MATHEMATICS OF COMPUTATION
Volume 87, Number 312, July 2018, Pages 2047–2056
http://dx.doi.org/10.1090/mcom/3283
Article electronically published on October 17, 2017

GENERATING RANDOM FACTORED IDEALS IN NUMBER FIELDS

ZACHARY CHARLES

Abstract. We present a randomized polynomial-time algorithm to generate an ideal and its factorization uniformly at random in a given number field. We do this by generating a random integer and its factorization according to the distribution of norms of ideals at most N in the given number field. Using this randomly generated norm, we can produce a random factored ideal in the ring of algebraic integers uniformly at random among ideals with norm up to N, in randomized polynomial time. We also present a variant of this algorithm for generating random factored ideals in function fields.

1. Introduction

We consider a generalization of the following problem: Given an integer $N > 0$, generate an integer in $[1, N]$ uniformly at random, along with its prime factorization, in polynomial time. Since there are currently no known polyonmial-time factorization algorithms, we cannot simply generate an integer and factor it. Instead, we can generate the prime factorization uniformly at random.

In his thesis, Bach gave a randomized polynomial time method to uniformly produce a factored integer in $[\frac{N}{2}, N]$ [2]. Bach's method uses only an expected number of $\log N$ primality tests in $[1, N]$ [3]. Since we can test for primality in polynomial time by the work of Agrawal, Kayal, and Saxena [1], Bach's algorithm runs in randomized polynomial time. In 2003, Kalai gave another method for doing this using a conceptually simpler but slower method. Kalai's algorithm uses an expected number of $O(\log(N)^2)$ primality tests [7]. In 2016, Lebowitz-Lockard and Pomerance gave a variant of Kalai's algorithm to produce random factored elements on $\mathbb{Z}[i]$ with norm at most N [9]. This paper gives a generalization of these algorithms that will produce random factored ideals in $\mathcal{O}_K$ for any number field K. The method given will be polynomial in N and the degree d, where we fix d and assume N tends to infinity.

We first discuss Kalai's algorithm and give a brief analysis. We then use these ideas to generate, in polynomial time, an integer in $[1, N]$ according to the distribution of norms of ideals in $\mathcal{O}_K$. We then discuss how to use this algorithm to produce random factored ideals of $\mathcal{O}_K$ with norm up to N uniformly at random using $O((\log N)^{d^2+d+1})$ primality tests. This algorithm is then modified to generate random factored ideals in function fields using $O((\log N)^{d^2+d+1})$ primality tests. While the algorithm described by Agrawal, Kayal, and Saxena runs in $O(\log^{15/2} N)$, Lenstra and Pomerance later developed a primality testing algorithm with run time $O(\log^6 N)$ [10]. Using the latter algorithm, our algorithms for generating random

Received by the editor December 14, 2016, and, in revised form, February 8, 2017.
2010 *Mathematics Subject Classification.* Primary 11Y16; Secondary 68Q25.

factored ideals in number fields and function fields have expected run times that are $O((\log N)^{d^2+d+7})$.

2. Kalai's algorithm

In his 2003 paper [7], Kalai presents the following algorithm.

Algorithm 1. *Input: A positive integer N.*
Output: A positive integer $r \in [1, N]$ and its prime factorization, produced according to a uniform distribution.

```
1. Generate a sequence $N \geq s_1 \geq s_2 \geq \cdots \geq s_l = 1$, where $s_1 \in \{1, \dots, N\}$ and $s_{i+1} \in \{1, \dots, s_i\}$ are chosen uniformly at random. Stop when $s_j = 1$.
2. Let $r$ be the product of the prime $s_i$'s.
3. If $r \leq N$ output $r$ with probability $\frac{r}{N}$.
4. If $r > N$ or we do not accept $r \leq N$, return to step 1.
```

