Generating Random Factored Gaussian Integers, Easily

Noah Lebowitz-Lockard
Advisor: Carl Pomerance

June 6, 2013

Abstract

We introduce an algorithm to generate a random Gaussian integer with the uniform distribution among those with norm at most N, along with its prime factorization. Then, we show that the algorithm runs in polynomial time. The hard part of this algorithm is determining a norm at random with a specific distribution. After that, finding the actual Gaussian integer is easy. We also consider the analogous problem for Eisenstein integers and quadratic integer rings.

1 Generating Random Factored Numbers, Easily

Consider the following problem:

Given a positive integer N, generate a random integer less than or equal to N with uniform distribution, along with its factorization in polynomial time. (In this context, polynomial time refers to a polynomial in the number of digits of N, not the size of N. So, the running time of our algorithm should be $O(\log^k N)$, for some real k.)

At first glance, this seems very simple. Simply choose a random integer in the range $[1, N]$ and factor it. However, there are no known polynomial time factorization algorithms. But, the problem does not explicitly state that we need to factor anything. We need a random factored number, not a method to factor random numbers.

In 1988, Eric Bach presented an algorithm without any factorizations at all in his paper "How to Generate Random Factored Numbers" [2]. Bach's algorithm requires $O(\log N)$ primality tests. In 2003, Adam Kalai presented another algorithm in his "Generating Random Factored Numbers, Easily", which used $O(\log^2 N)$ primality tests [1]. Though Kalai's algorithm is slower, it is also easier to understand and we shall spend the rest of the paper discussing modifications of it.

Also note that though we will not factor any numbers, every one of the algorithms presented in this paper will perform primality tests, which we can run in polynomial time. The two polynomial time primality algorithms are the Miller-Rabin Test and the Agrawal-Kayal-Saxena (AKS) Algorithm. A programmer running the algorithms in this paper should use either of the two primality tests just mentioned. Because Kalai's algorithm requires $O(\log^2 N)$ primality tests, using Miller-Rabin or AKS enables the algorithm to run in polynomial time. Either of these algorithms could be used as a subroutine within the program.

Strictly speaking, the Miller-Rabin Test can only check compositeness. Running Miller-Rabin once will either tell you that a given number is composite or the test will be inconclusive. If you run the test many times, and it is consistently inconclusive, then the given number is almost certainly prime. By running the test many times, we can be arbitrarily accurate, i.e. the probability of error is less than ϵ for a given positive real ϵ.

For a given function f, we define $\tilde{O}(f)$ as $O(f \log^k f)$, where k is some positive real number. Note that $\log^k f$ grows much more slowly than f. Running the Miller-Rabin Test on N once has a running time of $\tilde{O}(\log^2 N)$ assuming we perform multiplications using a Fast Fourier Transform [12]. If the Generalized Riemann Hypothesis is true, then the Miller-Rabin Test becomes deterministic after running it $O(\log^2 N)$ times. In this scenario, we could determine whether or not a given integer is prime in $\tilde{O}(\log^4 N)$ time [8].

AKS will tell you whether a given number is prime or composite with perfect accuracy, but it is significantly slower. AKS runs in $\tilde{O}(\log^{15/2} N)$ time [1]. Specifically, AKS runs in $O((\log N)^{15/2}(2 + \log\log N)^c)$ for some constant c. H. W. Lenstra and Carl Pomerance recently found a faster deterministic algorithm that runs in time $O((\log N)^6(2 + \log\log N)^c)$ [9]. If perfect accuracy is required, then we suggest Lenstra and Pomerance's algorithm. If you can settle for very good accuracy, then we suggest Miller-Rabin.

2 Kalai's Algorithm

Here is the algorithm from Adam Kalai's "Generating Random Factored Numbers, Easily" [7].

Algorithm 1. *Given a positive integer N, this algorithm produces a random positive integer $r \leq N$, along with its factorization, with uniform distribution.*

1. *Create a list of integers $s_1 \geq s_2 \geq \ldots \geq s_k = 1$, where s_1 is chosen uniformly at random in $[1, N]$ and if s_i has been chosen and $s_i > 1$, then s_{i+1} is chosen uniformly at random in $[1, s_i]$. Call this list S. The procedure terminates when 1 is chosen. When the procedure terminates, go to Step 2.*

2. *Let r be the product of the prime elements of S.*

3. *If $r > N$, return to Step 1. Otherwise, output r, along with its prime factorization, with probability r/N. If you did not output r, return to Step 1.*

For example, let $N = 100$. The algorithm might generate the list $[98, 41, 38, 3, 3, 1]$. When we multiply the prime elements of the list, we obtain 369, so we would create a new list. We might obtain $[70, 5, 5, 2, 1]$, in which case we would output 50 with probability 0.5. We have to prove two facts about this algorithm.

1. The algorithm generates each $r \leq N$ with probability $1/N$.

2. The algorithm expects to use $O(\log^2 N)$ primality tests.

How do we determine the probability of obtaining r? Consider the following "generalized prime factorization" of r:

$$r = \prod_{p \leq N} p^{\alpha_p}.$$

(Throughout this paper, p will always be a prime.)

For any $p \leq N$, p^{α_p} is the highest power of p that is a factor of r. Note that in our definition, $\alpha_p = 0$ if p is not a factor of r. In a normal prime factorization, we would simply ignore any prime that is not a factor of r, but in this case, we must include all primes less than or equal to N.

In order for our algorithm to output r, the list must contain exactly α_p copies of p for each $p \leq N$. But, what is the probability that the list contains α_p copies of p? Suppose we have not yet finished our list and every element of the list so far is greater than or equal to p. Then, it is still possible to add a copy of p to the list. The conditional probability of choosing p given that you are choosing some number in $[1, p]$ is $1/p$. The conditional probability of adding a number smaller than p is $1 - (1/p)$. The probability that the list contains n copies of p is equal to the probability of adding one copy of p n times in a row, then adding a number less than p, namely,

$$P(n \text{ copies of } p) = \frac{1}{p^n}\left(1 - \frac{1}{p}\right).$$

Let $P^*(r)$ be the probability that the product of the primes in a list is equal to r. In other words, $P^*(r)$ is the probability of outputting r if we ran Kalai's algorithm without Step 3. We can use the equation above to determine the initial probability that we obtain r:

$$P^*(r) = \prod_{p \leq N} P(\alpha_p \text{ copies of } p) = \prod_{p \leq N} \frac{1}{p^{\alpha_p}} \left(1 - \frac{1}{p}\right) = \prod_{p \leq N} \frac{1}{p^{\alpha_p}} \prod_{p \leq N} \left(1 - \frac{1}{p}\right).$$

Notice that the product of $1/p^\alpha$ for all $p \leq N$ is equal to $1/r$. Also note that the product of $1 - (1/p)$ is a function of N and not r. We can define M_N as this product:

$$M_N = \prod_{p \leq N} \left(1 - \frac{1}{p}\right).$$

We can express the product of obtaining r at the end of Step 2 more succinctly as

$$P^*(r) = \frac{M_N}{r}.$$

Here, r is any positive integer supported on the primes in $[1, N]$. Step 3 states that if $r \leq N$, then we should output r with probability r/N. Otherwise, we should not output r at all. $P(r)$ is equal to $P^*(r)$ times the conditional probability that the algorithm outputs r given that it is the product of the primes in the list. Here is the actual probability that we output r:

$$P(r) = P^*(r) \cdot \frac{r}{N} = \frac{M_N}{r} \cdot \frac{r}{N} = \frac{M_N}{N}.$$

Note that the probability that we output r does not actually depend on the value of r, as long as $r \leq N$. Thus, Kalai's algorithm outputs every number less than or equal to N with a uniform distribution.

Now, we have to prove that Kalai's algorithm requires an average of $O(\log^2 N)$ primality tests. We will do this by showing that the algorithm produces an average of $O(\log N)$ lists and by showing that the average list has $O(\log N)$ distinct elements.

We have just shown that the probability of the algorithm producing a list and outputting r is M_N/N. There are N possible numbers that we can output. The probability that the algorithm terminates is equal to $N(M_N/N)$, or M_N. Therefore, the expected number of lists we have to produce is M_N^{-1}. In order to estimate M_N, we introduce Mertens' Two Theorems. Our proofs of them come from [10]. The value of γ comes from [6].

Definition. For any positive real number x,

$$A(x) = \sum_{p \leq x} \frac{\log p}{p}.$$

Theorem 1. *For any real $x \geq 1$,*

$$A(x) = \log x + O(1).$$

If there were only finitely many primes, then the sum of $(\log p)/p$ for all prime $p \leq x$ would be bounded above by some constant C. But, $\log x$ diverges. Therefore, Theorem 1 provides an alternate proof for the infinitude of the primes.

Theorem 2. (Mertens' First Theorem) *As N approaches infinity, the sum of the reciprocals of the primes that are less than or equal to N becomes asymptotic to* $\log \log N$.

Proof. In order to take the sum of $1/p$ for all $p \leq N$, we will split it into two separate sums:

$$\sum_{p \leq N} \frac{1}{p} = \sum_{p \leq N} \left(\frac{\log p}{p}\right)\left(\frac{1}{\log p}\right) = \frac{1}{\log N}\sum_{p \leq N} \frac{\log p}{p} + \sum_{p \leq N} \frac{\log p}{p}\left(\frac{1}{\log p} - \frac{1}{\log N}\right).$$

Theorem 1 makes the first sum easy to handle:

$$\frac{1}{\log N}\sum_{p \leq N} \frac{\log p}{p} = \frac{A(N)}{\log N} = 1 + O\left(\frac{1}{\log N}\right).$$

The second sum is more difficult. It appears as though we should be able to write it in terms of $A(N)$, but it is difficult to see how. We can rewrite the argument by de-telescoping it, i.e. writing it as a sum of terms that contract:

$$\frac{1}{\log p} - \frac{1}{\log N} = \sum_{n=p}^{N-1}\left(\frac{1}{\log n} - \frac{1}{\log(n+1)}\right).$$

Therefore,

$$\sum_{p \leq N} \frac{\log p}{p}\left(\frac{1}{\log p} - \frac{1}{\log N}\right) = \sum_{p \leq N} \frac{\log p}{p}\sum_{n=p}^{N-1}\left(\frac{1}{\log n} - \frac{1}{\log(n+1)}\right).$$

We can switch the sums around. Instead, we are taking the sum of $(1/\log n) - (1 - \log(n+1))$ for all $p \leq n$ and all n from 2 to $N-1$:

$$\sum_{p \leq N} \frac{\log p}{p}\left(\frac{1}{\log p} - \frac{1}{\log N}\right) = \sum_{n=2}^{N-1}\left(\frac{1}{\log n} - \frac{1}{\log(n+1)}\right)\sum_{p \leq n} \frac{\log p}{p}$$

$$= \sum_{n=2}^{N-1} A(n)\left(\frac{1}{\log n} - \frac{1}{\log(n+1)}\right).$$

Observe that $1/\log t$, like all differentiable functions, is the integral of its own derivative:

$$\sum_{n=2}^{N-1} A(n)\left(\frac{1}{\log n} - \frac{1}{\log(n+1)}\right) = \sum_{n=2}^{N-1} A(n)\int_n^{n+1} \frac{1}{t \log^2 t} dt.$$

For any t in the half-open interval $[n, n+1)$, $A(t) = A(n)$ because a prime must be all primes are integers. Every prime that is less than or equal to t is also less than or equal to n. Therefore, $A(t)$ is a constant over the interval $[n, n+1]$, allowing us to move $A(n)$ inside the integral:

$$\sum_{n=2}^{N-1} A(n) \int_n^{n+1} \frac{1}{t \log^2 t} dt = \sum_{n=2}^{N-1} \int_n^{n+1} \frac{A(t)}{t \log^2 t} dt = \int_2^N \frac{A(t)}{t \log^2 t} dt.$$

Once again, we break our expression into a sum and put asymptotic estimates on both of its components:

$$\int_2^N \frac{A(t)}{t \log^2 t} dt = \int_2^N \frac{1}{t \log t} dt + \int_2^N \frac{A(t) - \log t}{t \log^2 t} dt.$$

For the first integral,

$$\int_2^N \frac{1}{t \log t} dt = \log \log N - \log \log 2.$$

Theorem 1 states that $A(t) = \log t + O(1)$. Therefore, $|A(t) - \log t|$ is bounded above by some positive constant C. Hence,

$$\left| \int_2^N \frac{A(t) - \log t}{t \log^2 t} dt \right| \leq C \left| \int_2^N \frac{1}{t \log^2 t} dt \right| = C \left(\frac{1}{\log 2} - \frac{1}{\log N} \right) = O(1).$$

Putting all of this together gives us our result:

$$\sum_{p \leq N} \frac{1}{p} = \log \log N + O(1).$$

□

Theorem 3. (Mertens' Second Theorem) *Let $\gamma = 0.5772\ldots$ be the Euler-Mascheroni Constant. Then,*

$$\lim_{N \to \infty} M_N \log N = e^{\gamma}.$$

Proof. Let x be a real number with absolute value less than 1. The Taylor Expansion of $\log(1+x)$ is

$$x - \frac{x^2}{2} + \frac{x^3}{3} + \ldots = -\sum_{k=1}^{\infty} \frac{(-x)^k}{k}.$$

