpha] \implies \mathfrak{b} = \mathbb{Z}[\alpha] \text{ or } \mathfrak{b} = \mathfrak{a}$$

for every ideal $\mathfrak{b}$.

Definition 2.1.14. An ideal $\mathfrak{a} \neq \mathbb{Z}[\alpha]$ with the property

$$rs \in \mathfrak{a} \implies r \in \mathfrak{a} \text{ or } s \in \mathfrak{a}$$

for all $r, s \in \mathbb{Z}[\alpha]$ is called a *prime ideal.*

Definition 2.1.15. A *degree one prime ideal* in $\mathbb{Z}[\alpha]$ is an ideal $\mathfrak{p}$ in $\mathbb{Z}[\alpha]$ such that

$$\mathbb{Z}[\alpha]/\mathfrak{p} \cong \mathbb{F}_p$$

for some prime number p.

Definition 2.1.16. For p a prime number and $r \in \mathbb{Z}$ we define

$$\mathfrak{a}_{p,r} = p\mathbb{Z}[\alpha] + (\alpha - r)\mathbb{Z}[\alpha].$$

Remark 2.1.17. It is clear that $\mathfrak{a}_{p,r}$ is an ideal in $\mathbb{Z}[\alpha]$.

Proposition 2.1.18. *If $f(r) \equiv 0 \bmod p$ then $\mathfrak{a}_{p,r}$ is a degree one prime ideal in $\mathbb{Z}[\alpha]$, and if $f(r) \not\equiv 0 \bmod p$ then $\mathfrak{a}_{p,r} = \mathbb{Z}[\alpha]$.*

Proof. Let

$$f(x) = (x - r)g(x) + s$$

with $g(x) \in \mathbb{Z}[x]$ and $s \in \mathbb{Z}$. Then $s = f(r)$ and

$$0 = f(\alpha) = (\alpha - r)g(\alpha) + f(r),$$

and hence $f(r) = -(\alpha - r)g(\alpha) \in \mathfrak{a}_{p,r}$.

Assume $f(r) \not\equiv 0 \bmod p$. Then $1 = \gcd\big(p, f(r)\big) \in \mathfrak{a}_{p,r}$, which implies $\mathfrak{a}_{p,r} = \mathbb{Z}[\alpha]$.

Assume $f(r) \equiv 0 \bmod p$. Since $\alpha - r \in \mathfrak{a}_{p,r}$ and $p \in \mathfrak{a}_{p,r}$, it follows that

$$\mathbb{Z}[\alpha]/\mathfrak{a}_{p,r} = \{\, 0 + \mathfrak{a}_{p,r},\ 1 + \mathfrak{a}_{p,r},\ \ldots,\ (p-1) + \mathfrak{a}_{p,r} \,\}.$$

Suppose these cosets were not distinct, say $u + \mathfrak{a}_{p,r} = v + \mathfrak{a}_{p,r}$ with $0 \leqslant u < v \leqslant p - 1$. Then $u - v \in \mathfrak{a}_{p,r}$, and we would have

$$u - v = p\,h(\alpha) + (\alpha - r)k(\alpha)$$

for some $h(x), k(x) \in \mathbb{Z}[x]$. It would follow that

$$u - v = p\,h(x) + (x - r)k(x) + q(x)f(x)$$

for some $q(x) \in \mathbb{Z}[x]$ and therefore

$$u - v = p\,h(r) + (r - r)k(r) + q(r)f(r) \equiv 0 \bmod p,$$

which is a contradiction. Therefore $|\mathbb{Z}[\alpha]/\mathfrak{a}_{p,r}| = p$, hence $\mathbb{Z}[\alpha]/\mathfrak{a}_{p,r} \cong \mathbb{F}_p$, and $\mathfrak{a}_{p,r}$ is a degree one prime ideal in $\mathbb{Z}[\alpha]$. □

Definition 2.1.19. For each prime p we define

$$R(p) = \{\, r \in \{\, 0, 1, \ldots, p-1 \,\} : f(r) \equiv 0 \bmod p \,\}.$$

Proposition 2.1.20. *If $\mathfrak{p}$ is a degree one prime ideal in $\mathbb{Z}[\alpha]$ then $\mathfrak{p} = \mathfrak{a}_{p,r}$ for some prime number p and some $r \in R(p)$.*

Proof. Let $\mathfrak{p}$ be a degree one prime ideal in $\mathbb{Z}[\alpha]$. Then $\mathbb{Z}[\alpha]/\mathfrak{p} \cong \mathbb{F}_p$ for some prime number p, so that

$$\mathbb{Z}[\alpha]/\mathfrak{p} = \{\, 0 + \mathfrak{p},\ 1 + \mathfrak{p},\ \ldots,\ (p-1) + \mathfrak{p} \,\}.$$

It follows that there is a unique r with $0 \leqslant r \leqslant p-1$ such that $\alpha + \mathfrak{p} = r + \mathfrak{p}$. Thus $\alpha - r \in \mathfrak{p}$, and it is clear that $p \in \mathfrak{p}$, and hence we have

$$\mathfrak{a}_{p,r} = p\mathbb{Z}[\alpha] + (\alpha - r)\mathbb{Z}[\alpha] \subseteq \mathfrak{p} \subsetneq \mathbb{Z}[\alpha].$$

Since $\mathfrak{a}_{p,r} \neq \mathbb{Z}[\alpha]$ it follows from proposition 2.1.18 that $\mathfrak{a}_{p,r}$ is a degree one prime ideal, hence $\mathfrak{a}_{p,r} = \mathfrak{p}$, and moreover $f(r) \equiv 0 \bmod p$, hence $r \in R(p)$. □

2.2 The Legendre Symbol

Definition 2.2.1. The *Legendre Symbol* is the group homomorphism

$$\left(\frac{\cdot}{p}\right) : \mathbb{F}_p^* \to \{\pm 1\},$$

where if $a \in \mathbb{F}_p^*$,

$$\left(\frac{a}{p}\right) = \begin{cases} +1 & \text{if } a \text{ is a quadratic residue mod } p. \\ -1 & \text{otherwise.} \end{cases}$$

2.3 Running Time Analysis

Definition 2.3.1. Let $f(x)$ and $g(x)$ be two functions with positive range, and n_0 be a positive constant. Suppose that for all $x \geqslant n_0$, $f(x) \leqslant Cg(x)$ for some constant C, then we say that $f(x) = O(g(x))$.

Definition 2.3.2. If the two functions $f(x)$ and $g(x)$ mentionned above satisfy

$$lim_{x \to \infty} \frac{f(x)}{g(x)} = 0,$$

then we say $f(x) = o(g(x))$.

Definition 2.3.3. An algorithm is said to run in *polynomial time* if there exists an integer d such that the number of bit operations required for the algorithm to terminate is bounded $O(k^d)$, where k is the length of the input.

Definition 2.3.4. Similarly to the above definition, we say an algorithm runs in *exponential time* if the number of bit operations required is $O(e^{ck})$

One useful way to classify time estimates can be given using the following definition:

Definition 2.3.5. Let n be a large positive integer (usually the input of the algorithm), $0 \leqslant \gamma \leqslant 1$ be a real number, and $c > 0$ be a constant. Define:

$$L_n(\gamma; c) = O(e^{c(\ln n)^{\gamma}(\ln \ln n)^{1-\gamma}}).$$

An algorithm is said to be an $L(\gamma)-$ algorithm if its running time is of the form $L(\gamma; c)$.

Note that, if $\gamma = 0$ we have, $L(0; c) = O((\ln n)^c)$, *i.e.*, polynomial time. And if $\gamma = 1$, $L(1; c) = O(n^c)$, *i.e.*, exponential time.

Definition 2.3.6. We say an algorithm runs in *subexponential time* if this is an $L(\gamma)-$algorithm for $0 < \gamma < 1$.

Chapter 3

The Quadratic Sieve

3.1 Kraitchik's Method

3.1.1 Introduction

Let $n = a \cdot b$ be an odd composite number. We can write n as a difference of squares in the following way:

$$n = \left(\frac{a+b}{2}\right)^2 - \left(\frac{a-b}{2}\right)^2.$$

The idea of writing n in this form, namely finding a pair (u, v) such that

$$n = u^2 - v^2,$$

is due to Fermat. For a general number n this method is no better than the trial division algorithm.

In the 1920s Maurice Kraitchik came up with an enhancement of Fermat's method [Kraitchik, 1922], [Kraitchik, 1924]. His idea is the basis of most of the modern factoring algorithms. Kraitchik's idea was to find a pair (u, v) such that:

$$u^2 \equiv v^2 \bmod n. \tag{3.1}$$

This idea has roots in the work of Gauss and Seelhoff but it was Kraitchik who brought it out of the shadows. For a more complete reference on the history of factoring

we refer the reader to [H.C. Williams, J.O. Shallit, 1994], [Pomerance, 1996], and [Montgomery, 1994b].

Congruence (3.1) leads to two types of solutions:

$$u \equiv \pm v \bmod n,$$

$$u \not\equiv \pm v \bmod n.$$

The first pair of solutions is not of interest for the factorization of n. But if we consider the second pair, we can see that $\gcd(u-v, n)$ is a nontrivial factor of n. The remaining part of this section deals with finding a pair of integers (u, v) that satisfies congruence (3.1).

The algorithm starts by considering the sequence $Q(x) = x^2 - n$ for $x = x_1, x_2, \ldots$. We then look for a subsequence of $\{x_i\}$ such that the product of the corresponding $Q(x)$'s is a square in $\mathbb{Z}$. Without loss of generality we rewrite the terms of such a subsequence as $x_1, x_2, \ldots, x_k$ i.e.,

$$\prod_{i=1}^{k} Q(x) = v^2,$$

and let

$$u = x_1 x_2 \cdots x_k.$$

Then:

$$\begin{aligned} u^2 &= x_1^2 x_2^2 \cdots x_k^2 \\ &\equiv (x_1^2 - n)(x_2^2 - n) \cdots (x_k^2 - n) \\ &= Q(x_1)Q(x_2) \cdots Q(x_k) \\ &= v^2 \bmod n \end{aligned}$$

and thus we found a solution for (3.1). The question remains: how to find such a subsequence?

3.1.2 Billhart and Morrison's Strategy

A systematic strategy for finding the subsequence $x_1, x_2, \ldots, x_k$ was found by John Brillhart and Michael Morrison [M.A. Morrison, 1975].

Definition 3.1.1. Let y be a positive real number. The positive integer a is said to be *y-smooth* if a has no prime factor greater than y.

Definition 3.1.2. Let y be a fixed smoothness bound. The set

$$B = \{\, p : p \text{ prime}, p \leqslant y, n \text{ a quadratic residue mod } p \,\} \cup \{-1\}$$

is called the *factor base (with respect to n and y)*.

Definition 3.1.3. For every y-smooth number m we associate an exponent vector

$$v(m) = (v_1, v_2, \ldots, v_{|B|})$$

where $m = \prod_{p_i \in B} p_i^{v_i}$.

The following algorithm finds the subsequence $x_1, x_2, \ldots, x_k$.

Algorithm 3.1.4 (Brillhart and Morrison).

Input: n (to be factored); $y \in \mathbb{Z}$ (smoothness bound); original sequence $\{x_i\}$

Output: A sequence of pairs (u, v), each satisfying congruence (3.1)

1. Form the factor base B.
2. Search the sequence $\{Q(x)\}$ for y-smooth $Q(x)$'s;
   gather at least $|B| + 1$ of them.
3. Form $v(m)$ for each y-smooth m.
4. Linear dependency mod 2 gives a sequence of pairs satisfying congruence (3.1)
   (see algorithm 3.1.6).

Remark 3.1.5. In step 2 of the algorithm, one could look at small arrays and keep track of the size of B. We stop when $|B|$ is sufficiently large. Since we have about a

50% chance of getting a non-trivial factorization of n from each pair (u, v) and since $|B| + 1$ vectors in $\mathbb{F}_2^{|B|}$ must be linearly dependent, we can guarantee finding at least one relation by gathering at least this many y-smooth numbers. We can increase our chances of getting a non-trivial factorization by gathering more y-smooth numbers.

We will discuss step 4 in an algorithm by itself since it is needed in future sections.

Algorithm 3.1.6 (Linear Algebra Step).