Let p be a prime number. Then there is a first number s_i produced in the range $\{1, 2, \dots, p\}$. It is chosen uniformly at random, so the probability that $s_i = p$ is $\frac{1}{p}$. Generating e factors p then occurs with probability $\frac{1}{p^e}(1 - \frac{1}{p})$. The probability of outputting a given r is then

$$\begin{aligned}\frac{r}{N} Pr\left[r = \prod_{p \leq n} p^{v_p(r)}\right] &= \frac{r}{N} \prod_{p \leq N} \left(\frac{1}{p}\right)^{v_p(r)} \left(1 - \frac{1}{p}\right)\\ &= \frac{r}{N}\frac{1}{r} \prod_{p \leq n} \left(1 - \frac{1}{p}\right)\\ &= \frac{M_N}{N}.\end{aligned}$$

Here $M_N = \prod_{p \leq N}(1 - 1/p)$. Therefore, if the algorithm terminates, it produces r and its prime factorization with probability $\frac{1}{N}$. Moreover, the probability that Kalai's algorithm terminates after a single round is M_N. We therefore expect M_N^{-1} trials before we output a number.

We now show that the expected number of primality tests is $O(\log(N)^2)$. Note that given some $s_i > 1$, the expected value of s_{i+1} is $\frac{s_i}{2}$. Therefore, a given list $s_1, \dots, s_l$ has expected length $O(\log N)$. We need an expected number of $O(\log N)$ primality tests for every round of the algorithm. Hence, we do an expected number of $O(M_N^{-1} \log N)$ primality tests overall. By Mertens' theorem, $M_N^{-1} = O(\log N)$. Therefore Kalai's algorithm uses $O(\log(N)^2)$ primality tests in expectation before the algorithm terminates.

3. Generating random factored norms

We now give a generalization of Kalai's algorithm that produces a random norm of an ideal in $\mathcal{O}_K$, along with its prime factorization. Let K be a number field of degree d, and N an integer satisfying $d << \log(N)$. The algorithm will use a polynomial number of primality tests. In order to do so, we need knowledge of how the rational primes split. In general, this is computationally efficient for primes not dividing the discriminant of K [11]. If $f(x)$ is the monic irreducible polynomial over $\mathbb{Q}$ determining K, then we can determine how p splits by factoring $f(x)$ modulo $\mathbb{Z}/p\mathbb{Z}$. Factoring polynomials over finite fields can be accomplished in randomized polynomial time [8]. The remaining prime numbers can be factored using a method of Chistov to factor polynomials in $\mathbb{Q}_p$ in polynomial time [4]. We therefore assume we can factor rational primes in K.

For any integer $r > 0$, let $D(r)$ denote the number of ideals in K of norm r. Let $p \in \mathbb{Z}$ be a rational prime with factorization $p\mathcal{O}_K = \prod_{j=1}^{m} \mathfrak{p}_j^{e_j}$. Let $N(\mathfrak{p}_j) = p^{f_j}$. Finding $D(p^e)$ can now be reduced to determining the number of solutions to $\sum_{i=1}^{m} c_i f_i = e$. This is an instance of the subset sum problem. We know that e is $O(\log N)$. Therefore, we can solve the subset sum problem in $O(m \log N)$ operations [5]. Since $\sum_{i=1}^{m} e_i f_i = d$, we know that $m \leq d$. The runtime for this becomes $O(\log N)$ as a result. Since $D(r)$ is multiplicative, we can calculate $D(r)$ for any r relatively efficiently (certainly in polynomial time).

We will generate r and keep r with probability proportional to $\frac{rD(r)}{N}$. In order to guarantee a well-defined probability, we need to bound $D(r)$ by a factor that can be incorporated in to the selection of the factors. We use the following result. Let $\Omega(r)$ denote the number of prime factors of r with multiplicity, and let $\Omega_d(r)$ be the number of prime factors (counting multiplicity) of r that are greater than d.

Proposition 1. $D(r) \leq d^{\Omega(r)}$.

Proof. Note that $D(r)$ is multiplicative. This is because prime ideals in $\mathcal{O}_K$ have norms that are powers of prime numbers [11]. Constructing an ideal of norm $r = \prod_p p^{e_p}$ amounts to constructing an ideal of norm p^{e_p} for each prime $p|r$.