Let $x = -1/p$ for some prime p. Making this substitution gives us

$$\log \left(1 - \frac{1}{p}\right) = -\left(\frac{1}{p} + \frac{1}{2p^2} + \ldots\right) = -\sum_{k=1}^{\infty} \frac{1}{kp^k}.$$

We can also separate out the first term and put the other terms into a separate sum. Ultimately, we will prove that for sufficiently large primes p, only the first term really affects

our proof. This makes sense on an intuitive level. As p approaches infinity, $1/p^2$ becomes meaningless next to $1/p$. We have

$$\log\left(1-\frac{1}{p}\right) = -\frac{1}{p} - \sum_{k=2}^{\infty}\frac{1}{kp^k}.$$

At this point, we take the logarithm of M_N in order to use this result.

$$\log M_N = \log\prod_{p\leq N}\left(1-\frac{1}{p}\right) = \sum_{p\leq N}\log\left(1-\frac{1}{p}\right) = -\sum_{p\leq N}\frac{1}{p} - \sum_{p\leq N}\sum_{k=2}^{\infty}\frac{1}{kp^k}.$$

Mertens' First Theorem states that the sum of $1/p$ for all $p \leq N$ is asymptotic to $\log\log N$. We will prove that the other sum in the equation above is on a smaller order and may be ignored. For any prime p,

$$\sum_{k=2}^{\infty}\frac{1}{kp^k} < \sum_{k=2}^{\infty}\frac{1}{2p^k} = \frac{1}{2}\sum_{k=2}^{\infty}\frac{1}{p^k} = \frac{1}{2p(p-1)} \leq \frac{1}{p^2}.$$

Therefore,

$$\sum_{p\leq N}\sum_{k=2}^{\infty}\frac{1}{kp^k} < \sum_{p\leq N}\frac{1}{p^2} < \sum_{n=1}^{\infty}\frac{1}{n^2}.$$

We can prove that the sum of the reciprocals of the squares converges. Note that the number m is equal to the integral of m over an interval of unit length. That allows us to put the integral into the last part of the equation:

$$\sum_{n=1}^{\infty}\frac{1}{n^2} = 1 + \sum_{n=1}^{\infty}\frac{1}{(n+1)^2} = 1 + \sum_{n=1}^{\infty}\frac{1}{(n+1)^2} = 1 + \sum_{n=1}^{\infty}\int_{n}^{n+1}\frac{1}{(n+1)^2}dx.$$

Observe that for any real x in the interval $[n, n+1]$, $1/x^2 > 1/(n+1)^2$. Therefore, the integral of $1/x^2$ on the range $[n, n+1]$ is greater than the integral of $1/(n+1)^2$. We can use this fact to set an integral as the upper bound for our sum:

$$\sum_{n=1}^{\infty}\frac{1}{n^2} = 1 + \sum_{n=1}^{\infty}\int_{n}^{n+1}\frac{1}{(n+1)^2}dx < 1 + \sum_{n=1}\int_{n}^{n+1}\frac{1}{x^2}dx = 1 + \int_{n=1}^{\infty}\frac{1}{x^2}dx = 2.$$

(To be exact, the sum of the reciprocals of the squares is $\pi^2/6$, but that is beyond the scope of this paper.) To summarize, we have two different limits:

$$\sum_{p\leq N}\frac{1}{p} \sim \log\log N,\ \sum_{p\leq N}\sum_{k=2}^{\infty}\frac{1}{kp^k} = O(1).$$

Therefore,

$$\lim_{N\to\infty}\frac{\log M_N}{\log\log N} = -1.$$

We raise e to both sides in order to obtain an asymptotic limit for M_N:

$$\lim_{N\to\infty} M_N = e^{-\log\log N} + O(1) = O(\log^{-1} N).$$

In the equation above, C is some constant. Though we will not prove it here, $C = e^{\gamma}$. Though it is difficult to prove, one could verify the value of C by calculating M_N and $\log^{-1} N$ for very large values of N. □

From this point onward, whenever we refer to Mertens' Theorem, we will be referring to his second theorem. We expect to create M_N^{-1} lists. As N approaches infinity, the ratio between M_N is asymptotic to $e^{\gamma}\log N$. Hence, $M_N = O(\log N)$ and Kalai's algorithm creates an average of $O(\log N)$ lists. But, how many distinct elements does a list contain? Let n be an integer that is less than or equal to N. The probability that the list contains at least one copy of n is $1/n$. If the list contains a copy of n, then we must check if n is prime exactly once. Otherwise, we do not have to check whether or not n is prime. The expected number of times the algorithm checks if n is prime when processing a given list is $1/n$. By additivity of expectation, the expected number of primality tests required for a given list is

$$1 + \frac{1}{2} + \ldots + \frac{1}{N} = \sum_{n=1}^{N} \frac{1}{n}.$$

We can write both an upper and lower bound for this sum. Consider the integral of $(1/x)dx$ as x goes from 1 to N. We can split this integral into a sum of smaller integrals as follows:

$$\int_1^N \frac{1}{x}dx = \sum_{n=1}^{N-1} \int_n^{n+1} \frac{1}{x}dx.$$

The minimum value of $1/x$ on the interval $[n, n+1]$ is $1/(n+1)$. The length of the interval $[n, n+1]$ is 1. Thus,

$$\sum_{n=1}^{N-1} \int_n^{n+1} \frac{1}{x}dx > \sum_{n=1}^{N-1} \frac{1}{n+1} = \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{N}.$$

Adding 1 to both sides gives us a useful inequality:

$$1 + \frac{1}{2} + \ldots + \frac{1}{N} < 1 + \int_1^N \frac{1}{x}dx = 1 + \log N.$$

The maximum value of $1/x$ on the interval $[n, n+1]$ is $1/n$, so that

$$\log N = \int_1^N \frac{1}{x}dx = \sum_{n=1}^{N-1} \int_n^{n+1} \frac{1}{x}dx < \sum_{n=1}^{N-1} \frac{1}{n} < \sum_{n=1}^{N} \frac{1}{n} = 1 + \frac{1}{2} + \ldots + \frac{1}{N}.$$

We can now write both an upper and lower bound for our sum. We have

$$\log N < \sum_{n=1}^{N} \frac{1}{n} < 1 + \log N.$$

Our takeaway is that

$$\sum_{n=1}^{N} \frac{1}{n} = O(\log N).$$

Thus, the expected number of primality tests the algorithm performs for a given list is $O(\log N)$. Since we've seen that the expected number of lists is $O(\log N)$, Kalai's algorithm does an average of $O(\log^2 N)$ primality tests.

3 The Gaussian Problem

Our new goal is to generate a random Gaussian integer with norm less than or equal to given integer N, along with its factorization into Gaussian primes. From here on, $N(z)$ will be the norm of the complex number z. Our plan of attack will be as follows:

1. Find a formula for $G(r)$, the number of Gaussian integers in the first quadrant with a given norm r.

2. Modify Kalai's Algorithm so that the probability of outputting a given r is proportional to $G(r)$.

3. For a given r, along with its factorization into rational primes, produce a random Gaussian integer with norm r, along with its factorization into Gaussian primes.

The Gaussian integers have four units, unlike the rational integers, which have 2. Throughout this section, we will only be concerned the outcome up to a unit. For simplicity, we may assume that the Gaussian primes we generate are all in the first quadrant. If not, we multiply them by a power of i and then they will be. However, this means that for a given output z, we cannot obtain iz, $-z$, or $-iz$. To rectify this, the reader may insert a Step 4 at the end of the algorithm. Step 4 multiplies the Gaussian integer that the algorithm was just outputted by i^k, where k is a random integer in the range $[0, 4]$. Units will not seriously concern us in this section.

In order to find a formula, we will introduce the function D and then prove that G is identical to D. We will do this by showing that G and D both possess a certain set of properties, then proving that any two functions that possess all of these properties are identical. In order to define D, we must define a few other functions first.

Definition. The divisor function d gives the number of positive divisors of a given integer.

Definition. Let r be a positive integer with the prime factorization used in the theorem above. Let r_1 be the largest factor of r which only contains primes that are congruent to 1 mod 4. Let r_3 be the largest factor of r which only contains primes that are congruent to 3 mod 4. Note that with this notation, there exists some nonnegative integer k such that $r = 2^k r_1 r_3$.

Definition. Let r be a positive integer. Let $D(r)$ be the following function:

$$D(r) = \begin{cases} d(r_1) & r_3 \text{ is a square,} \\ 0 & \text{otherwise.} \end{cases}$$

At this point, we shall define a new property of functions and prove that d, D, and G all have this property.

Definition. A function $f : \mathbb{N} \to \mathbb{R}$ is multiplicative if $f(mn) = f(m)f(n)$ for all relatively prime m, n.

Theorem 4. *Let r be a positive integer with prime factorization $p_1^{e_1} \dots p_m^{e_m}$. Then,*

$$d(r) = (e_1 + 1) \dots (e_m + 1).$$

Proof. Let r_0 be a factor of r. Any prime factor of r_0 must also be a prime factor of r. The exponent of p_i in r_0 can be any number that is less than or equal to e_i. In other words, $r_0 = p_1^{f_1} \dots p_m^{f_m}$, where each f_i is an integer in the range $[0, e_i]$. There are $e_i + 1$ possibilities for each f_i. Therefore, there are $(e_1 + 1) \dots (e_m + 1)$ factors of r. □

Theorem 5. *The function d is multiplicative.*

Proof. Let $a = p_1^{e_1} \dots p_n^{e_n}$ and $b = q_1^{f_1} \dots q_m^{f_m}$ be two relatively prime integers. Then, we can determine $d(ab) = d(a)d(b)$. It is easy to write out $d(a)d(b)$:

$$d(a)d(b) = (e_1 + 1) \dots (e_n + 1)(f_1 + 1) \dots (f_m + 1).$$

Because a and b are relatively prime, there do not exist integers i and j such that $p_i = q_j$. This makes writing ab especially easy:

$$ab = p_1^{e_1} \dots p_n^{e_n} q_1^{f_1} \dots q_m^{f_m}.$$

We then apply the divisor function to ab:

$$d(ab) = (e_1 + 1) \dots (e_n + 1)(f_1 + 1) \dots (f_m + 1) = d(a)d(b).$$

Hence, d is multiplicative. □

Theorem 6. *The function D is multiplicative.*

Proof. Once again, let $a = p_1^{e_1} \dots p_n^{e_n}$ and $b = q_1^{f_1} \dots q_m^{f_m}$. Suppose a_3 and b_3 are both squares. Then, $a_3 b_3$ is a square. hence, $D(ab) = d(a_1 b_1) = d(a_1)d(b_1) = D(a)D(b)$. Suppose either a_3 or b_3 is not a square. Then, $a_3 b_3$ is not a square. Hence, $(ab)_3$ is not a square. So, $D(ab) = 0 = D(a)D(b)$. Thus, D is multiplicative. □

To prove that G is multiplicative, we need a few results.

Definition. A Gaussian prime is a Gaussian integer z such that if $z = ab$, where a and b are Gaussian integers, then either a or b is a unit.

The fact that we can factor Gaussian integers comes from the following theorem:

Theorem 7. (Fundamental Theorem of Gaussian Integers) [11] *"The expression of an integer as a product of primes is unique, apart from the order of the primes, the presence of unities, and ambiguities between associated primes."*

At this point, we will present an outline of the proof of the Fundamental Theorem.

Definition. [5] Let R be a commutative ring. The subset I of R is an ideal if I is a group under addition and for any $a \in I$ and $r \in R$, ar is an element of I.

Definition. An element a of R is irreducible if it is not a unit and its only divisors are 1 and itself up to a unit.

It may seem as though we have just defined a prime, but in fact, for the Gaussian integers, the two concepts are one and the same.

Definition. An element $p \in R$ is prime if for any $a, b \in R$ such that $p|ab$, either $p|a$ or $p|b$.

Definition. A ring R is a unique factorization domain if for every $r \in R$, there is a unique way of writing r as the product of irreducible elements, up to a change in units.

Definition. A ring R is a principal ideal domain if for every ideal $I \subseteq R$, there exists some element $r \in R$ such that $I = \{rx \mid x \in R\}$.

Definition. A ring R is a Euclidean domain if there exists a function $\delta : R \to \mathbb{N}$ that satisfies the following two properties:

1. For any $a, b \in R \backslash \{0\}$, $\delta(ab) \geq \delta(a)\delta(b)$.
2. For any $a, b \in R$ with $b \neq 0$, there exist $q, r \in R$ such that $\delta(r) < \delta(b)$ or both $a = bq + r$ and $\delta(r) < \delta(b)$.

The Euclidean algorithm is an efficient method for finding the greatest common divisor of two numbers. When a ring is a Euclidean domain, it means that the Euclidean algorithm applies to the ring. Therefore, we can define an analogous notion of a gcd for the Gaussian integers.

