

California State University, San Bernardino

CSUSB ScholarWorks

Theses Digitization Project

John M. Pfau Library

2008

Factorization

Di Phan Reagan

Follow this and additional works at: <https://scholarworks.lib.csusb.edu/etd-project>



Part of the [Mathematics Commons](#)

Recommended Citation

Reagan, Di Phan, "Factorization" (2008). *Theses Digitization Project*. 3565.
<https://scholarworks.lib.csusb.edu/etd-project/3565>

This Thesis is brought to you for free and open access by the John M. Pfau Library at CSUSB ScholarWorks. It has been accepted for inclusion in Theses Digitization Project by an authorized administrator of CSUSB ScholarWorks. For more information, please contact scholarworks@csusb.edu.

FACTORIZATION

A Thesis

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Arts

in

Mathematics

by

Di Phan Reagan

December 2008

FACTORIZATION

A Thesis

Presented to the

Faculty of

California State University,


San Bernardino

by


Di Phan Reagan

December 2008


Approved by:



Chris Freiling, Committee Chair

12/2/2008
Date


Dan Rine, Committee Member


Laura Wallace, Committee Member


Peter Williams, Chair,
Department of Mathematics


Joe Chavez
Graduate Coordinator,
Department of Mathematics

ABSTRACT

Factorization and primality testing have blossomed in recent decades. This ancient factoring problem has a very important application in our modern society. The security of information transmission over the internet is dependent on the difficulty of factoring large numbers. Therefore this subject has become of great interest to government, business and those who are concerned with the secure transmission of information.

This paper reviews different methods of factoring. The focus will be on the two most efficient algorithms which are the Quadratic Sieve and Number Field Sieve. Background information such as definitions and theorems are given to help understand the concepts behind each method. Several examples are also given to help to illustrate the factorization process.

ACKNOWLEDGMENTS

I wish to thank Dr. Chris Freiling for all his help and support in the development and formation of this paper. I also want to thank my committee members Dr. Laura Wallace and Dr. Dan Rinne for their comments on the paper as well. Your assistance was very much appreciated.

Table of Contents

Abstract	iii
Acknowledgments	iv
1 Introduction	1
2 Different Methods of Factorization	3
2.1 Definitions and Theorems	3
2.2 Trial Division	4
2.3 Fermat's Algorithm	6
2.4 Pollard's Rho Method	7
2.5 Pollard's p-1 Method	9
2.6 Dixon's Algorithm	10
2.7 Quadratic Sieve	12
2.7.1 Initialization	12
2.7.2 Sieving	14
2.7.3 Linear Algebra	16
2.7.4 Factorization	17
2.7.5 Large Prime Variations and Multiple Polynomials	19
3 General Number Field Sieve	23
3.1 General Idea	23
3.2 Polynomial Selection	24
3.3 Sieving	24
3.3.1 The Rational Sieve	25
3.3.2 The Algebraic Sieve	25
3.4 Obstructions	26
3.5 Exponent Vectors	27
3.6 Matrix and Linear Algebra	30
3.7 Square Root in $Z[\alpha]$	30
Bibliography	35

Chapter 1

Introduction

The concept of prime numbers is quite old. It was first extensively studied by the ancient Greek mathematicians as early as 500 BC. By the time Euclid wrote the Elements in 300 BC, the concept of factorization of a composite number already existed. It is surprising that such an ancient topic has a very important application in our modern age. It is the RSA public key crypto-system that I am talking about. The name “RSA” comes from the initials of the originators, R.L. Rivest, A. Shamir and L.M. Adelman. RSA is one of many methods of encryption used to transmit secure information. It is based on Euler’s Theorem. The security of this method relies on the tremendous difficulty of factoring very large numbers.

The RSA encryption process starts with two distinct large primes p and q and their product $n = pq$. Let e be an integer that it is relatively prime to $\phi(n)$ where $\phi(n)$ is the Euler phi-function. From number theory we know there exists a unique $d \pmod{\phi(n)}$ such that $e \times d \equiv 1 \pmod{\phi(n)}$. The number e is called the encryption exponent while d is the decryption exponent. Only n and e are made available to the public, in particular to the sender. While the receiver is the one that created n from p and q , d from e through $e \times d \equiv 1 \pmod{\phi(n)}$. The receiver will use the private key d to retrieve the message.

Before a message is sent out, it is first converted to a number. Suppose each letter is assigned to a number, then a string of letters will be replaced with a string of numbers. To make a string of numbers easy to handle, it is necessary to keep the numbers in blocks. Each block is then converted to a single number. Let $M < n$ be a numerical version of the message block; it should be chosen relatively prime to n . The

sender uses n and e to encrypt the number M . Let E be the encrypted version of the message, defined by:

$$E = M^e \pmod{n}$$

To decode the message, one simply computes $E^d \pmod{n}$. To see how this works, we first recall Euler's theorem that says if M and n are relatively prime positive integers then $M^{\phi(n)} \equiv 1 \pmod{n}$. The original message M is then recovered by:

$$E^d \equiv (M^e)^d \equiv M^{ed} \equiv M^{1+k\phi(n)} \equiv M \times (M^{\phi(n)})^k \equiv M \times 1 \pmod{n}.$$

Notice that the private key d is needed above to reveal M . Since $e \times d \equiv 1 \pmod{\phi(n)}$, d is just the multiplicative inverse of e modulo $\phi(n)$ provided $\phi(n)$ is known. Because $\phi(n) = (p-1) \times (q-1)$, $\phi(n)$ can be calculated easily if p and q are known. The problem of decrypting the message therefore boils down to the factorization of n . The numbers p and q are chosen to be so large that modern methods cannot factor $n = pq$ in a reasonable amount of time. If an adversary were able to factor n , then the system would be broken.

In chapters 2 and 3, we look at different methods of factorization which include Trial Division, Fermat's algorithm, Pollard's Rho method, Pollard's $p-1$ method, Dixon's algorithm, Quadratic Sieve and General Number Field Sieve.

Chapter 2

Different Methods of Factorization

2.1 Definitions and Theorems

We start out this chapter with some of the definitions and theorems that we are going to use repeatedly throughout this paper. The following definitions and theorems can be found in the Elementary Number Theory book by James K. Strayer.

Definition 2.1. *A positive integer is said to be y -smooth if it does not have any prime factor exceeding y .*

Definition 2.2. *Let $n \in \mathbb{Z}$ with $n > 0$. The Euler phi-function, denoted $\phi(n)$, is the function defined by*

$$\phi(n) = |\{x \in \mathbb{Z} : 1 \leq x \leq n; \gcd(x, n) = 1\}|, \text{ where } |\cdot| \text{ denotes cardinality.}$$

Definition 2.3. *Let $x \in \mathbb{R}$ with $x > 0$. Then $\pi(x)$ is the function defined by*

$$\pi(x) = |\{p : p \text{ is prime; } 1 < p \leq x\}|.$$

Definition 2.4. *Let p be an odd prime number and let $a \in \mathbb{Z}$ with $p \nmid a$. The Legendre symbol, denoted $\left(\frac{a}{p}\right)$, is 1 if a is a quadratic residue modulo p . That is, if $x^2 \equiv a \pmod{p}$ for some $x \in \mathbb{Z}$. Otherwise, it is -1 if a is a quadratic nonresidue modulo p .*

Theorem 1. *(Fermat's Little Theorem)*

If p is an odd prime then

$$2^{p-1} \equiv 1 \pmod{p}.$$

Theorem 2. (General Form of Fermat's Theorem)

If p is a prime which does not divide b , then

$$b^{p-1} \equiv 1 \pmod{p}.$$

Theorem 3. (Euler's Theorem)

Let $a, m \in \mathbb{Z}$ with $m > 0$. If $\gcd(a, m) = 1$, then

$$a^{\phi(m)} \equiv 1 \pmod{m}.$$

Theorem 4. (Euler's Criterion)

Let p be an odd prime number and let $a \in \mathbb{Z}$ with $p \nmid a$. Then

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}.$$

Theorem 5. (Prime Number Theorem)

$$\lim_{x \rightarrow \infty} \frac{\pi(x) \ln x}{x} = 1.$$

For large x , the quantity $\frac{\pi(x) \ln x}{x} = 1$ is close to 1. That is to say the quantity $\pi(x)$ may be approximated by $\frac{x}{\ln x}$.

Theorem 6. (Chinese Remainder Theorem)

Let p_0, \dots, p_{l-1} be positive, pairwise coprime moduli with product $P = \prod_{i=0}^{l-1} p_i$. Let l respective residues x_i also be given. Then the system comprising the l relations and inequality

$$x \equiv x_i \pmod{p_i}, 0 \leq z < P$$

has a unique solution. Furthermore, this solution is given explicitly by the least nonnegative residue modulo P of

$$\sum_{i=0}^{l-1} x_i a_i P_i,$$

where $P_i = P/p_i$, and the a_i are inverses defined by $a_i P_i \equiv 1 \pmod{p_i}$.

2.2 Trial Division

Trial division is our first and simplest method for factoring an integer. Let n be the number to be factored. Trial division is based on the fact that if n has a factor other

than 1 and itself, then n must have a factor less than \sqrt{n} . We start out with a list of primes less than or equal to \sqrt{n} and try to divide them into n repeatedly. If none of the primes in the list divides into n evenly, then n is a prime. Otherwise, each time a prime divides n , we replace n by its quotient with that prime. Once we reach the point where the remaining unfactored portion is less than the square of the prime that we last used, then the unfactored portion is a prime, or else it is 1. In either case the factorization is complete.

For example, let $n = 3948$, $\sqrt{n} \approx 62$. We are going to start out with 2. Since 2 is a factor of n , we divide 2 out and the quotient is 1974. We realize that 2 still goes into n . We divide by 2 again and we have the quotient 987. Since 2 doesn't go into the remaining unfactored portion, we try the next prime 3. We divide 987 by 3 and the quotient is 329. Continuing in the same manner we find the next prime factor that goes into the remaining portion is 7 and the quotient is 47. Realizing that $47 < 7^2$, then 47 must be a prime. Therefore $n = 2^2 \times 3 \times 7 \times 47$.

