EFFICIENT FACTORING
AND
THE NUMBER FIELD SIEVE

Submitted to the Department of Mathematics in partial
fulfillment of the requirements for the degree of
Bachelor of Arts in Mathematics with Honors

Rebecca Saul

Advised by Professor Noam Elkies

March 21, 2022
Harvard University

Contents

1 Introduction 3

2 The Quadratic Sieve 5

2.1 The Building Blocks 5

2.1.1 Elementary Difference of Squares 5

2.1.2 Kraitchik's Improvements 5

2.1.3 Systematizing Difference of Squares 7

2.2 The Quadratic Sieve 9

2.2.1 Using a Sieve to Evaluate Smoothness 9

2.2.2 Choosing p_B and Arriving at a Complexity Estimate 12

2.2.3 The Quadratic Sieve Algorithm 14

2.3 Improvements on the Quadratic Sieve 15

2.3.1 Working with Logarithms 15

2.3.2 Multiple Polynomials 16

2.3.3 Large Primes 17

3 The Number Field Sieve 18

3.1 From the Quadratic Sieve to the Number Field Sieve 18

3.2 Results from Algebraic Number Theory 18

3.3 The Number Field Sieve 22

3.3.1 Finding $f(x)$ and m. 23

3.3.2 Building S 24

3.3.3 Taking Square Roots in $\mathbb{Z}[\alpha]$ 29

3.3.4 An Asymptotic Analysis of the Sieve Step and Optimal Values for d and Y 31

3.3.5 Full Algorithm Description 33

3.4 Improvements to the Number Field Sieve 35

3.4.1 Nonmonic Choices for f 36

3.4.2 Polynomial Pairs 36

3.4.3 The Special Number Field Sieve 36

4 Comparing Factoring Techniques 38

Acknowledgements

I would first like to thank my advisor, Professor Noam Elkies, who encouraged me to tackle the Number Field Sieve and provided invaluable guidance throughout the writing process. I would also like to thank my dad, who believed I was good at math even after I got a C on my first geometry quiz, my mom, who believed I was good at the rest of life even after I put my white shirt in the laundry and it came out pink, my brother Caleb, who imbued me with the tenacity needed to complete a thesis, and all the friends who listened to me go on and on about factoring and never once taped my mouth shut.

1 Introduction

There is a special collection of natural numbers, known as the prime numbers, which cannot be written as the product of two smaller natural numbers. The existence of such a collection raises many questions, including

1. Can we easily identify which natural numbers are prime?
2. What is the relationship between the prime numbers and the remaining (composite) natural numbers?

Mathematicians have excellent tools for primality testing. Probabilistic methods like the Miller-Rabin Test are both efficient and highly accurate in practice. From a theoretical perspective, the problem of prime recognition was proven to be in P[1] in 2002 due to a breakthrough algorithm posited by Manindra Agrawal, Neeraj Kayal, and Nitin Saxena[2].

Answers to our second question date back much further than 2002. A proof that the natural numbers greater than one factor uniquely as product of primes, up to reordering, is given in Euclid's *Elements*. So not only is it easy to recognize which numbers are prime, but we also know that every composite number can be written uniquely as a product of these primes. The question that follows naturally is then: given a natural number n, how do we find these decompositions? We know n can have at most one factor greater than $\sqrt{n}$, so one way to factor would be to try dividing n by each prime p up to $\sqrt{n}$. This process is known as trial division, and while it is extremely inefficient, it is guaranteed to work.

For many years, the knowledge that factoring was solved in theory was enough to satisfy most mathematicians. Yes, trial division was slow, rendering many composite numbers unfactorable in our lifetime, but we knew how to factor. There were many problems that did not yet have even a theoretical solution, and it was these questions, and not that of how to factor more quickly, that attracted the attention of mathematicians.

Then technology changed everything. Computers became broadly available, and their computational power appeared to double every two to three years, making executable in hours algorithms that would have previously taken millennia to finish. Suddenly, fast factoring seemed achievable. The president of the Association for Computing Machinery summarized the landscape well in 1984 when he remarked upon the task of factoring $2^{251}-1$: "Even 20 years ago... the search time was estimated to be about 10^{20} years. The number was factored in February of this year... in 32 hours[28]."

Not only did the factorization of large numbers become increasingly possible, but the importance of such factorizations grew as well. As computers developed, so did the field of cryptography, with mathematicians and computer scientists designing cryptographic systems based on the perceived difficulty of many problems in number theory, including factoring. Suddenly, it was critical to know exactly how capable we were of factoring large composites, as it would determine the security of many cryptographic protocols.

[1] i.e., solvable in polynomial time

The most relevant characteristic of a factoring method is the speed at which it can factor a number n. As a baseline, factoring algorithms are compared to the performance of trial division. In the simplest implementation of trial division, we divide n by every number up to $\sqrt{n}$ to identify the prime factors. This takes $O(\sqrt{n})$ steps. To evaluate factoring algorithms, however, we examine the running time in terms of the size of the representation of n in the computer, which is $b = \log(n)$. We can solve to see that $2^{b/2} = \sqrt{n}$, and so then factoring by trial division takes $O(2^{b/2})$ time. This means that as n increases, the time needed to perform trial division increases exponentially. Consequently, modern day factoring algorithms, which should improve on the performance of trial division, are expected to run in sub-exponential time, with an open question being if factoring, like multiplication, is possible in polynomial time on classical computers. In the following chapters, we will introduce and analyze two sub-exponential general purpose factoring algorithms, the Quadratic Sieve and the Number Field Sieve.

2 The Quadratic Sieve

2.1 The Building Blocks

2.1.1 Elementary Difference of Squares

Many people first become introduced to difference of squares factorization in the context of quadratic equations. Given equations of the form $x^2 - c^2 = 0$, we are taught to reduce the left-hand side to a product of the linear terms $(x + c)(x - c)$, to facilitate easy identification of the roots $(\pm c)$. This same technique can be used to quickly factor any number that can be written as a difference of squares.

Example 1. (Difference of Squares Factorization) [28] 8051, written as the difference of 8100 and 49, factors as follows:

$$\begin{aligned} 8051 &= 8100 - 49 \\ &= 90^2 - 7^2 \\ &= (90 + 7)(90 - 7) \\ &= 97 \cdot 83. \end{aligned}$$

In fact, any odd composite number[2] can be written as a difference of squares, and thus factored via this method, using the identity $ab = (\frac{1}{2}(a + b))^2 - (\frac{1}{2}(a - b))^2$. The difficulty in factoring a number n then becomes determining appropriate u, v such that $n = u^2 - v^2$.

Fermat's factorization method attempts to find u and v by making an initial guess of $u_1 = \lceil \sqrt{n} \rceil$, and if $u_1^2 - n$ is not a square, progressing to check $u_2 = u_1 + 1$, $u_3 = u_2 + 1$, and so on, until a difference of squares is identified. In the best case scenario, when n has a factor near $\sqrt{n}$, this method is very efficient; unfortunately, this is the case for only a tiny percentage of n. When n has no factors near $\sqrt{n}$, Fermat's method can perform significantly worse than trial division [28].

Example 2. (Fermat's Factorization Method) We factor $n = 2021$ using Fermat's technique. First we compute $u_1 = \lceil \sqrt{n} \rceil = 45$. We can see

$$45^2 - n = 2025 - 2021 = 4.$$

Four is a perfect square, so we can write $n = 45^2 - 2^2$ and obtain the factorization

$$2021 = 47 \cdot 43.$$

2.1.2 Kraitchik's Improvements

The primary source of inefficiency in Fermat's technique is the time it takes to identify u and v when n has no factor near its root. This difficulty was partially alleviated by Maurice Kraitchik, who, in the 1920s, suggested that instead of requiring $u^2 - v^2 = n$, it would suffice to find u, v such that $u^2 - v^2$ was a multiple of n [28]. Not all solutions to $u^2 \equiv v^2 \pmod{n}$ are interesting, but if this equivalence holds, and $u \not\equiv \pm v \pmod{n}$, then $n | (u + v)(u - v)$ while n does not divide either

[2] We are only interested in factoring odd composites, as it is easy to identify and divide out powers of two.

factor $(u \pm v)$. Then $\gcd(u-v, n)$ (or alternatively, $\gcd(u+v, n)$) must yield a nontrivial factor of n.

Using the Euclidean algorithm, $\gcd(a, b)$ can be computed in $O(\log(\min(a, b)))$ steps. This is extremely fast, meaning that if suitable u, v can be found, that is, u, v such that $u^2 \equiv v^2 \pmod{n}$ but $u \not\equiv \pm v \pmod{n}$, factorization will be simple under Kraitchik's method. How does this improve on Fermat's results? Kraitchik's requirement that $u^2 \equiv v^2 \pmod{n}$ is less stringent than Fermat's, which asks that $u^2 = v^2 + n$, and so it should be easier to find u, v to satisfy Kraitchik's equivalence than to satisfy Fermat's equality. But for this additional flexibility to be helpful, we need to be able to find u, v such that $u \not\equiv \pm v \pmod{n}$ as well. Luckily, this task is not too onerous.

Lemma 1. If n is odd and divisible by at least two different primes, then at least half of the solutions to $u^2 \equiv v^2 \pmod{n}$, with uv coprime to n, have $u \not\equiv \pm v \pmod{n}$, and thus result in $\gcd(u - v, n)$ non-trivial.

Proof: Following [30], we give proof in the case where $n = pq$, with p, q distinct primes, from which generalization is easy. Consider the equation $x^2 \equiv v^2 \pmod{p}$. The solutions $x \equiv \pm v \pmod{p}$ are immediate. Suppose that $v \equiv -v \pmod{p}$. Then $2v \equiv 0 \pmod{p}$. Since n is odd, p must be odd, so this implies $v \equiv 0 \pmod{p}$. However, uv is coprime to n, so this is a contradiction. Therefore $v \not\equiv -v \pmod{p}$, and the solutions $x \equiv \pm v \pmod{p}$ are distinct.

By similar reasoning, we see that $y^2 \equiv v^2 \pmod{q}$ has distinct solutions $y \equiv \pm v \pmod{q}$. Then by the Chinese Remainder Theorem, we can solve the following systems of congruences to get four distinct solutions for u, where $u^2 \equiv v^2 \pmod{n}$.

$$\begin{cases} u \equiv v \pmod{p}, \quad u \equiv v \pmod{q} & \Longrightarrow \quad u \equiv v \pmod{n} \\ u \equiv -v \pmod{p}, \quad u \equiv -v \pmod{q} & \Longrightarrow \quad u \equiv -v \pmod{n} \\ u \equiv v \pmod{p}, \quad u \equiv -v \pmod{q} & \Longrightarrow \quad u \equiv z \pmod{n} \\ u \equiv -v \pmod{p}, \quad u \equiv v \pmod{q} & \Longrightarrow \quad u \equiv -z \pmod{n} \end{cases}$$

The solutions $u \equiv \pm z \pmod{n}$ will result in $\gcd(u - v, n)$ nontrivial, so at least half the solutions of $u^2 \equiv v^2 \pmod{n}$ are nontrivial in the case $n = pq$. To generalize to the case where n is the product of more than two primes, take p to be one prime, and $q = \frac{n}{p}$. □

Now that we know that at least half of the solutions to $u^2 \equiv v^2 \pmod{n}$ will give us a nontrivial factor of n, we can feel confident that finding such a (u, v) pair is an appropriate factoring strategy. Kraitchik begins the search for u, v in the same place as Fermat, setting $u_1 = \lceil \sqrt{n} \rceil$, examining if $Q(u_1) = u_1^2 - n$ is a square, and progressing to $u_2 = u_1 + 1$ if it isn't. However, Kraitchik goes one step beyond Fermat — instead of just considering if $Q(u_i) = u_i^2 - n$ is a square, Kraitchik also asks whether products of some of the $Q(u_i)$'s are squares. We have a congruence

$$Q(u_{i_1}) \cdots Q(u_{i_k}) = (u_{i_1}^2 - n) \cdots (u_{i_k}^2 - n) \equiv u_{i_1}^2 \cdots u_{i_k}^2 \pmod{n},$$

so if $Q(u_{i_1}) \cdots Q(u_{i_k}) = v^2$ for some v, then letting $u = u_{i_1} \cdot u_{i_k}$, we obtain $u^2 \equiv v^2 \pmod{n}$, which is the relationship we desired. In this way, requiring congruence modulo n, rather than equality, gives Kraitchik many more possible pairs (u, v) which may lead to a nontrivial factorization of n than Fermat would have had.

Example 3. (Kraitchik's Method) [28] We factor $n = 2041$ using Kraitchik's method. Let

$$Q(u) = u^2 - n = u^2 - 2041$$

generate the **auxiliary numbers**. We begin with $u = \lceil\sqrt{2041}\rceil = 46$ and compute the first six auxiliary numbers, shown in Table 1.

u	46	47	48	49	50	51
Q(u)	75	168	263	360	459	560

Table 1: The first six auxiliary numbers

If $Q(u)$ is a square, we can factor n immediately using difference of squares. (This is Fermat's technique.) None of the first six auxiliary numbers are square in this example, so at this point, Fermat would need to keep searching. However, Kraitchik, who needs only an equivalence, and not an equality relationship, has additional tools to work with. Kraitchik uses trial division to factor the auxiliary numbers, which are smaller than n and thus easier to factor, and observes that several of them factor completely over small primes. In particular, $75 = 3 \cdot 5^2$, $168 = 2^3 \cdot 3 \cdot 7$, $360 = 2^3 \cdot 3^2 \cdot 5$, and $560 = 2^4 \cdot 5 \cdot 7$. A closer look at these factorizations reveals that the product of these four auxiliary numbers is itself a square. Letting $v = 2^5 \cdot 3^2 \cdot 5^2 \cdot 7$ and $u = 46 \cdot 47 \cdot 49 \cdot 51$, we have

$$v^2 = 75 \cdot 168 \cdot 360 \cdot 560 = (46^2 - 2041)(47^2 - 2041)(49^2 - 2041)(51^2 - 2041) \equiv u^2 \pmod{2041}.$$

Simplifying, we get that $u \equiv 311 \pmod{2041}$ and $v \equiv 1416 \pmod{2041}$. Then $u \not\equiv v \pmod{2041}$, so this (u, v) pair should give a nontrivial factor of 2041. And indeed, $\gcd(1416 - 311, 2041) = 13$ and $2041 = 13 \cdot 157$. This factors 2041 completely.

2.1.3 Systematizing Difference of Squares

In Example 3, we found v, and then u, by factoring several auxiliary numbers and observing from their decompositions that their product would be a square. However, we were fortunate in several respects when dealing with this example. Firstly, we only needed to compute six auxiliary numbers before we were able to find a subset that multiplied to a square - for a different n, one could imagine requiring many more auxiliary numbers before this is possible. Secondly, the auxiliary numbers we used to create v factored completely over the first four primes, so it was easy to recognize that their product would be a square. Yet with many more auxiliary numbers, and many more primes involved in the factorization of these auxiliary numbers, it could quickly become quite difficult to spot nice products of auxiliary numbers with the naked eye. This suggests that a more systematic approach is needed to use Kraitchik's method in practice.

A more methodical strategy for finding a subset of auxiliary numbers whose product is a square was first given by John Brillhart and Michael Morrison in 1975 [25]. They recognized that every positive integer m has a corresponding **exponent vector** $\mathbf{v}(m)$ determined by its prime factorization. If p_i is the i^{th} prime, then m can be written as $m = \prod_i p_i^{v_i}$. We then define $v(m) = (v_1, v_2, \ldots)$. This definition can be extended to include negative numbers by incorporating a 0^{th} coordinate at the start of the vector $v(m)$, which is 0 if m is positive or 1 if m is negative. Essentially, we can consider -1 to be the 0^{th} prime, and v_0 the exponent corresponding to it in the factorization of m.

Using exponent vector representation, we can see that the product of several non-zero integers being a square is equivalent to the sum of their exponent vectors equaling the zero vector modulo 2. This observation suggests that instead of recording the factorization of the auxiliary numbers once we compute them, what we really need to store is their exponent vectors modulo 2. Of course, as there are infinitely many primes, each exponent vector will have infinitely many entries, though only finitely many of them will be nonzero. To limit the scope of the problem, Brillhart and Morrison suggest one further simplification.

Definition 1. (Y-smooth) A number n is **Y-smooth** if it has no prime factors exceeding Y.