Theorem 8. *Primes and irreducibles are identical in a unique factorization domain.*

Theorem 9. *Every Euclidean domain is a principal ideal domain.*

The reason that this theorem is true is that the gcd of all the elements of an ideal must be contained in the ideal. Every element of the ideal must be a multiple of that element. Alternately, every multiple of the element must be contained in the ideal.

Theorem 10. *Every principal ideal domain is a unique factorization domain.*

Theorem 11. *The Gaussian integers form a Euclidean domain.*

Putting all these theorems and definitions together proves the Fundamental Theorem of Gaussian Integers. Because the Gaussian integers from a Euclidean domain, they must form a principal ideal domain and a unique factorization domain.

Theorem 12. *G is multiplicative.*

Proof. For a given integer n, let G_n be the set of all Gaussian integers with norm n. By definition, $G(n)$ is the size of G_n. Let a and b be two relatively prime integers. We shall prove that $G(ab) = G(a)G(b)$ by creating a bijection from $G_a \times G_b$ to G_{ab}. Define the function $f : G_a \times G_b \to G_{ab}$ as $f(x, y) = xy$.

To prove that f is a bijection, we must show that it is injective and surjective. Suppose $f(x, y) = f(x_0, y_0)$ for some $x_1, x_2 \in G_a$ and $y_1, y_2 \in G_b$. Then, $x_1y_1 = x_2y_2$. So, x_1y_1 and x_2y_2 have the same factorization into Gaussian primes. Let z be a Gaussian prime that divides x_1y_1. If the norm of z divides a, then z must divide both x_1 and x_2, but not y_1 and y_2. Otherwise, the norm of z divides b and z divides y_1 and y_2, but not x_1 and x_2. It is impossible for z to divide both a and b because a and b are relatively prime. Therefore, x_1 and x_2 are composed of the exact same Gaussian primes, implying that $(x_1, x_2) = (y_1, y_2)$. Every element of G_{ab} has at most one inverse. Therefore, f is injective.

Now, we must prove that f is surjective. Let $w \in G_{ab}$. By definition, $N(w) = ab$. Let p be a prime factor of ab where p^n is the largest power of p that divides ab. There exists some Gaussian integer with norm p^n that divides w. Either p^n divides a or p^n divides b. Let w_1 be the product of the Gaussian integers with norm p^n that divide a and w_2 be the product of the Gaussian integers with norm p^n that divide b. Then, w_1 has norm a and w_2 has norm b. Also, $w = w_1w_2$. All this implies that $f(w_1, w_2) = w$ and f is surjective. Hence, f is bijective and $G(ab) = G(a)G(b)$. By definition, G is multiplicative. □

At this point, we will determine $G(p^n)$ where p is a prime and n is a positive integer. First, we shall introduce the notion of a quadratic residue.

Definition. Let p be an odd prime and a be an integer that is not a multiple of p. We say that a is a quadratic residue mod p if the equation $x^2 \equiv a \bmod p$ has a solution. Otherwise, a is a quadratic non-residue. To express the quadratic residues compactly, we use a Legendre symbol:

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

The Legendre symbol can be easily calculated for specific values of a and p with a congruence.

Theorem 13. (Euler's Criterion) [4] *Let p be an odd prime. Then,*

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

For this section, we will only need to know whether or not -1 is a quadratic residue for a given prime p. This can be easily solved with the following theorem and its corollary. In Section 10, we will use other values of a.

Corollary 1. *Let p be an odd prime. Then, -1 is a quadratic residue mod p if and only if $p \equiv 1$ mod 4.*

Proof. Let $p \equiv 1 \bmod 4$. Then, $p-1$ is a multiple of 4. Therefore, $(p-1)/2$ is even and $(-1)^{\frac{p-1}{2}} = 1$. By the theorem above, -1 is a quadratic residue.

Suppose $p \equiv 3 \bmod 4$. Then, $p-1 \equiv 2 \bmod 4$ and $(p-1)/2$ is odd. Hence, $(-1)^{\frac{p-1}{2}} = -1$. By the theorem above, -1 is a quadratic non-residue. □

In order to address Point 1, we shall prove the following theorem, which determines the number of Gaussian integers with norm p, where p is a prime. We treat two Gaussian integers as equal if one divided by the other is a power of i. In other words, you could transform one Gaussian integer into the other simply by rotating it with a series of right angles.

Later in this section, we will introduce an algorithm for finding solutions to $x^2 \equiv a \bmod p$ in the case where such a solution exists.

Theorem 14. *A Gaussian prime in the first quadrant possesses exactly one of the following three properties:*

1. *The Gaussian integer is equal to $1+i$.*
2. *It is equal to p, where p is a prime that is congruent to 3 mod 4.*
3. *Its norm is p, where p is a prime that is congruent to 1 mod 4.*

Proof. $1+i$ has norm 2. Any factor of $1+i$ must have norm 1 or 2. But, $1+i$ is the only Gaussian integer in the first quadrant. Therefore, $1+i$ is a Gaussian prime.

Let p be a a prime that is congruent to 3 mod 4. We can prove that the only Gaussian factors of p are 1 and p. Let $a+bi$ be a Gaussian prime with norm p. Then, $a^2+b^2 = p$, which is impossible because the sum of two squares cannot be congruent to 3 mod 4. There are no Gaussian primes with norm p. p is a Gaussian prime.

Let p be a prime that is congruent to 1 mod 4. We already proved that -1 is a quadratic residue of p. Therefore, there exists some positive integer $x < p$ such that $x^2 \equiv -1 \bmod p$. Let z be the gcd of $x+i$ and p. The norm of $x+i$ is a multiple of p, but not p^2, while the norm of p is p^2. The norm of z must be the gcd of $|x+i|$ and p^2, which is p. So, p has some proper Gaussian prime factor, namely z. □

Theorem 15. *For a prime p, we have*

$$G(p) = \begin{cases} 2 & p \equiv 1 \bmod 4, \\ 0 & p \equiv 3 \bmod 4, \\ 1 & p = 2. \end{cases}$$

Proof. It is easy to see why this theorem is true for $p = 2$ and $p \equiv 3 \bmod 4$. The only Gaussian integers with norm 2 have the form $\pm 1 \pm i$. However, all of these numbers are considered the same because the quotient of any pair of them is a power of i. Thus, $G(p) = 2$.

Let $p \equiv 3 \bmod 4$. Suppose $a + bi$ has norm p. Then, $\sqrt{a^2 + b^2} = \sqrt{p}$. Hence, $a^2 + b^2 = p$. Every square is equivalent to 0 or 1 mod 4. So, $a^2 + b^2$ is 0, 1, or 2 mod 4. But, $p \equiv 3 \bmod 4$. Hence, $a^2 + b^2 = p$ has no solutions. If $p \equiv 3 \bmod 4$, then $G(p) = 0$.

Finally, suppose $p \equiv 1 \bmod 4$. Once again, suppose $a + bi$ has norm p and $a^2 + b^2 = p$. The Corollary states that -1 is a quadratic residue mod p. There exists some solution to the equation $x^2 \equiv -1 \bmod p$. Therefore, $x^2 + 1$ is a multiple of p. There exists some integer k such that $x^2 + 1 = kp$. We can factor $x^2 + 1$ in the Gaussian integers:

$$x^2 + 1 = (x + i)(x - i).$$

Thus, $(x + i)(x - i)$ is a multiple of p. But, p cannot divide either $x + i$ or $x - i$ because their imaginary components are ± 1, which is not a multiple of p. Both $x + i$ and $x - i$ share a common factor with p, that is not p itself. To find this common factor, simply calculate $\gcd(x + i, p)$. Suppose z is a Gaussian integer that is not a unit that divides p, but is not a multiple of p. Then, $|z|$ is a proper divisor of p^2. The only possibilities are $|z| = 1$ and $|z| = p$. Because z is not a unit, $|z| \neq 1$. Thus, $|z| = p$. Let $a + bi = z$. Then, $a + bi$ is a Gaussian integer with norm p. □

As an example of the construction in the proof above, consider $p = 41$. There exist some $a, b \in \mathbb{Z}_+$ such that $a^2 + b^2 = p$ because $p \equiv 1 \bmod 4$. One solution to $x^2 + 1 \equiv 0 \bmod 41$ is $x = 9$. So, we want to find the gcd of $9 + i$ and 41. Just as in $\mathbb{Z}$, we may use the Euclidean Algorithm:

$$\gcd(9 + i, 41) = \gcd(9 + i, 41 - 4(9 + i)) = \gcd(9 + i, 5 - 4i).$$

Note that $5 - 4i$ divides $9 + i$. Specifically, $9 + i = (5 + 4i)(1 + i)$. Hence, the gcd of $9 + i$ and $5 - 4i$ is $5 - 4i$. Hence, $5 - 4i$ is our Gaussian integer of norm 41. We confirm this fact by noting that $5^2 + 4^2 = 41$.

Now that we know $G(p)$, we can find $G(p^n)$.

Theorem 16. *For any prime p and positive integer n:*

$$G(p^n) = \begin{cases} (1 + (-1)^n)/2 & p \equiv 3 \bmod 4, \\ n + 1 & p \equiv 1 \bmod 4, \\ 1 & p = 2. \end{cases}$$

Proof. Let $p \equiv 1 \bmod 4$ and let z be a Gaussian integer with norm p^n. Then, z is the product of n Gaussian integers with norm p. But, there are exactly two Gaussian integer with norm p, up to a unit. These two integers have the forms $a + bi$ and $a - bi$, for some $a, b \in \mathbb{Z}_+$. Therefore, $z = (a + bi)^m(a - bi)^{n-m}$, where m is a nonnegative integer. Thus, m can range from 0 to n. So, there are $n + 1$ possibilities for z.

Let $p = 2$ and let z be a Gaussian integer with norm 2^n. Then, z is the product of n Gaussian integers with norm 2. However, $1 + i$ is the only Gaussian integer with norm 2. Hence, $z = (1 + i)^n$, implying that z is unique.

Let $p = 3$ and let z be a Gaussian integer with norm p^n. Once again, z is the product of n Gaussian integers with norm p. But, there are no Gaussian integers with norm p, implying that z cannot exist. □

Note that $D(p^n) = G(p^n)$. We introduce a theorem that shows that G and D are identical.

Theorem 17. *Let f and g be two multiplicative functions. If $f(p^n) = g(p^n)$ for all prime p and positive n, then f and g are identical.*

Proof. Let r be a positive integer with prime factorization $p_1^{e_1} \dots p_k^{e_k}$. By assumption, we have:

$$f(r) = f(p_1^{e_1} \dots p_k^{e_k}) = f(p_1^{e_1}) \dots f(p_k^{e_k}) = g(p_1^{e_1}) \dots g(p_k^{e_k}) = g(p_1^{e_1} \dots p_k^{e_k}) = g(r).$$

f and g are identical. □

Note that $G(p^n) = D(p^n)$ for all prime p and positive n. This proves the following result.

Theorem 18. *For a positive integer r,*

$$G(r) = \begin{cases} d(r_1) & r_3 \text{ is a square,} \\ 0 & \text{otherwise.} \end{cases}$$

In Kalai's algorithm, the probability that r is the product of the prime elements of a list was $1/N$. We want the probability to be proportional to $G(r)/N$. The next few section will be spent finding such an algorithm and proving that it accomplishes this task.

We still need a way to solve the congruence $x^2 \equiv -1 \bmod p$, where p is a prime that is congruent to 1 mod 4.

Algorithm 2. *Given an prime $p \equiv 1 \bmod 4$, this algorithm produces a solution to the congruence $x^2 \equiv -1 \bmod p$.*

1. *Choose a random integer a in the interval $[2, p - 2]$. Let $x \equiv a^{(p-1)/2} \bmod p$. If $x \equiv -1 \bmod p$, then output $a^{(p-1)/4}$. If $x \equiv 1 \bmod p$, choose a new value of a.*

The process for finding a square root of -1 is randomized. Let a be an integer. Euler's Criterion states that a is a residue if and only if $a^{(p-1)/2} \equiv 1 \bmod p$. Observe that $(a^{(p-1)/2})^2 \equiv a^{p-1} \equiv 1 \bmod p$. So, $a^{(p-1)/2}$ must be congruent to one of the two square roots

of 1, which are 1 and -1. If a is a quadratic non-residue, then $a^{(p-1)/2} \equiv -1 \bmod p$. Therefore, $x^{(p-1)/4}$ is one of the square roots of $-1 \bmod p$. Half of all elements of $\mathbb{F}_p^*$ are quadratic residues and the other half are non-residues. All our algorithm does is find a non-residue and raises it to an exponent of $(p-1)/4$. On average, we will have to choose 2 values of a. Once, you have found the square root a, simply take the gcd of $a+i$ and p.