In this method, all trial divisors do not have to be primes. Here is an example. Let $n = 774$ be the number to be factored. We trial divide by 2 and realize it is a divisor. Divide 2 into n and the quotient is 387. Since 2 does not go into the remaining unfactored portion, we try the next number which is 3. Divide 3 into 387 and the quotient is 129. The factor 3 goes into 129 one more time and the quotient is 43. This time we just divide 4, 5, 6 \dots consecutively into n without worrying whether they are primes. We see that 6 does not go into n evenly just simply means that prime factors of 6, which is 2 and 3 are already factored out of n previously. So dividing by 6 is a waste of time but it saves us from checking whether 6 is prime or not. The next trial is 7 and $7^2 > 43$, so therefore 43 is a prime. We have the complete factorization $774 = 2 \times 3^2 \times 43$. This version may take longer but it does end up with the factorization of n and is easier to apply. Since all the primes are odd except 2, we could compromise using 2 and all the odd numbers for trial divisors to speed up the process.

Trial division can be used for factoring or primality testing provided the number n is not too large. With a modern workstation, a number from 13-19 digits base 10 can be factored or proven to be a prime in less than one minute. Trial division can also be used to recognize smooth numbers. Recall that a number is said to be B-smooth if all of its primes in the factorization are less than or equal to B.

So how long does it take to factor a number n using trial division? The worst case is when n is a prime itself since we have to trial divide all numbers up to \sqrt{n} . If we only use prime divisors then it would take approximately $\pi(\sqrt{n}) \approx \frac{\sqrt{n}}{\ln(\sqrt{n})} = \frac{\sqrt{n}}{\frac{1}{2}\ln n} = \frac{2\sqrt{n}}{\ln n}$ divisions, by the prime number theorem. If we only use 2 and all the odd numbers as trial divisors then it would take approximately $\sqrt{n}/2$ divisions.

2.3 Fermat's Algorithm

Let n be the number to be factored. If n can be written in the form $n = x^2 - y^2$, then n can be immediately factored as $(x+y)(x-y)$. If $x-y > 1$, then we have succeeded in factoring n into two smaller factors. We notice that if n is odd and is also a product of two integers, then n can always be expressed as the difference of two perfect squares. To see this, let $n = ab$, where a, b are positive odd integers. Let

$$x = (a+b)/2 \text{ and } y = (a-b)/2.$$

Then $x^2 - y^2 = \frac{(a+b)^2 - (a-b)^2}{4} = ab = n$. Fermat's algorithm starts with x from $\lceil\sqrt{n}\rceil$, $\lceil\sqrt{n}\rceil + 1, \dots$ and checks whether $x^2 - n$ is a square, say y^2 . If that is the case, then $x^2 - y^2 = n$ or $n = (x+y)(x-y)$.

For example, let $n = 551$ be the number to be factored, $\lceil\sqrt{551}\rceil = 24$. We notice that $24^2 - 551 = 5^2$ or $(24+5)(24-5) = 551$. Therefore $551 = 29 \times 19$.

If a and b are primes, there will be a 50-50 chance that n will be factored. To see how this works we notice that with the above conditions, we have $x^2 \equiv y^2 \pmod{a}$ and $x^2 \equiv y^2 \pmod{b}$. Therefore $a \mid x^2 - y^2$ and $b \mid x^2 - y^2$, equivalently $a \mid (x-y)$ or $a \mid x+y$. Also $b \mid (x-y)$ or $b \mid x+y$. We have four cases to consider,

Case 1:

If $a \mid x-y$ and $b \mid x-y$, then $n \mid x-y$.

and $\gcd(n, x-y) = n$. We do not have a factoring of n .

Case 2:

If $a \mid x-y$, $a \nmid x+y$ and $b \mid x+y$, $b \nmid x-y$,

then $\gcd(n, x-y) = a$. We have found a factor a of n .

Case 3:

If $a \mid x + y$, $a \nmid x - y$ and $b \mid x - y$, $b \nmid x + y$,
then $\gcd(n, x - y) = b$. We have found a factor b of n .

Case 4:

If $a \mid x + y$, $a \nmid x - y$ and $b \mid x + y$, $b \nmid x - y$,
then $\gcd(n, x - y) = 1$. We do not have a factoring of n .

2.4 Pollard's Rho Method

The Pollard Rho factorization algorithm was introduced in 1975 [CP01]. It works well for numbers that have moderately sized prime divisors, around 10^5 to 10^{10} . When the number to be factored has prime divisors that are too big for trial division, this method may be useful since it is easy to understand and does not take a lot of storage in the computer. Once all the prime divisors are bigger than 10^{12} , we have to rely on other methods like the Quadratic Sieve Algorithm, the General Number Field Sieve, etc.

Let n be a composite integer that has a nontrivial divisor p . As an example, we let $n = 1313$. Consider a simple irreducible polynomial in x , like $f(x) = x^2 + 1$. Starting with a random integer $x_0 = 1$, we can create a sequence from the recursive definition:

$$x_i = f(x_{i-1}) \bmod n.$$

We get the sequence $x_1 = 2, x_2 = 5, x_3 = 26, x_4 = 677, x_5 = 93, x_6 = 772, x_7 = 1196, x_8 = 560, x_9 = 1107, x_{10} = 421, x_{11} = 1300, x_{12} = 170, x_{13} = 15, x_{14} = 226, x_{15} = 1183, \dots$

Since n is finite, there are only finite number of congruence classes modulo n . The above sequence will eventually have a repeat term and become cyclic. This behavior is therefore associated with the oval part of the Greek letter “ ρ ”, whereas the precyclic part is associated with the tail of the “ ρ ”. According to the birthday paradox [CP01], we expect to have a repeat term in approximately \sqrt{n} steps, which is about the same as trial division.

Here is a better way. Choose a factor p of n and denote $y_i = x_i \bmod p$. If we knew p (for example, $p = 13$) we could create the y_i 's as follows, where $y_{i+1} = f(y_i) \pmod{p}$

p): $y_0 = 1, y_1 = 2, y_2 = 5, y_3 = 0, y_4 = 1, y_5 = 2, y_6 = 5, y_7 = 0, y_8 = 1, y_9 = 2, y_{10} = 5, y_{11} = 0 \dots$

Since there are less congruence classes in modulo 13, we see more repeated terms in the y_i 's. It only takes 4 steps for the sequence to repeat this time. When $y_i = y_j$, then $x_i \equiv x_j \pmod{p}$. Therefore p divides $x_i - x_j$. Since p is also a factor of n , there is a good chance that $\gcd(n, x_i - x_j)$ is a non-trivial divisor of n . However, since we do not know the factor p , we have no access to the y_i sequence, therefore we have no idea when y_i will equal to y_j . Note that we do not need to know the values of y_i and y_j , we just need to determine two indices i, j where $y_i = y_j$.

So how do we go about searching for pairs (i, j) such that $y_i = y_j$ in order to compute $\gcd(n, x_i - x_j)$? The first cycle-finding method is called the Floyd cycle-finding algorithm [CP01]. Suppose $i < j$. We notice that if $y_i = y_j$, then for $m \geq i$, $y_m = y_{m+(j-i)} = y_{m+2(j-i)} \dots = y_{m+n(j-i)}$. Let $m \geq i$ such that m is divisible by $(j - i)$, so $y_m = y_{2m}$. The basic idea of the Pollard's Rho method is that instead of searching for all pairs of (i, j) and computing $\gcd(x_i - x_j, n)$, we will compute the sequence $\gcd(x_i - x_{2i}, n)$ until something other than 1 or n is found. One of the advantages of this method is that very little space is required. We only need to keep in memory the number n which is the number to be factored and the current pair x_i and x_{2i} . Even though many x_i 's need to be calculated twice, it is much better than trying to store all the x_i 's in an array. With this method, we are able to factor $n = 1313$ successfully by finding $\gcd(x_8 - x_4, n) = \gcd(560 - 677, 1313) = 13$. Therefore $1313 = 13 \times 101$.

Another form of a cycle-finding algorithm is due to R.P. Brent [Bre89]. As in the Floyd method, it does not store all the x_i 's but looks at the differences: $x_1 - x_3, x_3 - x_6, x_6 - x_{12}, \dots, x_{2^{n-1}} - x_j$ where $(2^{n+1} - 2^{n-1} \leq j \leq 2^{n+1} - 1)$.

This gives a systematic way of choosing a lot of pairs (i, j) to compute the $\gcd(x_i - x_j, n)$ by using each difference $j - i$ once and letting $i \rightarrow \infty$ at the same time. With this method, we are able to factor $n = 1313$ successfully by finding $\gcd(x_3 - x_7, n) = \gcd(1196 - 26, 1313) = 13$. Therefore $1313 = 13 \times 101$.

In both the Brent and the Floyd method, we have to compute $\gcd(x_i - x_j, n)$ many times to find a non-trivial divisor of n . We can save work by doing it in blocks. For example, we can compute ten successive values of $(x_i - x_j) \pmod{n}$ and then take the gcd of n with that product. Sometimes the gcd will be n . If that is the case, we may have

to go back to take the gcd of each factor individually with n in order to recover p . For example, $\prod (x_i - x_j) \pmod{1313} = 0$ with $1 \leq i \leq 15$ and $3 \leq j \leq 26$. The gcd of 1313 with the product of these ten successive values of $(x_i - x_j)$ will therefore be 1313. By going back to take the gcd of each factor with 1313, we are able to factor 1313 by using either $gcd(x_3 - x_7, n) = gcd(1196 - 26, 1313) = 13$ or $gcd(x_7 - x_{15}, n) = gcd(1196 - 1183, 1313) = 13$.

2.5 Pollard's p-1 Method

This algorithm was invented by John Pollard in 1974 and based on Fermat's Little Theorem which says that if p is an odd prime, then $2^{p-1} \equiv 1 \pmod{p}$. Therefore if $p - 1$ is a factor of M , then we also have $2^M \equiv 1 \pmod{p}$ due to Fermat's Theorem. Equivalently $p \mid 2^M - 1$. Let n be the integer to be factored and let p be one of its prime factors. We have p divides both n and $2^M - 1$. There is a good chance that n does not divide $2^M - 1$, in which case, $gcd(2^M - 1, n)$ is a nontrivial factor of n . To speed up the computation, we can take $gcd((2^M - 1) \pmod{n}, n)$ instead of $gcd(2^M - 1, n)$. Since exponentiation modulo n is very fast, this algorithm can find potential factors with great efficiency. Pollard's idea is to choose M so that it has many factors that are 1 less than a prime number. The suggestion is to let M be the least common multiple of the integers up to B for some choice of B . Therefore $M = lcm(1, 2, \dots, B) = \prod \{(p^\alpha) \mid p^\alpha \leq B\}$.