Brillhart and Morrison recommend designating some prime p_B and only considering auxiliary numbers that are p_B-smooth, i.e., that factor completely over the first B primes. These auxiliary numbers will yield truncated (and reduced modulo 2) exponent vectors in the vector space $\mathbb{F}_2^{B+1}$, which is $B+1$ dimensional. If we can find $B+2$ such auxiliary numbers, we know their corresponding exponent vectors will be linearly dependent in $\mathbb{F}_2^{B+1}$, and some linear combination of them will sum to the zero vector. As the only scalars in $\mathbb{F}_2^{B+1}$ are 0 and 1, this is equivalent to saying some subset of the $B+2$ vectors will sum to the zero vector. Since these are exponent vectors, the auxiliary numbers corresponding to this subset of vectors will thus multiply to a square.

We redo the problem from Example 3 to illustrate how to apply Brillhart and Morrison's insights in practice.

Example 4. (Brillhart and Morrison) We again factor $n = 2041$, and let $Q(u) = u^2 - 2041$ generate the auxiliary numbers. This time, we will allow for auxiliary numbers to be negative. This helps keep the absolute value of the auxiliary numbers small, making them easier to factor and break into exponent vectors. In this example, we will work with auxiliary numbers that are 5-smooth, so our exponent vectors will be in $\mathbb{F}_2^4$, as we are counting -1 as a prime.

u	$Q(u)$	Factorization	$v(u)$	$v(u) \pmod 2$
43	-192	$-1 \cdot 2^6 \cdot 3$	$(1, 6, 1, 0)$	$(1, 0, 1, 0)$
44	-105	$-1 \cdot 3 \cdot 5 \cdot 7$	$- - -$	$- - -$
45	-16	$-1 \cdot 2^4$	$(1, 4, 0, 0)$	$(1, 0, 0, 0)$
46	75	$3 \cdot 5^2$	$(0, 0, 1, 2)$	$(0, 0, 1, 0)$

Table 2: Auxiliary numbers with corresponding factorization and exponent vectors

Since $Q(44) = -105$ is not 5-smooth, we do not bother computing its exponent vector. However, by examining the fifth column, we can see that the exponent vectors (modulo 2) corresponding to $Q(43), Q(45)$, and $Q(46)$ already sum to the zero vector, suggesting correctly that for $v = 2^5 \cdot 3 \cdot 5$,

$$Q(43) \cdot Q(45) \cdot Q(46) = v^2.$$

Then letting $u = 43 \cdot 45 \cdot 46$, we generate the desired equivalence $u^2 \equiv v^2 \pmod{2041}$. Simplifying, we see that $u \equiv 1247 \pmod{2041}$, $v \equiv 480 \pmod{2041}$, and

$$\gcd(u - v, 2041) = \gcd(767, 2041) = 13$$

again yields a nontrivial factor of 2041. Note that we are a little lucky in this example because we only needed three vectors to find a linear dependence in $\mathbb{F}_2^4$, when in general we require five vectors in this space before any linear dependence is guaranteed.

Some questions remain about how to scale Brillhart and Morrison's method to larger n. When B is small, as in Example 4, it is feasible to identify a subset of exponent vectors, reduced modulo 2, that sum to zero by hand. But how difficult is it to identify such a subset when the number of vectors to choose from is much larger? Finding a linear dependence relation is a question of linear algebra which can be solved by the Wiedemann coordinate recurrence method in time $B^{2+o(1)}$ [29]. So once we identify $B + 2$ p_B-smooth auxiliary numbers, it is rather straightforward, using their exponent vectors, to find a subset whose product is a square.

A trickier problem is checking if the generated auxiliary numbers are p_B-smooth. To this point, we have been using trial division to accomplish this, but this strategy is efficient only when the auxiliary numbers are small; as x moves further from $\sqrt{n}$ the value of $Q(x) = x^2 - n$ grows and this approach becomes more and more costly. Since every auxiliary number needs to be checked for smoothness, and there may be many auxiliary numbers, advances in this area will have an outsized impact on the overall time necessary for factoring. In Section 2.2.1, we describe the improvements offered in determining smoothness by the Quadratic Sieve.

Finally, there is the challenge of choosing p_B. The smaller p_B is, the fewer smooth auxiliary numbers are needed before one can find a linear dependence relation in $\mathbb{F}_2^{B+1}$, and thus find a product of auxiliary numbers that are a square. However, choosing p_B small also decreases the chance that any given auxiliary number is p_B-smooth. So one could be forced to generate a large number of auxiliary numbers before finding enough smooth ones to proceed with finding a dependence relation. The ideal choice for p_B is then the solution to an interesting problem of optimization, which must strike a balance between these two extremes. The process for selecting p_B is detailed in Section 2.2.2; it is the final piece needed to give a complete description of the Quadratic Sieve algorithm.

2.2 The Quadratic Sieve

2.2.1 Using a Sieve to Evaluate Smoothness

If we wanted to check which integers in the range $[1, X]$ were Y-smooth, we would not proceed by trial division — we have a much more efficient solution, suggested by the Sieve of Eratosthenes. Beginning with the smallest prime, 2, we can divide every second number (i.e. every multiple of 2) by 2. We repeat this process with all the primes $p < Y$, dividing every p^{th} number in the sequence $1, \ldots, X$ by p. If we also divide out by small powers of p, where $p^e \leq Y$, we can see that at the end of this process, all the Y-smooth numbers in $[1, X]$ will have been transformed to 1. Only 1 in every p integers is divisible by p, so the time to divide the appropriate numbers in [1, X] by p is proportional to X/p. Then the time to sieve for every prime less than Y will be proportional to

$$X \sum_{p' \leq Y} \frac{1}{p'},$$

where p' is a prime or a power of a prime. We simplify the summation using the following theorem from Mertens in 1874:

Theorem 1. (Mertens) [20] For p prime,

$$\sum_{p \leq y} \frac{1}{p} = \log\log(y) + C + O\Big(\frac{1}{\log(y)}\Big),$$

where C is a constant.

From Mertens, we can see that the time to sieve $[1, X]$ for primes less than Y will be proportional to $X \log\log(Y)$. So for each integer in $[1, X]$, we are doing about $\log\log(Y)$ worth of work. This is much better than trial division, which takes $\pi(Y) = \#\{p \leq Y : p \text{ prime}\} \sim Y/\log(Y)$ work for each integer.

Under the method suggested by Brillhart and Morrison, we want to check for Y-smoothness on the sequence generated by $Q(x) = x^2 - n$, not $[1, X]$. Can the sieve technique still apply? What makes sieving efficient on $[1, X]$ is that the multiples of a prime or prime power p' appear at regular intervals. Fortunately, this property holds over the sequence $Q(x)$. Letting $x' = x + kp$, we have $x' \equiv x \pmod{p}$, and so $(x')^2 \equiv x^2 \pmod{p}$. Then, solutions to $Q(x) \equiv 0 \pmod{p}$, i.e. $x^2 \equiv n \pmod{p}$, give sequences $x, x \pm p, x \pm 2p, \ldots$ where the corresponding auxiliary numbers $Q(x)$ are divisible by p.

Solutions to $x^2 \equiv n \pmod{p}$ are well-understood. Assuming p and n are coprime and $p > 2$, then using Euler's criterion, we know that $x^2 \equiv n \pmod{p}$ has two solutions if $n^{(p-1)/2} \equiv 1 \pmod{p}$ and no solutions otherwise [7]. Further, this value can easily be checked using a fast-powering algorithm. If there are no solutions, no sieving needs to be done. If $x^2 \equiv n \pmod{p}$ has solutions, and $p \equiv 3 \pmod{4}$, the solutions $x \equiv \pm n^{(p+1)/4} \pmod{p}$ are given directly. In the case where $p \equiv 1 \pmod{4}$, we know of many algorithms, including those given by Cipolla-Lehmer and Tonelli-Shanks, that can compute solutions efficiently, in time $O(\log^3(p))$ and $O(\log^4(p))$ respectively [16]. The time to do these computations pales in comparison to the time needed to perform the sieve itself, as we will see in Section 2.2.2. Similar techniques, with equal efficiency, work for powers of odd primes and also in the case where $p = 2$, though $x^2 \equiv n \pmod{2}$ has at most one solution in this instance. Solving modulo powers of 2 is slightly more complicated, but still efficient when compared to sieving; if $x^2 \equiv n \pmod{2^k}$, and $k \geq 3$, we have four solutions $(x, -x, x + 2^{k-1}, -(x + 2^{k-1}))$ when $n \equiv 1 \pmod{8}$ and no solutions otherwise. Using Hensel's Lemma, we can lift solutions from the case where $x^2 \equiv n \pmod{2^{k-1}}$, finding a zero in $O(2n^2 \log^2(p))$ steps [4]. Thus obtaining solutions to $x^2 \equiv n \pmod{p^k}$ for $p^k \leq Y$ will not be an impediment to our algorithm.

Since we can solve $Q(x) \equiv 0 \pmod{p'}$, and from this solution easily locate auxiliary numbers that are divisible by p', for p' a prime or power of a prime, we can apply a sieve over primes up to Y to the sequence generated by the quadratic $Q(x) = x^2 - n$. Since multiples of primes appear at regular intervals in this sequence too, this will again require time proportional to $\log\log(Y)$ per integer in the sequence. So we retain our advantage over trial division in recognizing Y-smooth auxiliary numbers, and label this technique the Quadratic Sieve.

Example 5. (Quadratic Sieve) [19] We show how to use the sieve technique to factor $n = 87463$, with $p_B = 37$. First, we check for which primes p the equivalence $Q(x) = x^2 - n \equiv 0 \pmod{p}$ has solutions. We use the Legendre symbol $\left(\frac{n}{p}\right)$ to represent the value $n^{(p-1)/2}$.

p	2	3	5	7	11	13	17	19	23	29	31	37
$\left(\frac{n}{p}\right)$	1	1	-1	-1	-1	1	1	1	-1	1	-1	-1

Table 3: Primes for which n is a quadratic residue

Recall that $\left(\frac{n}{p}\right) = 1$ if and only if $x^2 - n \equiv 0 \pmod{p}$ has solutions. So the primes we need to sieve over are $P = \{2, 3, 13, 17, 19, 29\}$. What are the solutions to $x^2 \equiv n \pmod{p}$ for these primes?

p	2	3	13	17	19	29
x	1	1, 2	5, 8	7, 10	5, 14	12, 17

Table 4: Solutions to $x^2 \equiv 87463 \pmod{p}$.

Now we are ready to sieve. First we must generate the auxiliary numbers $Q(x) = x^2 - n$. We know $\lfloor\sqrt{87463}\rfloor = 295$, so letting $M = 30$, we generate $Q(x)$ for $x \in [295 - M, 295 + M]$. Then we can sieve through by each prime in P. We know that $265 \equiv 1 \pmod{2}$, so $2|Q(265)$, and every second number from there on. That is, $2|Q(267)$, $2|Q(269)$, and so forth. So we have identified each auxiliary number that we need to divide by 2. Similarly, $265 \equiv 1 \pmod{3}$, so $3|Q(265)$ and every third number from there on. Additionally, $266 \equiv 2 \pmod{3}$, so 3 also divides $Q(266)$ and every third number from there on. Now we have identified each auxiliary number that we need to divide by 3. We can continue on in this matter, identifying x equivalent to one of the solutions of p for each remaining $p \in P$, and then identifying the $Q(x)$ associated with it and its multiples to determine where to divide. Once we have done this for each $p \in P$, and small powers of the p's, the smooth auxiliary numbers will be transformed to 1.

There are six values of x for which $Q(x) = x^2 - n$ is smooth over P: 265, 278, 296, 299, 307, and 316. Their exponent vectors, reduced modulo 2, are:

x	-1	2	3	13	17	19	29
265	1	1	1	0	1	0	0
278	1	0	1	1	0	0	1
296	0	0	0	0	1	0	0
299	0	1	1	0	1	1	0
307	0	1	0	1	0	0	1
316	0	0	0	0	1	0	0

Table 5: Exponent vectors, reduced modulo 2, of the auxiliary numbers

We can see that taking $Q(265), Q(278), Q(296)$, and $Q(307)$, that is, the first, second, third, and fifth rows, produces the needed dependence. So we have $u = 265 \cdot 278 \cdot 296 \cdot 307 \equiv 34757 \pmod{n}$ and $v = \sqrt{(265^2 - n) \cdot (278^2 - n) \cdot (296^2 - n) \cdot (307^2 - n)} \equiv 28052 \pmod{n}$, with $u^2 \equiv v^2 \pmod{n}$. Taking $\gcd(u - v, n) = 149$, we get the factorization $n = 87463 = 149 \cdot 587$.

In this example we were fortunate that our relatively small choices for p_B, the largest prime we checked for smoothness over, and $2M$, the number of auxiliary numbers we sieved over, worked out. Section 2.2.2 will explain how to make intelligent choices for these variables.

2.2.2 Choosing p_B and Arriving at a Complexity Estimate

What is the optimal choice for p_B? Richard Schroeppel, in the late 1970s, proposed a way of answering this question using complexity theory [28]. Let Y estimate the value of p_B, and take X to be an upper bound on the size of auxiliary numbers used in factorization. Schroeppel's insight was to imagine the auxiliary numbers $Q(x)$ as a random sequence — an assumption that has held up pretty well in later surveys [29]. Following this assumption, we are able to compute an estimate for the number of steps needed to factor n via the method followed in Example 5 in terms of X and Y. Then we can minimize this expression, choosing Y as a function of X, to model the fastest possible factorization.

Recall that X represents an upper bound on the auxiliary numbers. If we sieve over the auxiliary numbers $Q(x) = x^2 - n$, where $x \in [n^{1/2}, n^{1/2} + n^{\epsilon}]$, $0 < \epsilon < 1/2$, then we have an approximate upper bound of $2n^{1/2+\epsilon}$ on the auxiliary numbers. So we can set $X = 2n^{1/2+\epsilon}$.

We need to find a group of auxiliary numbers whose product is a square — under the assumption that the sequence $Q(x)$ is basically random, this amounts to determining the expected number of random integers bounded by X needed to find a subset that multiply to a square. As in Example 5, we approach this problem by finding Y-smooth integers in our sequence of auxiliary numbers. To begin, we need to find one smooth integer.

Definition 2. Let $\psi(\mathbf{X}, \mathbf{Y})$ represent the number of Y-smooth integers in the interval [1, X].

The probability that a random positive integer up to X is $Y-$smooth is $\frac{\psi(X,Y)}{\lfloor X \rfloor} \approx \frac{\psi(X,Y)}{X}$. The expected number of random integers needed to find one Y-smooth integer is then the reciprocal of this number, $\frac{X}{\psi(X,Y)}$. However, it is not enough to find one Y-smooth integer.

Definition 3. Let $\pi(\mathbf{Y})$ denote the number of primes up to Y.

To find a linear dependence among the exponent vectors in $\mathbb{F}_2^{\pi(Y)+1}$, which indicates that a product of some subset of the auxiliary numbers is square, we need to find $\pi(Y) + 2$ auxiliary numbers which are Y-smooth. For simplicity, we round this to $\pi(Y)$. Then, we would expect to need $\pi(Y)X/\psi(X,Y)$ auxiliary numbers, assuming they appear randomly, to find enough Y-smooth numbers to factor. And once we generate an auxiliary number, we need to do some additional work to determine whether it is Y-smooth or not. As discussed in Section 2.2.1, using the Quadratic Sieve we can determine Y-smoothness in time $\log\log(Y)$ per auxiliary number. So the expected number of steps needed to find enough suitable auxiliary numbers to factor n is

$$\frac{\pi(Y)\log\log(Y)X}{\psi(X,Y)}.$$

We want to choose a value for Y, our estimate for p_B, that minimizes the number of steps needed to factor n. So we will choose Y as a function of X to minimize $\pi(Y)\log\log(Y)X/\psi(X,Y)$. We break down this term by making some estimates. Let $Y = X^{1/u}$. First, following [29], we approximate

$$\pi(Y)\log\log(Y) \approx X^{1/u}.$$

Next, we make an estimate for $\frac{X}{\psi(X,Y)}$. It is a result of Karl Dickman [10] that $\frac{\psi(X,X^{1/u})}{X} \sim \rho(u)$ for each fixed $u \geq 1$, where $\rho(u)$ is the Dickman-de Bruijn function, a continuous function that satisfies $u\rho'(u) = -\rho(u-1)$ for $u > 1$ and $\rho(u) = 1$ for $u \in [0,1]$. As u increases, by [9] we have

$$\rho(u) = \exp[-u\{\log(u) + \log\log(u) - 1 - \frac{1}{\log(u)} + \frac{\log\log(u)}{\log(u)} + O\Big(\frac{(\log\log(u))^2}{(\log(u))^2}\Big)\}].$$

Observe that the dominant term in this exponent is $-u(\log(u))$, which tells us that the behavior of $\rho(u)$ can be seen as similar to u^{-u}. E.R. Canfield, P. Erdős, and C. Pomerance showed in [12] that this property, that $\frac{\psi(X,X^{1/u})}{X} \sim \rho(u)$, which in turn resembles u^{-u}, remains mostly true as $X \to \infty$ and $u \to \infty$, as long as $X^{1/u} > \log(X)^{1+\epsilon}$. In such a case, we get

$$\frac{\psi(X,X^{1/u})}{X} = u^{-(1+o(1))u}$$

for any fixed $\epsilon > 0$. So we can approximate

$$\frac{X}{\psi(X,Y)} \approx u^u,$$

as $Y = X^{1/u}$. Then we reach the much cleaner estimate

$$\frac{\pi(Y)\log\log(Y)X}{\psi(X,Y)} \approx X^{1/u}u^u$$

for the number of steps required to factor n.