Here is an example. Let $p = 53$. We choose a random integer a in $[2, 51]$, say 6. Then, we calculate $6^{(53-1)/2} \equiv 6^{26} \equiv 1 \bmod 53$. We choose a new number. Let $a = 2$. We calculate $2^{26} \equiv -1 \bmod 53$. We output $2^{13} \equiv 30 \bmod 53$. Let's confirm this: $30^2 = 900 \equiv -1 \bmod 53$. Our algorithm produced 30, which is a square root of $-1 \bmod 53$.

Observe that to calculate $a^{(p-1)/2} \bmod p$, one does not actually have to calculate $a^{(p-1)/2}$. Use a fast modular exponentiation algorithm. To calculate a general $a^x \bmod b$, write the binary form of x. Then, obtain a raised to every power of 2 that is less than or equal to x by squaring the previous value and reducing it mod b. Finally, multiply the necessary powers of a together.

At this point, we know how many Gaussian integers have a given norm and how to generate Gaussian primes with a given norm. This leads us to an important conclusion about how to extend Kalai's Algorithm to the Gaussian integers. Generate each integer r with a probability proportional to $d(r_1)$. Output r if r is a Gaussian norm. Generate a random Gaussian integer with norm r with uniform distribution. At this point, we will demonstrate a procedure to generate the random Gaussian integers with a given norm and its prime factorization.

Algorithm 3. *Given a positive integer r, along with its factorization, this algorithm produces a random Gaussian integer with norm r, up to a power of i, with uniform distribution.*

1. *Let $z = 1$. For each p that divides r, do one of the the following three things.*
2. *If $p = 2$, multiply z by $(1+i)^{\alpha_2}$.*
3. *If $p \equiv 3 \bmod 4$, multiply z by $p^{\alpha_p/2}$.*
4. *If $p \equiv 1 \bmod 4$, determine the positive solutions to the equation $a^2 + b^2 = p$. Choose a random integer m in the interval $[0, \alpha_p]$. Multiply z by $(a+bi)^m(a-bi)^{\alpha_p - m}$.*

4 Choosing 1 Less Often

Definition. The 2-uniform probability distribution on $\{1, \ldots N\}$ is half as likely to choose 1 as it is to choose any other number.

Consider a 2-uniform version of Kalai's algorithm. Let the probability of choosing 1 in the interval $[1, N]$ be ρ. Then, the probability that we choose any other integer must be 2ρ. But, the probability that we choose some number must be equal to 1. We have

$$(2N-1)\rho = 1,$$

which implies that

$$\rho = \frac{1}{2N-1}.$$

We plan to choose integers from 1 to N, then multiply them by 2 and subtract 1, ensuring that we only obtain odd numbers. This begs the question, "Why not choose odd numbers from the get go?" For the next algorithm, we do just that. We get to replace every instance of $2s-1$ with s, making the algorithm easier to understand.

Algorithm 4. *Given a positive integer N, this algorithm produces a random positive integer $r \leq N$, along with its factorization, where the probability of obtaining r is proportional to $G(r)$.*

1. *Let M be the largest odd number that is less than or equal to N. Create a list $s_1 \geq s_2 \geq \ldots \geq s_k = 1$ of all odd numbers, where s_1 is 1 with probability $1/M$ and any odd element of $[3, N]$ with probability $2/M$. If s_i has already been chosen, then let s_{i+1} equal 1 with probability $1/s_i$ and any other odd integer in the interval $[3, s_i]$ with probability $2/s_i$.*

2. *Let r be the product of the prime s_i for each s_i in the list.*

3. *Multiply r by 2 with probability $1/2$. If you just added a 2, repeat this step. Otherwise, go to Step 4.*

4. *If $r > N$ or r_3 is not a square, do not output r and return to Step 1. Otherwise, output r with probability $rd(r_1)/(2^{\Omega_0(r)}N)$.*

How does this change the probability distribution? Given an integer r, we compute the probability that the algorithm outputs r. First, we must solve a simpler problem. We compute the probability that we obtain n copies of the number $s > 1$ in a given list. As long as it is possible to choose s, we will choose it with probability $2/s$. If we have chosen a number less than s, then the probability of choosing s is 0. We have to choose n s's in a row and then choose a number smaller than s. This occurs with a probability

$$P(n \text{ copies of } s) = \left(\frac{2}{s}\right)^n \left(1 - \frac{2}{s}\right).$$

Let $P^*(r)$ be the probability that the number r is produced in Step 2. Then,

$$P^*(r) = \prod_{p \leq N} \left(\frac{2}{p}\right)^{\alpha_p} \left(1 - \frac{2}{p}\right).$$

Once again, we can write the probability that r is the product of each prime $2p-1$ with the following equation:

$$P(r) = \prod_{p \leq N} \left(\frac{2}{p}\right)^{\alpha_p} \prod_{p \leq N} \left(1 - \frac{2}{p}\right).$$

We may keep the Kalai's definition of r, namely

$$r = \prod_{p \leq N} p^{\alpha_p}.$$

This allows us to write the probability of obtaining r in a more compact manner:

$$P(r) = \frac{1}{r} \prod_{p \leq N} 2^{\alpha_p} \prod_{p \leq N} \left(1 - \frac{2}{p}\right).$$

To simplify this further, we must introduce some new symbols.

Definition. Let the factorization of n be $p_1^{e_1} \dots p_k^{e_k}$. Then, $\Omega(n) = e_1 + \dots e_k$. In other words, $\Omega(n)$ is the sum of the number of prime factors of n, counted with multiplicity. Let $\Omega_0(n)$ be the sum of the number of odd primes factors of n, counted with multiplicity.

The use of the symbol Ω allows us to write our probability more succinctly:

$$P^*(r) = \frac{2^{\Omega_0(r)}}{r} \prod_{2p-1 \leq N} \left(1 - \frac{2}{2p-1}\right).$$

Note that the product of $1 - 2/(2p-1)$ for all $2p - 1 \leq N$ is independent of r. Call this number L_N. We use Ω_0, instead of Ω because only insert copies of 2 during Step 3. The probability has a new equation, namely

$$P^*(r) = \frac{2^{\Omega_0(r)} L_N}{r}.$$

Once again, we output r with probability r/N, leading to a simpler equation for $P(r)$:

$$P(r) = \frac{2^{\Omega_0(r)} L_N}{N}.$$

But, we can do better. We wanted the probability that we outputted r to be proportional to $d(r_1)$, not $2^{\Omega(r)}$. However, $d(r_1)$ is less than $2^{\Omega_0(r)}$. So, instead of outputting r with probability r/N, output r with probability $(rd(r_1))/(2^{\Omega_0(r)}N)$. Here is the probability that we obtain r:

$$P(r) = \frac{2^{\Omega(r)} L_N}{r} \cdot \frac{rd(r_1)}{2^{\Omega(r)}N} = \frac{L_N d(r_1)}{N.}$$

However, a probability can never be greater than 1. So, we must verify the following inequality:

$$rd(r_1) \leq 2^{\Omega_0(r)} N.$$

We already know that $r \leq N$. Therefore, it is sufficient to prove that $d(r_1) \leq 2^{\Omega_0(r)}$. This time, let the prime factorization of r be $2^k p_1^{e_1} \dots p_n^{e_n}$. Every factor of r_1 is also a factor of r. Therefore, $d(r_1) \leq d(r)$, giving us

$$d(r_1) \leq d(r/2^k) = (e_1 + 1) \dots (e_n + 1)$$

and

$$2^{\Omega_0(r)} = 2^{e_1+\ldots+e_n} = 2^{e_1} \ldots 2^{e_n}.$$

For any nonnegative integer m, $m+1 \leq 2^m$. Hence, $d(r_1) \leq 2^{\Omega(r)}$. Therefore,

$$\frac{rd(r_1)}{2^{\Omega(r)}N} \leq 1.$$

5 Proof That Algorithm 4 Works

To verify that Algorithm 4 works, we must prove two statements.

1. If r is a Gaussian norm, then the probability that the algorithm outputs r is proportional to $d(r_1)$. Otherwise, the algorithm does not output r at all.

2. The algorithm runs in polynomial time.

We proved Statement 1 in the previous section and showed that the probability that the algorithm outputs r is

$$P(r) = \frac{L_N G(r)}{N}.$$

In the previous section, we proved that the algorithm generates Gaussian norms with a probability proportional to the number of Gaussian integers that have that norm. Now, we must prove that the algorithm runs in polynomial time. To do this, we set an upper bound on the number of primality tests that the algorithm will require. This upper bound will have the form $O(\log^k N)$ for some positive k. Here is the probability that the algorithm will generate some Gaussian norm after making a list:

$$P(\text{output}) \sim \sum_{r \leq N} \frac{L_N G(r)}{N} = \frac{L_N}{N} \sum_{r \leq N} G(r).$$

By definition, $G(r)$ is the number of Gaussian integers of norm r. Therefore, the sum of $G(r)$ for all $r \leq N$ is equal to the number of Gaussian integers with norm $\leq N$. As N goes to infinity, this sum becomes asymptotic to the area of a the upper quarter of a circle with radius $\sqrt{N}$. Hence, we can make this substitution for large N:

$$P(\text{output}) = \frac{L_N}{N}\left(\frac{\pi N}{4}\right) = \frac{\pi L_N}{4} = O(L_N).$$

We can approximate L_N by noting that its formula is very similar to the formula for M_N^2:

$$L_N = \prod_{2<p\leq N} \left(1 - \frac{2}{p}\right).$$

Clearly, $4M_N^2 \geq L_N$. We have $L_N/M_N^2 \leq 4$. Here is the formula for M_N^2:

$$M_N^2 = \prod_{p \leq N} \left(1 - \frac{1}{p}\right)^2 = \prod_{p \leq N} \left(1 - \frac{2}{p} + \frac{1}{p^2}\right).$$

We will prove that L_N/M_N^2 is bounded below by a positive number:

$$\frac{L_N}{M_N^2} = 4 \prod_{2 < p \leq N} \left(1 - \frac{2}{p}\right) \left(1 - \frac{2}{p} + \frac{1}{p^2}\right)^{-1} = 4 \prod_{2 < p \leq N} \left(\frac{p-2}{p}\right) \left(\frac{p^2}{p^2 - 2p + 1}\right)$$

$$= 4 \prod_{2 < p \leq N} \frac{p^2 - 2p}{p^2 - 2p + 1} = 4 \prod_{2 < p \leq N} \left(1 - \frac{1}{(p+1)^2}\right) \leq 4 \prod_{2 < p \leq N} \left(1 - \frac{1}{p^2}\right) < 4 \prod_{p \text{ prime}} \left(1 - \frac{1}{p^2}\right).$$

For a given prime, p, we can expand $(1 - (1/p^2))^{-1}$:

$$\left(1 - \frac{1}{p^2}\right)^{-1} = 1 + \frac{1}{p^2} + \frac{1}{p^4} + \ldots.$$

Hence,

$$\frac{L_N}{M_N^2} < 4 \prod_{p \text{ prime}} \sum_{i=0}^{\infty} \frac{1}{p^i}.$$

For any integer n, there is exactly one $1/n^2$ term in the sum. Thus, the term on the right is equal to 4 times the sum of the reciprocals of the squares, which we already proved is at most 2. Now, we can plug this into our earlier inequality and multiply both sides by M_N^2:

$$L_N < 8M_N^2.$$

Therefore, $L_N = O(\log^2 N)$. The expected number of lists is on the order of $1/L_N$, which is $O(\log^2 N)$.

At this point, we have to determine how many distinct elements occur in a list. The probability that the list contains least one copy of $2p - 1$ is $2/(2p - 1)$.

$$E(\text{length of list}) = \sum_{\substack{2 < n \leq N \\ n \text{ odd}}} \frac{2}{n} = O(\log N).$$

The expected number of distinct elements in a list is $O(\log N)$. The expected number of lists is $O(\log^2 N)$. Therefore, the expected number of primality tests is $O(\log^3 N)$.

6 Improvement

We can make an improvement to this algorithm by adding one extra step. This improvement reduces the expected time from $O(\log^3 N)$ to $O(\log^2 N)$. Strictly speaking, this improvement

is not necessary for our algorithm. Our goal was simply to create a polynomial time algorithm and $O(\log^3 N)$ is polynomial time. However, a $O(\log N)$ time reduction is nothing to ignore. This section is important because it shows that our algorithm runs as quickly as Kalai's,

Here is the improvement. In the previous algorithm, we simply threw away r if r_3 is not a square. Instead, we divide r by a certain number T and output r/T with a certain probability. Once again, N is our input and M is the largest odd number that is less than or equal to N.