For example, let $n = 527$ be the number to be factored, let $B = 10$. The least common multiple of the integers up to 10 is $M(10) = 2^3 \times 3^2 \times 5 \times 7$. We want to compute $gcd(2^{2^3 \times 3^2 \times 5 \times 7} \pmod{527}, 527)$. Unfortunately this gcd turns out to be 527. For many cases, we can increase the bound B . In this case it does not work because $((2^{2^3})^{3^2})^5 \equiv 1 \pmod{527}$. The gcd in this case will always end up to be n no matter how high we increase the bound B . Notice that there is nothing special about the number 2 in this method. The number 2 can just be replaced with any a such that it is relatively prime to n . This time we want to try $a = 3$, $gcd(3^{2^3 \times 3^2 \times 5 \times 7} \pmod{527}, 527) = 31$. Therefore $527 = 31 \times 17$.

We notice from the above example that this method sometimes fails to give nontrivial factor of n . The $gcd(a^M - 1 \pmod{n}, n)$ sometimes yields 1 or n . In practice, the situation that happens more often is that the $gcd(a^M - 1 \pmod{n}, n) = 1$ and we usually deal with it by expanding the bound B and applying an extension called the

second stage. Let B' be the second bound, bigger than B . Let all the primes in $(B, B']$ be $Q_1 < Q_2 < \dots$. Previously we use the exponents $M(B)$. We now continue with all the exponents of the form $QM(B)$ with $Q \in (B, B']$. Notice that $QM(B) \mid M(B')$. Therefore what we are doing here is not the same as raising bound B to B' and trying to compute $\gcd(a^{M(B')} - 1 \pmod n, n)$ as above. What we are doing now is trying to retrieve more factors p of n with $p - 1$ of the form Qm where m is a factor of $M(B)$. Notice that $2^{Q_i M} \pmod n$ is fairly easy to find by recursion. For example, after we find the initial value $2^{Q_1 M(B)} \pmod n$, $2^{Q_2 M(B)} \pmod n$ can be found simply by multiplying $2^{Q_1 M(B)} \pmod n$ with $2^{(Q_2 - Q_1)M(B)} \pmod n$. Basically it is inexpensive to do this additional stage since the differences of the Q_i are much smaller than Q_i themselves and all the $2^{(Q_i - Q_{i-1})M(B)}$ can be precalculated.

The two above algorithms, Pollard Rho and Pollard $p-1$, are called probabilistic algorithms. We are no longer sure that they will succeed. However, when they don't succeed, we can often change parameters. It is an art to find the right parameter for these algorithms. For the Pollard Rho method, we can replace the function $x^2 + 1$ with any irreducible like $x^2 + 2$ or $x^2 + 3$. We can vary the parameter for the Pollard $p-1$ method by changing the base a and the smoothness bound B or apply second stage as described above.

2.6 Dixon's Algorithm

In the next two topics, we are going to focus on two methods that are considered the best for factoring much larger numbers. These two methods are the Quadratic Sieve and the Number Field Sieve. Before we go on to discuss the Quadratic Sieve, we are going to focus on a similar yet easier method called Dixon's algorithm. It is based on Fermat's idea that if we can find two random integers x and y such that $x^2 \equiv y^2 \pmod n$ then we can often factor n by finding $\gcd(x - y, n)$.

Dixon's Algorithm starts by letting $f(x) = x^2 \pmod n$. If we can find x such that $f(x) = y^2$ is a perfect square over the integers, then n may be factored since $x^2 \equiv y^2 \pmod n$. A perfect square $f(x)$ will be achieved through the means of exponent vectors, which we will now describe. If $f(x)$ is factored completely, then it has the form

$$f(x) = p_1^{e_1} \times p_2^{e_2} \times \dots \times p_m^{e_m}$$

We call (e_1, e_2, \dots, e_m) the exponent vector of $f(x)$. In the case that $f(x)$ is a perfect square, all the e_i 's will be even; usually most of them will be zero in any factorization of $f(x)$. The idea is to force this to happen by multiplying different $f(x)$'s together. For example, if $f(x_1) = p_1^{e_1} \times p_2^{e_2} \times \dots \times p_m^{e_m}$, $f(x_2) = p_1^{d_1} \times p_2^{d_2} \times \dots \times p_m^{d_m}$, then $f(x_1 x_2) = f(x_1)f(x_2) = p_1^{e_1+d_1} \times p_2^{e_2+d_2} \times \dots \times p_m^{e_m+d_m}$. Let $v(x)$ denote the exponent vector (mod 2) where $v(x) = (e_1 \pmod{2}, \dots, e_m \pmod{2})$ if $f(x) = p_1^{e_1} \times p_2^{e_2} \times \dots \times p_m^{e_m}$. Therefore $f(x_1 x_2) = f(x_1)f(x_2)$ is a square if and only if $\sum v(x_i)$ has zero entries.

Our plan is to choose a suitable smoothness bound B , then find several $f(x)$ that are B -smooth. We will record their exponent vectors $v(x)$. Then we will do Gaussian elimination modulo 2 on these vectors to find a subset whose sum is zero. From a linear algebra perspective, our goal boils down to finding linear dependency of the vectors $v(x)$. We know that a set of vectors must be linearly dependent when there are more of them than the dimension of the vector space. Therefore a sufficient condition for the existence of a product of $f(x)$'s to be square is having at least $\pi(B) + 1$ entries of $f(x)$ that are B -smooth.

After we find a collection of $v(x_i)$ where the sum of their entries are zero (mod 2), the product of corresponding $f(x_i)$'s will be a perfect square. Combining the $f(x_i)$ we have:

$$y^2 = f(x_1) \times f(x_2) \times \dots \times f(x_k) \equiv x_1^2 \times x_2^2 \times \dots \times x_k^2 \pmod{n}$$

or

$$y^2 \equiv (x_1 x_2 \dots x_k)^2 \pmod{n}.$$

Use Fermat's method by computing $\gcd(y - (x_1 x_2 \dots x_k), n)$ to figure out the factor of n . For example, let $n = 589$ be the number to be factored and $B = 10$. We only want to keep the $f(x)$ that factor into primes smaller than 10: $f(20) = 2^4 \times 5^2$, $f(21) = 3^2 \times 7^2$, $f(24) = 2^6 \times 3^2$, $f(25) = 2^2 \times 3^2$, $f(27) = 2^2 \times 5 \times 7$, $f(29) = 2^2 \times 3^2 \times 7$, $f(33) = 2^2 \times 5^3$, $f(34) = 3^4 \times 7$.

Right away we can see that $f(24) \times f(25) = (2^4 \times 3^2)^2$ is a perfect square. Therefore $19 = \gcd(24 \times 25 - 2^4 \times 3^2, 589)$ is a factor of $n = 589$. Similarly we can pair $f(34)$ with $f(29)$ which gives us $(2 \times 3^3 \times 7)^2$. Therefore $19 = \gcd(34 \times 29 - 2 \times 3^3 \times 7, 589)$.

The problem with this method is finding B -smooth values of $f(x)$. For a random x , the chance for $f(x)$ to be factored completely over the factor base is small if n is

large. That is why the next method, the Quadratic Sieve, becomes an improvement of Dixon's Algorithm. Nevertheless, for large enough values of n , Dixon's method beats Trial Division as well as the two Pollard methods.

2.7 Quadratic Sieve

As mentioned earlier, the Quadratic Sieve is associated with Dixon's Algorithm but a sieving procedure is incorporated in the method in order to find a collection of $f(x_i)$ that are B -smooth. Unlike Dixon's Algorithm that starts with a sequence of $f(x_i) = x_i^2 \pmod{n}$, the Quadratic Sieve computes $x^2 - n$ where x starts from the value $\lceil \sqrt{n} \rceil$ in order to keep $x^2 - n$ close to zero. The idea is that the smaller the value of $x^2 - n$, the more likely that it will be smooth. The goal is to obtain a sequence of smooth numbers of the form $x^2 - n$. Then, as in Dixon's Algorithm, we use linear algebra to find a subsequence $x_1^2 - n, x_2^2 - n, x_3^2 - n, \dots, x_k^2 - n$ where their product is a perfect square. Denote $\prod_1^k (x_i^2 - n) = a^2$, and $\prod_1^k x_i \pmod{n} = b$, therefore $a^2 \equiv b^2 \pmod{n}$. If $a \not\equiv \pm b \pmod{n}$, we can find a factor of n by computing $\gcd(a - b, n)$. The Quadratic Sieve has four steps: initialization, sieving, linear algebra, factorization.

2.7.1 Initialization

In this step, we need to set up the factor base which involves deciding on the bound, B . If B is chosen to be small, we don't have to find too many B -smooth values of $x^2 - n$ in order to produce a subset product that is a square. In addition, the matrix for the linear algebra step discussed later will be small. But B -smooth values of $x^2 - n$ are so special that we have to search hard for even one entry. On the other hand, if B is chosen to be large, we will more easily find them. Remember that our goal is to find a sequence of $x_i^2 - n$ that is B -smooth and combine them to create a square. Therefore finding B -smooth values of $x^2 - n$ may not be hard, but finding enough B -smooth values to find a dependency will be difficult. In addition, the matrix in the linear algebra step will be quite large. So it is a matter of balancing out these two conflicting forces.

The factor base consists of primes p up to B . If p divides $x^2 - n$ then $x^2 \equiv n \pmod{p}$. In other words, n is a quadratic residue. In this case, the Legendre symbol $\left(\frac{n}{p}\right) = 1$. There will be exactly $\frac{p-1}{2}$ incongruent quadratic residues and the same amount

for quadratic nonresidues for any prime $p \leq B$. We can use Euler's Criterion to detect those primes that would make n a quadratic residue.

Theorem 7. (*Euler's Criterion*)

Let p be an odd prime number and let $n \in \mathbb{Z}$ with $p \nmid n$. Then

$$\left(\frac{n}{p}\right) \equiv n^{(p-1)/2} \pmod{p}$$

As an example, we are going to try to factor $n = 18079$. Suppose we choose the smoothness bound B to be 40. The factor base would consist of 2, 3, 5, 13, 17, 23 since their Legendre symbols equal to 1. For example 5 belongs to the factor base since its Legendre symbol $\left(\frac{n}{p}\right) \equiv 18079^{(5-1)/2} \pmod{5} = 1$. Similarly, the rest of the primes smaller than $B = 40$ have $\left(\frac{n}{p}\right) = -1$. We also want to include -1 in the factor base since that allows us to choose $x < \sqrt{n}$ and $x^2 - n < 0$.