We want to choose u in a way that minimizes this expression. We take the logarithm of both sides so as to not work with exponents; this will not affect our estimate, as log is a strictly increasing function. Then we have the expression

$$\frac{1}{u}\log(X) + u\log(u),$$

which has derivative

$$\frac{-1}{u^2}\log(X) + 1 + \log(u).$$

This derivative will equal zero when $\log(X) = u^2(\log(u)+1)$. We take the logarithm of this equation, dropping lower degree terms, to see that

$$\log(u) \sim \frac{1}{2}\log\log(X).$$

Plugging this estimate for $\log(u)$ into the equation we obtained by setting the derivative to zero and solving for u, we see that

$$u \sim \Big(\frac{2\log(X)}{\log\log(X)}\Big)^{1/2}.$$

From here, we can estimate

$$\begin{aligned} Y &= X^{1/u} \\ &= \exp\Big(\frac{1}{u}\log(X)\Big) \\ &= \exp\Big(o(1)\Big(\frac{\log\log(X)}{2\log(X)}\Big)^{1/2}\log(X)\Big) \\ &= \exp\big((2^{-1/2} + o(1))(\log(X)\log\log(X))^{1/2}\big). \end{aligned}$$

We can also estimate

$$\begin{aligned}
u^u &= \exp(u\log(u)) \\
&\sim \exp\Big(\Big(\frac{2\log(X)}{\log\log(X)}\Big)^{1/2}\log\Big(\Big(\frac{2\log(X)}{\log\log(X)}\Big)^{1/2}\Big)\Big) \\
&\sim \exp\Big(\Big(\frac{2\log(X)}{\log\log(X)}\Big)^{1/2}\cdot\frac{1}{2}(\log(2)+\log\log(X)-\log\log\log(X))\Big) \\
&\sim \exp\Big(\Big(\frac{2\log(X)}{\log\log(X)}\Big)^{1/2}\cdot\frac{1}{2}(\log\log(X))\Big) \\
&\sim \exp\Big(\Big(\frac{\log(X)\log\log(X)}{2}\Big)^{1/2}\Big).
\end{aligned}$$

Combining this with the previous estimate, we can get an estimate for $X^{1/u}u^u$, the number of steps needed to perform the Quadratic Sieve to factor n. Then

$$X^{1/u}u^u = \exp\big((2^{1/2}+o(1))(\log(X)\log\log(X))^{1/2}\big).$$

Plugging in $X = 2n^{1/2+\epsilon}$, we get the following estimates for Y and the complexity of the sieve step:

$$\begin{aligned}
Y &= \exp\big((1/2+o(1))(\log(n)\log\log(n))^{1/2}\big) \\
X^{1/u}u^u &= \exp\big((1+o(1))(\log(n)\log\log(n))^{1/2}\big).
\end{aligned}$$

To simplify this expression, we introduce L-notation.

Definition 4. (L-notation) Define

$$\mathbf{L_n}[\alpha,\mathbf{c}] = \exp((c+o(1))(\log(n))^{\alpha}(\log\log(n))^{1-\alpha}),$$

where c is a positive constant and α is a constant with $0 \le \alpha \le 1$.

Then the sieve step of the Quadratic Sieve requires $L_n[1/2, 1]$ steps.

2.2.3 The Quadratic Sieve Algorithm

At last, we are ready to put forth an explicit algorithm for the Quadratic Sieve.

Algorithm 1. (Quadratic Sieve) Let n be a composite number divisible by at least two different primes. We proceed to find a nontrivial factorization of n.

1. Set $Y = \exp\big((\frac{1}{2}+o(1))\big(\log(n)\log\log(n)\big)^{\frac{1}{2}}\big)$, as discussed in Section 2.2.2.

2. Run through the sequence $Q(x) = x^2 - n$ until $\pi(Y)+2$ auxiliary values that are Y-smooth are found. Expect to need, by Section 2.2.2, $\{\pi(Y)n^{1/2+o(1)}\}/\{\psi(n^{1/2+o(1)}, Y)\}$ values of x before this occurs. The values of x chosen should be centered around $\lceil\sqrt{n}\rceil$, so that the auxiliary numbers $Q(x)$ remain small, and are more likely to be smooth. Do this step efficiently by generating a large quantity of auxiliary numbers first, and then using the Quadratic Sieve to check for smoothness, as in Example 5.

3. When at least $\pi(Y)+2$ smooth auxiliary numbers are identified, retrieve the exponent vectors of these numbers, reduced modulo 2, and use a technique from linear algebra, likely Wiedemann Coordinate Recurrence, to find a dependence among these vectors, which will correspond to a subsequence $Q(x_1), \dots Q(x_m)$ of auxiliary numbers that multiply to a square. Let this square be v^2. Recover v from the prime factorization given by the exponent vectors, and let $u = x_1 \cdots x_m$. Reduce both u and v modulo n.

4. We now have $u^2 \equiv v^2 \pmod{n}$. Check that $u \not\equiv \pm v \pmod{n}$; if this is the case, compute $\gcd(u-v, n)$ to retrieve a nontrivial factor of n. If $u \equiv \pm v \pmod{n}$, return to Step 2 and expand the values used for x to generate new smooth auxiliary numbers. Then move to Step 3 and look for a new linear dependence among the exponent vectors. Repeat this process until u and v are found such that $u^2 \equiv v^2 \pmod{n}$ but $u \not\equiv \pm v \pmod{n}$, at which point we can take the $\gcd(u-v, n)$ to find a nontrivial factor of n.

We now examine each step of our algorithm. Step 1 is only done once, and involves the computation of a constant, so it will involve negligible work — at worst, a small power of $\log(n)$. In Step 2 we perform the Quadratic Sieve, which by Section 2.2.2 is expected to take $L_n[1/2, 1]$ steps. To perform the sieve, we must also compute solutions to $x^2 \equiv n \pmod{p^k}$ for primes p such that $p^k \leq Y$; however, as noted in Section 2.2.1, these solutions can be found quite efficiently and in time that does not compare to the time involved in sieving. In Step 3 we use Wiedemann coordinate recurrence (or an equivalently fast method) to find a dependence among the exponent vectors of our smooth auxiliary numbers. The matrix of these exponent vectors will be approximately $\pi(Y) \times \pi(Y)$, which Wiedemann coordinate recurrence can tackle in time complexity $Y^{2+o(1)}$ [29]. This step makes a significant contribution to the runtime of the overall Quadratic Sieve algorithm, especially because, unlike the sieving step, there is no way to parallelize the implementation of the linear algebra step to gain a practical, although not asymptotic, speedup. But though it about equals it, the linear algebra of Step 3 still does not surpass the time required for the sieve in Step 2. Finally, in Step 4 the primary calculation is the gcd, which, as discussed in Section 2.1.2, can be done so efficiently via the Euclidean algorithm that its contribution to the overall running time of the algorithm is negligible. So the most time-intensive part of factoring via the Quadratic Sieve algorithm is performing the sieve itself, giving the overall algorithm a runtime of $L_n[1/2, 1]$.

2.3 Improvements on the Quadratic Sieve

In this section, we explore various alterations to the Quadratic Sieve. While the asymptotic complexity remains unchanged, as these changes mainly affect the $o(1)$ term, implementations of the Quadratic Sieve that make use of these methods see significant gains in performance over the traditional algorithm.

2.3.1 Working with Logarithms

One way to speed up the Quadratic Sieve is to reduce the amount of division necessary to identify smooth auxiliary numbers. Imagine we are sieving over $Q(x) = x^2 - n$ for $x \in (\lceil\sqrt{n}\rceil - M, \lceil\sqrt{n}\rceil + M)$. We can rewrite $Q(x) = (x + \lceil\sqrt{n}\rceil)^2 - n$, and sieve over $Q(x)$ for $x \in (-M, M)$. Then we can initialize a zero array of length $2M$, and for every prime p we are sieving over, add $\log(p)$ to the array at indices $x \pm kp$, where $x^2 \equiv n \pmod{p}$ and k is an integer. $Q(x) \approx M\sqrt{n}$ on the interval $(-M, M)$, so taking the natural map between $Q(x)$ and the locations in the array, any $Q(x)$ that are smooth should have value about $\frac{1}{2}\log(n)+\log(M)$ in the corresponding location of the array. Then we can restrict factoring via division to $Q(x)$ whose array look-ups approach $\frac{1}{2}\log(n)+\log(M)$, and avoiding doing division on any $Q(x)$ that are not smooth [34]. Using the array technique, the majority of divisions are replaced with addition operations, which are much less computationally expensive. Practically, this will improve the runtime of the algorithm, but since this division was not the main contributor to the complexity of the Quadratic Sieve, the overall asymptotic complexity estimate will be unchanged.

2.3.2 Multiple Polynomials

One challenge in finding the requisite number of smooth auxiliary numbers is that the more auxiliary numbers $Q(x)$ we consider, the further x moves from $\sqrt{n}$, and the larger $Q(x) = x^2-n$ becomes. As $Q(x)$ becomes larger, it is less and less likely to be smooth, which means we will have to look at an ever increasing number of auxiliary numbers to continue to find smooth ones. One way to alleviate this problem is to look at auxiliary numbers generated by multiple polynomials, not just $Q(x)$.

We have been thinking of the sequence of $Q(x)$'s as the being generated by the polynomial $x^2 - n$, where we choose x near $\lceil\sqrt{n}\rceil$. We can rewrite this expression as $Q(x) = (x + b)^2 - n$, where $b = \lceil\sqrt{n}\rceil$, so that we can choose x near zero. Peter Montgomery had the idea to generalize this quadratic, and use other quadratics of the form $Q_{a,b}(x) = (ax + b)^2 - n$, where $0 < b < \frac{a}{2}$ [34]. Choose $a = q^2$ and b so that $b^2 - n = ac$ for some integer c [16]. Then

$$\begin{aligned}
a^{-1}Q_{a,b}(x) &= a^{-1}((ax+b)^2-n)\\
&= a^{-1}(a^2x^2+2bax+b^2-n)\\
&= a^{-1}(a^2x^2+2bax+ac)\\
&= ax^2+2bx+c.
\end{aligned}$$

Furthermore,

$$\begin{aligned}
a^{-1}Q_{a,b}(x) &= (q^2)^{-1}((ax+b)^2-n)\\
&\equiv \big((ax+b)(q^{-1})\big)^2 \pmod{n}.
\end{aligned}$$

So

$$\big((ax+b)(q^{-1})\big)^2 \equiv ax^2+2bx+c \pmod{n}.$$

Thus we can let the polynomial $ax^2 + 2bx + c$ determine auxiliary numbers, and check the values it generates for smoothness. Since the quadratic $ax^2 + 2bx + c$ is equivalent to a square modulo n, we can apply the same trick of combining smooth auxiliary numbers to find u, v such that $u^2 \equiv v^2 \pmod{n}$, $u \not\equiv \pm v \pmod{n}$, and $\gcd(u-v, n)$ is nontrivial. By adjusting a and b, we can get many candidate polynomials for generating auxiliary numbers. So when the auxiliary numbers generated by a polynomial $Q_{a,b}(x)$ get so large that they are unlikely to be smooth, we can switch to another polynomial and generate another round of small, and more likely to be smooth, auxiliary numbers.

Multiple Polynomial Quadratic Sieve has several advantages. Computationally, it preserves the ability of the Quadratic Sieve algorithm to be parallelized across multiple processors, as each processor can sieve over auxiliary numbers generated by different polynomials, and contribute its results to a central database of smooth auxiliary numbers and their exponent vectors when it finishes [34]. By working with smaller auxiliary numbers, we can estimate a heuristic speedup by a factor of roughly $\frac{1}{2}\sqrt{\log(n)\log\log(n)}$ [16]. This translates to an algorithm that is about 17 times faster when n has around 100 digits, a real practical improvement, although asymptotically the change is only in the $o(1)$ term of our total estimate. Robert Silverman shows that the cost of changing the polynomial that generates the auxiliary numbers is small compared to the runtime benefits of parallelization, smaller auxiliary numbers, and smaller sieve sizes [34], cementing the beneficial status of the multiple polynomial version of the Quadratic Sieve.

2.3.3 Large Primes

Another improvement to the Quadratic Sieve involves a clever way of creating additional smooth auxiliary numbers. Imagine $Q(x) = x^2 - n$ generates our auxiliary numbers, and after sieving, we find that $Q(x_1)$ and $Q(x_2)$ are not p_B-smooth, i.e. are not transformed to one by the sieve process, but have an additional shared large prime factor L. So $Q(x_1) = x_1^2 - n = u \cdot L$ and $Q(x_2) = x_2^2 - n = v \cdot L$, where u, v are p_B-smooth. Then

$$\left(\frac{x_1 x_2}{L}\right)^2 \equiv u \cdot v \pmod{n},$$

meaning we have created an equivalence of the form $x^2 \equiv a \pmod{n}$ where a is p_B-smooth, which is exactly our goal when generating the auxiliary numbers anyway.

We can apply this insight to every auxiliary number $Q(x_i)$ by storing a list of tuples (x_i, L_i) that records the remaining large factor after each $Q(x_i)$ has been sieved. We can sort this list by L_i in $O(M\log(M))$ time, if M is the number of auxiliary numbers generated, and then do $O(M)$ pairwise comparisons to find auxiliary numbers with shared large factors. Once we identify numbers with shared L_i we can generate new pairs (x, a), where $x^2 \equiv a \pmod{n}$ and a is p_B-smooth, in the manner demonstrated above. It is a consequence of the problem often referred to as the “birthday paradox” that we expect to find many pairs of (x_i, L_i) with matching large prime factors L_i once we have generated enough auxiliary numbers [14].

If we can get additional smooth auxiliary numbers by matching large prime factors, we will need to generate fewer auxiliary numbers using quadratics, and do less sieving, to find enough relations to get the linear dependence we need. This slightly reduces the size of the typical auxiliary number used by the algorithm, only affecting the $o(1)$ term in the asymptotic complexity of the Quadratic Sieve but practically resulting in a speedup of more than a factor of two [5][22].

The large prime insight can be further extended to allow L to be the product of two large primes. This is known as the Double-Large Prime variation of the Quadratic Sieve, and has been shown to additionally improve the performance of the sieving algorithm [16].

This concludes our discussion of the Quadratic Sieve. In the next chapter, we will examine the Number Field Sieve, which improves on the framework of the Quadratic Sieve and reduces the runtime of the sieve step from $L_n[1/2, 1]$ to $L_n[1/3, (64/9)^{1/3}]$.

3 The Number Field Sieve

3.1 From the Quadratic Sieve to the Number Field Sieve

At the heart of both the Quadratic and the Number Field Sieve is Maurice Kraitchik's insight that if one can find u, v such that $u^2 \equiv v^2 \pmod{n}$ and $u \not\equiv \pm v \pmod{n}$, it is possible to obtain a nontrivial factor of n by computing $\gcd(u - v, n)$. Both algorithms use a sieve, and accompanying factor base, to construct the desired values u and v, but they differ in how they select numbers for use in the sieve. The Quadratic Sieve algorithm uses the function $Q(x) = x^2 - n$ to generate a long sequence of auxiliary numbers, some subset of which will multiply together to create v^2. Due to the quadratic nature of $Q(x)$, we immediately have $v^2 \equiv u^2 \pmod{n}$ for u equal to the product of the x-values corresponding to the auxiliary numbers that compose v^2. The fact that a subset of auxiliary numbers will multiply together to produce a square is guaranteed by linear algebra, as long as we can find $\pi(Y) + 2$ auxiliary numbers that are Y-smooth. Thus $Q(x)$ being quadratic creates a square on one side of our congruence, and the remaining square is generated using a linear algebra technique that requires only a sufficient quantity of smooth numbers.