Algorithm 5. *Given a positive integer N, this algorithm produces a random positive integer $r \leq N$, along with its factorization, with a probability proportional to $G(r)$. This algorithm serves the same function as Algorithm 4. However, Algorithm 4 requires $O(\log^3 N)$ primality tests, while this algorithm only requires $O(\log^2 N)$ primality tests.*

1. *Let M be the largest odd number that is less than or equal to N. Create a list $s_1 \geq s_2 \geq \ldots \geq s_k = 1$ of odd numbers, where s_1 is 1 with probability $1/M$ and any odd element of $[3, N]$ with probability $2/M$. If s_i has already been chosen, then let s_{i+1} equal 1 with probability $1/s_i$ and any other odd integer in the interval $[3, s_i]$ with probability $2/s_i$.*

2. *Let r be the product of the prime s_i for each s_i in the list.*

3. *Multiply r by 2 with probability $1/2$. If you just added a 2, repeat this step. Otherwise, go to Step 4.*

4. *Let T be the product of all distinct prime factors of r that are congruent 3 mod 4 and occur an odd number of times in the prime factorization of r. Let $R = r/T$. If $R \leq N$, output R with probability $Rd(R_1)/(2^{\Omega(r)}N)$. If you did not output R, return to Step 1.*

The change in this algorithm is that instead of throwing r away if it is not a Gaussian norm, we divide it by a number T and output r/T with a certain probability. To show that this is acceptable, we must prove three statements.

1. The number r/T is a Gaussian norm, whether or not r is not a Gaussian norm.

2. Our modification outputs every Gaussian norm R with a probability proportional to $d(R_1)$.

3. Our modification improves the expected running time by a factor of $\log N$.

Let p be a prime factor of r_3. Let p^k be the largest power of p that is a factor of r. If k is even, then p^k is also a factor of R because we do not divide by p. If k is odd, then we divide by p. In this case, p^{k-1} is the largest power of p that divides R. If p is a prime factor of r_3, then p occurs an even number of times in the prime factorization of R. Hence, R_3 is a square because every one of its prime factors occurs an even number of times. When R_3 is a square, R is a Gaussian norm.

Let R be a Gaussian norm. We can list all values of r such that $R = r/T$. By definition, R is formed from r by dividing every prime that is congruent to 3 mod 4 that occurs an odd number of times in the prime factorization of r from r. So, T can be any square-free product of prime numbers that are congruent to 3 mod 4 and are less than or equal to N.

Definition. Let $P^*(r)$ is the probability of arriving at r after Step 3. Let $\tilde{P}(r)$ be the probability of outputting r with our new algorithm. Let $r \mapsto R$ mean that R is the largest factor of r that is also a Gaussian norm.

Theorem 19. *Let r be a positive integer whose prime factors are less than or equal to N. Then, $P^*(r) = 2^{\Omega(r)} L_N / r$ for all r.*

In order to output R, we must obtain RT, where T can be any square-free product of prime numbers that are congruent to 3 mod 4 and are less than or equal to N. The probability of outputting R is the sum of $P^*(RT)$ for all T times the probability that the algorithm decides to output R upon selecting it. Hence,

$$\tilde{P}(R) = \frac{Rd(R_1)}{2^{\Omega(R)} N} \sum_{r \mapsto R} P^*(r) = \frac{Rd(R_1)}{2^{\Omega(R)} N} \sum_{r \mapsto R} \frac{2^{\Omega(r)} L_N}{r} = \frac{d(R_1) L_N}{N} \sum_{r \mapsto R} \frac{2^{\Omega(r/R)}}{(r/R)}.$$

Observe that r/R can be any square-free product of primes that are less than or equal to N and are congruent to 3 mod 4. Taking a sum over all possible r/R is equivalent to taking the sum of all of these products. Let $\mathcal{P}(N)$ be the set of all such products. The set of all possible values of r/R is independent of R. Let n be a positive integer and p be a prime. Then, $\Omega(np) = \Omega(n) + 1$. Thus,

$$\sum_{n \in \mathcal{P}(N)} \frac{2^{\Omega(n)}}{n} = \prod_{\substack{p \equiv 3(4) \\ p \leq N}} \left(1 + \frac{2}{p}\right).$$

The product of $1 + (2/p)$ for all prime $p \leq N$ is on the order of $\log^2 N$. However, we are only taking the product over the primes that are congruent to 3 mod 4. Hence, it is on the order of $\log N$. We can summarize all of this with

$$\tilde{P}(R) = \frac{d(R_1) L_N}{N} \prod_{\substack{p \equiv 3(4) \\ p \leq N}} \left(1 + \frac{2}{p}\right).$$

We can asymptotically estimate the product of $1 + (2/p)$ using a method similar to our proof of Mertens' Theorem [10]. First, we will consider all primes less than or equal to N, then we will modify our argument so that we only consider primes that are also congruent to 3 mod 4. We write the logarithm of $1 + (2/p)$ for a prime $p > 2$ using a Taylor Expansion:

$$\log\left(1 + \frac{2}{p}\right) = \frac{2}{p} - \frac{4}{2p^2} + \frac{8}{3p^3} - \ldots = -\sum_{k=1}^{\infty} \frac{1}{k} \left(-\frac{2}{p}\right)^k.$$

Then, we write the log of the product as a sum of log's:

$$\log\left(\prod_{\substack{p \leq N \\ p \equiv 3(4)}} \left(1 + \frac{2}{p}\right)\right) = \sum_{\substack{p \leq N \\ p \equiv 3(4)}} \log\left(1 + \frac{2}{p}\right) = -\sum_{\substack{p \leq N \\ p \equiv 3(4)}} \sum_{k=1}^{\infty} \frac{1}{k} \left(-\frac{2}{p}\right)^k.$$

For each p, there is an infinite sum. To deal with these infinite sums, we use a method similar to the one we used to prove Mertens' Theorem. We separate the first terms of all of those sums from the other terms as follows:

$$-\sum_{\substack{p\leq N\\ p\equiv 3(4)}}\sum_{k=1}^{\infty}\frac{1}{k}\left(-\frac{2}{p}\right)^k = 2\sum_{\substack{p\leq N\\ p\equiv 3(4)}}\frac{1}{p} - \sum_{\substack{p\leq N\\ p\equiv 3(4)}}\sum_{k=2}^{\infty}\frac{1}{k}\left(-\frac{2}{p}\right)^k.$$

Putting an upper bound on the second term is the easy part:

$$\left|\sum_{k=2}^{\infty}\frac{1}{k}\left(-\frac{2}{p}\right)^k\right| \leq \sum_{k=2}^{\infty}\frac{1}{k}\left(\frac{2}{p}\right)^k \leq \frac{1}{2}\sum_{k=2}^{\infty}\left(\frac{2}{p}\right)^k = \frac{2}{p(p-2)}$$

Observe that

$$\left|\sum_{\substack{p\leq N\\ p\equiv 3(4)}}\sum_{k=2}^{\infty}\frac{1}{k}\left(-\frac{2}{p}\right)^k\right| \leq \sum_{\substack{p\leq N\\ p\equiv 3(4)}}\frac{2}{p(p-2)} \leq \sum_{\substack{p\leq N\\ p\equiv 3(4)}}\frac{6}{p^2} < \sum_{n=1}^{\infty}\frac{6}{n^2}.$$

We established earlier that the sum of the reciprocals of the squares is a positive constant. Therefore, the sum written above is bounded above by a constant.

Once again, we need to estimate the sum of the reciprocals of the primes. However, we are only considering primes that are less than or equal to N and congruent to 3 mod 4. A partial summation argument similar to the proof of Mertens' First Theorem gives us

$$\sum_{\substack{p\leq N\\ p\equiv 3(4)}}\frac{1}{p} = \sum_{\substack{p\leq N\\ p\leq 3(4)}}\frac{\log p}{p}\cdot\frac{1}{\log p} = \frac{1}{\log N}\sum_{\substack{p\leq N\\ p\equiv 3(4)}}\frac{\log p}{p} - \int_2^N\left(\sum_{\substack{p\leq t\\ p\equiv 3(4)}}\frac{\log p}{p}\right)d\left(\frac{1}{\log t}\right).$$

The problem is that we no longer want the sum of $(\log p)/p$ for all primes $p \leq N$. We need to also have the additional condition that $p \equiv 3 \bmod 4$. There is a theorem that lets us handle this problem.

Theorem 20. [10] *Let a and m be relatively prime integers with $m > 0$. For any positive real number x,*

$$\sum_{\substack{p\leq x\\ p\equiv a(m)}}\frac{\log p}{p} = \frac{1}{\phi(m)}\log x + O(1),$$

where $\phi(m)$ is the number of numbers that are less than or equal to m and relatively prime to m.

Intuitively, Theorem 20 makes sense. Dirichlet's Theorem states that in the long run every integer that is relatively prime with m is equally likely to occur as the residue of a

prime mod m. Therefore, we would expect that the sum of $(\log p)/p$ would behave similarly. However, we will not prove that statement here. In our case, $a = 3$ and $m = 4$. Observe that $\phi(4) = 2$ because the only positive integers that are less than and relatively prime to 4 are 1 and 3. Hence,

$$\frac{1}{\log N} \sum_{\substack{p \leq N \\ p \equiv 3(4)}} \frac{\log p}{p} = \frac{1}{\log N} \left(\frac{1}{2} \log N + O(1) \right) = \frac{1}{2} + O\left(\frac{1}{\log N} \right).$$

We can also use Theorem 20 to evaluate the integral:

$$\int_2^N \left(\sum_{\substack{p \leq t \\ p \equiv 3(4)}} \frac{\log p}{p} \right) d\left(\frac{1}{\log t} \right) = \int_2^N \frac{1}{t \log^2 t} \left(\frac{1}{2} \log t + O(1) \right) dt$$

There exists some positive constant E such that

$$\int_2^N \frac{1}{t \log^2 t} \left(\frac{1}{2} \log t + O(1) \right) dt \leq \int_2^N \frac{1}{t \log^2 t} \left(\frac{1}{2} \log t + E \right) dt$$

$$= \int_2^N \frac{1}{2t \log t} dt + \int_2^N \frac{E}{t \log^2 t} dt = \frac{1}{2} \log \log N - \frac{1}{2} \log \log 2 + O\left(\frac{1}{\log N} \right).$$

Putting all this together gives us

$$\sum_{\substack{p \leq N \\ p \equiv 3(4)}} \frac{1}{p} = \frac{1}{\log N} \sum_{\substack{p \leq N \\ p \equiv 3(4)}} \frac{\log p}{p} - \int_2^N \left(\sum_{\substack{p \leq t \\ p \equiv 3(4)}} \frac{\log p}{p} \right) d\left(\frac{1}{\log t} \right)$$

$$= \frac{1}{2} + O\left(\frac{1}{\log N} \right) + \frac{1}{2} \log \log N - \frac{1}{2} \log \log 2 + O\left(\frac{1}{\log N} \right).$$

In other words,

$$\sum_{\substack{p \leq N \\ p \equiv 3(4)}} \frac{1}{p} = \frac{1}{2} \log \log N + O(1).$$

We now have

$$\log \left(\prod_{\substack{p \leq N \\ p \equiv 3(4)}} \left(1 + \frac{2}{p} \right) \right) = \log \log N + O(1).$$

Raising e to both sides allows us to estimate our product.

$$\prod_{\substack{p \leq N \\ p \equiv 3(4)}} \left(1 + \frac{2}{p} \right) \sim O(\log N).$$

This equation shows we have decreased the expected number of lists by a factor of $O(\log N)$. Therefore, we expect to make $O(\log N)$, instead of $O(\log^2 N)$ lists. The expected number of primality tests for a given list is $O(\log N)$. Hence, we expect to make $O(\log^2 N)$ primality tests, just like in Kalai's algorithm.

7 Eisenstein Integers

The Gaussian integers are the elements of the ring $\mathbb{Z}[\sqrt{-1}]$. But, what would would happen if we considered $\mathbb{Z}[\sqrt[3]{-1}]$? Then, we would have a new ring. The elements of $\mathbb{Z}[\sqrt[3]{-1}]$ are known as the Eisenstein integers and they form a triangular lattice in the complex plane. Let $\xi_3 = e^{2\pi i/3}$. We can write any Eisenstein integer as a linear combination of ξ_3 and 1. Here is the norm of an arbitrary Eisenstein integer:

$$N(x+y\xi_3) = N\left(x+y\left(-\frac{1}{2}+\frac{\sqrt{3}}{2}i\right)\right) = N\left(\left(x-\frac{y}{2}\right)+\frac{y\sqrt{3}}{2}i\right) = \left(x-\frac{y}{2}\right)^2+\left(\frac{y\sqrt{3}}{2}\right)^2$$

Simplifying this expression gives us our norm:

$$N(x+y\xi_3) = x^2-xy+y^2.$$

When we considered the Gaussian integers, we defined a function D that determined how many ways a given integer could be written as the sum of two squares. Now, we want to know how many ways a given integer can be written in the form x^2-xy+y^2. Once again, we answer this question for the primes and work our way up to an arbitrary integer r. The Gaussian integers had four units, while the Eisenstein integers have six (the powers of $e^{\pi i/3}$). Therefore, the actual answer that we obtain will be one sixth of the actual number of Eisenstein integers with norm r. Think of this as only considering the solutions that occur in the upper right sextant of the complex plane. Once again, units are not a serious issue.