As preparation for the sieving step, we need to figure out for what values of x does $p \mid x^2 - n$. That is we need to solve the congruences $x^2 \equiv n \pmod{p}$ for all the p in the factor base. Since $g(x) = x^2 - n$ may be divisible by p more than once, we will also solve $x^2 \equiv n \pmod{p^a}$.

For the first prime 2 and the odd n , we realize that $x^2 - n$ is divisible by 2 when x is odd. When $n \equiv 3$ or $7 \pmod{8}$ then $x^2 - n$ is divisible by 2 but not divisible by any higher power of 2. When $n \equiv 5 \pmod{8}$ then $x^2 - n$ is divisible by 4 but not divisible by 8. When $n \equiv 1 \pmod{8}$ then $8 \mid x^2 - n$. So $n \equiv 1 \pmod{8}$ is the most general case among the three and there is a way to convert the first two cases to the general one. For $n \equiv 5 \pmod{8}$ then multiply it with 5 to get $5n \equiv 25 \pmod{8}$ or $5n \equiv 1 \pmod{8}$. Similarly, if $n \equiv 3 \pmod{8}$ then multiply it with 3, and if it is congruent to 7 then multiply it by 7.

For $p \equiv 3 \pmod{4}$ or $p \equiv 5 \pmod{8}$, we can use the following theorem to solve for x .

Theorem 8. Let n be a quadratic residue modulo the prime p .

1. If $p = 4k + 3$, then $x \equiv n^{k+1} \pmod{p}$.
2. If $p = 8k + 5$ and $n^{2k+1} \equiv 1 \pmod{p}$, then $x \equiv n^{k+1} \pmod{p}$.
3. If $p = 8k + 5$ and $n^{2k+1} \equiv -1 \pmod{p}$, then $x \equiv (4n)^{k+1} \times \left(\frac{p+1}{2}\right) \pmod{p}$.

The following theorem is slightly slower than Theorem 7 but it can be used for any odd prime.

Theorem 9. *Let n be a quadratic residue modulo an odd prime p and let h be chosen so that the Legendre symbol $\left(\frac{h^2-4n}{p}\right)$ is -1 . Define a sequence v_1, v_2, \dots by the recursion*

$$v_1 = h$$

$$v_2 = h^2 - 2n$$

$$v_i = h \times v_{i-1} - n \times v_{i-2}.$$

Then we have

$$v_{2i} = v_i^2 - 2n^i$$

and

$$v_{2i+1} = v_i \times v_{i+1} - h \times n^i.$$

The solution to congruence $x^2 \equiv n \pmod{p}$ is : $x \equiv v_{(p+1)/2} \times \left(\frac{p+1}{2}\right) \pmod{p}$.

As mentioned earlier, we need to solve the congruences $x^2 \equiv n \pmod{p}$ for all p in the factor base. For example, we can use Theorem 7 to solve $x^2 \equiv 18079 \pmod{13}$. Since $p = 13 = 8k + 5$ with $k = 1$ and $n^{2k+1} = 18079^{2 \cdot 1 + 1} \equiv 1 \pmod{13}$, then $x \equiv n^{k+1} \pmod{p} = 18079^{(1+1)} \pmod{13} \equiv 3 \pmod{13}$.

2.7.2 Sieving

The purpose of this step is to locate smooth values for $x^2 - n$ as x changes. It works similar to the sieve of Eratosthenes. We are first going to review this sieve. Suppose we want to find all prime numbers less than or equal to certain bound X . By a lemma in number theory we know that if X is a composite number then X has a prime divisor less than or equal to \sqrt{X} . From a list of integers from 2 to X , we cross out all the multiples of all the primes up to \sqrt{X} but not the primes themselves. All the numbers that are left unmarked are primes. For the sieving step in Quadratic Sieve algorithm, we are only interested in the marked numbers. What it means is the more marked a number, the more primes that number is divisible by.

In order to locate the values of x such that $x^2 - n$ is divisible by p , we solve the congruence $x^2 - n \equiv 0 \pmod{p}$ as mentioned in the previous section. Once we find

the first values of x , $x = x_1$ and $x_2 = p - x_1$, for which $p \mid g(x)$, we can spot the other values of x with $p \mid g(x)$ by simply adding p to the first locations. A simple computation can explain why we can add p 's value to the first $g(x)$ that is divisible by p and the new entries $g(x+p)$ are still divisible by p . We have $x^2 - n \equiv 0 \pmod{p}$ or $x^2 - n = kp$. Then $g(x+p) = (x+p)^2 - n = x^2 + 2xp + p^2 - n = (x^2 - n) + (2xp + p^2) = kp + p(2x+p)$. So that $g(x+p)$ is divisible by p .

From the above example we have $x_1 = 3$ is the first solution to the congruence $x^2 - 18079 \equiv 0 \pmod{13}$. Since x starts from $\lfloor \sqrt{18079} \rfloor = 134$, the first value for $x^2 - n \equiv 0 \pmod{13}$ is $x_1 = 3 + 11 \cdot 13 = 146$, and the second value is $x_2 = 13 - 146 = -133 \equiv 140 \pmod{13}$. We have two paths to branch off starting from the initial solutions of the congruence $g(x) = x^2 - n$ to find the remaining locations for the two residue classes. For $x_1 = 146$, we have $x^2 - 18079 \equiv 0 \pmod{13}$, and the next place that $x^2 - 18079$ is divisible by 13 is $g(146 + 13) = g(159) = 159^2 - 18079 \equiv 0 \pmod{13}$. Similarly, the next place that $g(x)$ is divisible by 13 after the initial value $x_2 = 140$ is $g(140 + 13) = g(153)$. The same procedure is done for all the primes in the factor base.

As mentioned earlier, we are interested only in $g(x)$ entries that have a lot of marked primes, preferably small primes. The Quadratic Sieve helps to recognize smooth values of $g(x) = x^2 - n$. The sieve starts with values of $g(x)$. Every time that each $g(x)$ is divisible by a prime in the factor base, we replace the current value of $g(x)$ with its quotient by that prime. By the time we are done sieving values of $g(x)$ through the factor base, those that are left with value of 1 are B-smooth. Instead of using division, we can subtract $\log p$ from $\log(x^2 - n)$ each time p divides the corresponding $g(x)$. By the end of the sieving process, such smooth $g(x)$ will have value close to zero. Continued from the example above, we find the following values of $g(x) = x^2 - n$ that completely factor over the factor base after the sieving step:

$$139 = 2 \times 3^3 \times 23$$

$$148 = 3^2 \times 5^2 \times 17$$

$$158 = 3^4 \times 5 \times 17$$

$$166 = 3^6 \times 13$$

$$185 = 2 \times 3^3 \times 13 \times 23$$

$$192 = 5 \times 13 \times 17^2$$

$$198 = 5^3 \times 13^2$$

2.7.3 Linear Algebra

After the last step, we should have a collection of $g(x)$ values that factor completely over factor base. The goal of this next step is to use linear algebra to find a subset of these values such that their product is a square. Similar to what was mentioned earlier in section 2.6, each $g(x)$ if factored completely, can be expressed as:

$$g(x) = p_1^{e_1} \times p_2^{e_2} \times \cdots \times p_{\pi(B)}^{e_{\pi(B)}}$$

The factorization of each value $g(x)$ is recorded as:

$$v(x) = (e_1, e_2, \cdots, e_{\pi(B)})$$

Where e_i will be 0 if it is even, 1 if it is odd. Therefore each $g(x)$ is represented as a sequence of 0's and 1's. For example, if our factor base is $\{-1, 2, 3, 5, 13, 17, 23\}$ and $g(x) = 2 \times 5^3 \times 13^2$ then it is represented as $(0, 1, 0, 1, 0, 0, 0)$.

Finding a subset of $g(x)$'s such that their product is a square is therefore the same as finding those with their corresponding exponent vectors adding up to 0 (mod 2). The problem boils down to finding linear dependency in the set of vectors. We need to find more values of $g(x)$ than the number of elements in the factor base in order to ensure the dependency. If the bound for the smoothness is B , then $\pi(B) + 1$ B-smooth values of $g(x)$ would be sufficient. The task at hand now is to set up the matrix formed with these vectors.

Denote the matrix we are going to form by A . The rows of the matrix will be binary exponent vectors corresponding to the $\pi(B) + 1$ values of $g(x)$ that are B-smooth. Whereas the columns correspond to primes in the factor base. Notice that the first column of zeros that corresponds to positive signs of seven values of $g(x)$ is omitted for easy computation. All we need to do now is to look for x such that $A^T x = 0$. This problem can be solved by Gaussian elimination of the matrix A .

The matrices corresponding to the above smooth values of $g(x)$ are:

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$A^T = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

Using row reduction operations, the reduced row-echelon form of A^T is:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Assigning values to free variables, one of the solutions that we come up with is $(0, 1, 1, 0, 0, 0, 1)^T$ which implies that the sum of the second, third and seventh columns is zero. Therefore, from the solution we can tell what linear combination of the $g(x)$'s would give us the square, namely $148 \times 158 \times 198 = (3^3 \times 5 \times 3 \times 17 \times 13)^2$. Another solution that we have is $(1, 0, 0, 1, 1, 0, 0)^T$, corresponding to $139 \times 166 \times 185 = (2 \times 3^6 \times 13 \times 23)^2$.

2.7.4 Factorization

Up to this point, we have found a subset of $g(x) = x^2 - n$, whose product $(x_1^2 - n)(x_2^2 - n) \cdots (x_k^2 - n)$ is a square. From the exponent vectors of the $x_i^2 - n$, we can

calculate the prime factorization of the product $(x_1^2 - n)(x_2^2 - n) \cdots (x_k^2 - n)$ and therefore $\sqrt{(x_1^2 - n)(x_2^2 - n) \cdots (x_k^2 - n)}$.

Denote $a = \sqrt{(x_1^2 - n)(x_2^2 - n) \cdots (x_k^2 - n)} \pmod{n}$ and $b = x_1 x_2 \cdots x_k \pmod{n}$. We have $a^2 \equiv b^2 \pmod{n}$. If $a \not\equiv \pm b \pmod{n}$, then n can be factored by $\gcd(a - b, n)$.