Input: y; B; the sequence $S_1, \ldots, S_{N_v}$ of y-smooth values of $Q(x)$, with $N_v \geqslant |B| + 1$; the N_v exponent vectors $v(m)$ reduced mod 2

Output: A sequence of pairs $(u_1, v_1), \ldots, (u_{N_r}, v_{N_r})$, each pair satisfying congruence (3.1)

- The matrix V is formed by arranging the exponent vectors $v(m)$ in an $N_v \times |B|$ array and adjoining the $N_v \times N_v$ identity matrix on the right.
- The matrix V is row-reduced over $\mathbb{F}_2$.
- Let N_r be the number of rows of V in which the first $|B|$ entries are all zero. The rows in which not all the first $|B|$ entries are zero are now discarded, and in what remains the first $|B|$ (empty) columns are suppressed. The result is the $N_r \times N_v$ submatrix M.
- **for** j **from** 1 **to** N_r **do**
  - $u_j \leftarrow 1$; $w_j \leftarrow 1$:
  - **for** k **from** 1 **to** N_v **do**
    - **if** $M_{j,k} = 1$ **then**
      - $x_j \leftarrow \sqrt{S_k + n}$
      - $u_j \leftarrow u_j \cdot x_j$
      - $w_j \leftarrow w_j \cdot S_k$
    - **fi**:
  - **od**:
  - $v_j \leftarrow \sqrt{w_j}$

• **od:**

Having found a solution (u, v) for (3.1), we only need to check that $u \not\equiv \pm v \bmod n$, and in that case compute $\gcd(u - v, n)$ to get a nontrivial factor of n.

Remark 3.1.7. Concerning the first step of algorithm 3.1.6 and in accordance with remark 3.1.5, if we decide to gather k extra y-smooth numbers ($k > 1$) then we form a matrix with $N_v = |B| + k$.

Remark 3.1.8. The best algorithm known for reduction mod 2 is Block Lanczos (see [Montgomery, 1995]).

3.1.3 Running Time for Kraitchik's Method

It is conjectured (see Pomerance [1996] and references cited there) that factoring n via the Kraitchik polynomial method should take

$$O\left(\exp\left(\sqrt{2 \ln n \ln \ln n}\right)\right) = L_n(1/2; \sqrt{2})$$

steps; *i.e.*, Kraitchik's algorithm is an $L(1/2)$-algorithm. This claim was first made by Richard Schroeppel in unpublished work in the late 1970s.

3.1.4 Numerical Example of Kraitchik's Method

The program with which we computed this example is in Appendix B.

• Let $n = 34121$, and set $y = 50$.

• We form the factor base

$$B = \{ -1, 2, 5, 13, 17, 19, 23, 37, 41 \}.$$

Each member of B (except -1) is a prime p satisfying

$$\left(\frac{n}{p}\right) = +1.$$

• We gather $|B| + 11$ y-smooth values[1] of $Q(x)$ and put them in the set S.

$x = \lfloor\sqrt{n}\rfloor \pm r$	$Q(x) = x^2 - n$
$186 = 185 + 1$	$475 = 5^2 \cdot 19$
$189 = 185 + 4$	$1600 = 2^6 \cdot 5^2$
$193 = 185 + 8$	$3128 = 2^3 \cdot 17 \cdot 23$
$194 = 185 + 9$	$3515 = 5 \cdot 19 \cdot 37$
$198 = 185 + 13$	$5083 = 13 \cdot 17 \cdot 23$
$181 = 185 - 4$	$-1360 = -1 \cdot 2^4 \cdot 5 \cdot 17$
$179 = 185 - 6$	$-2080 = -1 \cdot 2^5 \cdot 5 \cdot 13$
$176 = 185 - 9$	$-3145 = -1 \cdot 5 \cdot 17 \cdot 37$
$175 = 185 - 10$	$-3496 = -1 \cdot 2^3 \cdot 19 \cdot 23$
$205 = 185 + 20$	$7904 = 2^5 \cdot 13 \cdot 19$
$211 = 185 + 26$	$10400 = 2^5 \cdot 5^2 \cdot 13$
$213 = 185 + 28$	$11248 = 2^4 \cdot 19 \cdot 37$
$167 = 185 - 18$	$-6232 = -1 \cdot 2^3 \cdot 19 \cdot 41$
$164 = 185 - 21$	$-7225 = -1 \cdot 5^2 \cdot 17^2$
$161 = 185 - 24$	$-8200 = -1 \cdot 2^3 \cdot 5^2 \cdot 41$
$159 = 185 - 26$	$-8840 = -1 \cdot 2^3 \cdot 5 \cdot 13 \cdot 17$
$157 = 185 - 28$	$-9472 = -1 \cdot 2^8 \cdot 37$
$221 = 185 + 36$	$14720 = 2^7 \cdot 5 \cdot 23$
$224 = 185 + 39$	$16055 = 5 \cdot 13^2 \cdot 19$
$227 = 185 + 42$	$17408 = 2^{10} \cdot 17$

$$S = \{\, 475, 1600, 3128, 3515, 5083, -1360, -2080, -3145, -3496, 7904,$$
$$10400, 11248, -6232, -7225, -8200, -8840, -9472, 14720, 16055, 17408 \,\}$$

[1] in this case, $k = 11$ in remark 3.1.7

- We then form the exponent vectors $v(m)$ for each $m \in S$ and construct the matrix V, reduced modulo 2, as shown below.

  The matrix V has $|S|$ rows and $|B| + |S|$ columns. In each row the first $|B|$ entries give the exponent vector $v(Q(x))$. The remaining columns form an $|S| \times |S|$ identity matrix, each column of which is associated with a member of S.

$$V = \begin{pmatrix}
\cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & 1 & \cdot & \cdot & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & 1 & \cdot & \cdot & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & 1 & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & \cdot & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & 1 & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & \cdot & 1 & \cdot & 1 & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & 1 & \cdot & \cdot & \cdot & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & 1 & \cdot & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & 1 & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & 1 & 1 & 1 & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot \\
1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\
\cdot & 1 & 1 & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot \\
\cdot & \cdot & 1 & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1
\end{pmatrix}$$

- We row-reduce V over $\mathbb{F}_2$, then discard all rows with any nonzero entries among the first $|B|$ columns. The remaining rows (with the first $|B|$ columns suppressed) give the matrix M, below.

$$M = \begin{pmatrix}
1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\
\cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & \cdot & \cdot & 1 & 1 & 1 \\
\cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & 1 & \cdot & 1 & \cdot \\
\cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & \cdot & 1 & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & 1 & \cdot & \cdot & 1 & 1 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & 1 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & \cdot & 1 & \cdot & 1 & 1 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & \cdot & 1 & 1 & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & \cdot & 1 & \cdot & 1 & \cdot & \cdot & 1 & 1 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & \cdot & 1 & 1 & 1 & 1 & \cdot & \cdot & 1 & 1 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & 1 & 1 & 1 & \cdot & 1 & \cdot & \cdot & \cdot
\end{pmatrix}$$

- The matrix M contains 12 relations, from which the following factorizations of n are obtained:

$$n = 229 \cdot 149 \qquad (7 \text{ out of } 12 \text{ trials});$$

$$n = 1 \cdot 34121 \qquad (5 \text{ out of } 12 \text{ trials}).$$

3.2 The Quadratic Sieve

3.2.1 The Heart of the Quadratic Sieve

The Sieve of Eratosthenes (section 1.3) can be modified to find the y-smooth values in a finite arithmetic sequence. Instead of identifying the primes, one could look at the numbers that have many marks. Intuitively, there should be a correlation between having many prime factors and having all small prime factors. In 1981, Pomerance had the idea of applying the Sieve of Eratosthenes in order to find the y-smooth values of the polynomial $Q(x) = x^2 - n$.

Sieving gains its speed by not considering sequentially all the numbers in a certain array. In every step we only consider the entries in the array that are separated by a distance k. The key to apply sieving to the array with entries $Q(x)$ is to be able to decide: given a prime p, for which values of x does p divide $Q(x)$?

Given n, the number to be factored, and p, a prime such that n is a quadratic residue mod p, there exit two residue classes $[a]$ and $[-a]$, satisfying:

$$Q(x) \equiv 0 \bmod p \iff x \equiv \pm a \bmod p.$$

Thus we only have to check the values of $Q(x)$ for $x = kp \pm a$ with $k \in \mathbb{Z}$.

Once we have sieved for all p in B and have gathered enough smooth $Q(x)$'s we apply algorithm 3.1.6 to factor n.

3.2.2 Running Time of the Quadratic Sieve

It is conjectured (see [Pomerance, 1996] and [A.K. Lenstra, H.W. Lenstra Jr., 1993] pp. 76–78) that the Quadratic Sieve should take

$$O\left(\exp\left(\sqrt{\ln n \ln \ln n}\right)\right) = L_n(1/2; 1)$$

steps to factor the number n.

Chapter 4

The Number Field Sieve

4.1 Introduction

Assume the following.

- n is the (odd, composite) number to be factored.
- $f(x)$ is an irreducible monic polynomial with coefficients in $\mathbb{Z}$.
- The discriminant of f is relatively prime to n.
- α is a root of f, *i.e.*, $f(\alpha) = 0$.
- m is a positive integer such that $f(m) \equiv 0 \bmod n$.
- $K = \mathbb{Q}(\alpha)$, the extension of $\mathbb{Q}$ generated by α.
- $\mathcal{O}_K$ is the ring of integers of K.
- $\varphi : \mathcal{O}_K \to \mathbb{Z}/n\mathbb{Z}$ is the ring homomorphism defined by $\varphi(h(\alpha)) = h(m) \bmod n$.

The general idea behind the NFS is to find a finite set S of pairs of relatively prime integers such that

$$\prod_{(a,b)\in S} (a - bm) = v^2 \text{ for some } v \in \mathbb{Z}, \text{ and} \tag{4.1}$$

$$\prod_{(a,b)\in S} (a - b\alpha) = \gamma^2 \text{ for some } \gamma \in \mathcal{O}_K. \tag{4.2}$$

Suppose $u \in \mathbb{Z}$ with $\varphi(\gamma) \equiv u \bmod n$; then

$$u^2 \equiv \varphi^2(\gamma) = \varphi(\gamma^2) = \varphi\Big(\prod_{(a,b)\in S}(a - b\alpha)\Big) = \prod_{(a,b)\in S}(a - bm) = v^2 \bmod n.$$

Having thus found a pair (u, v) satisfying $u^2 \equiv v^2 \bmod n$ we expect to have at least a 50% chance that $\gcd(u - v, n)$ is a nontrivial factor of n.

Remark 4.1.1. Several questions arise:

- How do we construct the polynomial $f(x)$?
- How do we find m?
- How is a set S satisfying (4.1) and (4.2) to be found?
- How can we find $\gamma \in \mathcal{O}_K$ such that $\gamma^2 = \prod_{(a,b)\in S}(a - b\alpha)$?
- How fast is this algorithm?

4.2 Finding the Polynomial

4.2.1 Introduction

The first step of the NFS is to find a polynomial f with integer coefficients and an integer m such that

$$f(m) \equiv 0 \bmod n,$$

with n being the number we want to factor.

Remark 4.2.1. Suppose that the polynomial f is found to be reducible, say

$$f(x) = g(x) \cdot h(x).$$

This in fact leads to a factorization of n see [A.K. Lenstra, H.W. Lenstra Jr., 1993] pp. 54, [J. Brillhart, M. Filaseta, A. Odlyzko, 1981]. In case the factorization is trivial, say $g(m)$ is divisible by n, then we replace f and d by g and $\deg g$ respectively.

This replacement improves the algorithm by reducing d. One also checks whether $\gcd(f'(m), n)$ is a nontrivial factor of n. In case this factor is trivial we have the option of replacing f and d by f' and $d-1$.

In practice the degree d of f is usually between 3 and 5. In [A.K. Lenstra, H.W. Lenstra Jr., 1993] the authors use an odd degree due to a limitation in the square root algorithm used, but in principle an even degree could also be used.

Several mathematical objects are associated with the polynomial; these objects will be discussed as we go along in the chapter. The most important is the *norm* map, $N : \mathbb{Q}(\alpha) \to \mathbb{Q}$ (see section 2.1.2). For $a, b \in \mathbb{Z}$, $b \neq 0$, it can be shown (see proposition 4.3.1 below) that

$$N(a - b\alpha) = b^d f(a/b).$$

All else being equal, one polynomial f is better than another if the norm values $N(a-b\alpha)$ are more likely to be smooth when a and b are relatively small. Experiments have shown that the time taken by NFS to factor n depends heavily on the choice of the polynomial f .

4.2.2 A Simple Approach

An easy way to generate f is as follows. Given the degree $d \geqslant 1$, set $m = \lfloor n^{1/d} \rfloor$. Now write n to the base m,

$$n = c_d m^d + c_{d-1} m^{d-1} + \cdots + c_1 m + c_0, \quad 0 \leqslant c_i \leqslant m - 1,$$

and then let

$$f(x) = c_d x^d + c_{d-1} x^{d-1} + \cdots + c_1 x + c_0.$$

Proposition 4.2.2. *For f generated in this fashion we have $c_d = 1$ and $c_{d-1} \leqslant d$.*

4.3 Finding the Squares

Set a smoothness bound y. Let $\mathcal{A}$ be the (finite) region of $\mathbb{Z} \times \mathbb{Z}$ in which we plan to sieve. We adopt the following notation.