Theorem 21. [4] *Let $d = b^2-4ac$. Then, there exist integral x and y that solve $ax^2+bxy+cy^2 = n$ if and only if $h^2 \equiv d \bmod 4n$ has a solution in h.*

For example, consider x^2+y^2. The discriminant is -4. To determine whether $x^2+y^2 = n$ has any solutions, we would see if we can solve $h^2 \equiv -4 \bmod 4n$. Any possible values of h would be even. Let $h' = h/2$. Then, $h'^2 \equiv -1 \bmod n$. As we saw before, if n is prime, then $h = 2$ or h is equivalent to 1 mod 4.

For the Eisenstein integers, we want to solve $x^2-xy+y^2 = n$. The discriminant is -3. Hence, we want to find all solutions to $h^2 \equiv -3 \bmod 4n$. Since $4n$ is even, h must be odd. Thus, $h^2 \equiv -3 \bmod 4$. We also want to determine when $h^2 \equiv -3 \bmod n$. We need to determine all primes for which -3 is a quadratic residue. This is equivalent to saying that both or neither of -1 and 3 are quadratic residues for a given prime p. 2 is a non-residue mod 3. For primes greater than 3, we must introduce a new theorem.

Theorem 22. (Quadratic Reciprocity Law) *Let p and q be odd primes. If both p and q are congruent to 3 mod 4, then*

$$\left(\frac{p}{q}\right) = -\left(\frac{q}{p}\right).$$

Otherwise,

$$\left(\frac{p}{q}\right) = \left(\frac{q}{p}\right).$$

Using the Quadratic Reciprocity Law, we can determine when 3 is quadratic residue. Simply let $q = 3$. Let p be an odd prime greater than 3. If $p \equiv 3 \bmod 4$, then

$$\left(\frac{3}{p}\right) = -\left(\frac{p}{3}\right).$$

Otherwise,

$$\left(\frac{3}{p}\right) = \left(\frac{p}{3}\right).$$

It is easy to determine whether or not p is a residue mod 3. 1 is a residue and 2 is a non-residue. The product of two residues or two non-residues is a residue, while the product of a residue and a non-residue is a non-residue. If $p \equiv 3 \bmod 4$ and $p \equiv 2 \bmod 3$, then 3 is a residue mod p. 3 is also a residue if $p \equiv 1 \bmod 4$ and $p \equiv 1 \bmod 3$. In any other case, 3 is a non-residue. Thus, whether or not 3 is a residue mod p only depends upon the value of p mod 12.

$$\left(\frac{3}{p}\right) = \begin{cases} 1 & p \equiv 1, 11 \bmod 12, \\ -1 & p \equiv 2, 5, 7 \bmod 12. \end{cases}$$

We already know when -1 is a residue mod p, namely

$$\left(\frac{-1}{p}\right) = \begin{cases} 1 & p \equiv 1, 2 \bmod 4, \\ -1 & p \equiv 3 \bmod 4. \end{cases}$$

-3 is a residue mod p if and only if both or neither of -1 and 3 are residues. We can combine our previous two formulae:

$$\left(\frac{-3}{p}\right) = \begin{cases} 1 & p \equiv 1, 7 \bmod 12, \\ -1 & p \equiv 2, 5, 11 \bmod 12. \end{cases}$$

Finally, note that $|2 + \xi_3| = \sqrt{4 - 2 + 1} = \sqrt{3}$. 3 is an Eisenstein norm. From this information, we can write a new theorem.

Theorem 23. *A prime number is the norm of an Eisenstein integer if and only if it is congruent to 1, 3, or 7 mod 12.*

Recall that with the Gaussian integers 2 was a special case in that there was exactly one Gaussian integer with norm 2. For the Eisenstein integers, 3 is the special case. We can enumerate the number of Eisenstein integers with a given norm with the following definitions and theorem.

Definition. Let $E(r)$ be the number of Eisenstein integers with the norm r.

Definition. Let r be a positive integer. Let r_1 be the largest factor of r which only contains primes that are congruent to 1 mod 3. Let r_2 be the largest factor of r which only contains primes that are congruent to 2 mod 3. With this notation, there exists some nonnegative integer k such that $r = 3^k r_1 r_2$.

Theorem 24. *Let r be a positive integer. Let $E(r)$ be the number of Eisenstein integers with norm r. Then,*

$$E(r) = \begin{cases} d(r_1) & r_2 \text{ is a square,} \\ 0 & \text{otherwise.} \end{cases}$$

Given a prime number p, we need a way of generating the Eisenstein integers with norm p. This is equivalent to finding a pair of nonnegative integers (x, y) such that $x^2 - xy + y^2 = p$. We can use the approach analogous to the one we used for Gaussian integers. In the Gaussian case, we wanted to solve $x^2 + y^2 = p$. So, we let $y = 1$. We found a value of x such that p divides $x^2 + 1$. Then, we let took the gcd of $x + i$ and p.

For the Eisenstein case, we may use a similar process. Let $y = 1$. Now, we want to find a value of x such that p divides $x^2 - x + 1$. However,

$$x^3 + 1 = (x + 1)(x^2 - x + 1).$$

Therefore, p must also divide $x^3 + 1$. We intend to find a solution to the congruence $x^3 \equiv -1 \bmod p$ with $x \not\equiv -1 \bmod p$. Before, we can do that, we have to show there is such a solution. Note that $p \equiv 1 \bmod 3$. Fermat's Little Theorem states that if $a < p$, then $a^{p-1} \equiv 1 \bmod 3$. In our situation, $p - 1$ is a multiple of 3. Therefore, $a^{(p-1)/3}$ is a cube root of 1 mod 3. So, $-a^{(p-1)/3}$ is a cube root of -1. This observation allows to create the following algorithm for cube roots:

Algorithm 6. *Given a prime $p \equiv 1$ mod 3, this algorithm produces a positive integer x that satisfies the congruence $x^3 \equiv -1$ mod p.*

1. *Choose a random value of a in the interval $[2, p - 2]$. Let $x \equiv a^{(p-1)/3}$ mod p. If $x \equiv \pm 1$ mod p, then output $-x$.*

The process for finding cube roots is randomized. But, one in every three elements of $\mathbb{F}_p$ has an order of at most $(p - 1)/3$. [Phrase this without using group theory.] So, the probability that a given value of a will be successful will be 1/3. On average, we will have to choose 3 values of a. Once, you have found the cube root a, simply take the gcd of $a + \xi$ and p.

Here is an example of this algorithm in action. Let $p = 61$. We choose a number a from 2 to 59, say 33. Next, we raise it to our exponent: $33^{(61-1)/3} = 33^{20} \equiv 1 \bmod 43$. Because we obtained 1, we have to choose a new number. Let $a = 6$. This time, $6^{20} \equiv 47 \bmod 43$. Our solution is -47, which is equivalent to 14 mod 61. To check this, we observe that $14^3 + 1 = 2745 = 45 \cdot 61$. Now, we take the gcd of $14 + \xi$ and 61, like so:

$$\gcd(14 + \xi, 61) = \gcd(14 + \xi, 5 - 4\xi) = 5 - 4\xi.$$

We could tell that $5 - 4\xi$ was the gcd because its norm is 41. Thus, $5 \pm 4\xi$ are the two distinct Eisenstein primes with norm 41.

8 Eisenstein Algorithm

In this section, define r_1 as the largest factor of r where every prime factor is congruent to 1 mod 3. Define r_3 as the largest factor of r where every prime factor is congruent to 2 mod 3. The number of Eisenstein integers in the first sextant with a norm r is $d(r_1)$ if r_3 is a square and 0 otherwise. We must modify our Gaussian algorithm accordingly.

In the Gaussian problem, 2 was the only special case. Now, however, we still cannot obtain 2. But, 2 is just another number that is congruent to 2 mod 3 and we must treat it as such. The probability that there are n copies of 2 in the list should be proportional to 2^n. We do not have to treat 2 any differently.

One special case is 3 because it is the only prime that is not congruent to 1 or 2 mod 3. The probability that there are k copies of 3 should be proportional to $1/3^k$. Define $\Omega_1(n)$ as the number of prime factors of n, excluding 3 up to multiplicity. Just as we used Ω_0 in the case of the Gaussian integers, we use Ω_1 here.

Algorithm 7. *Given a positive integer N, this algorithm produces a random positive integer $r \leq N$, along with its factorization, with probability proportional to $E(r)$.*

1. *Let M be the largest odd number less than or equal to N. Create a list $s_1 \geq s_2 \geq \ldots \geq s_k = 1$ of odd numbers, where s_1 is 1 with probability $1/M$, where M is the number of odd numbers less than or equal to N and any odd element of $[3, N]$ with probability $2/M$. If s_i has already been chosen, then let s_{i+1} equal 1 with probability $1/s_i$ and any other odd integer in the interval $[2, s_i]$ with probability $2/s_i$. If $s_i = 3$, discard it and go Step 2.*

2. *Add a 2 to S with probability $1/2$. If you just added a 2, repeat this step. Otherwise, go to Step 3.*

3. *Let r be the product of the prime s_i for each s_i in the list. If $r > N$ or r_3 is not a square, do not output r. Otherwise, output r with probability $rd(r_1)/(2^{\Omega_1(r)}N)$. If you did not output r, return to Step 1.*

We can improve our Eisenstein algorithm just as we did with the Gaussian integers by adding a step that is exactly the same. Simply replace the r_1 that refers to a product of primes that congruent to 1 mod 4 with an r_1 that refers to a product of primes that are congruent to 1 mod 3.

9 Quadratic Integer Rings

In this section, we shall generalize our algorithms for the Gaussian and Eisenstein integers to quadratic integer rings. Unfortunately, unique factorization is impossible for many of these rings, which renders the problem unsolvable *as we have stated it.* We will show that we can factor any ideal into prime ideals, then modify our previous algorithms so that they apply to ideals, rather than numbers.

We form a ring by adjoining $\mathbb{Z}$ with $\sqrt{D}$. In other words, $\mathbb{Z}[\sqrt{D}] = \{a+b\sqrt{D} \mid a, b \in \mathbb{Z}\}$. If $D \equiv 1 \bmod 4$, then this ring is contained in a larger ring, which the next two definitions elaborate [5].

Definition. Let K be a field containing $\mathbb{Q}$. An element $\alpha \in K$ is an algebraic integer if α is the root of some monic polynomial with integral coefficients.

Definition. For a square-free integer d, $\mathcal{O}$ is the largest subring of algebraic integers contained in $\mathbb{Q}[\sqrt{D}]$.

Theorem 25. *For a given square-free d, $\mathcal{O}_D = \mathbb{Z}[\omega]$, with*

$$\omega = \begin{cases} \sqrt{D} & D \equiv 2, 3 \ mod\ 4, \\ (1+\sqrt{D})/2 & D \equiv 1 \ mod\ 4. \end{cases}$$

Proof. Observe that

$$a + b\sqrt{D} = (a-b) + 2b\left(\frac{1+\sqrt{D}}{2}\right).$$

Hence, $\mathbb{Z}[\sqrt{D}] \subseteq \mathcal{O}$. Though we have called $\mathcal{O}$ a quadratic integer ring, we have gotten ahead of ourselves. Namely, we have not proved that $\mathcal{O}$ is actually a ring.

Now, we have to prove that $\mathcal{O}$ is actually a ring. If $D \equiv 2, 3 \bmod 4$, then $\mathcal{O} = \mathbb{Z}[\sqrt{D}]$, making it a ring. For $d \equiv 1 \bmod 4$, we still have $\mathcal{O}$ is a subset of $\mathbb{C}$, which is also a ring. To prove that $\mathcal{O}$ is a ring, we merely have to show that it is closed. For $a, b, c, d \in \mathbb{Z}$,

$$\left(a + b\left(\frac{1+\sqrt{D}}{2}\right)\right)\left(c + d\left(\frac{1+\sqrt{D}}{2}\right)\right) = \left(\left(a + \frac{b}{2}\right) + \frac{b\sqrt{D}}{2}\right)\left(\left(c + \frac{d}{2}\right) + \frac{d\sqrt{D}}{2}\right)$$

$$= \left(ac + \frac{2(ad+bc) + bd(D+1)}{4}\right) + \frac{(ad+bc+bd)\sqrt{D}}{2}$$

$$= \left(ac + \frac{bd(D-1)}{4}\right) + (ad+bc+bd)\left(\frac{1+\sqrt{D}}{2}\right).$$

Because $D \equiv 1 \bmod 4$, $(D-1)/4$ is an integer. Therefore, the answer has the desired form. To finish the proof, we show that $\mathcal{O}$ contains every subring of algebraic integers of $\mathbb{Q}[\sqrt{D}]$.

Let α be an algebraic integer in $\mathbb{Q}[\sqrt{D}]$. Then, $\alpha = a + b\sqrt{D}$, where a and b are both rational numbers. If $b = 0$, then α is a rational number. (The only rational roots of a monic polynomial with integer coefficients are integers.)