Say $p\mathcal{O}_K = \prod_{i=1}^{m} \mathfrak{p}_i^{e_i}$ for $\mathfrak{p} \subset \mathcal{O}_K$ prime. The norm is multiplicative, and since $N(p\mathcal{O}_K) = p^d$, we must have $\sum_{i=1}^{m} N(\mathfrak{p}_i)^{e_i} = d$. Fix e. Note that $D(p^e)$ will be at its largest when the $\mathfrak{p}$ are all distinct (i.e., $e_i = 1$) and all the $\mathfrak{p}_i$ have norm p. Then $D(p^e)$ will be the number of unordered sets of e elements taken from the d prime ideals. This is bounded above by the number ordered sets, which is given by $d^e = d^{\Omega(r)}$. □

We now present the main algorithm. For simplicity of analysis, we demonstrate the case where d is even and $N \equiv d - 1 \bmod d$. Let $k \in \mathbb{Z}$ be such that $N = kd + (d-1)$. These assumptions are not necessary, but help simplify minor details.

Algorithm 2. *Input: A positive integer N.*
Output: A random integer in $[1, N]$. The integer is generated according to the distribution of norms of ideals in $\mathcal{O}_K$ with norm up to N.

1. *Generate $\lfloor \frac{d}{2} \rfloor$ lists of integers as follows. For $b \in \{1, 3, 5, \ldots, d-1\}$, generate a list $N \geq s_{1,b} \geq s_{2,b} \geq \cdots \geq s_{l_b,b} = 1$. We take $s_{1,b} \in \{1, d+b, 2d+b, \ldots, kd+b\}$, where $s_{1,b} = 1$ with probability $\frac{1}{kd+1}$ and is any other element with probability $\frac{d}{kd+1}$. Take $s_{i+1,b}$ in $\{1, d+b, 2d+b, \ldots, s_{i,b}\}$ as 1 with probability $\frac{1}{s_{i,b}-b+1}$ and any other element with probability $\frac{d}{s_{i,b}-b+1}$.*
2. *Let r be the product of the prime $s_{i,b}$.*
3. *For all primes p between 1 and d, do the following: Multiply r by p with probability $\frac{p-1}{p}$, and continue to multiply by p with this probability until your first failure.*
4. *If $r \leq N$, keep r with probability $M_d(r)\frac{\psi(r)D(r)}{d^{\Omega_d(r)}N}$.*
5. *If you did not keep r, go to step 1.*

If d is odd, then we take $b \in \{1, 3, 5, \ldots, d-2\}$. This way we ensure that we are only picking odd numbers. If N is some other value mod d, say $N = kd + j$ for

$j < d-1$, then we can instead do the following. Let N' be the smallest number above N such that $N' \equiv d-1 \bmod d$. Run steps 1–3 of the algorithm above with N' substituted for N. Then run step 4 by rejecting all $r > N$ instead. For simplicity, we will analyze the algorithm in the case that $N \equiv d-1 \bmod d$.

Let $\overline{n}$ denote the residue of $n \bmod d$. We define $\psi(r)$ as follows. Let $\psi(r)$ have the same factors as r, except that for any prime $p|r, p > d$, replace the factor of p with a factor of $p - \overline{p} + 1$. Note that $\psi(r) \leq r$ and they share the same prime factors, with multiplicity, for $2 \leq p < d$.

We define $M_d(r)$ by:

$$M_d(r) = \prod_{2 \leq p < d} \alpha \left(\frac{1}{p-1} \right)^{v_p(r)},$$

$$\alpha = \binom{d + \lfloor \log N \rfloor - 1}{\lfloor \log N \rfloor}^{-1}.$$

4. Analysis of the algorithm

We wish to show that the probability of accepting r is a well-defined probability. Let $g(r)$ be the product of all prime factors p of r, with multiplicity satisfying $p > d$. Note that since $D(r)$ is multiplicative, we have

$$D(r) = D(g(r)) \prod_{2 \leq p \leq d} D(p^{v_p(r)}).$$

By the discussion in section 3, $D(p^{v_p(r)}) \leq \binom{d+v_p(r)-1}{v_p(r)}$. This follows from the fact that $D(p^{v_p(r)})$ is maximized when all the primes ideals lying above p are all distinct and have norm p. Then $D(p^e)$ equals the number of unordered sets of size e taken from the d prime ideals lying above p.