The Number Field Sieve modifies the principle behind the Quadratic Sieve in one critical way — it uses this linear algebra technique to generate squares on both sides of the congruence modulo n, thus eliminating the need for the auxiliary numbers to be generated by a quadratic function. Instead, the Number Field Sieve requires pairs of the form $(\theta, \phi(\theta))$, where θ lies in an algebraic number ring and ϕ is a homomorphism from that ring to $\mathbb{Z}/n\mathbb{Z}$. The goal is to find k pairs $(\theta_i, \phi(\theta_i))$ with the property that $\theta_1 \cdots \theta_k = \gamma^2$, for some γ in the number ring, and $\phi(\theta_1) \cdots \phi(\theta_k) \equiv v^2 \pmod{n}$, with $v \in \mathbb{Z}$. Letting $\phi(\gamma) \equiv u \pmod{n}$, we would then have

$$u^2 \equiv \phi(\gamma)^2 \equiv \phi(\gamma^2) \equiv \phi(\theta_1 \cdots \theta_k) \equiv \phi(\theta_1) \cdots \phi(\theta_k) \equiv v^2 \pmod{n},$$

at which point, after verifying that $u \not\equiv \pm v \pmod{n}$, we can apply Kraitchik's trick to find a nontrivial factor of n by computing $\gcd(u - v, n)$. In Section 3.3, we give the specifics of this algorithm and discuss its asymptotic advantages over the Quadratic Sieve. To do so, we rely on some results from algebraic number theory, which are presented in Section 3.2.

3.2 Results from Algebraic Number Theory

The linear algebra technique used in the Quadratic Sieve requires generating a sufficient number of smooth quadratic residues. To extend this practice to the Number Field Sieve, we need to develop some results about prime ideals in number rings. Let $f(x) = x^d + c_{d-1}x^{d-1} + \cdots + c_0$ be an irreducible polynomial in $\mathbb{Z}[x]$, and let α be a complex root of f. Let $\mathbb{Z}[\alpha] = \mathbb{Z}[x]/(f(x))$ be the number ring. For the remainder of this section, the phrase "prime ideals" will be used to refer to non-zero prime ideals.

Lemma 2. [31] Let $\mathcal{O}$ be the ring of algebraic integers in $\mathbb{Q}(\alpha)$. Then nonzero ideals in $\mathcal{O}$ factor uniquely into prime ideals over $\mathcal{O}$.

Proof: The ring of algebraic integers is a Dedekind domain. Every Dedekind domain has the property that all nonzero ideals factor uniquely into prime ideals. □

We define a few different measurements for ideals and elements.

Definition 5. (Element Norm) [16] Recall $f(x) = x^d + c_{d-1}x^{d-1} + \cdots + c_0$. Let $\alpha_1, \ldots, \alpha_d$ be the complex roots of $f(x)$, with $\alpha_1 = \alpha$, and let $\beta = s_0 + s_1\alpha + \cdots + s_{d-1}\alpha^{d-1} \in \mathbb{Q}[\alpha]$. Define the **norm** of β to be the determinant of the map that sends $x \to \beta x$. We have

$$N(\beta) = \prod_{j=1}^{d}(s_0 + s_1\alpha_j + \cdots + s_{d-1}\alpha_j^{d-1}).$$

The norm is a rational number, and if all the s_i are integers, then $N(\beta)$ is an integer as well.

Making use of the complex roots, we can write $f(x) = (x - \alpha_1)\cdots(x - \alpha_d)$. Letting $a, b \in \mathbb{Z}$, $b \neq 0$, we can get a new expression for the norm of $a + b\alpha$:

$$N(a + b\alpha) = (a + b\alpha_1)\cdots(a + b\alpha_d) = (-b)^d\Big(\frac{a}{-b} - \alpha_1\Big)\cdots\Big(\frac{a}{-b} - \alpha_d\Big) = (-b)^d f\Big(\frac{a}{-b}\Big).$$

Definition 6. (Trace) [16] Recall the descriptions of $f(x)$, $\alpha_1, \ldots, \alpha_d$, and α from Definition 5. Represent $\beta \in \mathbb{Q}(\alpha)$ by $\beta = s_0 + s_1\alpha + \cdots + s_{d-1}\alpha^{d-1}$, with $s_i \in \mathbb{Q}$. The **trace** of β is

$$\mathrm{tr}(\beta) = \sum_{j=1}^{d}(s_0 + s_1\alpha_j + \cdots + s_{d-1}\alpha_j^{d-1}).$$

The trace is $\mathbb{Q}$-linear, and if all the s_i are integers, then $\mathrm{tr}(\beta)$ is an integer as well.

Definition 7. (Order) [27] Let $\mathcal{O}$ be the ring of algebraic integers in $\mathbb{Q}(\alpha)$. Say $A \subset \mathcal{O}$ is an **order** of $\mathbb{Q}(\alpha)$ if it is a subring (with 1) with the property that the index $|\mathcal{O} : A|$ is finite.

Observe that $\mathbb{Z}[\alpha]$ is an order of $\mathbb{Q}(\alpha)$. It is to this order that we will eventually apply many of the definitions and results that follow.

Definition 8. (Ideal Norm) [11] Let $A \subset \mathcal{O}$ be an order as in Definition 7. The **ideal norm** $\mathfrak{N}\mathfrak{a}$ of a nonzero ideal $\mathfrak{a} \subset A$ is $|A/\mathfrak{a}|$.

Lemma 3. [11] If $\mathfrak{p} \subseteq \mathbb{Z}[\alpha]$ is a prime ideal, then $\mathfrak{p}$ contains a unique prime number p and $\mathbb{Z}[\alpha]/\mathfrak{p}$ is a finite field.

Proof: We give proof of this statement for a general order A. For a nonzero $x \in A$, we have $\#(A/xA) = |N(x)|$ [11]. As $|N(x)|$ is finite, this implies that for every nonzero ideal $\mathfrak{a} \subset A$, $A/\mathfrak{a}$ will also be finite. If $\mathfrak{p} \subset A$ a nonzero prime ideal of A, then not only is $A/\mathfrak{p}$ finite, but it is also an integral domain. Being a finite integral domain, $A/\mathfrak{p}$ is then a field, and so $\mathfrak{p}$ is a maximal ideal of A and thus must contain a unique prime number $p \in \mathbb{Z}$. □

Definition 9. (Degree) [11] The **degree** of a prime ideal $\mathfrak{p}$ is $[\mathbb{Z}[\alpha]/\mathfrak{p} : \mathbb{F}_p]$, where p is the unique prime associated with $\mathfrak{p}$.

Definition 10. ($R(p)$) [16] Recall $f(x) = x^d + c_{d-1}x^{d-1} + \cdots + c_0$. Let

$$R(p) = \{r \in [0, 1, \ldots, p-1] \mid f(r) \equiv 0 \pmod{p}\}.$$

Lemma 4. [11] There is a one-to-one correspondence between pairs (p, r), where p is a prime number and $r \in R(p)$, and degree one prime ideals $\mathfrak{p} \subseteq \mathbb{Z}[\alpha]$. For each pair (p, r), we have $\mathfrak{p} = (p, \alpha - r)$.

Proof: Assume $\mathfrak{p}$ is a degree one prime ideal. By definition, we have $\mathbb{Z}[\alpha]/\mathfrak{p} \cong \mathbb{F}_p$. Consider the map $\varphi : \mathbb{Z}[\alpha] \to \mathbb{F}_p$ with kernel $\mathfrak{p}$. Then φ sends α, a root of f, to $r \pmod{p}$, a root of $f \pmod{p}$. This gives a relationship between a prime ideal $\mathfrak{p}$ and a pair (p, r), with p a prime and $r \in R(p)$.

In the other direction, let p be a prime and $r \in R(p)$. There is a unique ring homomorphism $\phi : \mathbb{Z}[\alpha] \to \mathbb{F}_p$ that sends α to $r \pmod{p}$. The kernel of ϕ will be a first degree prime $\mathfrak{p}$ of $\mathbb{Z}[\alpha]$. This establishes an injective relationship in the other direction, completing the proof. □

Definition 11. ($e_{p,r}$) [11] Let $\gcd(a, b) = 1$. Let p be a prime and $r \in R(p)$, as defined in Definition 10. Define

$$e_{p,r}(a + b\alpha) = \begin{cases} \operatorname{ord}_p(N(a + b\alpha)) & a + br \equiv 0 \pmod{p} \\ 0 & \text{otherwise,} \end{cases}$$

where $\operatorname{ord}_p(k)$ is the number of factors of p in k. Observe that

$$N(a + b\alpha) = \pm \prod_{p \text{ prime}, r \in R(p)} p^{e_{p,r}(a+b\alpha)}.$$

We introduce a proposition that relates the ideal norm and the element norm in $\mathbb{Z}[\alpha]$ via a group homomorphism. Then, we give a corollary that relates the values taken by this homomorphism to $e_{p,r}$.

Proposition 1. [11] For each prime ideal $\mathfrak{p} \subset A$, with A an order as in Definition 7, there exists a group homomorphism $l_{\mathfrak{p}} : (\mathbb{Q}(\alpha))^* \to \mathbb{Z}$ such that

1. $l_{\mathfrak{p}}(\beta) \geq 0$ for all $\beta \in A \backslash \{0\}$.
2. If $\beta \in A \backslash \{0\}$, then $l_{\mathfrak{p}}(\beta) > 0 \Leftrightarrow \beta \in \mathfrak{p}$.
3. for each $\beta \in (\mathbb{Q}(\alpha))^*$, we have $l_{\mathfrak{p}}(\beta) = 0$ for all but finitely many p and

$$\prod_{\mathfrak{p}} (\mathfrak{N}\mathfrak{p})^{l_{\mathfrak{p}}(\beta)} = |N(\beta)|,$$

where the product is over all prime ideals $\mathfrak{p} \subset A$.

Proof: First, we show how to construct $l_{\mathfrak{p}}$, and then show that this homomorphism satisfies the listed properties. Let $\mathfrak{p} \subset A$ be a prime, and $\beta \in A \backslash \{0\}$. Since βA is of finite index in A, there is a finite chain of ideals

$$A = \mathfrak{a}_0 \supset \mathfrak{a}_1 \supset \cdots \supset \mathfrak{a}_t = \beta A$$

such that $\mathfrak{a}_{i-1}/\mathfrak{a}_i$ is a field, that is, there are no proper ideals between $\mathfrak{a}_{i-1}$ and $\mathfrak{a}_i$. We define $l_{\mathfrak{p}}(\beta)$ to be the number of i between $\{1, 2, \ldots, t\}$ such that $\mathfrak{a}_{i-1}/\mathfrak{a}_i \cong A/\mathfrak{p}$. It is a consequence of the Jordan-Hölder Theorem that $l_{\mathfrak{p}}(\beta)$ does not depend on the choice of the ideal chain $\mathfrak{a}_0, \ldots, \mathfrak{a}_t$.

If x and y are nonzero elements of A, and $\mathfrak{a}_0, \ldots, \mathfrak{a}_t$, $\mathfrak{b}_0, \ldots, \mathfrak{b}_u$, the corresponding chains of ideals, we can combine the chains to form a large chain $\mathfrak{a}_0, \ldots, \mathfrak{a}_t = x\mathfrak{b}_0, \ldots, x\mathfrak{b}_u$. This demonstrates that $l_{\mathfrak{p}}(xy) = l_{\mathfrak{p}}(x) + l_{\mathfrak{p}}(y)$. Every $z \in (\mathbb{Q}(\alpha))^*$ can be written as x/y, for $x, y \in A$. By letting $l_{\mathfrak{p}}(x/y) = l_{\mathfrak{p}}(x) - l_{\mathfrak{p}}(y)$, we can extend $l_{\mathfrak{p}}$ to a group homomorphism on $(\mathbb{Q}(\alpha))^*$. By construction, we can observe that property (1) holds.

Next we verify the existence of property (2). If $\beta \in \mathfrak{p}$, we can take $\mathfrak{a}_1 = \mathfrak{p}$, since we know that $\mathfrak{p}$ is maximal (see proof of Lemma 3). Then $\mathfrak{a}_0/\mathfrak{a}_1 = A/\mathfrak{p}$, and so by definition we know $l_\mathfrak{p}(\beta) \geq 1$. If $\beta \notin \mathfrak{p}$, since $\mathfrak{p}$ is maximal, we know that $\beta A + \mathfrak{p} = A$. Then $\beta y + z = 1$ for some $y \in A$, $z \in \mathfrak{p}$. In other words, we know that $z \equiv 1 \pmod{\beta A}$, meaning multiplication by z is the identity map from $A/\beta A \to A/\beta A$. Then $z \cdot (\mathfrak{a}_{i-1}/\mathfrak{a}_i) = \mathfrak{a}_{i-1}/\mathfrak{a}_i$ for all i. Since $z \in \mathfrak{p}$, if $\mathfrak{a}_{i-1}/\mathfrak{a}_i \cong A/p$, we would expect $z \cdot (\mathfrak{a}_{i-1}/\mathfrak{a}_i) = 0$, and so it must be that $\mathfrak{a}_{i-1}/\mathfrak{a}_i \ncong A/p$ for any i. Hence $l_\mathfrak{p}(\beta) = 0$ in the case that $\beta \notin \mathfrak{p}$. This concludes the proof of property (2).

It is sufficient to prove property (3) only in the case where $\beta \in A$, since $N(x/y) = N(x)/N(y)$ and $l_\mathfrak{p}(x/y) = l_\mathfrak{p}(x) - l_\mathfrak{p}(y)$. Recall from the proof of Lemma 3 that $|N(\beta)| = |A/\beta A| = \prod_{i=1}^{t} |\mathfrak{a}_{i-1}/\mathfrak{a}_i|$. Then, to show equivalence between this product and the one given in property (3), we need to show that for each i there is only one prime ideal $\mathfrak{p} \subset A$ with $\mathfrak{a}_{i-1}/\mathfrak{a}_i \cong A/\mathfrak{p}$, as $\mathfrak{N}\mathfrak{p} = |A/p|$. Let $y \in \mathfrak{a}_{i-1}$, $y \notin \mathfrak{a}_i$. As there is no proper ideal between $\mathfrak{a}_i$ and $\mathfrak{a}_{i-1}$, we have $yA + \mathfrak{a}_i = \mathfrak{a}_{i-1}$. Thus $yA/\mathfrak{a}_i \cong \mathfrak{a}_{i-1}/\mathfrak{a}_i$, and so multiplication by y gives a surjective map from $A \to \mathfrak{a}_{i-1}/\mathfrak{a}_i$. Letting $\mathfrak{p}$ be the kernel of this surjective map, we have $A/\mathfrak{p} \cong \mathfrak{a}_{i-1}/\mathfrak{a}_i$. Since $\mathfrak{a}_i$ is maximal in $\mathfrak{a}_{i-1}$, we know $\mathfrak{a}_{i-1}/\mathfrak{a}_i$ is a field with no proper ideals. Thus $\mathfrak{p}$ is a maximal ideal, and therefore a prime ideal. Since $\mathfrak{p}$ is also the annihilator of $\mathfrak{a}_{i-1}/\mathfrak{a}_i$, we know that $\mathfrak{p}$ is unique. Thus we have

$$|N(\beta)| = |A/\beta A| = \prod_{i=1}^{t} |\mathfrak{a}_{i-1}/\mathfrak{a}_i| = \prod_{\mathfrak{p}} |A/\mathfrak{p}|^{l_\mathfrak{p}(\beta)} = \prod_{\mathfrak{p}} (\mathfrak{N}\mathfrak{p})^{l_\mathfrak{p}(\beta)},$$

completing the proof of property (3). □

In the case where $A = \mathcal{O}$, with $\beta \in \mathcal{O}$, $\beta \neq 0$, we can use Lemma 2 to write $\beta\mathcal{O} = \mathfrak{p}_1\mathfrak{p}_2 \cdots \mathfrak{p}_t$. Letting $\mathfrak{a}_i = \mathfrak{p}_1\mathfrak{p}_2 \cdots \mathfrak{p}_i$, we can see that $l_\mathfrak{p}(\beta)$ is the exponent of the power of $\mathfrak{p}$ dividing $\beta\mathcal{O}$. Thinking of $l_\mathfrak{p}$ as the exponent of a prime ideal is helpful in intuiting the connection between $l_\mathfrak{p}$ and $e_{p,r}$, which corresponds to an exponent of a prime number. Corollary 1 further develops this connection.