- d is the degree of $f(x)$.
- K^* is the multiplicative group of non-zero elements of K.
- $\mathcal{O}_K^*$ is the group of units of $\mathcal{O}_K$.
- $N : K \to \mathbb{Q}$ is the norm map from K to $\mathbb{Q}$.

Proposition 4.3.1. *For all $a, b \in \mathbb{Z}$ we have $N(a - b\alpha) = b^d f(a/b)$.*

Proof. Given

$$f(x) = \prod_{i=1}^{d} (x - \alpha_i)$$

and substituting a/b for x we get

$$f\left(\frac{a}{b}\right) = \prod_{i=1}^{d} \left(\frac{a - b\alpha_i}{b}\right)$$

so that

$$b^d \cdot f\left(\frac{a}{b}\right) = \prod_{i=1}^{d} (a - b\alpha_i) = N(a - b\alpha). \qquad \square$$

Definition 4.3.2. An element $\beta \in \mathcal{O}_K$ is said to be *y-smooth* if its norm $N(\beta) \in \mathbb{Z}$ is y-smooth.

We choose the factor bases (rational and algebraic) and the quadratic characters. The "rational" sieving goes as follows:

- For each b we initialize an array X for the values $a - bm$ with $(a, b) \in \mathcal{A}$.
- For each prime $p \leqslant y$ we look only at the entry corresponding to an a satisfying $a \equiv bm \bmod p$ and we divide the value in this entry by the highest power of p that divides it.

As for the algebraic part: given $b \in \mathcal{A}$ with $b \not\equiv 0 \bmod p$, the integers a with $N(a - b\alpha) \equiv 0 \mod p$ are those such that $a \equiv br \bmod p$ for some $r \in R(p)$ (see definition 2.1.19). If $b \equiv 0 \bmod p$ then it is clear that there exists no a with $(a, b) \in \mathcal{A}$ and $\gcd(a, b) = 1$ such that $N(a - b\alpha) \equiv 0 \bmod p$.

The "algebraic" sieving proceeds as follows:

- For each b we initialize an array Y for the values $N(a - b\alpha)$ for all $(a, b) \in \mathcal{A}$.
- For each p in the factor base B we consider only the entries where $a \equiv br \bmod p$.

At this point we gather into the set S_P those pairs (a, b) such that

- $a - bm$ is y-smooth ($X(a) = \pm 1$),
- $a - b\alpha$ is y-smooth ($Y(a) = \pm 1$),
- $\gcd(a, b) = 1$.

Once we have gathered the set of smooth pairs we fill the matrix M and proceed to the linear algebra step.

If the set of pairs S_P is sufficently large, the linear algebra step will yield at least one set S contained in S_P such that:

$$\prod_{(a,b)\in S} (a - bm) = v^2 \tag{4.3}$$

$$\prod_{(a,b)\in S} N(a - b\alpha) = w^2 \tag{4.4}$$

for some $v, w \in \mathbb{Z}$. We have thus satisfied equation (4.1), a necessary condition; but condition (4.4) is far from sufficient to satisfy equation (4.2).

In what remains of this section we will introduce new conditions in order to get

$$\prod_{(a,b)\in S} (a - b\alpha)$$

a square in $\mathcal{O}_K$.

For each prime p we keep track of the value $r \in R(p)$ such that $a \equiv br \bmod p$.

Lemma 4.3.3. *For each prime p and each pair (a, b) in $\mathcal{A}$ with $\gcd(b, p) = 1$ there exists exactly one value $r \in R(p)$ such that $a \equiv br \bmod p$.*

Proof. This is clear. □

Definition 4.3.4. Let $a, b \in \mathbb{Z}$, $\gcd(a, b) = 1$, p a prime number and $r \in R(p)$. We define

$$e_{p,r}(a - b\alpha) = \begin{cases} \mathrm{ord}_p\big(N(a - b\alpha)\big) & \text{if } a \equiv br \bmod p, \\ 0 & \text{otherwise,} \end{cases}$$

where $p^{\mathrm{ord}_p(k)}$ is the highest power of p dividing k.

Remark 4.3.5. It is clear that

$$N(a - b\alpha) = \pm \prod_{p,r} p^{e_{p,r}(a - b\alpha)}.$$

Proposition 4.3.6. *Let S be a finite set of coprime integer pairs (a, b) such that*

$$\prod_{(a,b) \in S} (a - b\alpha)$$

is a square in K. Then for each prime p and each r in $R(p)$ we have

$$\sum_{(a,b) \in S} e_{p,r}(a - b\alpha) \equiv 0 \bmod 2.$$

Proof. This follows from corollary 4.3.9 below. For the detailed proof we refer the reader to [A.K. Lenstra, H.W. Lenstra Jr., 1993] pp. 58–60. □

Remark 4.3.7. Suppose the congruence above holds for all pairs (p, r) does it follow that equation (4.2) is true? Unfortunately not! Consider the example where $S = \{(-1, 0)\}$ and let $\sqrt{-1} \notin K$. We have

$$\prod_{(a,b) \in S} (a - b\alpha) = -1;$$

however,

$$e_{p,r}(a - b\alpha) = e_{p,r}(-1) = 0.$$

We recall below some basic facts about the non-zero prime ideals of the ring $\mathbb{Z}[\alpha]$. In what follows, $\mathfrak{p}$ denotes a prime ideal in $\mathbb{Z}[\alpha]$.

- The prime ideal $\mathfrak{p}$ contains a unique prime number p, and $\mathbb{Z}[\alpha]/\mathfrak{p}$ is a finite field of characteristic p, called the *residue class field* of $\mathfrak{p}$.
- The *norm* $\mathfrak{N}_\alpha\mathfrak{p}$ of $\mathfrak{p}$ is the number of elements of its residue class field, *i.e.*,

$$\mathfrak{N}_\alpha\mathfrak{p} = |\mathbb{Z}[\alpha]/\mathfrak{p}|.$$

- The *degree* of $\mathfrak{p}$ is the degree of $\mathbb{Z}[\alpha]/\mathfrak{p}$ as an extension of the field $\mathbb{F}_p$.
- As was shown in proposition 2.1.18 there is a one-to-one correspondence between the degree one prime ideals of $\mathbb{Z}[\alpha]$ and the ideals $\mathfrak{a}_{p,r}$ (see definition 2.1.16) with p a prime number and $r \in R(p)$.

If $\mathbb{Z}[\alpha] = \mathcal{O}_K$ then the non-zero ideals of $\mathbb{Z}[\alpha]$ factor uniquely into prime ideals and $e_{p,r}(a-b\alpha)$ is simply the exponent of the degree one prime ideal $\mathfrak{a}_{p,r}$ in the factorization of the ideal $(a-b\alpha)\mathcal{O}_K$. In order to extend this idea to the case where $\mathbb{Z}[\alpha] \neq \mathcal{O}_K$ we use the following result.

Proposition 4.3.8. *There exists, for each prime ideal* $\mathfrak{p}$ *of* $\mathbb{Z}[\alpha]$, *a group homomorphism* $l_\mathfrak{p} : K^* \to \mathbb{Z}$, *such that the following hold:*

- $l_\mathfrak{p}(x) \geqslant 0$ *for all* $x \in \mathbb{Z}[\alpha]$, $x \neq 0$;
- *if* x *is a non-zero element of* $\mathbb{Z}[\alpha]$, *then* $l_\mathfrak{p}(x) > 0$ *if and only if* $x \in \mathfrak{p}$;
- *for each* $x \in K^*$ *one has* $l_\mathfrak{p}(x) = 0$ *for all but finitely many* $\mathfrak{p}$, *and*

$$\prod_\mathfrak{p} (\mathfrak{N}_\alpha\mathfrak{p})^{l_\mathfrak{p}(x)} = |N(x)|,$$

where $\mathfrak{p}$ *ranges over the set of all prime ideals of* $\mathbb{Z}[\alpha]$.

For the proof of this proposition see [A.K. Lenstra, H.W. Lenstra Jr., 1993], p. 63. We mention here only the following corollary.

Corollary 4.3.9. *Let a and b be integers with* $\gcd(a,b) = 1$ *and let* $\mathfrak{p}$ *be a prime ideal of* $\mathbb{Z}[\alpha]$. *If* $\mathfrak{p}$ *is of degree one then* $l_{\mathfrak{p}}(a - b\alpha) = e_{p,r}(a - b\alpha)$, *where* $\mathfrak{p} = \mathfrak{a}_{p,r}$, *and if* $\mathfrak{p}$ *is not of degree one then* $l_{\mathfrak{p}}(a - b\alpha) = 0$.

Now we can rewrite proposition 4.3.6 and replace $e_{p,r}$ by the homomorphism $l_{\mathfrak{p}}$. But still the proposition does not guarantee getting an algebraic square; the hypotheses of proposition 4.3.6 do not imply (4.2).

4.3.1 Complicating Factors

Until now we have considered a set

$$S \subseteq \{ (a,b) \in \mathcal{A} : \gcd(a,b) = 1,\ a - b\alpha \text{ is } y\text{-smooth} \}$$

satisfying

$$\sum_{(a,b)\in S} l_{\mathfrak{p}}(a - b\alpha) \equiv 0 \bmod 2 \tag{4.5}$$

for all degree one prime ideals $\mathfrak{p}$.

This is plainly a necessary condition for (4.2), but there are circumstances under which (4.2) might nevertheless fail to hold.

1. The ideal $\prod_{(a,b)\in S}(a - b\alpha)\mathcal{O}_K$ may not be the square of an ideal, since we work with prime ideals of $\mathbb{Z}[\alpha]$ rather than prime ideals of $\mathcal{O}_K$.
2. Even if $\prod_{(a,b)\in S}(a - b\alpha)\mathcal{O}_K = \mathfrak{a}^2$ for some ideal $\mathfrak{a}$ of $\mathcal{O}_K$, the ideal $\mathfrak{a}$ need not be principal.
3. Even if $\prod_{(a,b)\in S}(a - b\alpha)\mathcal{O}_K = \gamma^2\mathcal{O}_K$ for some $\gamma \in \mathcal{O}_K$, it is not necessary that $\prod_{(a,b)\in S}(a - b\alpha) = \gamma^2$.

These difficulties may be rephrased as follows. Let

$$V_1 = \{\, \beta \in K^* : l_{\mathfrak{p}}(\beta) \equiv 0 \bmod 2 \text{ for every prime } \mathfrak{p} \text{ of } \mathbb{Z}[\alpha] \,\},$$

$$V_2 = \{\, \gamma \in K^* : \gamma \mathcal{O}_K = (\delta \mathfrak{a})^2 \text{ for some } \delta \in K^* \text{ and some } \mathcal{O}_K\text{-ideal } \mathfrak{a} \,\},$$

$$V_3 = \mathcal{O}_K^* K^{*2},$$

$$V_4 = K^{*2},$$

For $1 \leqslant k \leqslant 3$, the set V_{k+1} consists of those elements of V_k that pass obstacle k. Therefore

$$V_4 \subseteq V_3 \subseteq V_2 \subseteq V_1,$$

the difficulty being that some or all of these set inclusions need not be equalities. The relations coming from the relation matrix will give elements of V_1: condition (4.2) requires an element of V_4.

4.3.2 Quadratic Characters

In [A.K. Lenstra, H.W. Lenstra Jr., 1993], pp. 61-62, it is shown that V_1/V_4 is a vector space over $\mathbb{F}_2$ of dimension at most $\log_2 n$. This fact suggests a technique to bypass the difficulties enumerated above.

Proposition 4.3.10. *Let S be a finite set of coprime integer pairs (a, b) with the property that $\prod_{(a,b)\in S}(a - b\alpha)$ is the square of an element of K. Further let q be an odd prime number and $s \in R(q)$, such that $a - bs \not\equiv 0 \bmod q$, and $f(s) \equiv 0 \bmod q$, for each $(a, b) \in S$, and $f'(s) \not\equiv 0 \bmod q$; then*

$$\prod_{(a,b)\in S} \left(\frac{a - bs}{q}\right) = +1$$

where $\left(\frac{a-bs}{q}\right)$ is the Legendre symbol.

Definition 4.3.11. For the pair (q, s), with q a prime number and $s \in R(q)$, the quadratic character $\chi_{q,s} : \mathbb{Z}[\alpha] - \mathfrak{a}_{q,s} \to \{\pm 1\}$ is given by

$$\chi_{q,s}(h(\alpha)) = \left(\frac{h(s)}{q}\right).$$

Adleman's technique (see [Adleman, 1979]) is to define $\chi_{q_1,s_1}, \ldots, \chi_{q_k,s_k}$ with k sufficiently large so that if $\beta \in K^*$ and $\chi_{q_1,s_1}(\beta) = \cdots = \chi_{q_k,s_k}(\beta) = +1$ then it is "virtually certain" that $\beta \in K^{*2}$.