Using the above we find

$$\begin{aligned}
&M_d(r) \frac{\psi(r) D(r)}{d^{\Omega_d(r)} N} \\
&= \prod_{2 \leq p \leq d} \left[\alpha \left(\frac{1}{p-1} \right)^{v_p(r)} D(p^{v_p(r)}) \right] \frac{\psi(r)}{N} \frac{D(g(r))}{d^{\Omega_d(r)}} \\
&\leq \prod_{2 \leq p \leq d} \left[\alpha D(p^{v_p(r)}) \right] \frac{\psi(r)}{N} \frac{D(g(r))}{d^{\Omega_d(r)}} \\
&\leq \prod_{2 \leq p \leq d} \left[\binom{d + \lfloor \log N \rfloor - 1}{\lfloor \log N \rfloor}^{-1} \binom{d + v_p(r) - 1}{v_p(r)} \right] \frac{\psi(r)}{N} \frac{D(g(r))}{d^{\Omega(g(r))}} \\
&\leq 1.
\end{aligned}$$

Let $p > d$ be an odd prime with $p \equiv \overline{p} \bmod d$. Then we will produce exactly e factors of p with probability given by

$$\left(\frac{d}{p - \overline{p} + 1} \right)^e \left(1 - \frac{d}{p - \overline{p} + 1} \right).$$

Let r be some integer at most N. Recall that $\psi(r)$ is formed from r by replacing all prime factors $p|r$, $p > d$ by $p - \overline{p} + 1$. In particular, r and $\psi(r)$ have the same prime divisors for $p \leq d$. Let $Pr\left[s = \prod_{d < p \leq N} p^{v_p(r)} \right]$ denote the probability that after step 2 we have generated an integer s that is the product of the prime factors of r that are larger than d. The probability can be worked out as follows:

$$\begin{aligned}
Pr\left[s = \prod_{d<p\leq N} p^{v_p(r)}\right] &= \prod_{d<p\leq N} \left(\frac{d}{p-\overline{p}+1}\right)^{v_p(r)} \left(1-\frac{d}{p-\overline{p}+1}\right)\\
&= \prod_{d<p\leq N} \left(\left(\frac{d}{p-\overline{p}+1}\right)^{v_p(r)}\right) \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right)\\
&= d^{\Omega_d(r)} \prod_{d<p\leq N} \left(\frac{1}{p-\overline{p}+1}\right)^{v_p(r)} \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right)\\
&= \frac{d^{\Omega_d(r)}}{\psi(r)} \prod_{2\leq p\leq d} p^{v_p(r)} \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right).
\end{aligned}$$

Note that this last step used the fact that by definition of $\psi(r)$, we have

$$\prod_{d<p\leq N} \left(\frac{1}{p-\overline{p}+1}\right)^{v_p(r)} = \frac{\prod_{2\leq p\leq d} p^{v_p(r)}}{\psi(r)}.$$

Given any $r \leq N$, the probability that we generate r after step 3 is then given by

$$\begin{aligned}
&\frac{d^{\Omega_d(r)}}{\psi(r)} \prod_{2\leq p\leq d} \left[\left(\frac{p-1}{p}\right)^{v_p(r)} \left(1-\frac{p-1}{p}\right)\right] \prod_{2\leq p\leq d} p^{v_p(r)} \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right)\\
&= \frac{d^{\Omega_d(r)}}{\psi(r)} \prod_{2\leq p\leq d} (p-1)^{v_p(r)} \frac{1}{p} \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right).
\end{aligned}$$

