Integer Factorization

A comparison of two modern techniques:
The general number field sieve and
Shor's quantum algorithm

April 19, 2015

Prepared in fulfillment of the final course project for
CSI4105 - Design and Analysis of Algorithms II
by

Jafar Jahed
Saad Basif
Kevin Johnson

The University of Ottawa
Faculty of Engineering

Contents

For the readers convenience, the letter N will always be used to signify the integer desired to be factored. For brevity, a theorems and definitions section (5) has been included. If a term appears in *italics*, then the reader may consult section 5 for further information.

1 Factoring

Integer factorization is the decomposition of a composite number into a product of prime powers. This means that the number can be represented as the multiplication of prime numbers to certain powers. This factorization is also unique. Given a large integer, finding this factorization into prime powers in a tractable manner (in polynomial time) has not been figured out. The decision form of this problem is does the integer N have a prime number less than M, whereas the regular problem is in function form (finding the factors). That means there is a single output for each input to the problem, but it is more complicated than a yes or no, in this case the output is the set of factors for N. That sets it apart from NP-Complete problems, putting the problem in a different complexity class. The problem is in the FNP complexity class. It is not known, however whether the problem is in FP (meaning it is polynomial) or not, similar to how we are not sure if $P = NP$-complete. The problem is not known to be a polynomial runtime functional problem, which is why it is not in FP.

This paper covers information about the problem, its applications, two algorithms for the problem. The paper will explain why this problem is important in the computing field, primarily in cryptography. The algorithms covered are general number field sieve and Shors algorithm. Analysis will be performed on the general number field sieve, with respect to time and runtime complexity. With Shors algorithm, because it requires quantum computing, instead only a theoretical analysis will be done.

1.1 Brute Force Attempts

To understand the runtime of the brute force algorithm, we need to explain the algorithm first. In brute force, we need to check all possible numbers that are could be factors, so that would be all the numbers from 2 to $\sqrt{N}$. We stop at $\sqrt{N}$ because N can only have one factor that is larger than $\sqrt{N}$. Any two numbers larger than $\sqrt{N}$ multiplied together would produce a number larger than N. We would repeat that until the number resulting by dividing by the factor is 1, meaning we have found all the numbers factors.

The complexity becomes exponential when we go back to the idea that the complexity relies on the size of the input. So the complexity does not depend on the number N itself we mentioned earlier. The size of N is really the number of bits it takes to represent N, or $\lg(N) = k$ bits. The number that we can represent with k bits is 2^k. It is evident by looking at that, that the number we can represent when we increase k rises exponentially. Additionally, the number of factors we have is also the same as the number of bits, because in the worse case, all our factors can be 2, and so the factorization of our number would just look like 2^k. So if for the moment, we think of N being the size of the input, that is to say, the number we are factoring takes N bits to represent, our complexity is $O(2^n)$.

1.2 Some Applications

Integer factorization is the basis of most modern forms of cryptography and therefore is of huge importance in cyber security. The concept is used in encryption techniques such as RSA, to come up with secure numbers. Part of the process is multiplying two very large prime numbers together to come up with your RSA key. The whole idea works on the assumption that it would be very difficult to factor an incredibly large number (in the order of 1000 bits) if it only had two prime factors, because the problem is non polynomial (to our knowledge). Another stipulation to the system is that the factors have to both be large, one cant be very small, and the other very large, or the brute force algorithm may be able to crack it in a reasonable amount of time. So on that basis, if someone were to alternatively find an efficient way to do the factorization, the setup would no longer be secure. This would be a huge issue, because many computing systems rely on that property for security, not just RSA, so they would all become insecure unless an alternate cryptographic system built.

Consequently, if the solution was in FP, it could also help lead to proving $P = NP$ [1]. Another application of integer factorization is in calculating Free Fourier Transforms (referred to as FFT). The FFT is an algorithm used to calculate the Discrete Fourier Transform (DFT). The classic algorithm for calculating the DFT is polynomial time, but quite inefficient, the FFT algorithms can calculate the same results but much faster using integer factorization. The DFT is used in the field of image processing to perform spectral analysis on images [2], and image convolution [3]. The aforementioned applications are basically used for extracting data from images or for image manipulation. DFT has applications in other fields as well such as data compression [4].

2 The General Number Field Sieve

Let N be a positive composite integer that is not a prime power. The general number field sieve is a result of 30 years of evolution of same general idea: finding integers x, y such that $x \not\equiv y \bmod N$ but $x^2 \equiv y^2 \bmod N$. Once these integers are found, we know N divides $(x-y)(x+y)$. Since N is not prime by assumption, it is likely (about half the time) that one of $\gcd(x-y, N)$ or $\gcd(x+y, N)$ is non-trivial. The first algorithm to leverage this idea was designed by John Dixon at Carleton University [ref dixon]. The key insight was that producing congruences of squares modulo N could be done using a factor base of primes. The technique seeks integers with squares mod N are *smooth* over those primes. This simple idea has been extrapolated into what is now known to the fastest factoring algorithm ever implemented, the number field sieve.