From the above example, corresponding to the solution $(1, 0, 0, 1, 1, 0, 0)^T$ we have:

$$g(139) = 139^2 - n = 2 \times 3^3 \times 23$$

$$g(166) = 166^2 - n = 3^6 \times 13$$

$$g(185) = 185^2 - n = 2 \times 3^3 \times 13 \times 23$$

$$a = \sqrt{(139^2 - n)(166^2 - n)(185^2 - n)} \pmod{n}$$

$$a = 2 \times 3^6 \times 13 \times 23 \pmod{n}$$

$$a = 2046 \pmod{n}$$

$$b = 139 \times 166 \times 185 \pmod{n}$$

$$b = 2046 \pmod{n}$$

Unfortunately, $a \equiv b \pmod{n}$, so we cannot find the nontrivial factor of n by computing $\gcd(a - b, n)$. Corresponding to the other solution $(0, 1, 1, 0, 0, 0, 1)^T$ above, we have:

$$g(148) = 148^2 - n = 3^2 \times 5^2 \times 17$$

$$g(158) = 158^2 - n = 3^4 \times 5 \times 17$$

$$g(198) = 198^2 - n = 5^3 \times 13^2$$

$$a = \sqrt{(148^2 - n)(158^2 - n)(198^2 - n)} \pmod{n}$$

$$a = (3^3 \times 5^3 \times 13 \times 17)^2 \pmod{n}$$

$$a = 4636 \pmod{n}$$

$$b = 148 \times 158 \times 198 \pmod{n}$$

$$b = 1808 \pmod{n}$$

Since $a \not\equiv \pm b \pmod{n}$, n will then be factored by computing $\gcd(4636 - 1808, 18079) = 101$.

Therefore $18079 = 101 \times 179$.

2.7.5 Large Prime Variations and Multiple Polynomials

Among many suggestions for improvement, the two refinements that have been proved to improve the running time are the large prime variation and the multiple polynomial version [Bre89].

Based on the idea that if we remove all the primes up to B in the factorization of a number, the remaining factor of that number is a prime provided it is less than B^2 . With that in mind, we can utilize those numbers that are almost B -smooth except they have one slightly larger prime than B . The easy way to get rid of that large prime factor is to pair it up with another number with the same large prime factor. As a result, it is necessary to keep track of the large prime factors. If it just appears once, we discard it since we cannot use it to make a square. Notice that allowing one large prime in the interval $(B, B^2]$ for this variation is not the same as increasing the smoothness bound to B^2 . As a result we should not view this type of number as having long exponent vectors.

Suppose we have a pair of $x_i^2 - n$ values that satisfy the above condition, namely $x_1^2 - n = \prod p_i^{e_i} P(\text{mod } n)$, $x_2^2 - n = \prod p_i^{d_i} P(\text{mod } n)$ where $B < P < B^2$ and $p_i < B$. Then $(x_1 x_2)^2 \equiv \prod p_i^{e_i + d_i} P^2 \pmod{n}$. Since the exponent vectors are reduced mod 2, the contribution of P^2 to the exponent vector doesn't matter because it is reduced to 0 anyway. Therefore $(x_1^2 - n)(x_2^2 - n)$ can be thought of as B -smooth. Since it is hard to find the second large prime to match up with the first one, it is wise to set the limit for the range of the interval where the large prime will be kept, for instance $(B, 20B]$ or $(B, 100B]$. From the above example, we could have paired $(177^2 - 18079)$ with $(141^2 - 18079)$ to produce a smooth number. Since $177^2 - 18079 = 2 \times 5^3 \times 53$ and $141^2 - 18079 = 2 \times 17 \times 53$, their product contains all small factors smaller than $B = 40$ except 53^2 . The contribution of the factor 53^2 does not affect the smoothness of the product since it will be reduced to 0 mod 2 in the exponent vector.

There is also double-prime variation. The single prime variation is based on the idea that if an integer in the interval $(1, B^2]$ has all the prime factors larger than B , then it is the prime, while the double-prime version works with numbers in the interval $(B^2, B^3]$. Once we remove all the prime factors up to B and the remaining unfactored portion exceeds B^2 then a test can be done to decide whether the unfactored portion is a prime. Denote the unfactored portion Q . We can find out whether Q is a prime by checking whether $2^{Q-1} \equiv 1 \pmod{Q}$. If it does, then there is a good chance that Q is a prime. Q

will be a too big to be valuable anyway; therefore it will be discarded. If Q can be shown to be a composite, then it will be factored using some of the previous simple methods. Suppose $Q = q_1 * q_2$. Suppose there is some $x_i^2 - n$ that is almost B-smooth except for two prime factors larger than B , namely q_1 and q_2 . The goal is to search for some other $x_i^2 - n$ that uses q_1 , q_2 , or both. For example, suppose the factorizations of some $x_i^2 - n$ are: $q_1 S_1$, $q_2 S_2$, $q_1 q_2 S_3$ where S_1, S_2, S_3 are B-smooth. Notice that the product of the above factorizations is $q_1^2 q_2^2 S_1 S_2 S_3$ which may be considered to be B-smooth since the prime factors above B have even exponents.

The second improvement is due to Peter Montgomery [Bre89]. Based on the idea that the smaller $x^2 - n$ is, the easier it will be smooth. Therefore we want to keep $x^2 - n$ close to zero by starting x from $\lceil \sqrt{n} \rceil$. But as x values move away from $\lceil \sqrt{n} \rceil$, it is hard to find $x^2 - n$ smooth since it gets big rapidly. The multiple polynomial variation takes care of this problem by using many polynomials instead of just $x^2 - n$. Basically we just replace x with a linear function of x . The suggestion is to look at polynomials of the form

$$f(x) = ax^2 + 2bx + c$$

where a, b, c are integers with $n = b^2 - ac$. Then

$$\begin{aligned} a \times f(x) &= a^2 x^2 + 2abx + ac \\ &= a^2 x^2 + 2abx + b^2 - n \\ &= (ax + b)^2 - n \end{aligned}$$

Notice that if p is a factor of $f(x)$ then $p \mid (ax + b)^2 - n$ or n is a quadratic residue modulo p . Therefore the factor base consists of the same elements as in the basic Quadratic Sieve algorithm. It is nice that we can use various polynomials without having an affect on the factor base. Also since

$$(ax + b)^2 - n = a \times f(x),$$

instead of evaluating $(ax + b)^2 - n$ for smoothness, we can deal with $a \times f(x)$. If a is a square times a B-smooth number and $f(x)$ is B-smooth, then $a \times f(x)$ can be thought of as B-smooth especially when its exponent vector is reduced modulo 2. Finding values of $f(x)$ that are B-smooth is a matter of keeping $f(x)$ small. This depends on the choice of a, b, c and the sieving interval. We decided at the start that we only want to sieve over

the interval of length $2M$. In order to make the interval of length $2M$ fall precisely on $[-M, M]$, we choose $|b| \leq \frac{1}{2}a$.

Note that the minimum value of $f(x)$ is achieved at $x = -b/a$ and the corresponding value of $f(x)$ at that point is $f(x) = f(-b/a) = n/a$. The values of $f(x)$ at the end points are:

$$f(-M - b/a) = f(M - b/a) = aM^2 - n/a \quad (2.1)$$

Setting the above values equal to each other, we have $n/a = aM^2 - n/a$ or $a = \sqrt{2n}/M$. Therefore, by choosing $a \approx \sqrt{2n}/M$ we can force the range of $f(x)$ to be small for values of x in our sieving region. Next, choose b to be the solution of the congruence $b^2 \equiv n \pmod{a}$ with $|b| < \frac{a}{2}$ and $c = (b^2 - n)/a$. We now have all the coefficients a , b , and c , and we can form the function $f(x) = ax^2 + bx + c$. A suggestion is to take various $p \approx (2n)^{1/4}M^{1/2}$ with $(\frac{n}{p}) = 1$, and choose $a = p^2$. With that selection of a , it satisfies the requirement for a that it has to be a product of a square and B-smooth number and $a \approx \sqrt{2n}/M$.

Once we found the function $f(x)$, for each p in the factor base with $(\frac{n}{p}) = 1$, we need to solve the congruence $ax^2 + bx + c \equiv 0 \pmod{p}$ since we will proceed with the sieving like before to look for B-smooth values of function $f(x)$. This process of finding roots for the congruence is referred to as the initialization problem since it can be very time-consuming. Especially when we use various polynomials, this method may not turn out to be as advantageous as we thought.

Pomerance came up with the solution called self initialization to save the running time for the polynomial switching process. Let's look at the roots for the congruence

$$\begin{aligned} f(x) &\equiv 0 \pmod{p} \\ [(ax + b)^2 - n]a^{-1} &\equiv 0 \pmod{p} \\ (ax + b)^2 &\equiv n \pmod{p}. \end{aligned}$$

Let $t(p)$ be a squareroot of $n \pmod{p}$. Then $(a + bx) \equiv \pm t(p)$ or

$$x = (-b \pm t(p))a^{-1} \pmod{p}.$$

If a has k distinct factor primes, then there are 2^{k-1} choices for b based on the way b is chosen, namely $b^2 \equiv n \pmod{a}$. If we choose $a = p^2$ as mentioned earlier, then there is only one choice for b subject to the constraint that $|b| < \frac{1}{2}$. Whereas if we choose

a to be a product of ten primes then there will be 2^9 choices for b . Taking advantage of this, we can save time finding solutions of so many polynomials by using the same value of a . So for each value of a which is a product of 10 primes, we only need to compute $t(p)$ once but we can use it for 2^9 polynomials.

There is another advantage to using polynomials other than $x^2 - n$. If a is approximately $\sqrt{2n}/M$ then by (2.1), $f(x) = ax^2 + 2bx + c$ is bounded by $(M\sqrt{n})/\sqrt{2}$ on the interval $[-M, M]$. In contrast, $x^2 - n$ is bounded by approximately $2M\sqrt{n}$ on the interval $[\sqrt{n} - M, \sqrt{n} + M]$. The absolute value of $f(x)$ is therefore smaller in the first case by a factor of $2\sqrt{2}$. Being able to keep the values of $f(x)$ down is an advantage since it is more likely to be smooth.

Perhaps the best reason to use multiple polynomials is that the sieving can be done in parallel on different processors. Each machine is in charge of doing the sieving for its own polynomial. With this method, A.K. Lenstra and M.S. Manasse were able to factor 100-digit integers successfully using roughly 400 computers around the world for the sieving process.