Finally, the probability that we accept this r is then given by

$$\begin{aligned}
&M_d(r) \frac{\psi(r)D(r)}{d^{\Omega_d(r)}N} \frac{d^{\Omega_d(r)}}{\psi(r)} \prod_{2\leq p\leq d} (p-1)^{v_p(r)} \frac{1}{p} \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right)\\
&= \frac{D(r)}{N} \prod_{2\leq p\leq d} \alpha \left(\frac{1}{p-1}\right)^{v_p(r)} (p-1)^{v_p(r)} \frac{1}{p} \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right)\\
&= \frac{D(r)}{N} \prod_{2\leq p\leq d} \left(\frac{\alpha}{p}\right) \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right).
\end{aligned}$$

Note that other than $D(r)$, all terms depend only on d and N. Therefore, this generates a number with probability proportional to $D(r)$.

We now show that the algorithm above runs in polynomial time, with polynomial many primality tests and factorizations of rational primes. Summing over all r at most N, the probability that we generate an ideal is

$$\frac{\sum_{r\leq N} D(r)}{N} \prod_{2\leq p<d} \left(\frac{\alpha}{p}\right) \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right).$$

Let $Z_n = \prod_{d<p\leq N} \left(1-\frac{d}{p-\overline{p}+1}\right)$. By the Wiener-Ikehara Tauberian theorem (see [6] for reference), $\sum_{r\leq N} D(r)$ asymptotically approaches $C_K N$, where C_K is

the residue of the Dedekind zeta function of K at 1. Then, asymptotically, the expected number of trials is $O(\alpha^{-d} Z_N^{-1} \prod_{2 \leq p < d} p)$.

By direct computation, we have

$$\begin{aligned}\alpha^{-1} &= \binom{d + \lfloor \log N \rfloor - 1}{\lfloor \log N \rfloor} \\ &= (d - 1 + \lfloor \log N \rfloor)(d - 2 + \lfloor \log N \rfloor) \dots (1 + \lfloor \log N \rfloor) \\ &\leq (2 \log N)^{d-1}.\end{aligned}$$

We have d factors of α^{-1}, so this contributes $O(\log(N)^{d^2-d})$ to the expected number of trials. Note that the term $\prod_{2 \leq p < d} p$ contributes a term that is bounded by a constant d^d, and is therefore $O(1)$ in terms of N. We now wish to find the contribution of the remaining term in the probability calculation above.

Simple estimates show that

$$\begin{aligned}Z_N &\geq \prod_{d < p \leq 2d} \left(1 - \frac{d}{d+1}\right) \prod_{2d < p \leq N} \left(1 - \frac{d}{p - d}\right) \\ &\geq c \prod_{2d < p \leq N} \left(1 - \frac{2d}{p}\right).\end{aligned}$$

Here $c = \prod_{d < p \leq 2d} \left(1 - \frac{d}{d+1}\right) = (d+1)^{-d}$. By standard estimates, such as in [12], we have

$$\prod_{2d < p \leq N} \left(1 - \frac{2d}{p}\right)^{-1} = O(\log(N)^{2d}).$$

Hence we need $O(\log(N)^{d^2+d})$ trials before success. Since any given list has expected length $O(\log(N))$, this leads to $O(\log(N)^{d^2+d+1})$ primality tests.

We now use the above algorithm to generate random ideals of $\mathcal{O}_K$, uniformly at random among all ideals with norm up to N. As previously stated, we assume that we for any rational prime p, we can find the factorization $p\mathcal{O}_K = \prod_{i=1}^m \mathfrak{p}_i^{e_i}$ with $N(\mathfrak{p}_i) = p^{f_i}$. To find an ideal of norm p^e, it suffices to solve, as previously discussed, the following subset sum problem:

$$\sum_{i=1}^m c_i f_i = e.$$

Any solution to this corresponds to the ideal $\prod_{i=1}^m \mathfrak{p}^{c_i}$. Since $e \leq \log N$ and $m \leq d$, we can clearly find all solutions in $O(\log^d N)$ operations [5]. While this is not optimal, even the naive approach is dwarfed by the effort needed to generate the norm. After we find all solutions, we can then choose uniformly at random one of the solutions, which will give us one of the ideals of the desired norm. Therefore, the algorithm runs in a number of operations that is $O(\log^{d^2+d+1} N)$ primality tests and factorizations of $p\mathcal{O}_K$. The primality testing is the dominant part of this in terms of runtime. Since we can perform primality tests in $O(\log^6 N)$, this gives us a runtime that is $O(\log^{d^2+d+7} N)$ overall.