2.1 Dixon's Method

Let $F = \{2, 3, ..., p_t\}$ be a set of consecutive prime numbers. An integer z is F-smooth if $z = 2^{e_1} \cdots p_t^{e_t}$ where $e_1, ..., e_t \in \mathbb{Z}^+$. Dixons big idea was to collect a set of integers $A = \{x \in \mathbb{Z} \mid x^2 \text{ is } F\text{-smooth mod } N\}$ with $s = \mid A \mid$. Then write these squares in the form $x_i^2 = 2^{e_{1i}} \cdots p_t^{e_{ti}} \bmod N$ for $i = 1, ..., s$ and place them in a matrix as

$$\begin{pmatrix} e_{11} \bmod 2 & e_{21} \bmod 2 & \cdots & e_{t1} \bmod 2 \\ e_{12} \bmod 2 & e_{22} \bmod 2 & \cdots & e_{t2} \bmod 2 \\ \vdots & \vdots & \ddots & \vdots \\ e_{1s} \bmod 2 & e_{2s} \bmod 2 & \cdots & e_{ts} \bmod 2 \end{pmatrix}$$

If $s > t$ then there are more rows than colums. Then there exists a subset $U \subset A$ whos corresponding rows are linearly dependent, which means that they sum to the zero vector. This means that

$$(\prod_{x_i \in U} x_i)^2 \equiv \prod_{x_i \in U} x_i^2 \equiv 2^{2f_1} \cdots p_t^{2f_t} \equiv (2^{f_1} \cdots p_t^{f_t})^2 \bmod N \text{ where } f_j = \frac{\sum_{x_i \in U} e_{ji}}{2}$$

Let $x = \prod_{x_i \in U} x_i$ and $y = 2^{f_1} \cdots p_t^{f_t}$, then $x^2 \equiv y^2 \bmod N$.

2.2 Quadratic Sieve

The bottleneck of Dixon's Mathod is constructing the set $A = \{x \in \mathbb{Z} \mid x^2 \text{ is } F\text{-smooth mod } N\}$. Initially, this was done by randomly choosing integers less than N and squaring them. This was radically improved in 1981 by Carl Pomerance with his idea to instead use the factor base to find smooth integers mod N. His idea was the combination of two key observations: When trying to find smooth squares mod N, if x is an integer in the range $\sqrt{N} < x < \sqrt{N} + N^{\delta}$ for some small positive delta, then $x^2 - N \approx N^{1/2+\delta}$. This means that if we let $f(x) = x^2 - N$, then as x takes values starting from $\sqrt{N}$, then the values of $f(x)$ are small squares mod N. Secondly, if x_1, x_2 are a roots of $f(x) = x^2 - N$, then so are $x_1 + kp, x_2 + lp$ for every integer k, l. Keeping these two facts in mind the idea of the quadratic sieve is to initialize two arrays

$$R = [\left\lceil\sqrt{N}\right\rceil, \left\lceil\sqrt{N}\right\rceil + 1, ..., \left\lceil\sqrt{N}\right\rceil + N^{\delta}]$$

$$\text{LogR} = [\log(f(\left\lceil\sqrt{N}\right\rceil) \bmod N), \log(f(\left\lceil\sqrt{N}\right\rceil + 1) \bmod N), ..., \log(f(\left\lceil\sqrt{N}\right\rceil + N^{\delta}) \bmod N)]$$

where LogR$[i] = \log(f(R[i]))$ for all i. Then for each p_j in the factor base $F = \{2, 3, ..., p_t\}$, iterate through R checking if p divides $f(R[i]) \bmod N$, if it does subtract $\log(p_j)$ from LogR$[i + kp_j]$ for all $0 \leq k < \left\lfloor \frac{N}{p_j} \right\rfloor$ and skip to the next prime p_{j+1}. Once this is done for all primes in the factor base, scan through the array LogR and look for values which are approximately 0. These values are F-smooth mod N. This is because if a value satisfies $f(R[i]) = 2^{e_1} \cdots p_t^{e_t}$ then

$$\log(f(R[i])) - e_1 \log(2) - ... - e_t \log(p_t) = \log(\frac{f(R[i])}{2^{e_1} \cdots p_t^{e_t}}) = \log(1) = 0$$

Once these F-smooth integers $f(R[i])$ are found, they are placed in a matrix the same way as in Dixon's algorithm.

2.3 The Number Field Sieve

Followign the success of the Quadratic Sieve, researchers experiemented with various elements of the algorithm. One insight was that the polynomial $f(x) = x^2 - N$ used in the quadratic sieve doesn't necessarily have to be quadratic! The other key insight was that there are *rings* which have the similar notions of smoothness and may have "more" smooth elements that the regular integers. The following section describes such a ring. If the reader is confortable loosing some of the intuition behind the algorithm, they may choose to skip to section 2.3.2