If $b \neq 0$, then α is a root of $x^2 - 2ax + (a^2 - b^2D)$, its minimal polynomial. In this case, $2a$ and $a^2 + b^2$ are both integers. Hence, $4a^2$ and $4(a^2 - b^2D)$ are both integers, implying that $4b^2D$ is an integer. Because D is square-free, $2b$ must be an integer as well. Let $x = 2a$ and $y = 2b$. Then, $x^2 - y^2D \equiv 0 \bmod 4$. If $D \equiv 2, 3 \bmod 4$, then x and y are both even. If $D \equiv 1 \bmod 4$, then x and y are both even or both odd. $\square$

This theorem shows us that the Gaussian and Eisenstein integers both form quadratic integer rings. However, those were special cases because they were unique factorization domains. Quadratic integer rings do not generally satisfy this property. Instead, we try to produce an arbitrary ideal, along with its factorization. For each prime p that does not divide D, the ideal Ap represents every multiple of p in $\mathbb{Z}[\sqrt{D}]$.

Definition. For any two ideals I_1 and I_2 that are both contained in a ring R, the product of I_1 and I_2 is the set of all elements of R that can be written as a finite sum of elements of the form ab with $a \in I_1$ and $b \in I_2$.

Theorem 26. *The product of two ideals is also an ideal.*

Proof. Let I_1 and I_2 be ideals contained in a ring R. Let x and y be elements of I_1I_2. By definition, x and y can be written as finite sums of elements of the form ab with $a \in I_1$ and $b \in I_2$. So, $x + y$ also a finite sum of such elements. Hence, I_1I_2 is closed under addition. Let e be the additive identity of R. Then, $e \in I_1, I_2$ because they are both subgroups of R. Therefore, $e \in I_1I_2$. Note that I_1I_2 obeys the associative property because R obeys it. Let $x \in I_1I_2$. There exist $a_1, \ldots, a_n, b_1, \ldots, b_n$ such that $x = a_1b_1 + \ldots + a_nb_n$. Then, $-a_1, \ldots, -a_n \in I_1$ because I_1 is a group. Hence, $-x = (-a_1)b_1 + \ldots + (-a_n)b_n$ is also an element of I_1I_2. So, I_1I_2 is a group under addition.

Observe that I_1I_2 is closed under multiplication. Because I_1I_2 is a subring of R, it obeys all of the other properties of a ring except that it might not contain the multiplicative identity. Let $x = a_1b_1 + \ldots + a_nb_n \in I_1I_2$ and $r \in R$. We can prove $xr \in I_1I_2$. Consider a_ib_ir for some positive integer $i \leq n$. Then, b_ir is an element of I_2 because $b_i \in I_2$ and I_2 is an ideal. So, $a_i(b_ir)$ is an element of I_1I_2 because it is the product of an element of I_1 and an element of I_2. Hence, xr, which is the sum of all a_ib_ir, must also be an element of I_1I_2. Therefore, I_1I_2 is an ideal. □

Definition. [10] Let p be a prime number.

1. If $Ap = P_1P_2$, where P_1 and P_2 are distinct prime ideals, then p is split.
2. If Ap is a prime ideal, then p is inert.
3. If $Ap = P^2$, where P is a prime ideal, then p is ramified.

Theorem 27. *For any prime p and square-free integer D, p is split, inert, or ramified in $\mathcal{O}_D$.*

In the Gaussian integers, 2 is ramified, primes that are congruent to 1 mod 4 are split, and primes that are congruent to 3 mod 4 are inert. In the Eisenstein integers, 3 is ramified, primes that are congruent to 1 mod 3 are split, and primes that are congruent to 2 mod 3 are inert. For a given D, we can determine which primes satisfy which properties with two theorems from [11], presented without proof.

Theorem 28. *Let p be an odd prime. If p divides D, then p is ramified. If D is a quadratic residue mod p, then p is split. Otherwise, p is inert.*

Theorem 29. *If $D \equiv 2, 3 \bmod 4$, then 2 is ramified. If $D \equiv 1 \bmod 8$, then 2 is split. Otherwise, 2 is inert.*

Given these two theorems, we can determine whether a given prime p is ramified, split, or inert mod D. Instead of having r_1 and r_3, we introduce new symbols.

Definition. For two positive integers r and D, r_I is the largest divisor of r where every prime factor is inert in $\mathbb{Z}[\sqrt{D}]$, r_R is the largest divisor of r where every prime factor is ramified in $\mathbb{Z}[\sqrt{D}]$, and r_D is the largest divisor of r where every prime factor is split in $\mathbb{Z}[\sqrt{D}]$.

In order to tell whether a prime number is an inert, ramified, or split, we need an algorithm that determines whether or not a given prime p is a quadratic residue mod D. Before we can introduce this algorithm, we must introduce the Jacobi symbol, which is an extension of the Legendre symbol.

Definition. [3] Let m be an odd number with prime factorization $p_1^{e_1} \dots p_n^{e_n}$ and let a be any integer. Then, the Jacobi symbol is defined as

$$\left(\frac{a}{m}\right) = \prod_{i=1}^{n} \left(\frac{a}{p_i}\right)^{e_i},$$

where the right side is a product of Legendre symbols.

The difference between the Jacobi symbol and the Legendre symbol is that for the Legendre symbol the number on the bottom must be a prime, while for the Jacobi symbol, it does not have to be. However, if m is prime, then the Jacobi and Legendre symbols are equal. The Jacobi symbol is an extension of the Legendre symbol because it has more possible inputs, but it is equal to the Legendre symbol whenever the Legendre symbol is defined.

Here is an algorithm for the Legendre symbol. For a given prime p and integer a that is not a multiple of p, the algorithm determines whether or not $x^2 \equiv a \bmod p$ has an integral solution. Our algorithm calculates a Legendre symbol by calculating the corresponding Jacobi symbol. It takes advantage of the fact that the Quadratic Reciprocity Law applies to Jacobi, as well as Legendre, symbols. We base an algorithm upon continually reducing the size of the numbers involved until they are so small that we can calculate the Legendre symbol directly.

Algorithm 8. *Given a prime p and a positive integer $a < p$, this algorithm determines whether or not a is a quadratic residue mod p.*

1. *Reduce a mod p. Separate the powers of 2 out of the numerator. In other words, suppose $a \equiv 2^k r$ mod p, where r is odd. Then,*

$$\left(\frac{a}{p}\right) = \left(\frac{2}{p}\right)^k \left(\frac{r}{p}\right).$$

Observe that 2 is a quadratic residue mod p if and only if $p \equiv \pm 1$ mod 8 [11].

2. *At this point, $r < p$. Use the Quadratic Reciprocity Law to switch the places of r and p. If the denominator is composite, factor it into primes. If $r = 1$, go to Step 3. Otherwise, return to Step 1.*

3. *Multiply all of the Legendre symbols together.*

The following example illustrates our algorithm in action. Let $p = 307$ and $a = 119$. Then,

$$\left(\frac{119}{307}\right) = -\left(\frac{307}{119}\right) = -\left(\frac{307}{7}\right)\left(\frac{307}{17}\right) = -\left(\frac{6}{7}\right)\left(\frac{1}{17}\right) = -\left(\frac{2}{7}\right)\left(\frac{3}{7}\right)\left(\frac{1}{17}\right)$$
$$= \left(\frac{2}{7}\right)\left(\frac{7}{3}\right)\left(\frac{1}{17}\right) = \left(\frac{2}{7}\right)\left(\frac{1}{3}\right)\left(\frac{1}{17}\right) = 1 \cdot 1 \cdot 1 = 1.$$

Our algorithm shows that 119 is a quadratic residue mod 307. Note that in the above equation, every term refers to a Jacobi symbol, as well as a Legendre symbol.

Definition. Let r be a positive integer with prime factorization $2^k p_1^{e_1} \dots p_n^{e_n}$. We write $v_2(r) = k$. In other words, $v_2(r)$ is the exponent of 2 in the prime factorization of r.

We want to generalize Kalai's Algorithm to a quadratic integer ring $\mathcal{O}_D$. Here is an algorithm that produces the integer $r \leq N$ with a probability proportional to the number of ideals with norm r. We will elaborate on Step 5 shortly.

Algorithm 9. *Given integers N and D with N positive, this algorithm produces a random ideal with norm $\leq N$ in $\mathcal{O}_D$ with uniform distribution.*

1. *Let M be the largest odd number less than or equal to N. Create a list $s_1 \geq s_2 \geq \dots \geq s_k = 1$, where s_1 is 1 with probability $1/M$ and any odd element of $[3, N]$ with probability $2/M$. If s_i has already been chosen, then let s_{i+1} equal 1 with probability $1/s_i$ and any other odd integer in the interval $[3, s_i]$ with probability $2/s_i$.*

2. *Let r be the product of the prime s_i for each s_i in the list.*

3. *If $D \equiv 1 \bmod 8$, multiply by 2 with probability $3/4$. Otherwise, multiply by 2 with probability $1/2$. If you just added a 2, repeat this step. Otherwise, go to Step 4.*

4. *If $r > N$ or r_I is not a square, do not output r and return to Step 1. If you did not return to Step 1, check the value of $p \bmod 8$. If $D \equiv 1 \bmod 8$, output r with probability*

$$\frac{3}{4}\left(\frac{2}{3}\right)^{v_2(r)} \frac{rd(r_S)}{2^{\Omega_0(r)}N}.$$

Otherwise, output r with probability $rd(r_S)/(2^{\Omega_0(r)}N)$. If you did not output r, return to Step 1.

5. *Generate a random ideal with norm r.*

The instances of 3/4 in this algorithm may seem strange. However, there is an argument for them. In the Gaussian integers, 2 was ramified. In the Eisenstein integers, 2 was inert. In those cases, $rd(r_S)/2^{\Omega_0(r)}N$ could not be larger than 1. However, if 2 is split, then this is no longer true. Theorem 29 states that 2 is split in $\mathcal{O}_D$ if and only if $D \equiv 1 \bmod 8$.

Consider the following example. We want to choose a positive integer $r \leq N$ in $\mathcal{O}_D$. However, say $N = 2^k$ for some positive k and $D \equiv 1 \bmod 8$. So, 2 is split in $\mathcal{O}_D$. We choose $r = 2^k$. Then,

$$\frac{rd(r_S)}{2^{\Omega_0(r)}N} = \frac{2^k(k+1)}{2^k} = k+1 > 1.$$

A probability can never be greater than 1. That is why we need a special case for when 2 is split. We can prove that our new algorithm solves this problem. Let 2 be split and let r have the following prime factorization:

$$r = 2^{v_2(r)} \prod_{2<p\leq N} p^{\alpha_p}.$$

We cannot obtain an even number after the first two steps. We can, however, determine the probability of obtaining $r/2^{v_2(r)}$, which is the largest odd factor of r. For a given odd prime p, the conditional probability of obtaining a copy of p, given that we are only looking at odd numbers that are at most p is equal to $2/p$. Therefore, the probability of obtaining $r/2^{v_2(r)}$ is

$$\prod_{2<p\leq N} \left(\frac{2}{p}\right)^{\alpha_p} \left(1 - \frac{1}{p}\right) = \frac{2^{\Omega_0(r)} L_N}{r/2^{v_2(r)}},$$

where

$$L_N = \prod_{2<p\leq N} \left(1 - \frac{1}{p}\right).$$

During Step 3, we want to obtain exactly $v_2(r)$ copies of 2. The probability of multiplying r by 2 is always 2/3. We want to obtain $v_2(r)$ copies of 2, then choose not to add another 2. The probability of doing this is

$$\frac{1}{4}\left(\frac{3}{4}\right)^{v_2(r)}.$$

We now have the probability that we obtain r after Step 3:

$$P^*(r) = \frac{1}{4}\left(\frac{3}{4}\right)^{v_2(r)} \frac{2^{\Omega_0(r)} L_N}{(r/2^{v_2(r)})} = \frac{1}{4}\left(\frac{3}{2}\right)^{v_2(r)} \frac{2^{\Omega_0(r)} L_N}{r}.$$

Now, we have to prove that we output norms with the correct distribution. We want the probability of outputting r to be proportional to $d(r_S)$. We output r with probability

$$P(r) = \frac{3}{4}\left(\frac{2}{3}\right)^{v_2(r)} \frac{rd(r_S)}{2^{\Omega_0(r)}N} P^*(r) = \frac{3}{4}\left(\frac{2}{3}\right)^{v_2(r)} \frac{rd(r_S)}{2^{\Omega_0(r)}N} \cdot \frac{1}{4}\left(\frac{3}{2}\right)^{v_2(r)} \frac{2^{\Omega_0(r)} L_N}{r}$$

$$= \frac{3}{16}\left(\frac{d(r_S)L_N}{N}\right).$$

In Step 4, we output r with a certain probability. By definition, a probability must be at most one. Therefore, we need to prove the following:

$$\frac{3}{4}\left(\frac{2}{3}\right)^{v_2(r)} \frac{rd(r_S)}{2^{\Omega_0(r)}N} \leq 1.$$