Chapter 3

General Number Field Sieve

3.1 General Idea

Similar to both Dixon's method and the Quadratic Sieve, we try to factor n by using the plan of Fermat. That is, by finding a solution to $x^2 \equiv y^2 \pmod{n}$. But in the Quadratic Sieve we only need to work with one side of the congruence since the other side was already a square. In the General Number Field Sieve, we are going to find squares from both sides of the congruence. This results in a substantial savings in work, allowing us to factor even larger numbers.

Basically we work with a homomorphism map from the ring $Z[\alpha]$ to Z_n where α is the root of some monic and irreducible $f(x)$ of degree $d > 1$ in $Z[x]$. It will help to have d odd, usually $d = 5$ as will be explained later. We do not need to compute the complex number α numerically, all we need to know is α stands for one of the roots of f . An element in $Z[\alpha]$ can be written in the form $\sum_{i=0}^{d-1} a_i \alpha^i$. Suppose that $m \in Z$ satisfies $f(m) \equiv 0 \pmod{n}$. We have a natural ring homomorphism $\varphi: Z[\alpha] \rightarrow Z/nZ$ which is induced by $\varphi(\alpha) = m \pmod{n}$. Therefore, $\varphi(\sum_i a_i \alpha^i) = \sum_i a_i m^i \pmod{n}$. For this method, we only consider elements in $Z[\alpha]$ of the form $a - b\alpha$.

The main goal is to find a non-empty set S of pairs (a, b) of relatively prime integers such that we have the two following equations:

$$\prod_{(a,b) \in S} (a - bm) = v^2 \text{ is a square in } Z \quad (3.1)$$

$$\prod_{(a,b) \in S} (a - b\alpha) = \gamma^2 \text{ is a square in } Z[\alpha] \quad (3.2)$$

Let $u \in Z$ and $\varphi(\gamma) \equiv u \pmod{n}$. Then $u^2 \equiv \varphi(\gamma)\varphi(\gamma) \equiv \varphi(\gamma^2) \equiv \varphi(\prod_{(a,b) \in S} (a - b\alpha)) \equiv \prod_{(a,b) \in S} (a - bm) \equiv v^2 \pmod{n}$. If u and v are known, then as in Fermat's method we have a 50-50 chance of factoring n by computing $\gcd(u - v, n)$. Although this is the basic idea, we will have to modify this plan later.

3.2 Polynomial Selection

The first thing we need to do to factor a positive integer n with the Number Field Sieve algorithm is to find some monic polynomial f of degree d in $Z[x]$ and an integer m such that $f(m) \equiv 0 \pmod{n}$. We want m , as well as the coefficient of f , to be as small as possible. Experimentally, the choice of $d=5$ is acceptable for an integer n of around 130 digits. One method goes as follows. Set $m = \lfloor n^{1/d} \rfloor$ and write n in base m :

$$n = m^d + c_{d-1}m^{d-1} + \dots + c_0, \quad 0 \leq c_i < m.$$

Replacing m with x , we have a monic polynomial $f(x) = x^d + c_{d-1}x^{d-1} + \dots + c_0$ for which $f(m) \equiv 0 \pmod{n}$, since $f(m) = n$, and whose coefficients are on the order of $n^{1/d}$. This polynomial is monic but may not be irreducible. If we have nontrivial factorization $f(x) = g(x)h(x)$ in $Z[x]$, then n can be factored by $n = g(m)h(m)$ and we are done. If $f(x)$ is irreducible, we proceed to the next step.

For example, the m -base expansion of $n = 44,831$ is $n = 8^5 + 2 \cdot 8^4 + 7 \cdot 8^3 + 4 \cdot 8^2 + 3 \cdot 8 + 7$. This expression yields $f(x) = x^5 + 2x^4 + 7x^3 + 4x^2 + 3x + 7$.

3.3 Sieving

The main goal of this step is to find a set T of pairs (a, b) such that both $a - bm$ and $a - b\alpha$ are smooth. The "smooth" concept will be defined momentarily in the context of $Z[\alpha]$. The set T will be constructed from sets T_1 and T_2 which are collections of pairs (a, b) such that the numbers $a - bm$ and $a - b\alpha$ are smooth.

3.3.1 The Rational Sieve

The purpose of this step is to find a set T_1 which is a collection of $a - bm$ numbers that are y smooth where the parameter y will be chosen depending on n .

Let $U = \{(a, b) \mid a, b \in \mathbb{Z}, \gcd(a, b) = 1, |a| \leq u, 0 < |b| \leq u\}$

The number u will be chosen later and will depend on n . It has to be sufficiently big enough so that the set U contains a set S satisfying (3.1) and (3.2) simultaneously. For the moment we only focus on the rational side of finding a set (a, b) such that $a - bm$ is smooth. Denote this set by T_1 ,

$$T_1 = \{(a, b) \in U : a - bm \text{ is } y\text{-smooth.}\}$$

This set will be referred to as the rational base. Recall that an integer is y -smooth if all of its prime divisors are less than or equal to y . A prime p divides $a - bm$ if and only if $a - bm \equiv 0 \pmod{p}$, and therefore $a \equiv bm \pmod{p}$. The sieving procedure starts with an array of numbers $a - bm$ for fixed integer $b \in (0, u]$ and lets a range over the interval $[-u, u]$. For each prime $p \leq y$, we identify those values of $a - bm$ satisfying $a \equiv bm \pmod{p}$. As in the Quadratic Sieve, once such a pair is found, the value of $a - bm$ will then be divided by the highest power of the prime that divides it, and the quotient will then replace the location of $a - bm$. Then the value of a is immediately increased by p to give the next location where $p \mid a - bm$. By the end of the procedure, we scan for locations with 1. Such locations correspond to a number $a - bm$ that is y smooth. As in the Quadratic Sieve, we can speed up the sieving process by initializing the array with $\ln(a - bm)$ instead of $a - bm$, to subtract $\ln(p)$ instead of dividing by p . By the end of the procedure, we would look for values of $0 = \ln(1)$ instead of 1.

3.3.2 The Algebraic Sieve

In this step, we want to find a set T_2 of pairs (a, b) such that $a - b\alpha$ is smooth. An element $a - b\alpha \in \mathbb{Z}[\alpha]$ is y -smooth if its norm $N(a - b\alpha) \in \mathbb{Z}$ is y -smooth. Let's define the norm of an element of the form $a - b\alpha$. Let $\alpha_1 \cdots \alpha_d$ be the complex roots of the irreducible polynomial $f(x)$. Then $(a - b\alpha_1) \cdots (a - b\alpha_d)$ are the conjugates of $a - b\alpha$.

Define the norm by

$$N(a - b\alpha) = (a - b\alpha_1) \cdots (a - b\alpha_d) = b^d (a/b - \alpha_1) \cdots (a/b - \alpha_d) = b^d f(a/b) \text{ since } f(x) = (x - \alpha_1)(x - \alpha_2) \cdots (x - \alpha_d).$$

So the norm of $a - b\alpha$ is the same as substituting a and b for x and y in the homogeneous form of f , which is : $F(x, y) = x^d + c^{d-1}x^{d-1}y + \dots + c_0y^d = y^d f(x/y)$. Therefore, $N(a - b\alpha) = F(a, b)$. For products of numbers of the form $a - b\alpha$, we define the norm by $N(xy) = N(x)N(y)$. It is easy to check that this is well-defined.

The norm $N(a - b\alpha) \in Z$ is y -smooth if its prime divisors are less than or equal to y . As a result, we want to keep track of small primes p such that $p \mid N(a - b\alpha)$ or $N(a - b\alpha) \equiv 0 \pmod{p}$. Let $r = ab^{-1}$. Since $N(a - b\alpha) = F(a, b) = b^d f(a/b)$, p is a divisor of $N(a - b\alpha)$ when $f(r) \equiv 0 \pmod{p}$ or $a = br \pmod{p}$. Denote $R(p) = \{r \in [0, p-1] \text{ such that } f(r) \equiv 0 \pmod{p}\}$. The set $R(p)$ is computed for each prime p in the factor base.

Similar to the rational sieve, we want to find a set T_2 which is a collection of pairs (a, b) such that $a - b\alpha$ that is y -smooth. This set will be referred to as the algebraic base.

$$T_2 = \{(a, b) \in U : a - b\alpha \text{ is } y\text{-smooth}\}.$$

We start an array with the numbers $N(a - b\alpha)$ for each fixed b and let a vary in the interval $[-u, u]$. For each $p \leq y$ and each $r \in R(p)$, values of $N(a - b\alpha)$ that satisfy $a \equiv br \pmod{p}$ will be identified. The value of each $N(a - b\alpha)$ will then be divided by the highest power of the prime that divides it, and the quotient will then replace the entry for which the number was retrieved. Any location that contains the number 1 at the end of the procedure corresponds to a number $a - b\alpha$ that is y -smooth. Just like the above section, we can use the approximate logarithms to speed up this process. Once a pair (a, b) is identified, we increase a by p to get the next value of where $p \mid N(a - b\alpha)$.

Up to this point, we have found collections of T_1 and T_2 such that $a - bm$ and $a - b\alpha$ are smooth respectively. In reality, the sieving process is set up in a way that both arrays $(a - bm)$ and $N(a - bm)$ are working side by side. Pairs of (a, b) that are found by the end of the process that will make both $(a - bm)$ and $N(a - b\alpha)$ smooth. Denote this set by $T = T_1 \cap T_2$. This process is harder than the Quadratic Sieve since we need the same (a, b) from both sieves.

3.4 Obstructions

There are many issues regarding this construction of a square in $Z[\alpha]$. First of all, it is possible for an element of $Z[\alpha]$ to be a perfect square in I but not in $Z[\alpha]$. Here,

I is the ring of algebraic integers in the algebraic number field $Q[\alpha]$. That is to say I consists of elements of $Q[\alpha]$ that are the root of some monic polynomial in $Z[x]$. I is also known as a Dedekind domain which is an integral domain in which every nonzero proper ideal factors into a product of prime ideals. The following lemma is important since it is a handy tool to get an element in $Z[\alpha]$ from an element in I .

Lemma 10. *Let $f(x)$ be a monic irreducible polynomial in $Z[x]$, with roots α in the complex numbers. Let I be the ring of algebraic integers in $Q(\alpha)$, and let $\beta \in I$. Then $f'(\alpha)\beta \in Z[\alpha]$. [CP01]*

So instead of searching for $\prod_{(a,b) \in S} (a - b\alpha)$ to be a square in $Z[\alpha]$, we can get away with having that product to be square in I , namely γ^2 . Using Lemma 10, we have $f'(\alpha)\gamma$ is in $Z[\alpha]$. Therefore, $f'(\alpha)^2 \prod_{(a,b) \in S} (a - b\alpha)$ is a square in $Z[\alpha]$.