Appending a column to the relation matrix for each of these characters and then reducing modulo 2 gives relations producing (with very high probability) elements of K^{*2}. This gives an efficient procedure to produce elements in V_4 as required to satisfy equation (4.2)

4.3.3 Square Roots

Assuming γ^2 satisfying (4.2) has been found, the problem remains of computing γ. An efficient procedure is given by Couveignes in [A.K. Lenstra, H.W. Lenstra Jr., 1993], pp. 95-102, under the assumption that the degree of $f(x)$ is odd. A general algorithm is given in [Montgomery, 1994a].

A further discussion of this problem would take us too far afield. We will merely note that this phase of the algorithm is the subject of further research.

4.4 Running Time

The Number Field Sieve algorithm is an $L(1/3)$-algorithm. It requires

$$L_n\big(1/3; (64/9)^{1/3} + o(1)\big)$$

steps as $n \to \infty$. For a detailed analysis of the running time see [A.K. Lenstra, H.W. Lenstra Jr., 1993] section 11, pp. 80–84.

4.5 Numerical Examples

The procedures used in this section appear in appendix B.

The main procedure, `nfsv`, is invoked with the call

$$\texttt{nfsv}(n, m, f, p_{\max}, a_{\max}, n_{QC}, r_{SP}).$$

The output of `nfsv` is a list of integer pairs (u, v) satisfying

$$u^2 \equiv v^2 \bmod n.$$

4.5.1 A Small Example

For our example we will factorize

$$n = 19549.$$

- With

  $$d = 3,$$

  $$m = \lfloor n^{1/d} \rfloor = 26,$$

  the call

  $$\texttt{defpolp}(n, d, m)$$

  returns

  $$f(x) = x^3 + 2x^2 + 23x + 23.$$

- The parameter $p_{\max}$ is the upper bound on the primes in the (rational and algebraic) factor bases. With

  $$p_{\max} = 100$$

  the algorithm creates the rational factor base

  $$B_R = \{ 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97 \}$$

  and the algebraic factor base

$$B_A = \{ (5,2), (7,1), (11,8), (17,2), (19,9), (37,31), (53,43), (59,9), (59,14), (59,34), (61,48), (67,28), (79,42), (83,40), (89,54), (97,33) \}.$$

- The parameter $a_{\max}$ is the maximum value of $|a|$ for each value of b in the sieving step.
- The parameter n_{QC} is the number of quadratic characters to be used.
- The parameter r_{SP} determines the number smooth pairs to accumulate in the set S_P before commencing the linear algebra step. Sieving terminates when

$$|S_P| \geqslant r_{SP}(1 + |B_R| + 1 + |B_A| + n_{QC}).$$

With parameter values

$$a_{\max} = 3m = 78$$

$$n_{QC} = 20$$

$$r_{SP} = 2/3$$

the sieving proceeds for $1 \leqslant b \leqslant 20$, $-a_{\max} \leqslant a \leqslant +a_{\max}$, and terminates with

$$S_P = \{ (-43,1), (-13,1), (-10,1), (-6,1), (-3,1), (-1,1), (1,1), (2,1), (7,1), (8,1), (9,1), (14,1), (28,1), (-47,2), (-31,2), (-11,2), (-3,2), (-1,2), (5,2), (9,2), (-32,3), (-16,3), (-11,3), (-4,3), (2,3), (13,4), (-4,5), (7,5), (-5,6), (-5,7), (-27,8), (-13,8), (-16,9), (-29,11), (40,11), (41,13), (19,14), (-18,17), (-42,19), (-13,19), (-3,19), (-29,20) \}.$$

- The quadratic characters, which depend on the members of S_P, are chosen last. The algorithm gives

$$B_Q = \{\, (101,25), (103,63), (107,81), (109,101), (137,3),$$
$$(149,68), (157,131), (167,56), (181,112), (191,176),$$
$$(199,42), (211,4), (223,91), (227,43), (229,63),$$
$$(241,227), (251,100), (263,72), (271,76), (281,141) \,\}.$$

The next step is to construct the matrix M, which will be row-reduced over the field $\mathbb{F}_2$. The matrix M will have $|S_P|$ rows (one for each pair in S_P) and

$$1 + |B_R| + 1 + |B_A| + |B_Q| + |S_P|$$

columns.

The row of M corresponding to the pair (a, b) is constructed as follows.

- The entry in the first column is 0 if $a - bm > 0$ and 1 if $a - bm < 0$.
- The next $|B_R|$ columns correspond to the primes in B_R. The entry in the column corresponding to the prime p is the exponent of the highest power of p dividing $a - bm$, reduced modulo 2.
- The next entry is 0 if $N(a - b\alpha) > 0$, 1 if $N(a - b\alpha) < 0$.
- The next $|B_A|$ columns correspond to the primes in B_A. The entry in the column corresponding to the prime (p, r) is $e_{p,r}\big(N(a - b\alpha)\big)$, reduced modulo 2.
- The next $|B_Q|$ columns correspond to the quadratic characters in B_Q. The entry in the column corresponding to the quadratic character (q, s) is 0 if $a - bs$ is a quadratic residue modulo q, 1 if $a - bs$ is a quadratic nonresidue modulo q.

The last $|S_P|$ columns of M are initialized to be the $|S_P| \times |S_P|$ identity matrix.

The matrix M is now reduced over $\mathbb{F}_2$. Rows with a non-zero entry among the first $1 + |B_R| + 1 + |B_A| + |B_Q|$ columns are discarded, and in the remaining rows these columns are suppressed. What remains is the matrix $\widehat{M}$, with n_{RL} rows and $|S_P|$

columns. For $1 \leqslant j \leqslant n_{RL}$, row j of $\widehat{M}$ defines a subset S_j of S_P, namely, the set of pairs (a, b) for which the corresponding entry is 1.

With the parameter values as chosen above, the algorithm gives $n_{RL} = 7$. The sets $S_1, \ldots, S_7$ corresponding to the successive rows of $\widehat{M}$ are shown below, together with the computation of the values of u and v they define.

$$\begin{aligned}
\mathbf{S_1}\text{:}\ \prod_{(a,b)\in S_1}(a-b\alpha) &= (-43-\alpha)(1-\alpha)(2-\alpha)(14-\alpha)(28-\alpha)(-3-2\alpha)(5-2\alpha)\\
&\quad (-32-3\alpha)(-4-3\alpha)(13-4\alpha)(-4-5\alpha)(-5-7\alpha)\\
&\quad (-27-8\alpha)(-13-8\alpha)(-16-9\alpha)(-29-11\alpha)(40-11\alpha)\\
&\quad (41-13\alpha)(-18-17\alpha)(-42-19\alpha)(-13-19\alpha)\\
&= -844713683732320550049841025111\\
&\qquad -10829594568154848312453234790 08\,\alpha\\
&\qquad\quad -26296944852417420760614671 8392\,\alpha^2\\
&= \left(\tfrac{7184112683576897}{43} + \tfrac{12039205230256056}{43}\,\alpha + \tfrac{4942658030594364}{43}\,\alpha^2\right)^2\\
u &= \left(\tfrac{7184112683576897}{43} + \tfrac{12039205230256056}{43}\,m + \tfrac{4942658030594364}{43}\,m^2\right) \bmod n\\
&= 14973
\end{aligned}$$

$$\begin{aligned}
\prod_{(a,b)\in S_1}(a-bm) &= 19733281015109735570907939559430400000000000\\
&= 1404751971527704800000^2
\end{aligned}$$

$$v = 1404751971527704800000 \bmod n = 14973$$

$$n = \gcd(n, u-v)\cdot\gcd(n, u+v) = 19549 \cdot 1$$

$$\mathbf{S_2}: \prod_{(a,b)\in S_2} (a - b\alpha) = (-13 - \alpha)(1 - \alpha)(-11 - 2\alpha)(5 - 2\alpha)(-32 - 3\alpha)$$
$$(13 - 4\alpha)(7 - 5\alpha)(-27 - 8\alpha)(40 - 11\alpha)(41 - 13\alpha)$$
$$= 4516448867681 + 5986277651168\,\alpha + 2042595881832\,\alpha^2$$
$$= (2192033 + 1431384\,\alpha + 4396\,\alpha^2)^2$$

$$u = (2192033 + 1431384\,m + 4396\,m^2) \bmod n = 17030$$

$$\prod_{(a,b)\in S_2} (a - bm) = 61029755588223922500 = 7812154350^2$$

$$v = 7812154350 \bmod n = 2519$$

$$n = \gcd(n, u - v) \cdot \gcd(n, u + v) = 1 \cdot 19549$$

$$\mathbf{S_3}: \prod_{(a,b)\in S_3} (a - b\alpha) = (-10 - \alpha)(1 - \alpha)(2 - \alpha)(9 - \alpha)(14 - \alpha)(-4 - 3\alpha)$$
$$(13 - 4\alpha)(-4 - 5\alpha)(-13 - 8\alpha)(-16 - 9\alpha)(-29 - 11\alpha)$$
$$(40 - 11\alpha)(-42 - 19\alpha)(-13 - 19\alpha)$$
$$= -407346051769736467$$
$$- 4252312743698369186\,\alpha$$
$$- 340165426170141719\,\alpha^2$$
$$= (571070353 + 727245143\,\alpha + 172368054\,\alpha^2)^2$$

$$u = (571070353 + 727245143\,m + 172368054\,m^2) \bmod n = 5965$$

$$\prod_{(a,b)\in S_3} (a - bm) = 5126189466957908154624000000 = 715974124320002$$

$$v = 715974124320002 \bmod n = 13923$$

$$n = \gcd(n, u - v) \cdot \gcd(n, u + v) = 173 \cdot 113$$

$$\mathbf{S_4}\colon \prod_{(a,b)\in S_4}(a-b\alpha) = (-6-\alpha)(1-\alpha)(2-\alpha)(9-\alpha)(14-\alpha)(-11-2\alpha)(5-2\alpha)$$
$$(-32-3\alpha)(13-4\alpha)(-4-5\alpha)(7-5\alpha)(-5-6\alpha)(-27-8\alpha)$$
$$(-13-8\alpha)(-29-11\alpha)(40-11\alpha)(41-13\alpha)(-18-17\alpha)$$
$$(-42-19\alpha)(-13-19\alpha)$$
$$= 45320911824317286577001834 23$$
$$+ 43290678686403802233876500 87\,\alpha$$
$$- 61043696730258748675749 93\,\alpha^2$$
$$= (-21542537715167 - 16079423894440\,\alpha + 4332513950153\,\alpha^2)^2$$
$$u = (-21542537715167 - 16079423894440\,m$$
$$+ 4332513950153\,m^2) \bmod n = 1986$$

$$\prod_{(a,b)\in S_4}(a-bm) = 460291011049964608887029492665589760000 00$$
$$= 21454393746968582400 0^2$$
$$v = 2145439374696858240 00 \bmod n = 1647$$
$$n = \gcd(n, u-v)\cdot\gcd(n, u+v) = 113\cdot 173$$

$$\mathbf{S_5}\colon \prod_{(a,b)\in S_5}(a-b\alpha) = (-1-\alpha)(14-\alpha)$$
$$= -14 - 13\,\alpha + \alpha^2$$
$$= \left(\tfrac{32}{43} + \tfrac{74}{43}\,\alpha + \tfrac{9}{43}\,\alpha^2\right)^2$$
$$u = \left(\tfrac{32}{43} + \tfrac{74}{43}\,m + \tfrac{9}{43}\,m^2\right) \bmod n = 3824$$

$$\prod_{(a,b)\in S_5}(a-bm) = 324 = 18^2$$
$$v = 18 \bmod n = 18$$
$$n = \gcd(n, u-v)\cdot\gcd(n, u+v) = 173\cdot 113$$

$$\mathbf{S_6}: \prod_{(a,b)\in S_6} (a - b\alpha) = (7-\alpha)(28-\alpha)(-47-2\alpha)(-11-2\alpha)(-4-3\alpha)(2-3\alpha)$$
$$(-4-5\alpha)(-5-6\alpha)(-16-9\alpha)(40-11\alpha)(41-13\alpha)$$
$$(-18-17\alpha)(-42-19\alpha)$$
$$= -5998236206293235607$$
$$- 7161498572225516936\,\alpha$$
$$- 136175388372866648 4\,\alpha^2$$
$$= (511327523 + 772929358\,\alpha + 271248854\,\alpha^2)^2$$
$$u = (511327523 + 772929358\,m + 271248854\,m^2) \bmod n = 11054$$

$$\prod_{(a,b)\in S_6} (a - bm) = 14350652435184094681344 0000 = 11979420868800^2$$
$$v = 11979420868800 \bmod n = 8495$$
$$n = \gcd(n, u - v) \cdot \gcd(n, u + v) = 1 \cdot 19549$$

$$\mathbf{S_7}: \prod_{(a,b)\in S_7} (a - b\alpha) = (8-\alpha)(9-\alpha)(-47-2\alpha)(-4-5\alpha)(7-5\alpha)$$
$$(-5-7\alpha)(40-11\alpha)(-42-19\alpha)$$
$$= 132977776609 + 12677694045 0\,\alpha - 264391391\,\alpha^2$$
$$= \left(\tfrac{4461941}{43} + \tfrac{3709109}{43}\,\alpha - \tfrac{1035376}{43}\,\alpha^2\right)^2$$
$$u = \left(\tfrac{4461941}{43} + \tfrac{3709109}{43}\,m - \tfrac{1035376}{43}\,m^2\right) \bmod n = 3300$$

$$\prod_{(a,b)\in S_7} (a - bm) = 12311416792166976 = 110956824^2$$
$$v = 110956824 \bmod n = 16249$$
$$n = \gcd(n, u - v) \cdot \gcd(n, u + v) = 1 \cdot 19549$$

4.5.2 The Effect of Quadratic Characters

As seen above the use of the quadratic characters ensures (in practice) that we will get an algebraic square. To emphasize this fact we present the results obtained from the same Maple procedure but with the quadratic character columns disabled (*i.e.*, filled entirely with zeros). In the ouptut below, we show the smooth pair (a, b) corresponding

to each nonzero entry in the row, we also print its quadratic character array. The reader will notice that the sum modulo two of the quadratic characters' array is not necessarely zero (not a square). Although in this example we realise one successful factorization, there is no reason to beleive that this method will work with higher values of n.