Corollary 1. [11] Let a, b be coprime integers, and let $\mathfrak{p} \subseteq \mathbb{Z}[\alpha]$ be a prime ideal. If $\mathfrak{p}$ does not have degree one, then $l_\mathfrak{p}(a + b\alpha) = 0$. If $\mathfrak{p}$ is of degree one, and corresponds to the pair (p, r), with p a prime integer and $r \in R(p)$, then $l_\mathfrak{p}(a + b\alpha) = e_{p,r}(a + b\alpha)$.

Proof: Let $\mathfrak{p} \subseteq \mathbb{Z}[\alpha]$ be a prime ideal, and let p be the unique prime integer contained in $\mathfrak{p}$, which exists by Lemma 3. Assume $l_\mathfrak{p}(a + b\alpha) > 0$. We use the quotient map $\mathbb{Z}[\alpha] \to \mathbb{Z}[\alpha]/\mathfrak{p}$ throughout the proof.

By Proposition 1, we then know that $a + b\alpha \in \mathfrak{p}$, and so $a + b\alpha = 0 \in \mathbb{Z}[\alpha]/\mathfrak{p}$. If $p|b$, then $b\alpha = 0$ in $\mathbb{Z}[\alpha]/\mathfrak{p}$, which would imply that $a = 0$ in this quotient ring as well, suggesting that $p|a$. However, a and b are assumed to be coprime, so this is a contradiction. Therefore it must be the case that b maps to a nonzero element of $\mathbb{Z}[\alpha]/\mathfrak{p}$. Recall from Lemma 3 that $\mathbb{Z}[\alpha]/\mathfrak{p}$ is a field, and so there exists some b' which is the inverse of the image of b in $\mathbb{Z}[\alpha]/\mathfrak{p}$. We know that $b' \in \mathbb{F}_p \subset \mathbb{Z}[\alpha]$. Since $a + b\alpha = 0 \in \mathbb{Z}[\alpha]/\mathfrak{p}$, we can solve for α and see that $\alpha = -ab'$ in this field. As $-ab' \in \mathbb{F}_p$, this implies that every element in $\mathbb{Z}[\alpha]$ maps into $\mathbb{F}_p$ under the quotient map $\mathbb{Z}[\alpha] \to \mathbb{Z}[\alpha]/\mathfrak{p}$, confirming that $\mathfrak{p}$ has degree one.

If $\mathfrak{p}$ corresponds to the pair (p, r), we can see that r is also determined by the equation $a + br \equiv 0 \pmod{p}$. Since the pair (p, r) is unique, this confirms that $\mathfrak{p}$ is the unique prime ideal of $\mathbb{Z}[\alpha]$ containing both p and $a + b\alpha$. Then the statement that $l_{\mathfrak{p}}(a + b\alpha) = e_{p,r}(a + b\alpha)$ follows by comparing the power of p on both sides of the equation in part three of Proposition 1, with $\beta = a + b\alpha$. On the left hand side, we are looking for ideals $\mathfrak{p}'$ with $\mathfrak{N}\mathfrak{p}' = p^k$ such that $a + b\alpha \in \mathfrak{p}'$, so that $l_{\mathfrak{p}'}(a + b\alpha) > 0$. This means that $\mathfrak{p}'$ must be a prime ideal containing both p and $a + b\alpha$, and the only such ideal is $\mathfrak{p}$. So we have $p^{l_{\mathfrak{p}}(a+b\alpha)}$ on the left hand side. On the right hand side, we have $|N(a + b\alpha)| = |\prod_{p \text{ prime}, r \in R(p)} p^{e_{p,r}(a+b\alpha)}|$. There is only one value of r such that $e_{p,r}(a + b\alpha) \neq 0$, and so on the right hand side we have $p^{e_{p,r}(a+b\alpha)}$. Since the left and right hand sides are equal, we can conclude that $l_{\mathfrak{p}}(a + b\alpha) = e_{p,r}(a + b\alpha)$. This completes the proof. □

Finally, we introduce a lemma that gives a natural relationship between an element in $\mathcal{O}$, the ring of algebraic integers of $\mathbb{Q}(\alpha)$, and an element in $\mathbb{Z}[\alpha]$.

Lemma 5. [16] Let $f(x)$ be a monic irreducible polynomial in $\mathbb{Z}[x]$, with α a complex root of f. Denote by $\mathcal{O}$ the ring of algebraic integers in $\mathbb{Q}(\alpha)$. Let $\gamma \in \mathcal{O}$. Then $f'(\alpha)\gamma \in \mathbb{Z}[\alpha]$.

Proof: Let

$$f(x)/(x - \alpha) = \sum_{j=0}^{d-1} \beta_j x^j,$$

with each $\beta_j \in \mathbb{Q}(\alpha)$. By [36, Proposition 3-7-12], originally attributed to Euler, we know that $\beta_0/f'(\alpha), \ldots, \beta_{d-1}/f'(\alpha)$ forms the dual basis to the $\mathbb{Q}$-basis $1, \alpha, \ldots, \alpha^{d-1}$ with respect to the pairing $\langle x, y\rangle = \operatorname{tr}(x, y)$. Furthermore, we know that each $\beta_j \in \mathbb{Z}[\alpha]$ and that the trace of $\alpha^k\beta_j/f'(\alpha)$ equals 1 if $j = k$ and 0 otherwise. Using our basis, we can find rational numbers $s_0, \ldots, s_{d-1}$ such that

$$\gamma = \sum_{j=0}^{d-1} s_j \cdot (\beta_j/f'(\alpha)).$$

Multiplying both sides by α^k, we get $\gamma\alpha^k = \sum_{j=0}^{d-1} s_j \cdot (\alpha^k\beta_j/f'(\alpha))$. Then taking the trace of each side, we have

$$\operatorname{tr}(\gamma\alpha^k) = \sum_{j=0}^{d-1} s_j \cdot \operatorname{tr}(\alpha^k\beta_j/f'(\alpha)),$$

since the trace is $\mathbb{Q}$-linear. Recalling from [36, Proposition 3-7-12] that the term $\operatorname{tr}(\alpha^k\beta_j/f'(\alpha))$ is 0 unless $j = k$, and 1 in the case of equality, the only non-zero term in the summand is s_k, and we see that $\operatorname{tr}(\gamma\alpha^k) = s_k$ for k equalling $0, 1, \ldots, d - 1$. Since γ and α are both in $\mathcal{O}$, the product $\gamma\alpha^k$ is as well, and so by Definition 6 we know $\operatorname{tr}(\gamma\alpha^k) \in \mathbb{Z}$. We also know $\operatorname{tr}(\gamma\alpha^k) = s_k$, and so we can conclude $s_k \in \mathbb{Z}$ for all s_k. Given this fact, and the fact from Euler that each $\beta_k \in \mathbb{Z}[\alpha]$, we can observe that $f'(\alpha)\gamma = \sum_{j=0}^{d-1} s_j\beta_j \in \mathbb{Z}[\alpha]$, completing the proof. □

3.3 The Number Field Sieve

We first preview the major steps in the algorithm, before getting into the details of each phase.

Algorithm 2. (Number Field Sieve Overview) [16] Let n be an odd, composite integer that is not a prime power.

1. Let $f(x) = x^d + c_{d-1}x^{d-1} + \cdots + c_0$ be an irreducible polynomial in $\mathbb{Z}[x]$, and let α be a complex root of f.
2. Let $\mathbb{Z}[\alpha] = \mathbb{Z}[x]/(f(x))$ be the number ring. Elements in $\mathbb{Z}[\alpha]$ are polynomials in α with integer coefficients. Since $f(\alpha) = 0$, these polynomials can have degree at most $d - 1$.
3. Let m be an integer such that $f(m) \equiv 0 \pmod{n}$.
4. Define $\phi : \mathbb{Z}[\alpha] \to \mathbb{Z}/n\mathbb{Z}$, where
$$\phi(a_0 + a_1\alpha + \cdots + a_{d-1}\alpha^{d-1}) = a_0 + a_1 m + \cdots + a_{d-1}m^{d-1} \pmod{n}.$$
5. Consider elements $\theta \in \mathbb{Z}[\alpha]$ which take the form $\theta = a + b\alpha$, with $a, b \in \mathbb{Z}$. We assume $\gcd(a, b) = 1$ to avoid trivial redundancy.
6. Find a set S of coprime integer pairs (a, b) such that
   (a)
   $$\prod_{(a,b)\in S} (a + bm) = v^2$$
   (b)
   $$\prod_{(a,b)\in S} (a + b\alpha) = \gamma^2$$
   with $v \in \mathbb{Z}$ and $\gamma \in \mathbb{Z}[\alpha]$.
7. Compute $v = \sqrt{v^2}$ and $\gamma = \sqrt{\gamma^2}$.
8. Let $u = \phi(\gamma)$. Compute $\gcd(u - v, n)$.

The description of the Number Field Sieve given in Algorithm 2 leaves many questions unanswered. Why should we expect this procedure to be more powerful than the Quadratic Sieve? How do we find $f(x)$ and m? How do we build the set S? How do we take square roots in $\mathbb{Z}[\alpha]$? The next four subsections will address these issues, and in Section 3.3.5 we will finally give a more complete description of the algorithm.

3.3.1 Finding $f(x)$ and m.

The first step in finding $f(x)$ is deciding upon d, the desired degree of the polynomial. In Section 3.3.4, we will describe a heuristic argument that suggests a choice of $d \sim \left(\frac{3\log(n)}{\log\log(n)}\right)^{1/3}$. In practice, we often choose d to be 4 or 5 when n is approximately 130 digits [16][17].

Once d has been determined, constructing $f(x)$ and finding m is straightforward. Perhaps counterintuitively, we fix m first, letting $m = \lfloor n^{1/d} \rfloor$. Then we write n in base m, so that

$$n = m^d + c_{d-1}m^{d-1} + \ldots + c_1 m + c_0,$$

and $0 \leq c_i < m$ for all c_i. This expression suggests a natural choice for $f(x)$, namely

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

as this construction guarantees $f(m) \equiv 0 \pmod{n}$. However, nothing about our construction ensures that $f(x)$ is irreducible. Nearly all polynomials are irreducible, but in the case that $f(x)$ is not, we can use the polynomial-time algorithm of Lenstra, Lenstra, and Lovász [23] to write $f(x) = g(x)h(x)$. Then we have $n = f(m) = g(m)h(m)$, and it is a result of Brillhart, Filaseta, and Odlyzko that this factorization of n will be nontrivial [8]. Since our ultimate goal is to factor n, the "failure" of producing a reducible $f(x)$ should instead be interpreted as an efficient, successful completion of the task at hand. So in this unlikely scenario, we are not really troubled, as we are still able to obtain the desired factorization of n.

Finally, we mention that it is not necessary to numerically compute α, the chosen complex root of f, until Step 7 of Algorithm 2 [16]. For now, it suffices to use the symbol α as a placeholder. In any case, computing the roots of a polynomial is a fairly simple task compared to the work done in Step 6 of the algorithm, where S is built.

3.3.2 Building S

Elements (a, b) in the set S must satisfy two criteria. The first is that the product $\prod_{(a,b)\in S}(a+bm)$ must be equal to a square. The task of identifying the correct subset of numbers $G(a, b) = a + bm$ that will do this is analogous to the task of identifying the subset of auxiliary numbers $Q(x) = x^2 - n$ that multiplied to a square in the Quadratic Sieve, and we can use an approach similar to that used in the Quadratic Sieve to tackle this problem. That is, for some Y, we wish to identify numbers $G(a, b)$ that are Y-smooth, and compute their exponent vectors modulo 2. Once $\pi(Y) + 2$ smooth numbers have been found, we can use linear algebra to find a subset that will multiply to a square. Again, we can use a sieve to quickly identify Y-smooth numbers. Since we have two variables to work with in $G(a, b)$, as compared to the one variable of $Q(x)$, we can fix a, sieve over several values of b, then fix a new value of a, and sieve again over values of b, repeating this process until enough Y-smooth numbers $G(a, b)$ have been discovered. Thus the process for finding a set S that satisfies the first criterion is familiar.

However, we also need S to satisfy a second criterion — we need $\prod_{(a,b)\in S}(a+b\alpha) = \gamma^2$, for $\gamma \in \mathbb{Z}[\alpha]$. Though the method for satisfying this criterion is broadly similar to that used for the first criterion, there are a few additional challenges, as we are now working in $\mathbb{Z}[\alpha]$, rather than $\mathbb{Z}$. To perform sieving and do linear algebra, we need some way of relating elements in $\mathbb{Z}[\alpha]$ with elements in $\mathbb{Z}$. We use the norm of $a + b\alpha$, defined in 5, to do this conversion. Since the norm is multiplicative, meaning that $N(\beta\beta') = N(\beta)N(\beta')$, we cannot have $\prod_{(a,b)\in S}(a + b\alpha) = \gamma^2 \in \mathbb{Z}[\alpha]$ unless $N(\prod_{(a,b)\in S}(a + b\alpha)) = u^2$ for some $u \in \mathbb{Z}$. This provides some motivation for utilizing $N(a + b\alpha)$ in the Number Field Sieve construction.

Letting

$$F(x, y) = x^d + c_{d-1}x^{d-1}y + \cdots + c_0 y^d = y^d f\left(\frac{x}{y}\right)$$

be the homogeneous form of f, we can write $N(a + b\alpha) = F(a, -b)$. If we wanted to find a set S of integer pairs (a, b) so that $\prod_{(a,b)\in S} F(a, -b) = u^2$, for $u \in \mathbb{Z}$, we could do this through sieving and exponent vectors the same way we were able to find a set S' such that $\prod_{(a,b)\in S'}(a+bm) = v^2$. Recalling that $G(a,b) = a+bm$, we could even guarantee that the same set S satisfied both criteria by sieving over values of the function $H(a,b) = F(a,-b)G(a,b)$ and creating a doubly-long exponent vector that records the parity of primes in $F(a,-b)$ and $G(a,b)$ separately, then performing the linear algebra step when $2\pi(Y)+3$ smooth values are identified.

Unfortunately, though it is necessary that $N(\prod_{(a,b)\in S}(a+b\alpha))$ be a square if $\prod_{(a,b)\in S}(a+b\alpha)$ is a square, this condition is not sufficient – it is possible for this norm to be square without the product in the number ring being square. Take, for example, the number 5 in the ring $\mathbb{Z}[i]$. We can write $5 = (2+i)(2-i)$ and see that 5 is not a square, whereas $N(5) = 25 = 5^2$ is. The problem here is that the two prime factors of 5 in $\mathbb{Z}[i]$, which are $(2+i)$ and $(2-i)$, have the same norm, even though they are distinct. If we want to work with norms, instead of dealing with elements in $\mathbb{Z}[\alpha]$ directly, we will need some way of preserving this type of distinction in order to manufacture products β where both $\beta \in \mathbb{Z}[\alpha]$ and $N(\beta) \in \mathbb{Z}$ are squares. To do this, we have to record some additional information in our exponent vectors.

Recall the definition of $R(p)$ from Definition 10. If $\gcd(a,b) = 1$, we can see that $F(a,-b) \equiv 0 \pmod p$ if and only if $a \equiv -br \pmod p$ for some $r \in R(p)$. This means that in our sieve, when we identify some prime p that divides $F(a,-b)$, we know there is a corresponding $r \in R(p)$ such that $a+br \equiv 0 \pmod p$. We can keep track of this r in our exponent vector as well, creating separate coordinate entries (p,r) for each $r \in R(p)$. This means that our exponent vectors will contain $\pi(Y)$ coordinates that record the parity of primes in $G(a,b)$, followed by Y' coordinates recording the parity of primes associated with a specific r in $F(a,-b)$, with $Y' = \#\{(p,r) : p < Y, p \text{ prime}, r \in R(p)\}$. We expect $Y' \approx \pi(Y)$, as most p only have r associated with them [11]. We will refer to this special representation associated with the sieving of $F(a,-b)$ as an extended exponent vector. Example 6 shows how documenting this added information can preserve distinctions between primes in $\mathbb{Z}[\alpha]$ with the same norm.

Example 6. We revisit our earlier discussion of $\mathbb{Z}[i] = \mathbb{Z}[x]/(x^2+1)$, and the question of whether or not $(2+i)(2-i) = 5 \in \mathbb{Z}[i]$ is a square. The elements $2-i$ and $2+i$ each have norm 5, so we will carry primes up to 5 in our exponent vectors. The first step is then to compute $R(p) = \{0 \le r < p \mid r^2+1 \equiv 0 \pmod p\}$ for $p = 2, 3$, and 5. We get $R(2) = \{1\}$, $R(3) = \emptyset$, and $R(5) = \{2,3\}$, so our extended exponent vectors will have three coordinates corresponding to pairs (p,r): $(2,1)$, $(5,2)$, and $(5,3)$.