Note that this changes our basic plan, as explained in 3.1. Our old plan was to find $\prod(a - bm)$ to be square in Z and $\prod(a - b\alpha)$ to be a square in $Z[\alpha]$. Since f and m are constructed by the base m algorithm, $f(m) = n$ or $1 < f'(m) < n$ and also we can assume that $\gcd(f'(m), n) = 1$. Therefore, multiplying (3.1) by $f'(m)^2$ will give us $f'(m)^2 \prod(a - bm)$ which is a square in Z . Our new plan will be to find $f'(m)^2 \prod(a - bm)$ a square in Z and the corresponding $f'(\alpha)^2 \prod(a - b\alpha)$ a square in $Z[\alpha]$. To do this, it is sufficient to force the product $\prod(a - b\alpha)$ to be a square of an algebraic integer.

3.5 Exponent Vectors

The main goal of this step is to find a non-empty set S of coprime integer pairs that satisfy both (3.1) and (3.2) simultaneously. In order to achieve this, we use linear algebra together with the rational and algebraic factor bases to locate $S \subset T$. For a number to be a square, all the primes in its factorization have to have even powers. Let $B = \pi(y)$ where $\pi(y)$ denotes the number of primes up to y . Suppose there are more than $B + 1$ elements in T_1 with the choice of parameters u and y , we can use linear algebra to find a dependency over F_2 . If w is a y -smooth integer, then $w = \prod_i p_i^{e_i}, 0 \leq i \leq B$. The exponent vector $e(w)$ is defined by :

$$e(w) = (e_0 \pmod{2}, e_1 \pmod{2}, \dots, e_B \pmod{2})$$

The product of all the numbers w is a square when $\sum e(w) = 0 \in F_2^{B+1}$. With the same idea, we can combine $B + 1$ values of $a - bm$ which are y -smooth, forming $e(a - bm)$.

We can then find a non-trivial linear dependence relation with coefficients 0 and 1. The product $\prod_{(a,b) \in S} (a - bm)$ is a square in Z when we find $\sum_{(a,b) \in S} e(a - bm) = 0$ in F_2^{B+1} .

In the same manner, we can use the idea of exponent vectors to multiply a set of norms $N(a - bm)$ to find a square. Since different elements in $Z[\alpha]$ can have the same norm, it is necessary to keep track of $r = ab^{-1} \in R(p)$ for each p that divides $N(a - b\alpha)$. For each pair (p, r) , the exponent $e_{p,r}(a - b\alpha)$ is defined to be the number of factors p in the factorization of $N(a - b\alpha)$ if $a \equiv br \pmod{p}$. If $a \not\equiv br \pmod{p}$, then $e_{p,r}(a - b\alpha)$ is defined to be 0. Therefore, we have:

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

As an example, consider $f(x) = x^2 + 3$, with $B = 5$. Then $R(2) = \{1\}$, $R(3) = \{0\}$, $R(5) = \{\emptyset\}$. We consider three pairs (a, b) such that their norms $F(a, b)$ are 5-smooth. These pairs are: $F(1, 1) = 4 = 2^2$, $F(3, 1) = 12 = 2^2 \cdot 3$, $F(3, -1) = 12 = 2^2 \cdot 3$. If we only went by these prime factorizations, then we might choose $(3+i)(3-i)$ whose norm 12^2 is a perfect square. But this would not give us what we want because $(3+i)(3-i) = 10$ is not a square. We can also tell that $(3+i)(3-i)$ is not a square based on the sum of their exponent vectors.

Component vectors of 5-smooth members corresponding to the two pairs (p, r) : $(2, 1)$, and $(3, 0)$ are:

$$F(1, 1) = 4 \text{ has the exponent vector } (2, 0)$$

Since first of all we want to check whether $a \equiv br \pmod{p}$ with $(a, b) = (1, 1)$ and $(p, r) = (2, 1)$. Because the answer is yes, then the exponent vector of $e_{2,1}(1 - \alpha)$ is the exponent of 2 in the factorization of $F(1, 1) = 4$.

Next we do the same for $e_{3,0}(1 - \alpha)$. Since $a \not\equiv br \pmod{p}$ with $(a, b) = (1, 1)$ and $(p, r) = (3, 0)$, the exponent in this case is 0. This gives $F(1, 1)$ an exponent vector of $(2, 0)$. Similarly, $F(3, 1) = 12$ has the exponent vector $(2, 1)$, and $F(3, -1) = 12$ has the exponent vector $(2, 0)$.

Since the sum of the exponent vectors modulo 2 of $(3+i)$, $(3-i)$ is $(0, 1)$, it allows us to see that their product is not a square. At the same time, even when we have $\sum_{(a,b) \in S} e_{p,r}(a - b\alpha) \equiv 0 \pmod{2}$, there is no guarantee that the product of corresponding norms will be square in $Z[\alpha]$. In the above example, we have the exponent vector of

$F(1, 1)F(3, -1)$ is $(0, 0)$, yet $(1 + i)(3 - i) = 4 + 2i$ is not a square in $Z[i]$. Similarly, the converse of the following lemma is a necessary but not sufficient condition for the product of $(a - b\alpha)$ to be a square in $Z[\alpha]$. The extent to which the converse fails will be supplemented with the use of quadratic character base which will be discussed later.

Lemma 11. *If S is a set of coprime integer pairs a, b such that each $a - b\alpha$ is y -smooth, and if $\prod_{(a,b) \in S} (a - b\alpha)$ is a square of an element in I , the ring of algebraic integers in $Q[\alpha]$, then*

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

We still have obstructions, since converse of Lemma 11 does not hold. That is, the sum of our exponent vectors might be zero and still not have a square in I . This can be overcome with the use of quadratic characters. This idea is due to Adleman and based on the Legendre symbol. If p is an odd prime and if $\left(\frac{a}{p}\right) = 1$, then a is a quadratic residue modulo p . Similarly $\left(\frac{a}{p}\right) = -1$, if a is a quadratic nonresidue modulo p . Both occur with equal likelihood. So if a is a square, then for any odd prime p we have $\left(\frac{a}{p}\right) = 1$. Although the converse of the above statement is not true, we just want to apply the idea probabilistically. Suppose X is a finite set of k odd primes and $a \in Z$. Suppose also that $\left(\frac{a}{p}\right) = 1$ for each prime in X . The probability of a not being a square is about 2^{-k} . Therefore, if k is large and if $\left(\frac{a}{p}\right)$ is always equal to 1 for primes $p \in X$, then a has high probability of being a square. We want to incorporate this idea with the algebraic integers $a - b\alpha$ through the following lemma.

Lemma 12. *Let $f(x)$ be a monic, irreducible polynomial in $Z[x]$ and let α be a root of f in the complex numbers. Suppose q is an odd prime number and s is an integer with $f(s) \equiv 0 \pmod{q}$ and $f'(s) \not\equiv 0 \pmod{q}$. Let S be a set of coprime integer pairs (a, b) such that q does not divide any $a - bs$ for $(a, b) \in S$ and $f'(\alpha)^2 \prod_{(a,b) \in S} (a - b\alpha)$ is a square in $Z[\alpha]$. Then*

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

Just as Lemma 11, the result of this theorem alone is a necessary but not sufficient condition to test for squareness in $Z[\alpha]$. For those pairs (a, b) that satisfy Lemma 11 and Lemma 12 simultaneously, there is a good chance that the product of $a - b\alpha$ is a square of some element in the algebraic ring I . Pairs of (q, s) satisfying lemma 12 are referred to as character base.

3.6 Matrix and Linear Algebra

After section 3.3, we found $T = T_1 \cap T_2$ such that:

$$T = \{(a, b) : \gcd(a, b) = 1, |a| \leq u, 0 < b \leq u, (a - bm) \text{ and } N(a - b\alpha) \text{ are } y\text{-smooth}\}$$

define

$$B = \pi(y)$$

$$B' = \#\{(p, r) : p \text{ is a prime number, } p \leq y, r \in R(p)\}$$

$$B'' = \lfloor 3(\log n) / \log 2 \rfloor$$

Each column of the matrix corresponds to binary vector $e_{(a,b)}$ for each pair (a, b) and has entries as follows: the first entry would be the sign of $a - bm$ where the entry will receive 0 if $a - bm$ is positive and 1 if it is negative. The next B entries would be the exponents modulo 2 of all the primes up to y in the factorization of $a - bm$. The next B' entries would be exponents vectors as described in section 3.4. The next B'' entries are determined by $(\frac{a+bs}{q})$ as (q, s) runs over the character base. The corresponding entry to each pair (s, q) would be 0 if $(\frac{a+bs}{q}) = 1$ and 1 if $(\frac{a+bs}{q}) = -1$

If enough (a, b) pairs are found such that they exceed $1 + B + B' + B''$ then the vectors $e(a, b)$ for (a, b) are linearly dependent. Therefore a nonempty subset S of T has been found such that $\sum_{(a,b) \in S} e(a, b) = 0$ in $F_2^{1+B+B'+B''}$. Such S will make $\prod_{(a,b) \in S} f'(m)^2(a - bm)$ and $f'(\alpha)^2 \prod_{(a,b) \in S} (a - b\alpha)$ perfect squares in Z and $Z[\alpha]$ respectively.

3.7 Square Root in $Z[\alpha]$

Up to this point we have found a set of S of coprime integer pairs (a, b) such that $f'(\alpha)^2 \prod_{(a,b) \in S} (a - b\alpha) = \gamma^2$ for $\gamma \in Z[\alpha]$ and $f'(m)^2 \prod_{(a,b) \in S} (a - bm) = v^2$. Notice that this γ and v are slightly different from those in (3.1) and (3.2) because of the reasons that we discussed in 3.5 above. Suppose there is an integer u such that $\varphi(\gamma) \equiv u \pmod{n}$. Then $u^2 \equiv [\varphi(\gamma)]^2 \pmod{n}$ or $u^2 \equiv \varphi[f'(\alpha)^2 \prod_{(a,b) \in S} (a - b\alpha)] \pmod{n}$. Then $u^2 \equiv [f'(m)^2 \prod_{(a,b) \in S} (a - bm)] \pmod{n}$ or $u^2 \equiv [f'(m)v]^2 \pmod{n}$. Therefore with a probability of approximately to $\frac{1}{2}$, $\gcd(u - f'(m)v, n)$ will be a non-trivial factor of n . Since $v \in Z$, we would not have a problem taking square root of $f'(m)^2 \prod_{(a,b) \in S} (a - bm) = v^2$ to find v especially when we are only concerned with the residue $v \pmod{n}$. Actually we already

have $\prod(a - bm)$ expressed as a product of primes to even powers, so we can find v by cutting these powers in half, then multiplying by $f'(m)$. Unlike taking square root of v^2 , trying to take square root of γ^2 in the number ring $Z[\alpha]$ is by no means easy because we are dealing with a very big number and we cannot take advantage of the modulo n property to simplify the problem.