$\mathbf{S_1}$:	(0 1 1 0 1 1 1 0 0 1 1 0 0 1 0 0 1 0 0 0)	$-12-x$
	(0 1 0 0 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1)	$-5-2x$
	(1 1 1 0 0 0 1 1 1 0 0 1 0 1 1 0 0 1 0 1)	$-3-2x$
	(1 0 1 1 1 0 1 0 0 1 1 0 1 1 1 1 1 1 1 1)	$1-2x$
	(0 1 1 0 1 0 1 0 0 1 1 1 1 0 0 1 1 1 1 0)	$-10-3x$
	(0 0 1 0 1 1 0 0 0 0 1 0 0 1 1 0 1 0 0 0)	$-2-3x$
	(1 1 0 0 1 1 0 0 0 1 1 1 0 1 0 0 1 1 0 1)	$-6-5x$
	(1 1 1 0 0 1 1 1 0 0 1 1 0 0 0 1 1 1 1 0)	$-5-6x$
	(1 0 1 1 0 0 1 1 1 0 1 1 0 0 0 1 1 1 1 0)	$-25-7x$
	(0 0 1 0 1 1 0 0 1 0 0 1 1 1 0 0 0 1 0 1)	$-3-11x$
	(0 1 0 0 1 1 0 1 1 1 0 1 0 0 1 0 0 0 1 0)	$11-12x$
	(0 0 0 1 0 1 1 1 0 1 1 1 0 0 0 1 0 0 0 0)	$-6-13x$
	(1 1 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 1)	

In this case the product

$$\prod_{(a,b)\in S_1} (a - b\alpha)$$

is not a square in $\mathbb{Q}(\alpha)$ and thus the algoritm fails. Rows 2, 3, 6, 7, 8, and 9 likewise fail to yield an algebraic square. Row 4 gives an algebraic square that yields a proper factorization of n, whereas row 5 gives an algebraic square that fails to give a non-trivial factorization.

$\mathbf{S_4}$:	(0 1 1 1 0 0 0 1 1 1 0 0 0 0 1 0 0 1 0 0)	$-1-x$
	(1 1 0 1 0 1 1 0 1 1 0 1 1 1 1 0 0 0 0 0)	$1-3x$
	(0 0 0 1 0 0 0 1 0 0 1 1 1 1 1 1 0 0 1 1)	$13-3x$
	(1 1 1 0 1 1 0 0 1 0 0 1 0 1 0 1 1 1 0 1)	$5-4x$
	(1 1 0 0 1 1 0 0 0 1 1 1 0 1 0 0 1 1 0 1)	$-6-5x$
	(1 0 0 0 0 1 1 1 1 1 1 0 1 0 0 0 1 1 0 0)	$-8-9x$
	(0 1 1 1 0 1 1 1 1 0 1 0 0 0 0 0 1 1 1 0)	$-26-11x$
	(0 1 0 0 1 1 0 1 1 1 0 1 0 0 1 0 0 0 1 0)	$11-12x$
	(0 0 0 1 0 1 1 1 0 1 1 1 0 0 0 1 0 0 0 0)	$-6-13x$
	(0 0 1 1 1 1 0 0 0 0 1 0 1 0 0 1 0 1 1 1)	$-2-13x$
	(0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0)	

$$\prod_{(a,b)\in S_4}(a-b\alpha) = (-1-\alpha)(1-3\alpha)(13-3\alpha)(5-4\alpha)(-6-5\alpha)$$
$$(-8-9\alpha)(-26-11\alpha)(11-12\alpha)(-6-13\alpha)(-2-13\alpha)$$
$$= -167968697172 - 61478355559\,\alpha + 186527349381\,\alpha^2$$
$$= (120867 - 48058\,\alpha + 123709\,\alpha^2)^2$$

$$u = 4241$$

$$v = 3895$$

$$n = \gcd(n, u-v)\cdot\gcd(n, u+v) = 173\cdot 113$$

$\mathbf{S_5}$:

(0	0	1	0	1	0	0	0	0	0	1	1	1	0	1	1	1	1	1	0)	$-x$
(0	0	1	1	0	0	0	1	0	1	0	0	1	1	0	1	1	0	0	1)	$-9-2x$
(1	0	1	1	1	0	1	0	0	1	1	0	1	1	1	1	1	1	1	1)	$1-2x$
(0	0	1	0	1	1	0	0	0	0	1	0	0	1	1	0	1	0	0	0)	$-2-3x$
(1	1	0	0	1	1	0	0	0	1	1	1	0	1	0	0	1	1	0	1)	$-6-5x$
(0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0)	$6-5x$
(1	0	1	0	1	0	0	0	1	0	0	1	0	1	0	0	1	1	0	0)	$8-5x$
(1	1	1	0	0	1	1	1	0	0	1	1	0	0	0	1	1	1	1	0)	$-5-6x$
(1	0	1	1	0	0	1	1	1	0	1	1	0	0	0	1	1	1	1	0)	$-25-7x$
(1	1	0	1	1	0	0	0	1	1	0	1	1	1	1	1	1	1	0	0)	$-4-7x$
(1	0	1	1	1	0	1	0	1	1	0	0	1	0	1	1	0	0	1	1)	$-26-9x$
(1	0	0	0	0	1	1	1	1	1	1	0	1	0	0	0	1	1	0	0)	$-8-9x$
(0	1	0	0	1	1	0	1	1	1	0	1	0	0	1	0	0	0	1	0)	$11-12x$
(0	0	0	1	0	1	1	1	0	1	1	1	0	0	0	1	0	0	0	0)	$-6-13x$
(0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0)	

$$\prod_{(a,b)\in S_5}(a-b\alpha) = (-\alpha)(-9-2\alpha)(1-2\alpha)(-6-5\alpha)(6-5\alpha)(8-5\alpha)$$
$$(-5-6\alpha)(-25-7\alpha)(-4-7\alpha)(-26-9\alpha)(-8-9\alpha)$$
$$(-11-12\alpha)(-6-13\alpha)$$
$$= -18297905386779 - 6654570482880\,\alpha + 20372759621476\,\alpha^2$$
$$= (1325991 - 617666\,\alpha + 1301046\,\alpha^2)^2$$

$$u = 8660$$

$$v = 10889$$

$$n = \gcd(n, u-v)\cdot\gcd(n, u+v) = 1\cdot 19549$$

Chapter 5

The Discrete Logarithm Problem

5.1 Introduction

We begin this section by the statement of the discrete logarithm problem. Let G be a multiplicative group and let $a, b \in G$. The *discrete logarithm problem* is to find an integer x such that

$$a^x = b. \tag{5.1}$$

Such an integer x is the *discrete logarithm of a to the base b*. Some literature refers to the integer x as the *index of a base b*.

Remark 5.1.1. Suppose y is a solution for the discrete logarithm in equation (5.1), and suppose w is the order of a in G. Then clearly $y + kw$, for any $k \in \mathbb{Z}$, is also a solution.

Two questions arise:

1. Does the integer x exist?
2. If it does, how to find it?

Lemma 5.1.2 below answers the first question. As for the computation of the discrete logarithm, we discuss briefly in section 5.2 how to achieve this by sieving. We present

here some lemmas that will be used in the latter section. For the proof of lemmas 5.1.4 and 5.1.5 we refer the reader to [Weber, 1997]

Lemma 5.1.2. *Let G be a cyclic group of order r, let*

$$r = \prod_{i=1}^{z} p_i^{e_i}$$

be the prime factorization of r, and let $a, b \in G$. Then (5.1) *is solvable if and only if, for each j in the range $1 \leqslant j \leqslant z$ and each k in the range $1 \leqslant k \leqslant e_j$,*

$$b^{r/p_j^k} \neq 1$$

implies

$$a^{r/p_j^k} \neq 1.$$

In order to prove this lemma we need the following:

Lemma 5.1.3. *Let G be a cyclic group of order r and let $t \in \mathbb{Z}$ be a divisor of r. Then*

1. *$G^t = \{ b^t : b \in G \}$ is a subgroup of G.*
2. *$a^{r/t} = 1$ if and only if a is a t^{th} power in G.*

Proof of Lemma 5.1.3. 1. This is easy to check.

2. Let g be a generator of G and let x be such that $g^x = a$. If

$$a^{r/t} = 1$$

then

$$g^{xr/t} = 1.$$

Thus t divides x.

Conversely, let $a = g^{kt}, k \in \mathbb{Z}$. We have

$$a^{r/t} = g^{ktr/t} = g^{kr} = 1. \qquad \square$$

Proof of Lemma 5.1.2. Let g be a generator of G. Suppose there exist j, k such that

$$b^{r/p_j^k} \neq 1$$

and

$$a^{r/p_j^k} = 1.$$

Let $U_{j,k}$ be the subgroup of p_j^k-powers of G. By lemma 5.1.3, $a \in U_{j,k}$ but $b \notin U_{j,k}$, and therefore equation (5.1) does not have a solution. Conversely, suppose y is a solution of equation (5.1) and assume

$$b^{r/p_j^k} \neq 1.$$

Since $b = a^x$, we have $a^{xr/p_j^k} \neq 1$ and thus $a^{r/p_j^k} \neq 1$. □

Lemma 5.1.4. *Let G be a cyclic group of order r, let g be a generator of G, let $a, b \in G$, and let t be a divisor of r. Let $k, l \in \mathbb{Z}$ with $\gcd(t, l) = 1$ such that*

$$a^k b^l = d^t$$

for some $d \in G$. Then

$$x \equiv -k/l \bmod t$$

is a solution to

$$\bar{a}^x = \bar{b}$$

in G/G^t.

Lemma 5.1.5. *Let G, a, and b be as above. Suppose*

1. *$b = a^{x_0}$ for some $x_0 \in \mathbb{Z}$,*
2. *$\bar{a}^{x_j} = \bar{b}$ in $G/G^{p_j^{e_j}}$ for $1 \leqslant j \leqslant n$, and*
3. *the integer x simultaneously satisfies*

$$\begin{aligned} x &\equiv x_1 \bmod p_1^{e_1}, \\ &\vdots \\ x &\equiv x_n \bmod p_n^{e_n}. \end{aligned}$$

Then $a^x = b$.

We now give a brief description of some applications of the discrete logarithm problem to cryptography.

5.1.1 Application in Cryptography

One-way functions

Diffie and Hellman were the first to apply the Discrete Logarithm problem to cryptography [W. Diffie, M. Hellman, 1976]. They use it as a good example of a *one-way function*. Informally, an invertible function f is a one-way function if, given x, it is "easy" to compute $y = f(x)$, but, given y, it is "hard" to compute $x = f^{-1}(y)$. One-way functions were first used by Needham in a secure method to store passwords in a multiuser computer system [Wilkes, 1968]. Another important use of the discrete logarithm as a one-way function was introduced by Diffie and Hellman in a method to generate a secret key over an insecure channel [W. Diffie, M. Hellman, 1976].

Diffie-Hellman key selection protocol

Suppose two parties A (Alice) and B (Bob) want to agree on a secret key using *only* an insecure channel of communication.