Since $5|N(2-i) = F(2,1)$, we need to identify $r \in R(5)$ such that $2+r \equiv 0 \pmod 5$. Since $r = 3$ in this case, the exponent vector for $2-i$ takes the form $(0,0,1)$. We also have $5|N(2+i) = F(2,-1)$, but in this case the $r \in R(5)$ that makes $2-r \equiv 0 \pmod 5$ is 2, so the exponent vector for $(2+i)$ is $(0,1,0)$. Adding the exponent vectors for $2-i$ and $2+i$, we get $(0,1,1) \not\equiv (0,0,0) \pmod 2$, which corresponds to the fact that $(2+i)(2-i) = 5$ is not a square in $\mathbb{Z}[i]$.

Note that if we had not kept separate coordinates in the exponent vector for each $r \in R(p)$, the exponent vectors for both $2+i$ and $2-i$ would have been $(0,1)$, the sum of which would have been equivalent to $(0,0) \pmod 2$.

Theorem 2 states that if a product is a square in $\mathbb{Z}[\alpha]$, the sum of the extended exponent vectors of the norms of its factors, with coordinates for each pair (p,r) with $r \in R(p)$, instead of just for each prime p, will be congruent to the zero vector modulo 2. This suggests that requiring this more stringent restriction on the exponent vectors will help to ensure that products found in the linear algebra step with square norms in $\mathbb{Z}$ are more likely to correspond to squares in $\mathbb{Z}[\alpha]$.

Recall the meaning of $e_{p,r}(a+b\alpha)$ from Definition 11. Observe that $e_{p,r}(a+b\alpha)$ describes the value of the $(p,r)^{th}$ coordinate in the extended exponent vector. We are now ready to state Theorem 2.

Theorem 2. [11] Let S be a finite set of coprime integer pairs (a,b) such that $\prod_{(a,b)\in S}(a+b\alpha)=\gamma^2$, for some $\gamma \in \mathbb{Q}(\alpha)$. Then for each prime p and $r \in R(p)$, we have

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

Proof: Assume $\prod_{(a,b)\in S}(a+b\alpha)=\gamma^2$ and let $\mathfrak{p}$ be the degree one prime ideal corresponding to the pair (p,r), which exists by Lemma 4. Recall the group homomorphism $l_{\mathfrak{p}} : (\mathbb{Q}(\alpha))^* \to \mathbb{Z}$ from Proposition 1. Then we can write

$$\sum_{(a,b)\in S} e_{p,r}(a+b\alpha) = \sum_{(a,b)\in S} l_{\mathfrak{p}}(a+b\alpha) = l_{\mathfrak{p}}\Big(\prod_{(a,b)\in S}(a+b\alpha)\Big) = l_{\mathfrak{p}}(\gamma^2) = 2l_{\mathfrak{p}}(\gamma) \equiv 0 \pmod 2,$$

where the first equality follows from Corollary 1 and the second and fourth equalities follow from the fact that $l_{\mathfrak{p}}$ is a group homomorphism over a multiplicative group. □

In order for our sieving-linear algebra approach to guarantee success, we would need the converse of Theorem 2 to hold as well. Unfortunately, there are cases where it is possible to satisfy such a condition on the extended exponent vector of the norm without producing a square in $\mathbb{Z}[\alpha]$. However, these end up being the minority of cases [11, Theorem 6.7], and we will see that we are able to overcome this lack of a full converse with some additional machinery.

We briefly reframe the problem at hand to make clear the cases where the converse of Theorem 2 fails. We are trying to find a set S of coprime integer pairs (a,b) such that $\prod_{(a,b)\in S} a+b\alpha=\gamma^2$, for $\gamma \in \mathbb{Z}[\alpha]$. We approach this task by replacing each element $a+b\alpha$ with its norm and using a sieve, looking for Y-smoothness of the norm over a factor base consisting of pairs (p,r), with p a prime number and $r \in R(p)$. If we find enough candidates $a+b\alpha$ with smooth norms, we can use linear algebra to find a subset S of pairs (a,b) such that $\sum_{(a,b)\in S} e_{p,r}(a+b\alpha) \equiv 0 \pmod 2$ for all pairs (p,r). Recalling that each pair (p,r) corresponds to a unique degree one prime ideal $\mathfrak{p}$ (Lemma 4) gives us a better sense of what this sieve is actually doing. If $\mathfrak{p}$ is the ideal specified by (p,r), then $e_{p,r}(a+b\alpha)$ is the number of factors of $\mathfrak{p}$ in the ideal $(a+b\alpha)\mathcal{O}$, where $\mathcal{O}$ is the ring of algebraic integers in $\mathbb{Q}(\alpha)$. By Lemma 2, nonzero ideals in $\mathcal{O}$ factor uniquely into prime ideals over $\mathcal{O}$; in our sieve, we are looking for pairs (a,b) such that the ideal $(a+b\alpha)\mathcal{O}$ factors completely over the degree one prime ideals of $\mathbb{Z}[\alpha]$ with norms below a threshold Y.

Let $\prod_{(a,b)\in S}(a+b\alpha)=\beta$, and suppose $\sum_{(a,b)\in S} e_{p,r}(a+b\alpha) \equiv 0 \pmod 2$ for all (p,r). There are four reasons why we might not have $\beta=\gamma^2$ for any $\gamma \in \mathbb{Z}[\alpha]$. If $\mathbb{Z}[\alpha]=\mathcal{O}$, it is clear by our construction of the exponent vectors that the linear algebra step will produce β such that $\beta\mathcal{O}=J^2$, for $J \subset \mathcal{O}$ an ideal. The earliest potential obstruction thus occurs in the case where $\mathbb{Z}[\alpha] \subsetneq \mathcal{O}$. Obstructions 2-4 reveal that even if $\beta\mathcal{O}=J^2$ for an ideal $J \subset \mathcal{O}$, we cannot immediately conclude that $\beta = \gamma^2$ for $\gamma \in \mathbb{Z}[\alpha]$ an element. The obstructions are listed below.

Obstructions:

1. If $\mathbb{Z}[\alpha] \subsetneq \mathcal{O}$, since our factor base only contains prime ideals in $\mathbb{Z}[\alpha]$, we may not have $\beta\mathcal{O} = J^2$ for $J \subset \mathcal{O}$ an ideal.
2. Even if $\beta\mathcal{O} = J^2$ for $J \subset \mathcal{O}$ an ideal, J may not be a principal ideal.
3. Even if $\beta\mathcal{O} = \gamma^2\mathcal{O}$ for $\gamma \in \mathcal{O}$ an element, that does not mean $\beta = \gamma^2$.
4. Even if $\beta = \gamma^2$ for $\gamma \in \mathcal{O}$, we might not have $\gamma \in \mathbb{Z}[\alpha]$.

We begin by addressing the last possible obstacle to our goal, obstruction 4. Suppose we have $\beta = \gamma^2$ for $\gamma \in \mathcal{O} \backslash \mathbb{Z}[\alpha]$. We can use Lemma 5 from Section 3.2 to handle this obstruction. If we have $\beta = \gamma^2$ for $\gamma \in \mathcal{O} \backslash \mathbb{Z}[\alpha]$, Lemma 5 tells us that $f'(\alpha)\gamma \in \mathbb{Z}[\alpha]$. Then

$$f'(\alpha)^2\gamma^2 = f'(\alpha)^2\beta = f'(\alpha)^2 \prod_{(a,b)\in S} (a + b\alpha)$$

is the square of an element in $\mathbb{Z}[\alpha]$. We can let $u = \phi(f'(\alpha)\gamma)$. If $\prod_{(a,b)\in S}(a + bm) = v^2$, we can define $w = vf'(m) \pmod n$, which gives us $u^2 \equiv w^2 \pmod n$. We can assume $\gcd(f'(m), n) = 1$, as otherwise, we would have a factorization of n. Then, multiplying v by $f'(m)$ to get w will not affect the likelihood of factoring n by computing $\gcd(u - w, n)$. So Lemma 5 gives an extremely straightforward solution to the fourth possible obstruction.

Obstructions 1-3 arise because, due to the nature of the exponent vectors we use in our sieve, we are currently trying to procure the squareness of $\beta\mathcal{O}$ as an ideal, where $\beta = \prod_{(a,b)\in S}(a + b\alpha)$, instead of dealing directly with the squareness of β. Adding a few components to these exponent vectors, however, can help us to specify properties of β itself, and not just its associated ideal.

The strategy for overcoming the first three obstructions is due to Adleman [1] who adapts a square-recognition method more familiar in the integers. In the integers, if a number is a square, it must also be a square in $\mathbb{Z}/q\mathbb{Z}$, for any odd prime q. We can get information about an integer m by asking if m is a quadratic residue modulo q, i.e. if $\left(\frac{m}{q}\right) = 1$. If m is a quadratic residue modulo $q_1, \ldots, q_k$, where each $q_i < |m|$ and the q_i's are distinct odd primes, we can pretty confident that m is indeed a square, as the probability of this happening when m is not square is about 2^{-k},[3] which is small for k large. When factoring via the Number Field Sieve, we wish to use a similar trick to determine if the algebraic integer β is square. Lemma 6 shows how this is possible.

Lemma 6. [16] Let $f(x)$ be a monic, irreducible polynomial in $\mathbb{Z}[x]$ and let α be one of its complex roots. Suppose q is an odd prime number and s is an integer such that $f(s) \equiv 0 \pmod q$ but $f'(s) \not\equiv 0 \pmod q$. Let S be a set of coprime integer pairs (a, b) such that $q \nmid a + bs$ for all $(a, b) \in S$ and $f'(\alpha)^2 \prod_{(a,b)\in S}(a + b\alpha) = \gamma^2$, with $\gamma \in \mathbb{Z}[\alpha]$. Then

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

[3] This probability comes from the fact that about half of quadratic residues are 1, and by the Chinese Remainder Theorem behavior modulo different primes q_i is independent.

Proof: Consider the homomorphism $\phi_q : \mathbb{Z}[\alpha] \to \mathbb{Z}/q\mathbb{Z}$, where $\phi_q(\alpha)$ is the residue class $s \pmod q$. We can write

$$\phi_q(\gamma^2) \equiv f'(s)^2 \prod_{(a,b)\in S} (a + bs) \not\equiv 0 \text{ (mod } q),$$

as $f'(s) \not\equiv 0 \pmod q$ and $a + bs \not\equiv 0 \pmod q$ by the hypothesis. Then $\left(\frac{\phi_q(\gamma^2)}{q}\right) = \pm 1$. Since ϕ_q is a homomorphism, we can write

$$\left(\frac{\phi_q(\gamma^2)}{q}\right) = \left(\frac{\phi_q(\gamma)^2}{q}\right) = 1,$$

since a square is clearly a quadratic residue. As $\left(\frac{f'(s)^2}{q}\right) = 1$ as well, using the fact that the Legendre symbol is multiplicative, we can deduce that

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

□

Once again, this time via Lemma 6, we have given a necessary condition to find a set S from which we can get a product that is a square in $\mathbb{Z}[\alpha]$, when what we are really looking for is a sufficient condition. However, there is strong evidence to suggest that by combining the condition of Lemma 6 with the condition from Theorem 2, it will be overwhelmingly likely that $f'(\alpha)^2 \prod_{(a,b)\in S} (a + b\alpha) = \gamma^2$. We introduce some notation to make this conjecture clear.

Definition 12. [11] Define

$$V = \{\beta \in \mathbb{Q}(\alpha)^* : l_{\mathfrak{p}}(\beta) \equiv 0 \text{ (mod 2) for all prime ideals } \mathfrak{p} \subset \mathbb{Z}[\alpha].\}$$

V is a multiplicative group.

Definition 13. [11] Define $K = \{x^2 : x \in \mathbb{Q}(\alpha)^*\}$. Since $l_{\mathfrak{p}}$ is a homomorphism, we can see that $K \subset V$. Furthermore, the quotient V/K is seen to be a vector space over $\mathbb{F}_2$.

Definition 14. [11] Let $\mathfrak{q} \subset \mathbb{Z}[\alpha]$ be a degree one prime with $f'(\alpha) \notin \mathfrak{q}$. Define $\chi_{\mathfrak{q}} : \mathbb{Z}[\alpha] \to \{\pm 1\}$ to be the composition of ϕ_q from the proof of Lemma 6 with the Legendre symbol over q. (Here q is the unique prime corresponding to $\mathfrak{q}$.) We can see that χ_q induces a homomorphism from V/K to $\{\pm 1\}$, which with a slight abuse of notation we'll again denote by χ_q.

Elements in V can be found by doing sieving/linear algebra with the conditions mentioned in Theorem 2. We would like to find elements in $K \subset V$, however, as these are the squares. Buhler, Lenstra, and Pomerance [11, Theorem 6.7] showed that $\dim_{\mathbb{F}_2}(V/K)$ is small — this is what allows us to claim that the converse to Theorem 2 holds in most cases. This fact, combined with a choice of prime ideals $\mathfrak{q}_1, \ldots, \mathfrak{q}_k$ corresponding to odd primes $q_1, \ldots, q_k$, with k sufficiently large, makes it highly likely that the functions $\chi_{\mathfrak{q}_1}, \ldots, \chi_{\mathfrak{q}_k}$ span $\text{Hom}(V/K, \{\pm 1\})$. If they do span, then the condition that $\chi_{\mathfrak{q}_i}(\beta) = 1$ for all $\mathfrak{q}_i$ is both necessary and sufficient for guaranteeing that β is a square. This suggests that, with k sufficiently large, $q_1, \ldots, q_k$ odd primes such that $q_i \nmid N(a + b\alpha) \forall (a, b) \in S$, and $s_j \in R(q_j)$ for $j = 1, \ldots k$ such that $f'(s_j) \not\equiv 0 \pmod q_i$, then satisfying both

1. $\sum_{(a,b)\in S} e_{p,r}(a+b\alpha) \equiv 0 \pmod 2 \forall (p,r)$
2. $\prod_{(a,b)\in S} \left(\frac{a+bs_j}{q_j}\right) = 1$ for $j = 1, \ldots, k$

almost certainly means that

$$\prod_{(a,b)\in S} (a+b\alpha) = \gamma^2$$

for some $\gamma \in \mathcal{O}$, giving us sufficiency in practice.

How large does k need to be? By placing some conditions on the polynomial $f(x)$ defined in Section 3.3.1 — namely, requiring its degree d to satisfy $d^{2d^2} < n$ and its coefficients c_j to satisfy $|c_j| < n^{1/d}$ — we can choose $k = \lfloor 3\log(n) \rfloor$ [16]. Note that by this estimate, $k \ll \pi(Y)$ and $k \ll Y'$, so this is not really a significant expansion of the size of our exponent vectors. Finally, we choose the primes q_j to be as small as possible.

Despite having several additional intricacies, the idea for building S in the Number Field Sieve is similar to the process used in the Quadratic Sieve to find the subset of auxiliary numbers which will multiply to a square. In the Quadratic Sieve, we collected smooth auxiliary numbers, recorded their factorizations in exponent vectors, and then used linear algebra over these exponent vectors to find a linearly dependent subset, indicating that the corresponding auxiliary numbers would multiply to a square. The same idea is used in the Number Field Sieve, but with a much more complicated exponent vector to adjust for all the potential obstacles described in this section. Letting Y be our smoothness bound, the first $\pi(Y)+1$ coordinates in the exponent vector represent $\text{ord}_p(a+bm) \pmod 2$ for the primes $p_0, p_1, \ldots, p_{\pi(Y)}$, where we count -1 as the 0^{th} prime. Letting $Y' = \#\{(p,r) : p \text{ prime}, p < Y, r \in R(p)\}$, the next Y' coordinates in the exponent vector correspond to $e_{p,r}(a+b\alpha) \pmod 2$ for every pair (p,r) contributing to the count of Y'. Finally, the last k coordinates are determined by the value of $\left(\frac{a+bs_i}{q_i}\right)$ for the pairs (q_i, s_i) picked according to the conditions in Lemma 6. If $\left(\frac{a+bs_i}{q_i}\right) = 1$, we enter 0 into the coordinate, and if $\left(\frac{a+bs_i}{q_i}\right) = -1$, we enter the value 1, to work with the group of two elements in the additive, rather than multiplicative, context. Though this exponent vector has considerably more pieces than the exponent vector used in the Quadratic Sieve, each piece is necessary to ensure that S satisfies both the criterion laid out for it in Algorithm 2.