There are several methods of dealing with this part of the Number Field Sieve. One of the approaches is suggested by Couveignes 1993 [BLP93]. We have γ^2 represented as an element of $Z[\alpha] \cong Z[x]/f(x)$. By reducing the coefficients of this polynomial modulo p (where p is an odd prime) we create a perfect square in $Z_p[x]/f(x)$. This perfect square in $Z_p[x]/f(x)$ may have several square roots. One of them will have the coefficients of γ modulo p . That is the one we want. If we can determine the coefficients of $\gamma \pmod{p}$ for enough primes p , then we can use the Chinese Remainder Theorem to recover γ . We start by choosing odd primes p such that $f(x)$ is irreducible modulo p . This causes $Z_p[x]/f(x)$ to be a finite field. Since there are at most two square roots of an element of a field, this limits the number of square roots that we have to distinguish.

First of all, we want to solve for $\gamma \pmod{p}$ (that is for the coefficients of γ modulo p). For the time being, we are going to focus on how to compute square roots in a finite field. We are going to use concepts of quadratic residue and quadratic non residue together with an extension of Euler's criterion and the Sylow 2-subgroup to achieve what we want.

Definition 3.1. *Let F_{p^k} be a finite field with p^k elements where p is an odd prime. An element $\tau \in F_{p^k}^*$ is called a quadratic residue if there is an element θ in $F_{p^k}^*$ such that $\theta^2 = \tau$. It is called quadratic non-residue otherwise.*

Theorem 13. *(Euler's Criterion) Let F_{p^k} be a finite field with p^k elements where p is an odd prime. An element $\tau \in F_{p^k}^*$ is a quadratic residue if and only if $\tau^{(p^k-1)/2} = 1$ and is a quadratic non-residue if and only if $\tau^{(p^k-1)/2} = -1$.*

Consider the finite field F_{p^k} of size p^k . Denote $p^k = q$. Since p^k is odd, $p^k - 1 = q - 1$ can be expressed as $2^s t$ where t is odd. Let τ be a quadratic residue and $(\tau^{\frac{t+1}{2}})^2 = \tau^{t+1} = \tau^t \tau$. We conclude that τ^t is a quadratic residue in $F_{p^k}^*$. Then there exists an element $c \in F_{p^k}^*$ such that $c^2 = \tau^t$. Notice that the square root of τ can be

expressed as $\tau^{\frac{t+1}{2}} c^{-1}$ since

$$(\tau^{\frac{t+1}{2}} c^{-1})^2 = (\tau^{t+1} c^{-2}) = \tau^{t-1} \tau^{-t} = \tau.$$

Our immediate goal now is to find c from $c^2 = \tau^t$. We are going to show such an element c has order dividing 2^s and belongs to Sylow 2-subgroup S_{2^s} of F_p^* . Since τ is quadratic residue, $(\frac{\tau}{p}) = 1$. Using Euler's Criterion for the quadratic residue τ :

$$\tau^{\frac{p-1}{2}} = 1$$

$$\tau^{\frac{2^s t}{2}} = 1$$

$$\tau^{2^{s-1} t} = 1$$

$$(\tau^t)^{2^{s-1}} = 1.$$

Therefore τ^t has order dividing 2^{s-1} and so does c^2 since $c^2 = \tau^t$. It follows that c has order dividing 2^s . From abstract algebra we know that every element of the Sylow 2-subgroup S_{2^s} has order dividing 2^s and vice versa. We also notice that if g is a quadratic non-residue in $F_{p^k}^*$, then $g^{\frac{p-1}{2}} = -1$, $g^{\frac{2^s t}{2}} = -1$, and $g^{2^{s-1} t} = -1$. Then $(g^t)^{2^{s-1}} = -1$ so $(g^t)^{2^s} = 1$. Then g^t has order exactly 2^s . Therefore g^t is a generator of the Sylow 2-subgroup S_{2^s} . In particular, the Sylow 2-subgroup S_{2^s} will look like $\{1, g^t, g^{2t}, \dots, g^{(2^s-1)t}\}$. Since half of elements in $F_{p^k}^*$ are quadratic non-residues, a direct search for such g will end quickly. Once we find a quadratic non-residue and generate the Sylow 2-subgroup, we can search for an element c whose square is equal to τ^t . Once c is found, a square root of τ is just $\tau^{\frac{t+1}{2}} c^{-1}$.

For our purpose, the τ we are interested in is the reduction of $\gamma^2 \in Z[x]/f(x)$ modulo p . The problem we face is that we have two square roots γ_i for each prime p_i . We need to determine which of these two square roots gives the coefficients of γ modulo p_i . Earlier, we defined the norm of an element in $Z[x]/f(x)$. We can apply the same definition to get the norm of an element in $Z_p[x]/f(x)$. If we start with an element μ in $Z[x]/f(x)$, then reduce the coefficients of the polynomial modulo p , we get an element ν in $Z_p[x]/f(x)$. In this case $N(\nu)$ will just be the reduction of $N(\mu)$ modulo p . We notice that, by definition, $N(-1) = N(-1 + 0\alpha) = (-1)^d = -1$. If d is odd, as was promised from the start, then $N(-x) = N(-1)N(x) = -N(x)$. Therefore the two square roots of τ can be distinguished by their norms.

Now the norm of γ^2 is known. In fact we have its prime factorization in even powers. By cutting these powers in half, we can determine the norm of one of its square roots, which we designate as γ . By reducing this norm modulo p , we can then check which square root of τ has the correct norm. Computing the norm of the square roots of τ is easy since we can start with a choice of d (the non-residue) whose norm is known.

At this point we have found the square root γ in terms of γ_i in different finite fields. Let's remind ourselves that our goal is to search for u^2 and v^2 where

$$v^2 = f'(m)^2 \prod_{(a,b) \in S} (a - bm)$$

$$\text{and } u^2 \equiv \varphi[f'(\alpha)^2 \prod_{(a,b) \in S} (a - b\alpha)] \pmod{n}.$$

with $\gamma^2 = f'(\alpha)^2 \prod_{(a,b) \in S} (a - b\alpha)$ and $\varphi(\gamma) \equiv u \pmod{n}$. Also recall that γ is a polynomial in $Z[x]/f(x)$, and that we have access to the coefficients of this polynomial modulo p for several values of p . To pass from γ to u we need to replace the variable x with the integer m . Performing this calculation modulo p will immediately give us the value of $u \pmod{p}$. Therefore, instead of computing γ and then applying φ to give us u , we can save time by going directly after u .

We can calculate u by using Chinese Remainder Theorem because we have access to the system of congruences:

$$\begin{aligned} u &\equiv u_1 \pmod{p_1} \\ u &\equiv u_2 \pmod{p_2} \\ &\vdots \\ u &\equiv u_{r-1} \pmod{p_{r-1}} \end{aligned}$$

Where u_i is the image of γ_i under the φ mapping such that $\varphi(\gamma_i) \equiv u_i \pmod{p^i}$. Therefore $u = \sum_{i=0}^{r-1} u_i a_i P_i \pmod{P}$. Denote $\sum_{i=0}^{r-1} u_i a_i P_i = z$, $u = z \pmod{P}$. There is one final problem to overcome. If u is large to start with and z is much bigger than u then using z to calculate u could be a problem. Fortunately, we have a different approach to calculate u . If we round z/P to an integer $r = \lfloor \frac{z}{P} + \frac{1}{2} \rfloor$ then we can have $u = z - rP$. We can calculate r without having to deal with a very large z as follows:

$$\frac{z}{P} = \frac{\sum_{i=0}^{r-1} u_i a_i P_i}{P} = \sum_{i=0}^{r-1} \frac{u_i a_i}{p_i}.$$

This gives us r by computing $\lfloor \frac{z}{P} + \frac{1}{2} \rfloor$. Once r is computed, we can determine u modulo n by observing that $u \equiv z - rP \equiv \sum_{i=0}^{r-1} u_i a_i P_i - rP \pmod{n}$. Then the computation of u can be carried out modulo n .

All we have to do now is find $\gcd(u-v, n)$ and there is a 50-50 chance of factoring n . If the factorization fails then we don't have to start the process all over again. Most of the work is in the sieving and the linear algebra steps. We can throw out one of the smooth exponent vectors that was used in the linear dependency found in section 3.6. Then, as long as we did a small amount of over-sieving there should be additional linear dependencies remaining. We can find them by repeating the linear algebra step. Most of this work can also be saved. So coming up with a fresh pairs u, v is not that difficult and gives us additional chances to factor n .

Bibliography

- [BLP93] J. P. Buhler, H. W. Lenstra, Jr., and Carl Pomerance. Factoring integers with the number field sieve. In *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Math.*, pages 50–94. Springer, Berlin, 1993.
- [Bre89] David M. Bressoud. *Factorization and Primality Testing*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1989.
- [Bri98] M. E. Briggs. An Introduction to the General Number Field Sieve. Master's thesis, Virginia Polytechnic Institute, 1998.
- [CP01] Richard Crandall and Carl Pomerance. *Prime Numbers*. Springer-Verlag, New York, 2001. A computational perspective.
- [Fra67] John B. Fraleigh. *A First Course in Abstract Algebra*. Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont., 1967.
- [Gib93] Peter Giblin. *Primes and Programming*. Cambridge University Press, Cambridge, 1993. An introduction to number theory with computing.
- [LEF04] Ron Larson, Bruce H. Edwards, and David C. Falvo. *Elementary Linear Algebra*. Houghton Mifflin Company, 2004.
- [Str94] James K. Strayer. *Elementary Number Theory*. Waveland Press, Inc, 1994.