5. Function fields

The analogy between number fields and function fields suggests that there should be an analogous algorithm for function fields. In particular, there are well-known randomized polynomial-time algorithms for factoring over $\mathbb{F}_q[t]$. To generate a factored random polynomial in $\mathbb{F}_q[t]$, we could simply generate one at random and then factor it in randomized polynomial time. Much more elegant ways exist that generate the factorization at random, the same idea used by the methods above.

Therefore, we would expect the ability to translate the algorithm above to arbitrary function fields. Fix a function field K of degree d and $N > 0$. We want to generate a random ideal $I \subset \mathcal{O}_K$ with norm $r(t) \in \mathbb{F}_q[t]$ of degree at most N, along with the factorization of I. We will use the fact that we can perform primality testing over $\mathbb{F}_q[t]$ in polynomial time. We will also assume that for any irreducible polynomial $f(t) \in \mathbb{F}_q[t]$, we can factor $f(t)\mathcal{O}_K$ in polynomial time. Since $\mathcal{O}_K$ is a Dedekind domain, this holds for all $f(t)$ not dividing the discriminant, so there are only finitely many $f(t)$ that need to be factored as a one-time operation.

As in number fields, the main obstacle is generating the norm $g(t)$ of I with probability proportional to the number of ideals with this norm. Once we can do this, then we can use our ability to factor $g(t)$ over $\mathcal{O}_K$ and solve the corresponding subset sum problem to generate an ideal of $\mathcal{O}_K$ uniformly at random with its factorization.

Let $g \in \mathbb{F}_q[t]$. Then we can consider g to be a number written base q by looking at its coefficients. Let $n(g)$ denote this number. Let $D(g)$ denote the number of ideals in $\mathcal{O}_K$ with norm g. For any number $n \in \mathbb{Z}$, using its q-ary expansion, we can form a corresponding element $\beta(n) \in \mathbb{F}_q[t]$.

We present the following algorithm for generating a norm $g \in \mathbb{F}_q[t]$ with probabity proportional to $D(g)$. It is virtually identical to the algorithm above, except that our concept of primality of a number p is replaced by primality of the corresponding element $\beta(p) \in \mathbb{F}_q[t]$.

Algorithm 3. *Input: A positive integer N.*
Output: A random element in $F_q[t]$ with degree at most N. The polynomial is generated according to the distribution of norms of ideals in $\mathcal{O}_K$ with degree of their norm at most N.

1. *Generate $\lfloor \frac{d}{2} \rfloor$ lists of integers as follows. For $b \in \{1, 3, 5, \ldots, d-1\}$, generate a list $q^{N+1} > s_{1,b} \geq s_{2,b} \geq \cdots \geq s_{l_b,b} = 1$. We take $s_{1,b} \in \{1, d+b, 2d+b, \ldots, kd+b\}$, where $s_{1,b} = 1$ with probability $\frac{1}{kd+1}$ and is any other element with probability $\frac{d}{kd+1}$. Take $s_{i+1,b}$ in $\{1, d+b, 2d+b, \ldots, s_{i,b}\}$ as 1 with probability $\frac{1}{s_{i,b}-b+1}$ and any other element with probability $\frac{d}{s_{i,b}-b+1}$.*
2. *Let r be the product of the $s_{i,b}$ such that $\beta(s_{i,b})$ is prime.*
3. *For all integers p between 1 and d such that $\beta(p)$ is prime, do the following: Multiply r by p with probability $\frac{p-1}{p}$, and continue to multiply by p with this probability until your first failure.*
4. *If $r < q^{N+1}$, return $\beta(r)$ with probability $M_d(r)\frac{\psi(r)D(\beta(r))}{d^{\Omega_d(\alpha(r))}q^N}$.*
5. *If you did not return $\beta(r)$, go to step 1.*