3.3.3 Taking Square Roots in $\mathbb{Z}[\alpha]$

After successfully building S, we have a set of coprime integer pairs (a,b) such that

$$f'(\alpha)^2 \prod_{(a,b)\in S} (a+b\alpha) = \gamma^2$$

for $\gamma \in \mathbb{Z}[\alpha]$ and

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

for $v \in \mathbb{Z}$. By letting $\phi(\gamma) = u \pmod n$, we can see that

$$u^2 \equiv \phi(\gamma^2) \equiv \phi\Big(f'(\alpha)^2 \prod_{(a,b)\in S} (a+b\alpha)\Big) \equiv \phi(f'(\alpha)^2) \prod_{(a,b)\in S} \phi(a+b\alpha) \equiv f'(m)^2 \prod_{(a,b)\in S} (a+bm) \equiv v^2 \pmod n.$$

Then we can factor n by computing $\gcd(u - v, n)$. However, this operation is contingent on being able to determine u from γ^2. Finding $v = \sqrt{v^2}$ in the integers is relatively straightforward, as we have the prime factorization of $v^2/(f'(m)^2)$ stored in the exponent vectors of the sieve step, and can use this to quickly compute $v \pmod{n}$. Unfortunately, there is not a similarly elegant solution for finding $\gamma = \sqrt{\gamma^2}$, the natural intermediary in the process to compute u.

There are several different algorithms in the literature relating to obtaining $\sqrt{\gamma^2}$, following both direct and heuristic approaches, and the choice of which to use frequently comes down to desired ease of implementation. Here we briefly sketch Nguyen's adaptation of Peter Montgomery's algorithm. For more details, refer to [26].

Algorithm 3. (Square Roots in $\mathbb{Z}[\alpha]$) [26] Let $\mathcal{O}$ be the ring of algebraic integers of $\mathbb{Q}(\alpha)$, and let $\mathcal{I}$ be the abelian group of fractional ideals in $\mathcal{O}$. For $x_1, \ldots, x_j \in \mathbb{Q}(\alpha)$, we will let $\langle x_1, \ldots, x_j \rangle$ denote the element of $\mathcal{I}$ generated by $x_1, \ldots, x_j$. Let $\gamma \in \mathbb{Z}[\alpha]$ be the square that we want to find the root of.

1. Transform γ to make the fractional ideal $\langle \gamma \rangle$ simpler. By simpler, we refer to a reduction in the product of the norms of the numerator and denominator of $\langle \gamma \rangle$.

   Recall that γ is produced by our sieving step, and so $\gamma = \prod_{(a,b)\in S}(a + b\alpha)$. Because γ is a square, any $\gamma' = \prod_{(a,b)\in S}(a + b\alpha)^{e_{(a,b)}}$, for $e_{(a,b)} \in \{\pm 1\}$, will also be a square in $\mathbb{Q}(\alpha)$. The identity
   $$\Big[\phi\Big(\sqrt{\prod_{(a,b)\in S}(a - b\alpha)^{e_{(a,b)}}}\Big)\Big]^2 \equiv \Big[\sqrt{\prod_{(a,b)\in S}(a - bm)^{e_{(a,b)}}}\Big]^2 \pmod{n}$$
   allows us to replace γ with γ'. By carefully selecting the signs of $e_{(a,b)}$, we can ensure that $\langle \gamma' \rangle$ is simpler than $\langle \gamma \rangle$.

   We thus let $\gamma = \gamma'$.

2. Compute $\sqrt{\langle \gamma \rangle}$ using the prime ideal factorization of $\langle \gamma \rangle$, which can be derived from the prime factorization of $F(a, -b)$ for each $(a, b) \in S$. The prime factorizations of the $F(a, -b)$'s are stored in the exponent vectors of the sieving step.

3. From $\sqrt{\langle \gamma \rangle}$, approximate $\sqrt{\gamma}$ using lattice reductions. We can iteratively compute a sequence of algebraic integers $\delta_1, \ldots, \delta_L$ in $\mathcal{O}$, with accompanying signs $s_1, \ldots, s_L$ in $\{\pm 1\}$, such that $\theta = \gamma \prod_{i=1}^{L} \delta_i^{-2s_i}$ is an algebraic integer of decreasing size. Since the right hand side is a square, we can see that θ will also be a square. We iterate until θ is sufficiently "small."

4. Once θ is sufficiently small, it is feasible to compute $\sqrt{\theta}$ by a brute force method, at which point we can write
   $$\sqrt{\gamma} = \sqrt{\theta} \prod_{i=1}^{L} \delta_i^{s_i}.$$
   One method for finding $\sqrt{\theta}$ involves solving for $\sqrt{\theta}$ modulo several primes p, and then using the Chinese Remainder Theorem to combine these solutions.

Nguyen [26] provides a brief discussion of the algorithm's complexity in terms of $|S|$, the size of the set S generated in Section 3.3.2. Steps 1 and 2, the simplification of γ and the computation of $\sqrt{< \gamma >}$, can both be proven to take at most $O(|S|)$ time. For Step 3, the approximation of $\sqrt{\gamma}$ using an iterative lattice reduction procedure, he is able to show that the number of iterations is $O(|S|)$; however, there is no provable bound on the time required for each iteration. In practice, we know that overall time required for this step is about $O(|S|)$, and this is supported by the argument that since our lattice reductions are occurring on matrices with small dimensions, each iteration should take very little time. As with Step 3, we are similarly unable to bound the cost of Step 4, computing $\sqrt{\theta}$, this time because we are unable to tell how many different primes we will need to use in the Chinese Remainder Theorem component.

Though we lack a provable complexity, we know experimentally that the algorithm's run time is approximately linear in the size of S. It is worth noting that this difficulty in establishing a complexity bound is not unique to Nguyen's approach, but a challenge for many of the heuristic algorithms used to compute square roots in $\mathbb{Z}[\alpha]$ [26]. In practice, however, these approaches have been found to be among the most efficient and are widely used. Further, it has been well established [26][16][35] that the time to compute $\sqrt{\gamma^2}$ is minor compared to the sieving step in the Number Field Sieve algorithm.

3.3.4 An Asymptotic Analysis of the Sieve Step and Optimal Values for d and Y

The Number Field Sieve borrows much of its architecture from the Quadratic Sieve, and as such, the approach to its asymptotic analysis remains fairly similar. Following [16], we thus begin by briefly recapping the notation and complexity arguments used for the Quadratic Sieve.

To analyze the Quadratic Sieve, we worked under the assumption that the auxiliary numbers generated by our quadratic $Q(x) = x^2 - n$ formed a random sequence of integers bounded by X. Letting Y be equivalent to the largest prime we sieved for, we denoted by $\psi(X, Y)$ the number of Y-smooth integers in the interval $[1, X]$. Then the probability of a random integer (auxiliary number) being smooth was $\psi(X, Y)/X$, and the expected number of auxiliary numbers needed to find one smooth number was the reciprocal, $X/\psi(X, Y)$. We needed roughly $\pi(Y)$ smooth integers to proceed with the linear algebra step, so this quantity was multiplied by $\pi(Y)$. Finally, to check if a given auxiliary number was Y-smooth via sieving required $\log(\log(Y))$ work. This brought the total work of the sieving step to

$$\frac{\pi(Y) \log\log(Y) X}{\psi(X, Y)}.$$

The value of X is determined by our factoring method, but we were able to choose Y as a function of X to minimize the amount of work required in the sieving step. We calculated an optimal value for Y to be

$$Y = \exp((2^{-1/2} + o(1))(\log(X) \log\log(X))^{1/2}).$$

Using this result, we obtained an estimate for the amount of work needed in the sieve step solely in terms of X, getting

$$\frac{\pi(Y) \log\log(Y) X}{\psi(X, Y)} \approx \exp((2^{1/2} + o(1))(\log(X) \log\log(X))^{1/2}).$$

In L-notation, we can express this estimate for the work of the Quadratic Sieve as $L_X[1/2, 2^{1/2}]$.

Much of this argument for the Quadratic Sieve carries over to the Number Field Sieve. We retain the assumption that the numbers we are sieving over can be considered a random sequence. Additionally, we are still looking for Y-smooth numbers, and we still expect it to take $X/\psi(X,Y)$ attempts before we generate one. The advantage of the Number Field Sieve, as we are about to see, is that X is smaller, making the proportion of Y-smooth numbers in $[1, X]$ higher. Further, we know that auxiliary numbers are products of two smaller numbers, further increasing the likelihood that they are smooth, though following [16], we will mostly ignore this fact as it has a very limited impact on the asymptotic analysis. So in conclusion, as compared to the case of the Quadratic Sieve, the estimate for $X/\psi(X,Y)$ with the Number Field Sieve is smaller, reducing the amount of work in the sieve step.

Returning to the parallels between the Quadratic Sieve and Number Field Sieve, the time it takes to sieve a number and determine if it is Y-smooth is still $\log\log(Y)$. However, we are working with exponent vectors that are about twice as large in the Number Field Sieve, and thus will need many more smooth numbers before we can proceed to the linear algebra step. Section 3.3.2 indicates that instead of $\pi(Y)$ smooth numbers, we will need roughly $\pi(Y)+Y'+k$ smooth numbers. But we know that $Y' \approx \pi(Y)$ and that $k \ll \pi(Y)$, so the number of Y-smooth numbers needed in the Number Field Sieve is still of order of magnitude $\pi(Y)$, and continuing to use $\pi(Y)$ in our heuristic estimate does not significantly impact its accuracy. (Critically, our heuristic analysis in the Quadratic Sieve case is based on the fact that $\pi(Y) \sim Y/\log(Y)$ by the Prime Number Theorem, and we can still say $2\pi(Y) \sim Y/\log(Y)$ with the Number Field Sieve [16].) Putting this all together, we can retain $\pi(Y)\log\log(Y)X/\psi(X,Y) = L_X[1/2, 2^{1/2}]$ as an estimate for the number of steps in the Number Field Sieve algorithm and $Y = \exp((2^{-1/2} + o(1))(\log(X)\log\log(X))^{1/2})$ as an estimate for Y.

But what is X? In the Quadratic Sieve, we used the function $Q(x) = x^2 - n$ to generate auxiliary numbers, which gave us a bound of $X = 2n^{1/2+\epsilon}$. In the Number Field Sieve, however, the numbers that we want to evaluate for smoothness are generated by the polynomial $H(x,y) = F(x,-y)\ G(x,y)$, where $F(x,y) = x^d + c_{d-1}x^{d-1}y + \ldots + c_0y^d$, $G(x,y) = x + ym$, and the integer m and the coefficients c_j are as defined in Section 3.3.1. We know that $c_j < m < n^{1/d}$ by construction. Say $|x|, |y|$ are bounded by some M. Then $H(x,y)$ is a polynomial with $2(d+1)$ terms, homogeneous with degree $d+1$, with coefficients bounded by $n^{2/d}$, giving us the bound

$$|H(x,y)| < 2(d+1)n^{2/d}M^{d+1} = X.$$

M^2 represents the number of auxiliary numbers $H(x,y)$ we generate; certainly, we do not want $M^2 > L_X[1/2, 2^{1/2}]$, the total number of steps we expect the sieving to take based on our computation for the Quadratic Sieve, and so we can use this as a bound on M.

Our expression for X and our estimate for the number of steps required in the sieving step indicate that our choice for d affects the size of X, and the size of X determines the number of steps needed in the Number Field Sieve. Thus we want to choose d to minimize X. To deal with fewer exponents, however, we will work with $\log(X)$ rather than X. Substituting in our bound on M, we get that

$$\log(X) \sim \log(2(d+1)) + \frac{2}{d}\log(n) + (d+1)\sqrt{\frac{1}{2}\log(X)\log\log(X)}.$$

The term $\log(2(d+1))$ makes a negligible contribution to the sum, so in future analysis we will drop it from the equation. Further, we take $n \to \infty$, and assuming that $d \to \infty$ as well, we change $d+1$ to d to simplify the arithmetic. We will briefly assume that our expression for $\log(X)$ is an equation and take the derivative with respect to d, letting X' represent the derivative of X. This gives

$$\frac{X'}{X} = \frac{-2}{d^2}\log(n) + \sqrt{\frac{1}{2}\log(X)\log\log(X)} + \frac{dX'(1+\log\log(X))}{4X\sqrt{\frac{1}{2}\log(X)\log\log(X)}}.$$

Setting $X' = 0$, the last term disappears and we can solve for d, getting

$$d = (2\log(n))^{1/2}\Big(\frac{1}{2}\log(X)\log\log(X)\Big)^{-1/4}.$$

Plugging this value back in, we have

$$\log(X) = 2(2\log(n))^{1/2}(1/2\log(X)\log\log(X))^{1/4}.$$

We can divide by $\log(X)^{1/4}$ to get that $\log(X)^{3/4} = 2(2\log(n))^{1/2}(1/2\log\log(X))^{1/4}$, from which we can obtain the estimate $\frac{3}{4}\log\log(X) \sim \frac{1}{2}\log\log(n)$. With this approximation, we can get an estimate for X fully in terms of n, giving us $\log(X)^{3/4} \sim 2(2\log(n))^{1/2}(1/3\log\log(n))^{1/4}$, or, further simplified,

$$\log(X) \sim \frac{4}{3^{1/3}}(\log(n))^{2/3}(\log\log(n))^{1/3}.$$

Using these estimates for $\log(X)$ and $\log\log(X)$, we obtain values for d, Y, and $L_X[1/2, 2^{1/2}]$, the complexity of the sieving step of the Number Field Sieve. We get

$$d \sim \Big(\frac{3\log(n)}{\log\log(n)}\Big)^{1/3},$$

$$Y = \exp(((8/9)^{1/3} + o(1))(\log(n)^{1/3}\log\log(n)^{2/3})),$$

and

$$L_X[1/2, 2^{1/2}] = \exp\Big(\big((64/9)^{1/3} + o(1)\big)(\log(n))^{1/3}(\log\log(n))^{2/3}\Big) = L_n[1/3, (64/9)^{1/3}].$$

The overall complexity of the sieve step compares favorably to the complexity of the Quadratic Sieve step, which is $L_n[1/2, 1]$, as we have reduced the exponent on the most consequential term, $\log(n)$, by a factor of $2/3$.

3.3.5 Full Algorithm Description

We are now ready to expand on the description of the Number Field Sieve given in Algorithm 2.

Algorithm 4. (Full Number Field Sieve) [16] Let n be an odd, composite integer that is not a prime power.

1. Setup:

   (a) Following the results of Section 3.3.4, set $d = \lfloor (3\log(n)/\log\log(n))^{1/3} \rfloor$ and $Y = \lfloor \exp((8/9)^{1/3}(\log(n))^{1/3}(\log\log(n))^{2/3}) \rfloor$. Observe that $d^{2d^2} < n$, as indicated by Section 3.3.2.

(b) Compute $f(x)$ and m as in Section 3.3.1. Check that $f(x)$ is irreducible; if $f(x) = a(x)b(x)$, return $n = f(m) = a(m)b(m)$ as a factorization and terminate the algorithm.

(c) Let α be a complex root of $f(x)$. Define $\phi : \mathbb{Z}[\alpha] \to \mathbb{Z}/n\mathbb{Z}$, where

$$\phi(a_0 + a_1\alpha + \cdots + a_{d-1}\alpha^{d-1}) = a_0 + a_1 m + \cdots + a_{d-1}m^{d-1} \pmod{n}.$$

(d) Set $F(x, y) = y^d f(x/y)$ and $G(x, y) = x + my$.

(e) For prime numbers $p < Y$, compute the sets

$$R(p) = \{r \in [0, \ldots, p-1] \mid f(r) \equiv 0 \pmod{p}\}.$$

Let $Y' = \sum_{\text{primes } p<Y} |R(p)|$.

(f) As per Section 3.3.2, set $k = \lfloor 3\log(n) \rfloor$. Compute the first k primes $q_1, \ldots, q_k$ greater than Y such that $\exists s_j \in R(q_j)$ with $f'(s_j) \not\equiv 0 \pmod{q_j}$, and store the pairs (q_j, s_j).

2. Sieve: Use a sieve to find a set S' of coprime integer pairs (a, b) such that the product $H(a, b) = F(a, -b)G(a, b)$ is Y-smooth. Keep $|a|$ and $|b|$ as low as possible to increase the chance that $H(a, b)$ is Y-smooth. Collect elements for S' until $|S'| > \pi(Y) + Y' + k + 2$.

3. Linear Algebra: Construct the exponent vectors for elements $(a, b) \in S'$ as discussed in Section 3.3.2. Use a technique from linear algebra, such as Wiedemann coordinate recurrence, to identify a linearly dependent subset of these exponent vectors. Define the subset S of S' to contain the pairs (a, b) which correspond to the exponent vectors in the linearly dependent subset.