- Let G be a cyclic group and g a generator of G.
- Let M be a large integer.
- Alice chooses a random strictly positive integer $x_A \leqslant M$ (her secret key), computes g^{x_A} (her public key), and sends it to Bob.
- Bob does the same as Alice, chooses $x_B \leqslant M$ (his secret key), computes g^{x_B} (his public key), and sends it to Alice.
- The Diffie-Hellman secret key is $K_{AB} = g^{x_A x_B}$.

Note that both Alice and Bob can compute the secret key having each other's public key and each his/her own secret key. Suppose Zorro wants to know the secret key Alice and Bob agreed on knowing only the public information. Zorro has to solve the following problem:

$$\textit{Given } g^{x_A} \textit{ and } g^{x_B}, \textit{ compute } K_{AB} = g^{x_A x_B}.$$

Solving this problem is known as breaking the Diffie-Hellman scheme. It is not yet known if this is equivalent to breaking the discrete logarithm problem. Clearly the converse is true.

We also note that in order for the communication to be secure, g has to have at least one "large" prime divisor.

Many other cryptosystems are based on the supposed difficulty of the discrete log problem. We refer the reader to [McCurley, 1990] and references mentioned there.

5.2 Discrete Logarithms via the Number Field Sieve

As we have seen in the previous chapter, the Number Field Sieve is used to factor large integers. A variation of the NFS can be used to compute discrete logarithms in finite fields. In this chapter we will outline the main steps of a simplified version of the NFSDL (Number Field Sieve Discrete Logarithm) algorithm used by Weber [1997] to attack McCurley's Challenge [McCurley, 1990].

The main goal is to compute a q^{th} power in a large finite field F. Two quadratic polynomials are generated and with these polynomials are associated two quadratic number fields. We compute a q^{th} power in each of these number fields and we use them to obtain two integers k, l such that $a^k b^l$ is a q^{th} power in F. Having solved

$$x \equiv \log_a b \bmod q^{e_q} \tag{5.2}$$

for each prime power q^{e_q} exactly dividing $p-1$, the Chinese Remainder Theorem gives a solution of (5.2) modulo $p-1$, and this solution x is a solution of equation (5.1). In the remainder of this section we describe this process.

5.2.1 Outline

In this section we will restrict ourselves to the case $G = \mathbb{F}_p^*$. Our approach will be a simplified version of that in [Weber, 1997] (which itself is an adaptation of Montgomery's "two-quadratic" variant [Montgomery, 1993]).

In the algorithm below, it is assumed that $p-1$ is not divisible by the square of any "large" prime; if q is a prime such that $q^2 \mid p-1$ then q is sufficiently small that computing $\log_a b \bmod q$ is acceptably fast (via, *e.g.*, the Pohlig-Hellman algorithm [S. Pohlig, M. Hellman, 1978]).

Our task is to solve the congruence

$$a^x \equiv b \bmod p. \tag{5.3}$$

Without loss of generality we will assume $a \not\equiv \pm 1 \bmod p$ and $b \not\equiv a \bmod p$.

1. Irreducible quadratic polynomials $g_1(x), g_2(x) \in \mathbb{Z}[x]$ are constructed, such that

$$g_1(m) \equiv g_2(m) \equiv 0 \bmod p,$$

where (as in the standard Number Field Sieve) m is a conveniently chosen positive integer. Let α_1, α_2 be roots of g_1, g_2, and let $\mathcal{O}_1, \mathcal{O}_2$ be the rings of integers of the fields $\mathbb{Q}(\alpha_1), \mathbb{Q}(\alpha_2)$, respectively.

For simplicity in the following discussion we will assume g_1 and g_2 are monic, so that $\mathbb{Z}[\alpha_1] \subseteq \mathcal{O}_1$ and $\mathbb{Z}[\alpha_2] \subseteq \mathcal{O}_2$.

As in the standard Number Field Sieve, there exist homomorphisms

$$\varphi_1 : \quad \mathcal{O}_1 \to \mathbb{F}_p \qquad\qquad \varphi_2 : \quad \mathcal{O}_2 \to \mathbb{F}_p$$
$$h(\alpha_1) \mapsto h(m) \bmod p \qquad\qquad h(\alpha_2) \mapsto h(m) \bmod p.$$

The polynomials g_1 and g_2 will be used in computing additive character values (see section 5.2.4 below). With two quadratic polynomials the number of values is

$$N_C = \deg g_1 + \deg g_2 = 4.$$

2. Factor bases F_1 and F_2 are constructed, consisting of degree one prime ideals in $\mathcal{O}_1$ and $\mathcal{O}_2$ respectively.

   For simplicity we will assume $a\mathcal{O}_1$ is F_1-smooth and $b\mathcal{O}_2$ is F_2-smooth.

3. A set C of pairs of integers is constructed, with $|C| > |F_1| + |F_2| + N_C$, such that $(c + d\alpha_1)\mathcal{O}_1$ is F_1-smooth and $(c + d\alpha_2)\mathcal{O}_2$ is F_2-smooth for each pair (c, d) in C. Sieving is performed with respect to the members of F_1 and F_2 (just as with the algebraic factor base in the Number Field Sieve) to determine these "smooth pairs". By convention, $(a, 0) \notin C$ and $(b, 0) \notin C$. Define

$$\widehat{C} = C \cup \{ (a, 0), (b, 0) \}.$$

4. The following computation is performed for each "large" prime q dividing $p - 1$.

   - The matrix M is constructed, with one row for each pair $(c, d) \in \widehat{C}$.

     - The first $|F_1|$ entries of the row are

       the exponent vector of a with respect to F_1 if $(c, d) = (a, 0)$,
       the exponent vector of 1 with respect to F_1 if $(c, d) = (b, 0)$,
       the exponent vector of $c + d\alpha_1$ with respect to F_1 otherwise.

     - The next $|F_2|$ entries of the row are

       the exponent vector of 1 with respect to F_2 if $(c, d) = (a, 0)$,
       the exponent vector of b with respect to F_2, if $(c, d) = (b, 0)$,
       the exponent vector of $c + d\alpha_2$ with respect to F_2, otherwise.

     - The next N_C entries are the additive character values.

- The last $|C|$ entries are the entries in the corresponding row of the $|C| \times |C|$ identity matrix.

- M is row-reduced modulo q, giving a set of relations on the pairs in C. Each such relation R defines a polynomial $h_R(x)$, such that $h_R(\alpha_1)$ and $h_R(\alpha_2)$ are q^{th} powers in $\mathbb{Q}(\alpha_1)$ and $\mathbb{Q}(\alpha_2)$, respectively.

- Choosing a relation in which $(b, 0)$ appears with exponent $e_{b,0} \not\equiv 0 \bmod q$, we have

$$a^{e_{a,0}} 1^{e_{b,0}} \prod_{(c,d)\in C} (c + d\alpha_1)^{e_{c,d}} = \gamma_1^q,$$
$$1^{e_{a,0}} b^{e_{b,0}} \prod_{(c,d)\in C} (c + d\alpha_2)^{e_{c,d}} = \gamma_2^q,$$

for some $\gamma_1 \in \mathcal{O}_1$, $\gamma_2 \in \mathcal{O}_2$. Thus

$$a^{e_{a,0}} \prod_{(c,d)\in C} (c + dm)^{e_{c,d}} \equiv g_1^q \bmod p,$$
$$b^{e_{b,0}} \prod_{(c,d)\in C} (c + dm)^{e_{c,d}} \equiv g_2^q \bmod p,$$

for some $g_1, g_2 \in \mathbb{Z}$. Taking $k = e_{a,0}$, $l = -e_{b,0}$, we have

$$a^k b^l \equiv (g_1/g_2)^q \bmod p.$$

By lemma 5.1.4, if

$$x \equiv -k/l \bmod q$$

then

$$a^x b^{-1} \equiv y^q \bmod p$$

for some $y \in \mathbb{F}_p^*$, so that

$$x \equiv \log_a b \bmod q.$$

5.2.2 Reduction by Sieving

For large values of b it can be impractical to use a factor base F_2 with respect to which $b\mathcal{O}_2$ is smooth. In such cases we apply sieving to decompose b as the product modulo p of powers of small primes.

Let a factor base P of (small) prime numbers be given. If we can find integer exponents e_s such that

$$b \equiv \prod_{s \in P} s^{e_s} \bmod p \tag{5.4}$$

then we can reduce the original problem (5.3) to several smaller problems, each of the form

$$a^{x_s} = s \bmod p \tag{5.5}$$

for some prime $s \in P$.

The decomposition (5.4) can be found efficiently by sieving (see [Weber, 1997] p. 26), as follows. Assume for some $t \in \mathbb{Z}$ that $\sqrt{p}/2 \leqslant t \leqslant \sqrt{p}$, and let

$$t' = \left\lfloor \frac{p}{t} \right\rfloor + 1.$$

We define two linear homogeneous polynomials

$$f_1(x, y) = x + t'y,$$
$$f_2(x, y) = tx + (tt' - p)y.$$

We now sieve for $(x, y) \in \mathbb{Z} \times \mathbb{Z}$ such that f_1 and f_2 are simultaneously P-smooth. Having found such a pair we have

$$x + t'y = \prod_{s \in P} s^{e_s},$$
$$tx + tt'y = t(x + t'y) \equiv \prod_{s \in P} s^{e'_s} \bmod p.$$

Thus,

$$t \equiv \frac{\prod_{s \in P} s^{e'_s}}{\prod_{s \in P} s^{e_s}} \bmod p,$$

which is a P-smooth representation of $t \bmod p$.

To achieve the decomposition (5.4), we use the extended Euclidean algorithm to find $u, v \in \mathbb{Z}$ such that $b \equiv v/u \bmod p$, with v and u both close to $\sqrt{p}$. Applying the procedure above to u and v separately yields a P-smooth representation of $b \bmod p$.

Remark 5.2.1. Weber [1997] shows that the values of f_1 and f_2 that are tested are are of magnitude $O(\sqrt{p})$.

5.2.3 Sieving for Smooth Pairs

Sieving in NFSDL is similar to the "algebraic" sieving in the NFS. We construct a set C of integer coprime pairs (c, d) such that $(c + d\alpha_1)$ and $(c + d\alpha_2)$ are smooth with respect to F_1 and F_2, respectively.

The sieving is based on the following observation. Let $\mathcal{O}_i$ be one of $\mathcal{O}_1$, $\mathcal{O}_2$, and suppose $\mathfrak{r}$ is a prime ideal in $\mathcal{O}_i$ having norm the rational prime r. Then

$$\mathfrak{r} \mid (c + d\alpha)\mathcal{O}_i \iff \mathfrak{r} \mid (c + r + d\alpha)\mathcal{O}_i.$$

We continue sieving until

$$|C| > |F_1| + |F_2| + N_C$$

in order to assure linear dependence.

5.2.4 Additive Characters

The algorithm produces relations on the set $\widehat{C}$, each of which gives a polynomial $h(x) \in \mathbb{Z}[x]$ such that the norms of $h(\alpha_1)$, $h(\alpha_2)$ are q^{th} powers in $\mathbb{Z}$. Of course, this is not a sufficient condition for $h(\alpha_1)$ and $h(\alpha_2)$ to be q^{th} powers themselves. In close analogy to the situation described in section 4.3.1, there are complicating factors to

be dealt with. To circumvent these difficulties we use the mechanism of "additive characters", due to Schirokauer [1993].

The following remarks apply with $i = 1, 2$.

We introduce the requirements that q does not ramify in $\mathcal{O}_i$ (*i.e.*, $q\mathcal{O}_i$ is the product of distinct prime ideals in $\mathcal{O}_i$) and that the order of the group of ideal classes of $\mathcal{O}_i$ is not a multiple of q. (These restrictions must be taken into account in the construction of g_i.) Then

$$q\mathcal{O}_i = \prod_{j=1}^{r} \mathfrak{p}_j$$

with $\mathfrak{p}_1, \ldots, \mathfrak{p}_r$ distinct prime ideals in $\mathcal{O}_i$. We define

$$\epsilon_i = \operatorname{lcm}\{\, N(\mathfrak{p}_j) - 1 : j = 1, \ldots, r \,\}.$$

Then

$$\gamma^{\epsilon_i} \equiv 1 \bmod \mathfrak{p}_j, \quad j = 1, \ldots, r$$

for all $\gamma \in \mathcal{O}_i$, and it follows that

$$\gamma^{\epsilon_i} - 1 \in q\mathcal{O}_i.$$

We define the mapping

$$\begin{aligned} \lambda_i : \mathcal{O}_i &\to q\mathcal{O}_i/q^2\mathcal{O}_i \\ \gamma &\to \gamma^{\epsilon_i} - 1 + q^2\mathcal{O}_i. \end{aligned}$$

Supposing that

$$\lambda_i(c + d\alpha_i) = qy_i + qz_i\alpha_i + q^2\mathcal{O}_i$$

with $y_i, z_i \in \mathbb{Z}$, the values y_1, z_1, y_2, z_2 are the additive character values corresponding to the pair (c, d).