First, we note that $r \leq N$. Observe that

$$d(r_S) = (v_2(r)+1)d(r_S/2^{v_2(r)}) \leq (v_2(r)+1)d(r/2^{v_2(r)}) = (v_2(r)+1)\prod_{2<p\leq N}(\alpha_p+1)$$

and

$$2^{\Omega_0(r)} = \prod_{2<p\leq N} 2^{\alpha_p}.$$

For any α_p, $\alpha_p + 1 \leq 2^{\alpha_p}$. But, we cannot make the statement $d(r_S) < 2^{\Omega_0(r)}$ because of the $v_2(r)+1$ term. We can write

$$d(r_S) \leq (v_2(r)+1)2^{\Omega_0(r)}.$$

So far, we have established that $r \leq N$ and $d(r_S) \leq (v_2(r)+1)2^{\Omega_0(r)}$. To finish the proof, we show that

$$\frac{3}{4}\left(\frac{2}{3}\right)^{v_2(r)} \leq \frac{1}{v_2(r)+1} \leq \frac{2^{\Omega_0(r)}N}{rd(r_S)}.$$

Let $n = v_2(r)$. We can prove this inequality through induction on n. For this argument, we will check the inequality for $n = 0$ and 1 by hand. Then, we will prove it for $n > 1$ using induction. For $n = 0$, we have

$$\frac{3}{4}\left(\frac{2}{3}\right)^{0} = \frac{3}{4} \leq 1 = \frac{1}{0+1}.$$

For $n = 1$, we have

$$\frac{3}{4}\left(\frac{2}{3}\right)^{1} = \frac{1}{2} = \frac{1}{1+1}.$$

Now, we use induction. Suppose the inequality is true for some positive n. We want to prove it for $n+1$. We write

$$\frac{3}{4}\left(\frac{2}{3}\right)^{n+1} = \frac{2}{3}\cdot\frac{3}{4}\left(\frac{2}{3}\right)^{n} \leq \frac{2}{3}\left(\frac{1}{n+1}\right) = \frac{2}{3}\left(1+\frac{1}{n+1}\right)\left(\frac{1}{n+2}\right).$$

For any positive integer n, $1 + 1/(n+1) \leq 3/2$. Therefore,

$$\frac{3}{4}\left(\frac{2}{3}\right)^{n+1} \leq \frac{1}{n+2},$$

completing our inductive proof. We have finally proved

$$\frac{3}{4}\left(\frac{2}{3}\right)^{v_2(r)}\frac{rd(r_S)}{2^{\Omega_0(r)}N} \leq 1,$$

finishing our proof.

There is an alternate process one can use if 2 is split. In Step 3, replace 3/4 with $(n+3)/(2n+4)$, where n is the number of copies of 2 that the algorithm has already multiplied. Then, Step 4 outputs r with probability $rd(r_S)/(2^{\Omega_0(r)}N)$, regardless of whether or not 2 is decomposed.

Just as before, we can make this algorithm run more quickly. In this case, we define T as the largest square-free product of inert prime factors of r for which $v_p(r)$ is odd. Then, we divide r/T and output this number instead.

Step 5 requires some exposition. Given an integer r, along with its factorization, how do we generate a random ideal with norm r? First, we solve the analogous problem for p^n, where p is a prime and n is a nonnegative integer. Recall that in the case of the Gaussian integers, we had three possibilities. If $p = 2$, then output $(1+i)^n$. If $p \equiv 3 \bmod 4$, then output $p^{n/2}$. If $p \equiv 1 \bmod 4$, then output $(a+bi)^k(a-bi)^{n-k}$, where k is a random integer in the range $[0, n]$. For a general ring, we do something similar. In the cases below, (x) refers to the ideal generated by x and (x_1, x_2) refers to the ideal generated by x_1 and x_2.

Algorithm 10. *Given a prime p, a positive integer n, and an integer D, this algorithm produces a random ideal in $\mathcal{O}_D$ with norm p^n with uniform distribution, assuming such an ideal exists.*

1. *If p is inert, output $(p)^{n/2}$.*
2. *If p is ramified, there exists some ideal P such that $(p) = P^2$. Specifically, $P = (a + b\sqrt{D})$, where $a^2 + b^2D = p$. To find a and b, solve the congruence $a^2 \equiv p \bmod d$. Output P^n.*
3. *If p is split, then there exists some x such that $x + \sqrt{D}$ has a norm that is a multiple of p. Specifically, $x^2 \equiv -D \bmod p$. Output $(x + \sqrt{D}, p)^k(x - \sqrt{D}, p)^{n-k}$, where k is a random integer in the interval $[0, n]$.*

In order to perform Step 1, we need a square root algorithm.

Algorithm 11. [3] *Given a prime p and a quadratic residue a, this algorithm produces a solution to the congruence $x^2 \equiv a \bmod p$.*

1. *Check the value of $p \bmod 8$. If $p \equiv 3 \bmod 4$, go to Step 2. If $p \equiv 5 \bmod 8$, go to Step 3. If $p \equiv 1 \bmod 8$, go to Step 4.*
2. *Output $a^{(p+1)/4} \bmod p$.*
3. *Let $c \equiv a^{(p+3)/4} \bmod p$. If $c \equiv a \bmod p$, output $a^{(p+3)/8} \bmod p$. Otherwise, output $2^{(p-1)/4}a^{(p+3)/8} \bmod p$.*

4. *Pick random elements of* $[2, p-1]$ *until you find a quadratic non-residue mod* p. *Call this number* d. *Write* $p-1$ *as* $2^s t$, *where* t *is odd. Let* $A = a^t \bmod p$ *and* $D = d^t \bmod p$. *Let* $m = 0$.

5. *For each positive integer* $i < s$, *increase* m *by* 2^i *if* $(AD^m)^{2^{s-1-i}} \equiv -1 \bmod p$. *Output* $a^{(t+1)/2} D^{m/2} \bmod p$.

Proof. Earlier we mentioned this formula about the Legendre symbol:

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

In the algorithm above, a is always a quadratic residue mod p, meaning that the Legendre symbol is equal to a. Hence, we now have a congruence, namely

$$a^{(p-1)/2} \equiv 1 \bmod p.$$

Suppose $p \equiv 3 \bmod 4$. Then, $p + 1 \equiv 0 \bmod 4$, which implies that $(p+1)/4$ is an integer. Let $x \equiv a^{(p+1)/4}$. We can prove that x is a solution by evaluating $x^2 \bmod p$:

$$x^2 \equiv a^{(p+1)/2} \equiv a^{(p-1)/2} a \equiv a \bmod p.$$

Suppose $p \equiv 5 \bmod 8$. Then, $(p+3)/8$ is an integer. Let $c \equiv a^{(p+3)/4} \bmod p$. Note that $a^{(p+3)/4} = a^{(p-1)/4} a$. Earlier, we established that $a^{(p-1)/2} \equiv 1 \bmod p$. Therefore, $a^{(p-1)/4}$ must be a square root of $1 \bmod p$, implying that it is congruent to ± 1. We obtain

$$c \equiv (\pm 1)a \equiv \pm a \bmod p.$$

If $c \equiv a \bmod p$, then $x = a^{(p+3)/8}$ is a solution to the congruence $x^2 \equiv a \bmod p$. Assume $c \equiv -a \bmod p$. If we multiply our current value of x by the square root of $-1 \bmod p$, then we will have a solution.

We can prove that $2^{(p-1)/2} \equiv -1 \bmod p$. By our theorem above, this is equivalent to showing that 2 is a quadratic non-residue mod p. Hardy and Wright state that 2 is a quadratic non-residue for all primes that are congruent to $\pm 3 \bmod 8$ [6]. Hence, 2 is a quadratic non-residue mod p and $x = 2^{(p-1)/2} a^{(p+3)/8}$ is a solution to the congruence $x^2 \equiv a \bmod p$.

Finally, we have to prove that the algorithm works for $p \equiv 1 \bmod 8$. The part of the algorithm that assumes this congruence can actually apply to any odd prime p. However, for other values of p, we have faster methods.

At the beginning of the for loop, $i = 1$. Fermat's Little Theorem states that any number raised to the power of $p - 1$ is congruent to 1 mod p. Thus,

$$(AD^m)^{2^{s-1}} = (a^t d^{mt})^{2^{s-1}} \equiv (ad^m)^{2^{s-1} t} \equiv (ad^m)^{(p-1)/2} \equiv \pm 1 \bmod p.$$

If $(AD^m)^{2^{s-1}} \equiv 1 \bmod p$, then we do not change m. Otherwise, we let $m = 2$. At that poin, $(AD^m)^{2^{s-1}} \equiv 1 \bmod p$. The process continues as follows: whenever we increment i, $(AD^m)^{2^{s-i-1}}$ is either still equivalent to 1 mod p or it is equivalent to -1. If it is equivalent

to $-1 \bmod p$, then we increase m by 2^i and it is equivalent to 1 again. By the time Step 4 is complete, we have a value of m satisfying the congruence

$$(AD^m)^{s-1-(s-1)} \equiv AD^m \equiv 1 \bmod p.$$

By definition, m is a sum of powers of 2 with positive exponent. Therefo re, m is even. Let $x \equiv a^{(t+1)/2} D^{m/2} \bmod p$. To confirm that x is a solution to $x^2 \equiv a \bmod p$, we use our results to evaluate $x^2 \bmod p$:

$$x^2 \equiv a^{t+1} D^m \bmod p.$$

We already showed that $AD^m \equiv 1 \bmod p$. Multiplying both sides of the congruence by A will simplify it:

$$Ax^2 \equiv a^{t+1}(AD^m) \equiv a^{t+1} \bmod p.$$

By definition, $A \equiv a^t \bmod p$. Making this substitution gives us

$$a^t x^2 \equiv a^{t+1} \bmod p.$$

By assumption, a and p are relatively prime. Therefore, we may divide both sides of the congruence by a^t, finishing our proof:

$$x^2 \equiv a \bmod p.$$

For any integer a that is a quadratic residue mod p, our algorithm produces a solution to the congruence $x^2 \equiv a \bmod p$. □

10 Conclusion

Adam Kalai and Eric Bach both solved a seemingly contradictory problem. Their algorithms generate a random integer r less than or equal to a given number N with uniform distribution, along with the factorization of r, without actually factoring any numbers. We have generalized Kalai's algorithm to the Gaussian integers, the Eisenstein integers, and to arbitrary quadratic integer rings. Each time we used the same approach. Choose an integer r with a probability proportional to the number of number of elements of the ring with norm r. Then, we generate a random element of the ring with norm r. The central idea remained the same: generate the primes, then build the "factored" number out of them.

When you run the original versions of the algorithms in this paper, they require $O(\log^3 N)$ primality tests. For the Gaussian, Eisenstein, and quadratic cases, we observed that the algorithms generate a lot of integers which are not actually norms of any elements. Then, it simply discarded those norms. Because most integers are not norms of any of the rings we studied, this proved to be a waste. So, we found a way of turning integers that were not norms into integers that were without interfering with the probability distribution of the norms. This process cut the running time to $O(\log^2 N)$ primality tests.

A primality test requires $O((\log N)^6(2 + \log\log N)^c)$ time for some constant c, so our algorithms, like Kalai's, runs in at most $O((\log N)^8(2 + \log\log N)^c)$ time. Whether or not a

polynomial time factorization algorithm exists is still unknown. If so, then there is a much easier solution to the problems we have solved: Choose a random integer less than or equal to N with the correct distribution, factor it, and generate primes in the specific ring with the correct norm. Though such an algorithm would run in polynomial time, it would not necessarily be as fast as ours.

11 Bibliography

1. Agrawal, M., Kayal, N., Saxena, N., PRIMES is in P, *Annals of Mathematics*, volume 160(2), pp 781 − 793, 2004.

2. Bach, E., How to Generate Factored Random Numbers, *SIAM Journal of Computing*, volume 17, pp 179 − 193, 1988.

3. Crandall, R., Pomerance, C., *Prime Numbers: A Computational Perspective*, Springer-Verlag, New York, NY, 2001.

4. Davenport, H., *The Higher Arithmetic, Eighth Edition*, Cambridge University Press, Cambridge, UK, 2008.

5. Dummit, D., Foote, R., *Abstract Algebra, Third Edition*, Wiley and Sons, Hoboken, NJ, 2004.

6. Hardy, G. H., Wright, E. M., *An Introduction to the Theory of Numbers, Sixth Edition*, Oxford University Press, Oxford, UK, 2008.

7. Kalai, A., Generating Random Factored Numbers, Easily, *Journal of Cryptology*, volume 16(4), pp 287 − 289, 2003.

8. Koblitz, N., *A Course in Number Theory and Cryptography, Second Edition*, Springer-Verlag, New York, NY, 1994.

9. Lenstra, H. W., Pomerance, C., *Primality testing with Gaussian periods*, to appear, 2011.

10. Pollack, P., *Not Always Buried Deep: A Second Course in Elementary Number Theory*, AMS Press, Providence, RI, 2009.

11. Ribenboim, P., *Classical Theory of Algebraic Numbers*, Springer-Verlag, New York, NY, 2001.

12. Schoof, R., Four primality testing algorithms, *Algorithmic Number Theory: Lattices, Number Fields, Curves, and Cryptography*, Cambridge University Press, Cambridge, UK, 2008.