4. Square Roots: We now have a set S such that

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

and

$$f'(\alpha)^2 \prod_{(a,b)\in S} (a + b\alpha) = \gamma^2$$

for $v \in \mathbb{Z}$, $\gamma \in \mathbb{Z}[\alpha]$. Use the prime factorization in the exponent vectors from Step 3 to find $v \equiv \sqrt{v^2} \pmod{n}$. Use Algorithm 3 in Section 3.3.3 to compute $\gamma = \sqrt{\gamma^2}$.

5. GCD: Let $u \equiv \phi(\gamma)$. Check that $u \not\equiv \pm v \pmod{n}$, and if this is the case, return $\gcd(u - v, n)$ a nontrivial factor of n. If $u \equiv \pm v \pmod{n}$, return to the Step 2 and expand the set S', then move to Step 3 and find a new set S with linearly dependent exponent vectors.

We now examine each step in our algorithm to arrive at an overall complexity estimate. The setup step mostly involves a series of one-time computations of constants, which will take negligible time compared to the sieving and linear algebra steps. The two most tasking parts of the setup phase are determining if $f(x)$ is irreducible and computing the sets $R(p)$. Checking $f(x)$ for irreducibility can be done in polynomial time via the algorithm of Lenstra, Lenstra, and Lovász [23]. And there are several algorithms for finding roots of a polynomial over finite fields which run in time polynomial in d, the degree of the polynomial — see [33] for details. We then see that the setup phase of the algorithm involves a relatively small amount of work compared to the steps that follow.

The sieving phase, as we saw in Section 3.3.4, will take $L_n[1/3, (64/9)^{1/3}]$ steps. Following that, the linear algebra step, while not insignificant, can be done in time $(Y + Y' + k)^{2+o(1)}$ using Wiedemann coordinate recurrence, and as we saw with the Quadratic Sieve, this does not surpass the time spent sieving. Though it is difficult to give an exact estimate for the complexity of the square root step, our discussion in Section 3.3.3 gives us the heuristic $O(|S|)$, and we know experimentally that the time spent in this step is trivial compared to the time spent in Steps 2 and 3. Finally, the computation of the GCD in Step 5 via the Euclidean algorithm will also take negligible time. Thus, as with the Quadratic Sieve, the most expensive part of the Number Field Sieve algorithm is performing the sieve step, and so we give the algorithm as a whole a complexity of $L_n[1/3, (64/9)^{1/3}]$.

3.4 Improvements to the Number Field Sieve

As with the Quadratic Sieve, there are many enhancements that can be made to the traditional Number Field Sieve algorithm to improve its practical performance without affecting the asymptotic complexity estimate. Some of the suggested changes will be familiar from our discussion of the Quadratic Sieve. For example, the Large Prime Variation of the Quadratic Sieve, where we store auxiliary numbers that are nearly smooth with the exception of one large prime factor, and use pairs of nearly smooth numbers with identical large prime factors to create new, completely smooth, auxiliary numbers, can be adapted without much difficulty to the Number Field Sieve context.

There has also been research into a multiple polynomial version of the Number Field Sieve. Letting d be the degree, and $m = \lfloor n^{1/d} \rfloor$, we can write $n = m^d + c_{d-1}m^{d-1} + \ldots + c_1 m + c_0$, and let $f(x) = x^d + c_{d-1}x^{d-1} + \ldots + c_1 x + c_0$. For small integers i and j, the family of polynomials

$$f_{i,j}(x) = f(x) + jx^2 - (mj - i)x - mi$$

has the desired property that $f_{i,j}(m) \equiv 0 \pmod{n}$ for all i, j. So finding additional polynomials is not too hard. The challenge is in dealing with the fact that since the pairs (p, r) will differ depending on the polynomial, introducing additional polynomials will require expanding the sieve's factor base and thus extending our exponent vector. In turn, longer exponent vectors make the linear algebra step more difficult. However, some progress has been made in overcoming these difficulties [15] and experimental results suggest that utilizing multiple polynomials may be advantageous in practice [18].

There have been additional proposals to speed up the runtime of the Number Field Sieve algorithm based on some of the aspects it does not share with its predecessor, the Quadratic Sieve. The remainder of this section previews the most promising of these suggestions, which relate to the selection of polynomials used in the algorithm. As it turns out, the impact of polynomial selection can be so large that it is worth expending additional time and resources at the beginning of the algorithm to optimize choices before proceeding to the sieve stage [16].

3.4.1 Nonmonic Choices for f

As we saw in Section 3.3.4, the key factor explaining the advantage of the Number Field Sieve relative to the Quadratic Sieve is the size of the auxiliary numbers it evaluates for smoothness, and the size of these auxiliary numbers is determined by the values of m and $c_1, \ldots, c_{d-1}$, the coefficients of $f(x)$. In Section 3.3.1, we saw how to find a natural, monic, polynomial $f(x)$ for use in the algorithm. But working with nonmonic polynomials requires only extending our exponent vector by one coordinate [16]. By redefining $m = \lceil n^{1/(d+1)} \rceil$, writing $n = c^d m^d + \ldots + c_1 m + c_0$, and letting

$$f(x) = c_d x^d + \ldots + c_1 x + c_0,$$

we reach a nonmonic polynomial with coefficients c_i bounded by $n^{1/(d+1)}$, instead of $n^{1/d}$. This is a decrease by a factor of about $n^{1/(d^2+d)}$, which translates to an asymptotic improvement of about $\log(n)^{1/6}$ [16]. As $n \to \infty$, this factor is insignificant in the overall asymptotic complexity, but at the scale of the numbers we currently have the ability to factor, the practical implications are meaningful.

3.4.2 Polynomial Pairs

In the Number Field Sieve, we sieve over auxiliary numbers of the form $H(x, y) = F(x, -y)G(x, y)$, where $F(x) = y^d f(x/y)$ and $G(x) = yg(x/y)$, with $g(x) = x - m$. However, we do not need to force the degree of $g(x)$ to equal one; we could use any other irreducible polynomial with $g(m) \equiv 0 \pmod{n}$ and sieve over $\mathbb{Z}[\alpha']$, where α' is a root of $g(x)$, in an identical manner to how we sieved over the part of $H(x, y)$ corresponding to $F(x, -y)$. The benefit of this approach is that a number near x written as the product of two numbers near $x^{1/2}$ is much more likely to be smooth than a random number near x. The closer $F(x, y)$ and $G(x, y)$ are in degree, the more our auxiliary numbers will resemble the product of two numbers near $x^{1/2}$. Letting $F(x, y)$ and $G(x, y)$ have equal degree then increases the probability of smoothness by a factor of about $(1.46)^{(\log(n)/(\log\log(n)))^{1/3}}$, which is theoretically quite significant [16]. Unfortunately, in practice it can be difficult to find good polynomial pairs $f(x)$ and $g(x)$, as although counting arguments suggest that good pairs should exist, we don't have a method for finding them more efficient than exhaustive search [16].

3.4.3 The Special Number Field Sieve

In rare but often interesting cases, we can find extraordinarily good polynomials with unusually small coefficients. This is usually due to the number n we wish to factor taking a special form, such as being a Cunningham number.[4] In such instances, we deploy a variation of our algorithm known as the Special Number Field Sieve, which is able to further exploit the remarkably small coefficients of the polynomials we are working with [16]. The polynomials used in the Special Number Field Sieve produce smaller auxiliary numbers, which in turn lower the sieve's complexity estimate to $L_n[1/3, (32/9)^{1/3}]$ [28]. For more on the Special Number Field Sieve, we refer to [14].

[4] Cunningham numbers take the form $b^n \pm 1$, where b and n are integers and b is not a perfect power. Cunningham numbers occur throughout mathematics; for example, for p a prime, the Cunningham number $p^n - 1$ is the size of the unit group of a finite field.

This completes our discussion of the Number Field Sieve. In the final chapter, we will review the experimental performances of the algorithms we have described so far, and briefly mention other notable, non-sieve, factoring techniques.

4 Comparing Factoring Techniques

Having detailed both the Quadratic and Number Field Sieves, we now briefly compare the two. Before the advent of the Number Field Sieve, the Quadratic Sieve algorithm was considered the premier general-purpose factoring algorithm. RSA-129, which has 129 decimal digits, is the largest number first factored with the Quadratic Sieve, this feat having been accomplished by Atkins, Graff, Lenstra, and Leyland in 1994 [3]. All larger numbers factored as part of the RSA Factoring Challenge (RSA-130 and beyond) were subsequently first factored using a Number Field Sieve. The current record for largest general form factored integer is RSA-250, with 250 decimal digits, factored in February of 2020 by Boudot et al. [6], though the Special Number Field Sieve has been used successfully with applicable numbers of as many as 320 digits [13]. Though it had already been factored by the Number Field Sieve, Patrick Konsor succeeded in factoring RSA-140, which has 140 decimal digits, using the Quadratic Sieve in June of 2020, and this currently stands as the record for largest integer factored via the Quadratic Sieve [21].

It has been demonstrated experimentally that the Quadratic Sieve is superior for factoring numbers with fewer than 100 digits, while the Number Field Sieve is preferential for numbers with more than 130 digits. However, the exact crossover point between the algorithms is unclear, as within the indicated ambiguous range, performance of both algorithms is highly sensitive to details of programming/implementation and computer hardware [28].

Finally, this thesis would be remiss if it did not note the existence of two very important non-sieve factoring algorithms which have not been mentioned as of yet. First is the Elliptic Curve Method of Hendrik Lenstra, which has a complexity of $L_p[1/2, 2^{1/2}]$, with p being the smallest prime factor of n, the number we wish to factor. The Elliptic Curve Method excels at finding factors much less than $\sqrt{n}$ but still larger than what would be feasible to uncover by trial division, and also uses considerably less space than the linear algebra step of sieving methods [16]. If n has no small factors, the Elliptic Curve Method performs about as well as the Quadratic Sieve.

Also worthy of mention is Peter Shor's polynomial-time algorithm for factorization on quantum computers [32]. While a polynomial-time algorithm unquestionably surpasses a subexponential one, we are currently limited in our ability to produce quantum computers capable of executing Shor's algorithm. Though impressive factorizations have been reported, some studies place the largest number that can currently be factored on most quantum computers as low as 21 [24].

For large numbers lacking a special form which can be exploited by the Special Number Field Sieve, the standard Number Field Sieve is considered to be the best general-purpose factoring algorithm for classical computers. The Number Field Sieve expands on the approach of the Quadratic Sieve to incorporate number rings, reducing the size of the numbers used in the algorithm and thus further reducing its already sub-exponential run time. Yet while the Number Field Sieve currently reigns as the best factoring technique, the existence of Shor's algorithm suggests that faster factoring, whether on classical or quantum computers, may be in our future.

References

[1] Leonard M. Adleman. Factoring numbers using singular integers. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, STOC '91, page 64–71, New York, NY, USA, 1991. Association for Computing Machinery.

[2] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2):781–793, 2004.

[3] Derek Atkins, Michael Graff, Arjen K. Lenstra, and Paul C. Leyland. The magic words are squeamish ossifrage. In Josef Pieprzyk and Reihanah Safavi-Naini, editors, *Advances in Cryptology — ASIACRYPT'94*, pages 261–277, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[4] Eric Bach and Jeffrey Shallit. *Algorithmic Number Theory: Efficient algorithms.* Foundations of Computing. MIT Press, Cambridge, MA, 1996.

[5] Henk Boender and Herman J. J. te Riele. Factoring integers with large-prime variations of the quadratic sieve. *Experimental Mathematics*, 5(4):257–273, 1996.

[6] Fabrice Boudot, Pierrick Gaudry, Aurore Guillevic, Nadia Heninger, Emmanuel Thomé, and Paul Zimmermann. Comparing the difficulty of factorization and discrete logarithm: a 240-digit experiment. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020*, volume 12171 of *Lecture Notes in Computer Science*, pages 62–91, Santa Barbara CA, United States, August 2020. Springer.

[7] David M. Bressoud. *Factorization and Primality Testing.* Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1989.

[8] John Brillhart, Michael Filaseta, and Andrew Odlyzko. On an irreducibility theorem of a. cohn. *Canadian Journal of Mathematics*, 33(5):1055–1059, 1981.

[9] N.G. de Bruijn. The asymptotic behaviour of a function occuring in the theory of primes. *Journal of the Indian Mathematical Society*, 15:25–32, 1951.

[10] N.G. de Bruijn. On the number of positive integers $\leq x$ and free of prime factors $> y$, ii. *Indagationes Mathematicae (Proceedings)*, 69:239–247, 1966.

[11] J. P Buhler, H. W Lenstra, and Carl Pomerance. Factoring integers with the number field sieve. In *The Development of the Number Field Sieve*, Lecture Notes in Mathematics, pages 50–94. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[12] E.R. Canfield, Paul Erdös, and Carl Pomerance. On a problem of Oppenheim concerning "factorisatio numerorum". *Journal of Number Theory*, 17(1):1–28, 1983.

[13] Greg Childers. Factorization of a 1061-bit number by the special number field sieve. *IACR Cryptol. ePrint Arch.*, 2012:444, 2012.

[14] Henri Cohen. *A Course in Computational Algebraic Number Theory.* Graduate Texts in Mathematics 138. Springer Berlin Heidelberg : Imprint: Springer, Berlin, Heidelberg, 1993.

[15] D Coppersmith. Modifications to the number field sieve. *Journal of Cryptology*, 6(3):169–180, 1993.

[16] Richard E. Crandall and Carl B. Pomerance. *Prime Numbers: A Computational Perspective.* Springer, 1st edition, 2001.

[17] Marije Elkenbracht-Huizing. An implementation of the number field sieve. *Experimental Mathematics*, 5(3):231–253, 1996.

[18] Marije Elkenbracht-Huizing. A multiple polynomial general number field sieve. In *Algorithmic Number Theory*, Lecture Notes in Computer Science, pages 99–114. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[19] Alexander Hulpke. Factorization of $n = 87463$ with the quadratic sieve. `https://www.math.colostate.edu/~hulpke/lectures/m400c/quadsievex.pdf`, 2004.

[20] G. J. O. Jameson. *Some important Dirichlet series and arithmetic functions*, page 56–97. London Mathematical Society Student Texts. Cambridge University Press, 2003.

[21] Patrick Konsor. Useless accomplishment: Rsa-140 factorization with quadratic sieve. `https://www.mersenneforum.org/showthread.php?t=25676`, Jun 2020.

[22] A Lenstra, H Lenstra, Jr, M Manasse, and J Pollard. The number field sieve. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing*, STOC '90, pages 564–572. ACM, 1990.

[23] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[24] Enrique Martin-Lopez, Anthony Laing, Thomas Lawson, Roberto Alvarez, Xiao-Qi Zhou, and Jeremy L O'Brien. Experimental realization of Shor's quantum factoring algorithm using qubit recycling. *Nature Photonics*, 6(11):773–776, 2012.

[25] Michael A. Morrison and John Brillhart. A method of factoring and the factorization of F_7. *Mathematics of Computation*, 29(129):183–205, 1975.

[26] Phong Nguyen. A Montgomery-like square root for the Number Field Sieve. In *Algorithmic Number Theory*, Lecture Notes in Computer Science, pages 151–168. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[27] M. Pohst and H. Zassenhaus. *Algorithmic Algebraic Number Theory.* Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1989.

[28] Carl Pomerance. A tale of two sieves. *Notices Amer. Math. Soc*, 43:1473–1485, 1996.

[29] Carl Pomerance. Smooth numbers and the quadratic sieve. In *Algorithmic Number Theory: Lattices, Number Fields, Curves and Cryptography*, volume 44 of *Math. Sci. Res. Inst. Publ.*, pages 69–81. Cambridge Univ. Press, Cambridge, 2008.

[30] Hans Riesel. *Prime Numbers and Computer Methods for Factorization.* Progress in Mathematics, 126. Springer Science+Business Media, LLC, New York, New York, 1994.

[31] Pierre Samuel and Allan J Silberger. *Algebraic Theory of Numbers: Translated from the French by Allan J. Silberger*. Dover Publications, New York, 2008.

[32] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[33] Victor Shoup. Factoring polynomials over finite fields: Asymptotic complexity vs. reality. In *Proceedings of the IMACS Symposium*, 1993.

[34] Robert D. Silverman. The multiple polynomial quadratic sieve. *Mathematics of Computation*, 48(177):329–339, 1987.

[35] Emmanuel Thomé. Square root algorithms for the number field sieve. In *Arithmetic of Finite Fields*, volume 7369 of *Lecture Notes in Computer Science*, pages 208–224, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[36] Edwin Weiss. *Algebraic number theory*. International Series in Pure and Applied Mathematics. McGraw-Hill, New York, 1963.