Let $\overline{n}$ denote the residue of $n \bmod d$. We define $\psi(r)$ as follows. Factor $\beta(r)$ in to primes g_i. For each g_i such that $n(g_i) > d$, replace the factor of $n(g_i)$ with

$n(g_i) - \overline{n(g_i)} + 1$. Note that $\psi(r) \leq r$ and $\beta(r), \beta(\psi(r))$ share the same factors g_i such that $2 \leq n(g_i) \leq d$.

We define $M_d(r)$ by

$$M_d(r) = \prod_{\substack{f \in \mathbb{F}_q[t],\ f \text{ is prime} \\ 2 \leq n(f) < d}} \alpha\left(\frac{1}{n(f)-1}\right)^{v_{n(f)}(r)},$$

$$\alpha = \binom{d + N\lfloor \log q \rfloor - 1}{N\lfloor \log q \rfloor}^{-1}.$$

An almost identical argument to the one above shows that this produces r with probability proportional to $D(\beta(r))$. Moreover, the algorithm produces a norm with probability

$$\frac{\sum_{g \in \mathbb{F}_q[t], \deg(g) \leq N} D(g)}{q^N} \prod_{\substack{g \in \mathbb{F}_q[t] \\ 2 \leq n(g) \leq d}} \frac{\alpha}{n(g)} \prod_{\substack{g \in \mathbb{F}_q[t] \\ d < n(g) < q^{N+1}}} \left(1 - \frac{d}{n(g) - \overline{n(g)} + 1}\right).$$

We first want to analyze $\sum_{g \in \mathbb{F}_q, \deg(g) \leq N} D(g)$. Let $\mathcal{D}_K^+$ denote the set of effective divisors of V, where K is the function field of the projective variety V. Note that the zeta function for K can be written as

$$\zeta_K(s) = \sum_{D \in \mathcal{D}_K^+} \frac{1}{q^{\deg(D)s}}.$$

If we let a_n denote the number of $g \in \mathbb{F}_q[t]$ of degree n, then this becomes

$$\zeta_K(s) = \sum_{n \geq 0} \frac{a_n}{q^{ns}}.$$

Since $\zeta_K(s)$ converges absolutely for $\mathrm{Re}(s) > 1$ and has a simple pole at $s = 1$, the analogue of the Wiener-Ikehara Tauberian theorem for function fields implies that

$$\sum_{n \leq N} a_n = \Theta(q^N).$$

Let Z_N be given by

$$Z_N = \prod_{\substack{d < n(g) < q^{N+1} \\ g \text{ prime}}} \left(1 - \frac{d}{n(g) - \overline{n(g)} + 1}\right).$$

Note that the expected number of trials of the algorithm is

$$O(\alpha^{-d} Z_N^{-1} \prod_{\substack{2 \leq n(g) \leq d \\ g \text{ prime}}} n(g)).$$

By an analogous argument to the above, we have at most d factors of α which contribute at most $(2N\log(q))^{d-1}$ to the expected number of trials, while the last product in the estimate contributes an amount bounded by d^d. Therefore, it suffices to bound Z_N^{-1}. Using a version of the prime number theorem for $\mathbb{F}_q[x]$, one can prove in an analogous way to the proof above that Z_N^{-1} contributes $O((N\log(q))^{2d})$ iterations in expected value.

Therefore, we require $O((N \log q)^{d^2+d})$ trials before success in expectation. Each of the $\frac{d}{2}$ lists has expected length $O(N \log q)$, so we require $O((N \log q)^{d^2+d+1})$ primality tests. Note that this gives us a randomized algorithm that is polynomial in the logarithm of the size of the input (as there are q^N possible norms of the ideal to generate). Since primality testing can be performed in time that is $O((N \log q)^6)$, this gives us an overall runtime that is $O((N \log q)^{d^2+d+7})$.