In [Schirokauer, 1993] it is shown how the condition $\lambda_i(\gamma) = 0$ ensures that γ is a q^{th} power in $\mathcal{O}_i$.

5.2.5 Running Time

The NFSDL algorithm is an $L(1/3)$-algorithm. It is estimated in [Schirokauer, 1993] that the running time is

$$L_p\big(1/3; (64/9)^{1/3}\big).$$

Appendix A

Some Lemmas

Lemma A.0.2. *If*

$$F(x) = \frac{x\sqrt{x}}{\ln x}$$

and

$$G(x) = \int_2^x \frac{\sqrt{t}}{\ln t}\, dt$$

then

$$F(x) > \frac{5}{4} G(x)$$

for all $x > e^7$.

Proof. Define $H(x) = F(x) - \frac{5}{4}G(x)$. Then

$$H'(x) = \frac{\sqrt{x}\,(\ln x - 6)}{6 \ln^2 x},$$
$$H''(x) = \frac{(\ln x - 4)^2 + 8}{12\sqrt{x}\,\ln^3 x},$$

so that H is concave up for $x > 1$, with a minimum at $x = e^6$. Since $H(e^7) = 12.283\ldots$ it follows that $H(x) > 0$ for $x > e^7$. □

Lemma A.0.3. *There exists a positive constant* C *such that*

$$\sum_{\substack{p \text{ prime} \\ p \leqslant x}} \sqrt{p} \;\geqslant\; C \frac{x\sqrt{x}}{\ln x}$$

for all $x \geqslant 2$.

Proof. By the Prime Number Theorem we may choose $y > e^7$ such that

$$\frac{2}{3}\frac{x}{\ln x} \leqslant \pi(x) \leqslant \frac{4}{3}\frac{x}{\ln x}$$

for all $x \geqslant y$. Defining $F(x)$ and $G(x)$ as in Lemma 1 and applying Abel summation, we have

$$\begin{aligned}
\sum_{p \leqslant x} \sqrt{p} &= \pi(x)\sqrt{x} - \frac{1}{2}\int_2^x \frac{\pi(t)}{\sqrt{t}}\,dt \\
&= \pi(x)\sqrt{x} - \frac{1}{2}\int_2^y \frac{\pi(t)}{\sqrt{t}}\,dt - \frac{1}{2}\int_y^x \frac{\pi(t)}{\sqrt{t}}\,dt \\
&\geqslant \frac{2}{3}\cdot\frac{x\sqrt{x}}{\ln x} - \frac{1}{2}\int_2^y \frac{\pi(t)}{\sqrt{t}}\,dt - \frac{1}{2}\cdot\frac{4}{3}\int_y^x \frac{\sqrt{t}}{\ln t}\,dt \\
&= \frac{2}{3}F(x) - \frac{1}{2}\int_2^y \frac{\pi(t)}{\sqrt{t}}\,dt - \frac{2}{3}\Big(G(x) - G(y)\Big) \\
&= \frac{2}{3}\Big(F(x) - G(x)\Big) + \frac{2}{3}G(y) - \frac{1}{2}\int_2^y \frac{\pi(t)}{\sqrt{t}}\,dt \\
&\geqslant \frac{2}{3}\Big(F(x) - \frac{4}{5}F(x)\Big) + \frac{2}{3}G(y) - \frac{1}{2}\int_2^y \frac{\pi(t)}{\sqrt{t}}\,dt \\
&= \frac{2}{15}F(x) + \left(\frac{2}{3}G(y) - \frac{1}{2}\int_2^y \frac{\pi(t)}{\sqrt{t}}\,dt\right) \\
&= C_1 F(x) + C_2
\end{aligned}$$

and the lemma follows. □

Theorem A.0.4. *The series*

$$B = \sum_{p \text{ prime}} \left(\ln\left(1 - \frac{1}{p}\right) + \frac{1}{p}\right)$$

is convergent, and

$$\sum_{\substack{p \text{ prime} \\ p \leqslant x}} \frac{1}{p} = \ln\ln x + \gamma + B + o(1),$$

where γ is Euler's constant.

Proof. This is Theorem 427 of [Hardy & Wright, 1960]. □

Appendix B

Maple Procedures

The procedures below are written in MAPLE 7.