6. Further work

The algorithm above uses similar ideas to Kalai's algorithm. Bach's algorithm for generating factored integers runs in fewer primality tests, and so one could ask whether the ideas of Bach could be adapted to this setting in order to reduce the number of primality tests required. In general, one could determine ways to make the above algorithm run faster, in particular by reducing the number of primality tests required.

The algorithms above work for a fixed number or function field. In particular, their runtime is polynomial treating the degree of the field extension as constant. It remains an open question whether there is an algorithm for generating random factored ideals that runs in polynomial time, irrespective of the degree of the field. The methods above would have to be altered significantly to do so, due to their exponential dependence on the degree d.

There are other generalizations of this problem that could be considered. Due to the natural way in which principal ideals arise in number fields, one could ask for a variant of this algorithm that generates principal ideals uniformly at random. Clearly we could use the above algorithm to generate an ideal uniformly at random, and then only accept it if it is principal. Since this occurs with probability $1/h$, where h is the class number, this would result in a polynomial-time algorithm provided we had a polynomial-time way to recognize principal ideals. Unfortunately, this is a difficult question in arbitrary number fields, as there are no known polynomial-time algorithms to detect whether an ideal in an arbitrary number field is principal.

References

[1] M. Agrawal, N. Kayal, and N. Saxena, *PRIMES is in P*, Ann. of Math. (2) **160** (2004), no. 2, 781–793, DOI 10.4007/annals.2004.160.781. MR2123939

[2] E. Bach, *Analytic Methods in the Analysis and Design of Number-Theoretic Algorithms*, ACM Distinguished Dissertations, MIT Press, Cambridge, MA, 1985. MR807772

[3] E. Bach, *How to generate factored random numbers*, SIAM J. Comput. **17** (1988), no. 2, 179–193, DOI 10.1137/0217012. Special issue on cryptography. MR935336

[4] A. L. Chistov, *Efficient factoring polynomials over local fields and its applications*, Proceedings of the International Congress of Mathematicians, Vol. I, II (Kyoto, 1990), Math. Soc. Japan, Tokyo, 1991, pp. 1509–1519. MR1159333

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed., MIT Press, Cambridge, MA; McGraw-Hill Book Co., Boston, MA, 2001. MR1848805

[6] S. Ikehara, *An extension of landau's theorem in the analytical theory of numbers*, Journal of Mathematics and Physics **10** (1931), no. 1-4, 1–12.

[7] A. Kalai, *Generating random factored numbers, easily*, J. Cryptology **16** (2003), no. 4, 287–289, DOI 10.1007/s00145-003-0051-5. MR2002046

[8] D. E. Knuth, *The Art of Computer Programming. Vol. 2: Seminumerical Algorithms*, 3rd edition, Addison-Wesley Publishing Co., Reading, Mass.-London-Don Mills, Ont, 1997. Original edition review: MR0286318

[9] N. Lebowitz-Lockard and C. Pomerance, *Generating random factored Gaussian integers, easily*, Math. Comp. **85** (2016), no. 297, 503–516, DOI 10.1090/mcom/3000. MR3404459

[10] H. W. Lenstra, Jr., *Primality testing with gaussian periods*, Proceedings of the 22nd Conference Kanpur on Foundations of Software Technology and Theoretical Computer Science (London, UK), FST TCS '02, Springer-Verlag, 2002.

[11] J. S. Milne, *Algebraic number theory*, 2013, Available at www.jmilne.org/math/, p. 161.

[12] J. B. Rosser and L. Schoenfeld, *Approximate formulas for some functions of prime numbers*, Illinois J. Math. **6** (1962), 64–94. MR0137689

Department of Mathematics, University of Wisconsin-Madison, Madison, Wisconsin 53706

E-mail address: zcharles@math.wisc.edu