Implementation of Kraitchik's Algorithm

```
with(linalg,vector,submatrix):
with(numtheory,legendre):
############################################################
nnreln := proc (M, nr, nc)
local nn, r, c:
nn := 0:
for r from 1 to nr do
for c from 1 to nc do
    if M[r,c] <> 0 then nn := r fi
od od:
return(nn)
end:
############################################################
N := ithprime(500)*ithprime(1000):  sN := iroot(N,2):
d := iroot(N,4):
Y := 1000:
print('N'=N,'sN'=sN,'d'=d, 'Y'=Y):
### N is the integer we are interested in factoring
### d is the "spacing" before and after sN
### I will hold the primes less than or equal to Y in B.
### I am trying to find out which numbers are Y-smooth
### in an array of integers of the form Q(x) := x^2 - N.
### I will take the values of x somehow centered around
### the root of N.
B := []:  q := 2:
while q <= Y do
    if legendre(N,q) = +1 then B := [op(B),q] fi:
    q := nextprime(q):
od:
nB := nops(B):
```

```
S := []:  nS := 0:
Q := matrix(2,d):
sD := +1:              ### sign ###
bD := +0:              ### base ###
while nS < 11+nB do
    if sN + sD*(bD + 1) > 0 then
        for k from 1 to d do
            x := sN + sD*(bD + k):
    Q[1,k] := x^2 - N:
    Q[2,k] := abs(Q[1,k]):
        od:
        for p in B do     ###  non-sieve  ###
            for k from 1 to d do
                while Q[2,k] mod p = 0 do
                    Q[2,k] := Q[2,k] / p:
                od:
            od:
        od:
        for k from 1 to d do
            if Q[2,k] = 1 then S := [op(S),Q[1,k]] fi
        od:
        nS := nops(S):
    fi:
    sD := -sD:  if sD > 0 then bD := bD + d fi:
od:
print('B'=B):
print('S'=S):
print('nS'=nS,'nB'=nB,'nC'=1+nB+nS):
#########################################################
### Now we proceed to fill the exponent matrix.
M := matrix(nS, 1+nB+nS):
### Here I am going to fill the signs of the Q(x)
### in the matrix
for s from 1 to nS do
    if S[s] < 0 then M[s,1] := 1 else M[s,1] := 0 fi:
    for b from 1 to nB do
        w := S[s]:  M[s,1+b] := 0:
        while w mod B[b] = 0 do
            w := w / B[b]:
            M[s,1+b] := (M[s,1+b] + 1) mod 2:  ## note ##
        od:
    od:
    for b from 1 to nS do
        if b = s then M[s,1+nB+b] := 1
           else M[s,1+nB+b] := 0 fi:
    od:
od:
#####################################################
#####  Reduce matrix M mod 2
nC := 1+nB+nS:
r := 0:
for c from 1 to nC do
    h := 0:
    for j from nS by -1 to r+1 do
```

```
        if M[j,c] <> 0 then h := j fi
    od:
    if h <> 0 then
        r := r + 1:
        if h <> r then
            for k from c to nC do
                M[r,k] := (M[r,k] + M[h,k]) mod 2
            od:
        fi:
        for j from 1 to nS do
            if j <> r then
                if M[j,c] <> 0 then
                   for k from c to nC do
                        M[j,k] := (M[j,k] - M[r,k]) mod 2
                    od
                fi
            fi
        od
    fi
od:
nNR := nnreln(M,nS,1+nB):  nRL := nS - nNR:
#########################################################
M := submatrix(M,nNR+1..nNR+nRL,1+nB+1..nC):
print('nNR'=nNR,'nRL'=nRL):
for j from 1 to nRL do
    u := 1:
    w := 1:
    for k from 1 to nS do
        x := isqrt(S[k]+N):
        u := u*x^M[j,k]:
        w := w*S[k]^M[j,k]:
    od:
    v := isqrt(w):
    lprint([igcd(N,u+v),igcd(N,u-v)]):
od:
#########################################################
```

```
#####  True sieving  #####
with(linalg,vector,submatrix):
with(numtheory,legendre):
#########################################################
nnreln := proc (M, nr, nc)
### This is a procedure to find out how many
### non-relations we have.
### By a non-relation we mean a row that is not totally
### reduced to zeros i.e. not a square

local nn, r, c:
nn := 0:
for r from 1 to nr do
for c from 1 to nc do
    if M[r,c] <> 0 then nn := r fi
### If we find a one in the row,
### it means that this is a non-relation.
od od:
return(nn)
end:
#########################################################
N := ithprime(50000)*ithprime(100000):  sN := iroot(N,2):
d := iroot(N,4):
### N is the number to be factored
### sN is the square root of N it is where we start to
### Sieve with interval of length d on both sides
Y := 1000:
### Y is the smoothness bound.
print('N'=N,'sN'=sN,'d'=d, 'Y'=Y):
B := []:  q := 2:
### B is the array where we hold the primes we select
### We want primes such that N is a quadratic residue in order
### for the polynomial Q(x) = x^2 - N to factor.
while q <= Y do
### go until the smoothness bound
    if legendre(N,q) = +1 then B := [op(B),q] fi:
    q := nextprime(q):
od:
nB := nops(B):
S := []:  nS := 0:
### S is the array that will eventually collect
### the Y-smooth integers
Q := matrix(2,d):
sD := +1:              ### sign ###
bD := +0:              ### base ###
while nS < 11+nB do
### we continue sieving until we have at least
### 11 more relations than primes
    if sN + sD*(bD + 1) > 0 then
### We do not sieve on values of x that are negative
```

```
### because Q(-x) = Q(x) so it has already been considered!
#####  fill Q  #####
        for k from 1 to d do
            x := sN + sD*(bD + k):
    Q[1,k] := x^2 - N:
    Q[2,k] := abs(Q[1,k]):
### negative number could also be Y-smooth but they
### reduce to -1 so we take their absolute value
        od:
#####  Sieve  #####
x := 'x':
x0 := sN + sD*bD:
### We start sieving at sN in intervals of length d.
### We first sieve on the right of sN then on its left.
### Our next step is on the right of sN + d then
### on the left of sN - d etc.
### sD determins the signs of the movement +1 => right, -1 => left
### We call the starting point of the sieve bD
        for p in B do
### for all primes in the factor base
    w := Factors(x^2-N) mod p:   ### we know N is a residue ###
    for i from 1 to 2 do
### N has too roots mod p thus 2 cases i = 1 and 2
        a := p - subs(x=0,w[2][i][1]):
### a^2 \asym N mod p
        k0 := sD*(a - x0) mod p:
        hmin := ceil((1 - k0)/p):
        hmax := floor((d - k0)/p):
        for h from hmin to hmax do
### while sieving, the values of x are such that xk = x1 + (k-1)sD
### where 1 <= k <= d
### Let k \asym k0 mod p, so k = k0 + tp
### thus 1 <= k0 + tp <= d, and 1 - k0 <= tp <= d - k0
### so we could find that the bounds for t are
### ceil((1 - k0)/p) and floor((d - k0)/p)
### We called these hmin and hmax respecitvely

k := k0 + h*p:
### We are only sieving on the values of x such that x = a +/- kp
                    while Q[2,k] mod p = 0 do
                        Q[2,k] := Q[2,k] / p
                    od:
        od:
    od:
        od:
#####  Retrieve Smooth Q  #####
        for k from 1 to d do
            if Q[2,k] = 1 then S := [op(S),Q[1,k]] fi
        od:
        nS := nops(S):
    fi:
    sD := -sD:  if sD > 0 then bD := bD + d fi:
### swithching direction of sieving and/or
### advancing the base point
```

```
od:
print('B'=B):
print('S'=S):
print('nS'=nS,'nB'=nB,'nC'=1+nB+nS):
############################################################
### Now we proceed to fill the exponent matrix.
M := matrix(nS, 1+nB+nS):
### The first column is for the sign of Q(x)
### The next nB columns are for the primes powers
### of Q(x) reduced mod 2
### The last nS colums are used to append an nSxnS identity matrix
### Here we are going to fill the signs of the Q(x) in the matrix
for s from 1 to nS do
    if S[s] < 0 then M[s,1] := 1 else M[s,1] := 0 fi:
    for b from 1 to nB do
        w := S[s]:  M[s,1+b] := 0:
### making a copy of S in order not to lose the information
### in it and initializing the matrix to zeros
        while w mod B[b] = 0 do
            w := w / B[b]:
            M[s,1+b] := (M[s,1+b] + 1) mod 2:
### We only need to know if Q(x) is a square so we only care about
### prime powers reduced mod 2
        od:
    od:
### Filling the identity matrix
    for b from 1 to nS do
        if b = s then M[s,1+nB+b] := 1 else M[s,1+nB+b] := 0 fi:
    od:
od:
#################################################################
#####  Reduce matrix M mod 2
nC := 1+nB+nS:
r := 0:
### The reduction procedure
### for each column starting from the first
### look for the first row with a one, copy it to row r
### (at first r = 0 then we increase it as we move)
### make all other entries below row r zeros by addition mod 2
### move to the next column
for c from 1 to nC do
    h := 0:
    for j from nS by -1 to r+1 do
        if M[j,c] <> 0 then h := j fi
    od:
### looking for the first one below row r
    if h <> 0 then ### we found a non-zero entry
        r := r + 1:
### indicates that we have already reduced all rows down to row r
        if h <> r then
            for k from c to nC do
                M[r,k] := (M[r,k] + M[h,k]) mod 2
            od:
        fi:
```

```
### copying the row with the non-zero entry (row h) to row r
        for j from 1 to nS do
            if j <> r then
                if M[j,c] <> 0 then
                    for k from c to nC do
                        M[j,k] := (M[j,k] - M[r,k]) mod 2
### reducing mod 2 all entries in column c below r
                    od
                fi
            fi
        od
    fi
od:
nNR := nnreln(M,nS,1+nB):  nRL := nS - nNR:
### nNr number of non relations (rows not all zeros)
### nRL number of relations (rows with all zeros)
#############################################################
M := submatrix(M,nNR+1..nNR+nRL,1+nB+1..nC):
### We are only interested in the submatrix formed
### from the original matrix by taking the continuity
### of the relation rows
print('nNR'=nNR,'nRL'=nRL):
h := 0:
for j from 1 to nRL do
    u := 1:
    w := 1:
    for k from 1 to nS do
        x := isqrt(S[k]+N):
### recovering x from Q(x)
        u := u*x^M[j,k]:
### u = x1 . x2 . ... . xj
        w := w*S[k]^M[j,k]:
### w = Q(x1) . Q(x2) . ... Q(xj)
    od:
    v := isqrt(w):
    s := igcd(N,u+v):
    t := igcd(N,u-v):
    if s <> 1 and t <> 1 then
### s = 1 or t = 1 => factorization failed
h := h + 1:
if h = 1 then print([s,t]) fi:
###  h = 1 means we already have a factorization
    fi:
od:
print('Hits'=h):
### the number of times we succeeded
```

```
#######################################################################
#####  Number Field Sieve -- By the Book  #############################
#######################################################################

with(linalg,matrix,rowdim):  with(numtheory,legendre):

#######################################################################
#####  Subordinate procedures  ########################################
#######################################################################

defpolp := proc (n, d, m)       ###  Generate f from n, d, m  ###
local f, u, j, c:
f := 0:
u := n:
for j from 0 to d do
    c := modp(u,m):
    f := f + c*x^j:
    u := (u - c)/m:
od:
return(sort(f,x))
end:

#######################################################################

nnreln := proc (M, nr, nc)      ###  Count the non-relations  ###
local nn, r, c:
nn := 0:
for r from 1 to nr do
for c from 1 to nc do
    if M[r,c] <> 0 then nn := r fi
od od:
return(nn)
end:

#######################################################################

nfsqrt := proc (y, gg)       ###  y = RootOf(f),  gg in Z[y]  ###
local w, t:
w := factors(t^2-gg,y):
if rowdim(w[2]) = 1 then return(0)
                    else return(solve(w[2][1,1],t)) fi
end:

#######################################################################
#####  The main procedure  ############################################
#######################################################################

nfsv := proc (n, m, f, pMax, aMax, nQC, rSP)

local RFB, nRP, AFB, nAP, nCO, SP, nSP, QCB, nCM, nNR,
```

```
      nRL, X, Y, M, UV, alpha, Df, f1, a, a1, a2, b, c,
      e, g, gg, h, j, k, k1, k2,p, q, r, s, u, v, vv, w, qsok:

alpha := RootOf(f):  f1 := diff(f,x):  Df := discrim(f,x):

#########################################################################
#####  Construct factor bases RFB, AFB  #################################
#########################################################################

RFB := [ ]:                                  ###  Rational primes  ###
for p from 1 to pMax do
    if isprime(p) then
        RFB := [op(RFB),p]
    fi
od:
nRP := nops(RFB):

AFB := [ ]:                                 ###  Algebraic primes  ###
for p in RFB do
    if Df mod p <> 0 then
    for r from 0 to p-1 do
    if subs(x=r,f) mod p = 0 then
        AFB := [op(AFB),[p,r]]
    fi
    od
    fi
od:
nAP := nops(AFB):

nCO := 1 + nRP + 1 + nAP + nQC:

#########################################################################
#####  Construct list of smooth pairs SP  ###############################
#########################################################################

a1 := -aMax:  a2 := +aMax:

X := array(a1..a2):
Y := array(a1..a2):

SP := [ ]:  nSP := 0:  b := 0:

while nSP < rSP*nCO do

    b := b + 1:

    for a from a1 to a2 do
        X[a] := a - b*m:
        Y[a] := resultant(f,a-b*x,x):   ###  Norm (a - bx)  ###
    od:

    for p in RFB do  ###  Visit X[a] <==> a - bm = 0 mod p  ###
        k1 := ceil((a1-b*m)/p):
        k2 := floor((a2-b*m)/p):
```

```
        for k from k1 to k2 do
            a := b*m + k*p:
            if X[a] <> 0 then
            while X[a] mod p = 0 do
                X[a] := X[a] / p
            od
            fi
        od
    od:

    for w in AFB do  ###  Visit Y[a] <==> a - br = 0 mod p  ###
        p := w[1]:
        r := w[2]:
        k1 := ceil((a1-b*r)/p):
        k2 := floor((a2-b*r)/p):
        for k from k1 to k2 do
            a := b*r + k*p:
            if Y[a] <> 0 then
            while Y[a] mod p = 0 do
                Y[a] := Y[a] / p
            od
            fi
        od
    od:

    for a from a1 to a2 do
        if abs(X[a]) = 1 then
        if abs(Y[a]) = 1 then
        if igcd(a,b) = 1 then
            SP := [op(SP),[a,b]]:  nSP := nSP + 1:
        fi
        fi
        fi
    od:

od:

######################################################################
#####  Construct list of quadratic characters QCB  #############
######################################################################

QCB := [ ]:  q := pMax:
while nops(QCB) < nQC do
    q := nextprime(q):
    if igcd(q,n) = 1 then
        qsok := false:
        for s from 0 to q-1 do
        if not qsok then
            if subs(x=s,f) mod q = 0 then
            if subs(x=s,f1) mod q <> 0 then
                qsok := true:
                for w in SP do
                    a := w[1]:  b := w[2]:
                    if (a - b*s) mod q = 0 then qsok := false fi:
```

```
                od:
                if qsok then QCB := [op(QCB),[q,s]] fi:
            fi
            fi:
        fi:
        od:
    fi:
od:

#######################################################################
#####  Construct relation matrix M  ###################################
#######################################################################

M := matrix(nSP,nCO+nSP):  #  nCO = 1 + nRP + 1 + nAP + nQC  ###

for j from 1 to nSP do

    w := SP[j]:  a := w[1]:  b := w[2]:  u := a - b*m:

    if u < 0 then M[j,1] := 1 else M[j,1] := 0 fi:

    for k from 1 to nRP do
        p := RFB[k]:
        e := 0:
        while u mod p = 0 do
            u := u / p:
            e := e + 1
        od:
        M[j,1+k] := e mod 2
    od:

    u := resultant(f,a-b*x,x):              ###  Norm (a - bx)  ###

    if u < 0 then M[j,1+nRP+1] := 1 else M[j,1+nRP+1] := 0 fi:

    for k from 1 to nAP do
        w := AFB[k]:  p := w[1]:  r := w[2]:
        e := 0:
        if (a - b*r) mod p = 0 then
            while u mod p = 0 do
                u := u / p:
                e := e + 1
            od:
        fi:
        M[j,1+nRP+1+k] := e mod 2
    od:

    for k from 1 to nQC do
        w := QCB[k]:  q := w[1]:  s := w[2]:
        if legendre(a-b*s,q) = +1 then M[j,1+nRP+1+nAP+k] := 0
                                  else M[j,1+nRP+1+nAP+k] := 1 fi
    od:

    for k from 1 to nSP do
```

```
    if k = j then M[j,1+nRP+1+nAP+nQC+k] := 1
             else M[j,1+nRP+1+nAP+nQC+k] := 0 fi
    od:

od:

#########################################################################
#####  Reduce M modulo 2  ###############################################
#########################################################################

nCM := nCO + nSP:          ###  nCO = 1 + nRP + 1 + nAP + nQC  ###

r := 0:
for c from 1 to nCM do
    h := 0:
    for j from nSP by -1 to r+1 do
        if M[j,c] <> 0 then h := j fi
    od:
    if h <> 0 then
        r := r + 1:
        if h <> r then
            for k from c to nCM do
                M[r,k] := (M[r,k] + M[h,k]) mod 2
            od
        fi:
        for j from 1 to nSP do
            if j <> r then
            if M[j,c] <> 0 then
            for k from c to nCM do
                M[j,k] := (M[j,k] - M[r,k]) mod 2:
            od
            fi
            fi
        od
    fi:
od:

#########################################################################
#####  Construct pairs (u,v)  ###########################################
#########################################################################

nNR := nnreln(M,nSP,nCO):  nRL := nSP - nNR:  UV := [ ]:

for j from 1 to nRL do
    gg := 1:
    vv := 1:
    for k from 1 to nSP do
    if M[nNR+j,nCO+k] <> 0 then
        w := SP[k]:  a := w[1]:  b := w[2]:
        gg := rem(gg*(a - b*x),f,x):
vv := vv*(a - b*m):
    fi:
    od:
    g := nfsqrt(alpha,subs(x=alpha,gg)):
```

```
    u := subs(alpha=m,g) mod n:
    v := sqrt(vv) mod n:
    UV := [op(UV),[u,v]]:
od:

return(UV)

end:

#########################################################################
```

Bibliography

L.M. Adleman. A subexponential algorithm for the discrete logarithm problem with applications to cryptography. *Proc. of the 20th Annual IEEE Symposium on Foundations of Computer Science*, 55-60, 1979.

A.K. Lenstra, H.W. Lenstra Jr. *The development of the Number Field Sieve.* Springer-Verlag, New York, 1993.

Hardy & Wright. *An Introduction to the Theory of Numbers, 4th ed.* Oxford University Press, Oxford, 1960.

H.C. Williams, J.O. Shallit. Factoring integers before computers. *Mathematics of Computation 1943 - 1993 Fifty Years of Computational Mathematics (W. Gautshi, ed.) Proc. Sympos. Appl. Math*, 48, pp. 481 - 531, 1994.

J. Brillhart, M. Filaseta, A. Odlyzko. On an irreducibility theorm of a. cohen. *Can. J. Math*, 33, pp. 1055 - 1059, 1981.

M. Kraitchik. Théorie des nombres. *Gauthier-Villars*, 1, 1922.

M. Kraitchik. Recherche sur la théorie des nombres. *Gauthier-Villars*, 1924.

J. Brillhart M.A. Morrison. A method of factorization and the factorization of f_7. *Math. Comp.*, 29, pp. 183 - 205, 1975.

K. McCurley. The discrete logarithm problem. *Cryptology and Computational Number Theory*, 42: 49-74, 1990.

P.L. Montgomery. Number field sieve with two quadratic polynomials. *Centrum for Wiskunde en Informatica*, 1993.

P.L. Montgomery. Square roots of products of algebraic numbers. *Proc. Symp. in Appl. Math.*, 48, pp. 567-571, 1994a.

P.L. Montgomery. A survey of modern integer factorization algorithms. *CWI Quarterly* 7, 4, pp. 337 - 366, 1994b.

P.L. Montgomery. A block lanczos algorithm for finding dependencies over $gf(2)$. *EUROCRYPT '95, LNCS 921*, pp. 106, 120, 1995.

C. Pomerance. A tale of two sieves. *Notices of the AMS*, 43, Number 12, p. 1473-1485, 1996.

S. Pohlig, M. Hellman. An improved algorithm for computing logarithms over $gf(p)$ and its cryptographic significance. *IEEE Trans. on Information Theory*, 24, pp. 106-110, 1978.

O. Schirokauer. Discrete logarithms and local units. *Phil. Trans. R. Soc. Lond. A*, 345, pp. 409 - 423, 1993.

W. Diffie, M. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, 22, pp. 472 - 492, 1976.

D. Weber. *On the Computation of Discrete Logarithms in Finite Fields.* Dissertation, Universitat des Saarlandes, 1997.

M.V. Wilkes. *Time-sharing computer systems*, volume p. 91. American Elsevier, New York, 1968.