2.3.1 Algebraic Intuition

Let $f(x)$ be a monic irreducible polynomial of degree d in $\mathbb{Z}[x]$. By the *fundamental theorem of algebra*, $f(x)$ has exactly d complex roots $\theta_1, ..., \theta_d$. Choose one of these roots θ, then one may consider the set $\mathbb{Z}[\theta]$ of all $\mathbb{Z}$-linear combinations of the elements $\{1, \theta, \theta^2, ..., \theta^{d-1}\}$. Given $g = a_0 + a_1\theta + ... + a_n\theta^n$ and $h = b_0 + b_1\theta + ... + b_m\theta^m$ in $\mathbb{Z}[\theta]$ with $m < n < d$, let $G(x), H(x) \in \mathbb{Z}[x]$ such that $G(\theta) = g, H(\theta) = h$. Note that we may write $G(x)H(x) = C(x)f(x) + D(x)$ where $C(x), D(x) \in Z[x]$ and $\deg(f(x)) > \deg(D(x))$. Therefore $G(\theta)H(\theta) = C(\theta)f(\theta) + D(\theta) = 0 + D(\theta) = D(\theta)$. If we let $d = D(\theta)$, then with the operations

$$\begin{aligned} g + h &= (a_0 + b_0) + (a_1 + b_1)\theta + \cdots + (a_n + b_n)\theta^n \\ g \cdot h &= d \end{aligned}$$

the set $\mathbb{Z}[\theta]$ forms a ring. This ring is actually a subring of the *field* $\mathbb{Q}(\theta)$ which is isomorphic to $\mathbb{Q}[x]/(f)$ and thus the two operations decribed above are inherited from the field $\mathbb{Q}(\theta)$. Besides making the following definition of a "norm" function on $\mathbb{Q}(\theta)$, this fact will generally be ignored for the purposes of this report. Recall that f has precisely d roots $\theta_1, ..., \theta_d$ over $\mathbb{C}$ and we chose one of them and called it θ. For $i = 1, ..., d$ let $\sigma_i : \mathbb{Q}(\theta) \longrightarrow \mathbb{Q}(\theta_i)$ be defined by the action; $\sigma_i(\mathbb{Q}) = \mathbb{Q}$ and $\sigma_i(\theta) = \theta_i$. It turns out that these σ_i are precisely all the embedding of $\mathbb{Q}(\theta)$ into $\mathbb{C}$, but again this can be ignored. Define the "norm" function N on $\mathbb{Q}(\theta)$ to be a set map $\mathbb{Q}(\theta) \xrightarrow{N} \mathbb{C}$ with action $\alpha \mapsto \sigma_1(\alpha)\sigma_2(\alpha), ..., \sigma(\alpha)$. It is a standard result from algebra that N is a multiplicative function which maps $Q(\theta)$ to $\mathbb{Z}$ and thus likewise for $\mathbb{Z}[\theta]$. This will norm function N will be a very important link between the two rings $\mathbb{Z}$ and $\mathbb{Z}[\theta]$.

Since $\mathbb{Z}[\theta]$ is a ring, we may consider *ideals* in $Z[\theta]$. Furthermore, since $\mathbb{Z}[\theta]$ is in fact a *Dedekind domain*, an ideal I of $\mathbb{Z}[\theta]$ may be uniquely factored, up to order, as the product $I = \mathfrak{p}_1^{e_1}\mathfrak{p}_2^{e_2} \cdots \mathfrak{p}_t^{e_t}$ of *prime ideals* $\mathfrak{p}_1, \mathfrak{p}_2, ..., \mathfrak{p}_t \in \mathbb{Z}[\theta]$. Two more facts are necissary before it will become apparent why all this algebra is required. We may also define the norm N on an ideal of $I \in \mathbb{Z}[\theta]$ as $N(I) = [\mathbb{Z}[\theta] : I]$. That is, the number of cosets of I in $\mathbb{Z}[\theta]$. It turns out that this definition agrees with our earlier definition in the sense that $\mid N(\alpha) \mid = N(\langle\alpha\rangle)$. Secondly, if $\mathfrak{p}$ is an ideal of $\mathbb{Z}[\theta]$ such that $N(\mathfrak{p}) = p$ for some prime p, then $\mathfrak{p}$ is a prime ideal. Conversly, if $\mathfrak{p}$ is a prime ideal of $\mathbb{Z}[\theta]$, then $N(\mathfrak{p}) = p^e$ for prime $p \in \mathbb{Z}$ and possitve exponent e. The point is of all this is, for $\alpha \in \mathbb{Z}[\theta]$ with factorization $\alpha = \mathfrak{p}_1^{e_1}\mathfrak{p}_2^{e_2} \cdots \mathfrak{p}_t^{e_t}$, then

$$\begin{aligned}
\left\|N(\alpha)\right\| &= N(\langle\alpha\rangle) \\
&= N(\mathfrak{p}_1^{e_1}\mathfrak{p}_2^{e_2} \cdots \mathfrak{p}_t^{e_t}) \\
&= N(\mathfrak{p}_1)^{e_1} N(\mathfrak{p}_2)^{e_2} \cdots N(\mathfrak{p}_2)^{e_t} \\
&= (p_1^{f_1})^{e_1}(p_2^{f_2})^{e_2} \cdots (p_t^{f_t})^{e_t} \\
&= p_1^{f_1+e_1} p_2^{f_2+e_2} \cdots p_t^{f_t+e_t}
\end{aligned}$$

for some (not necissarly destinct) primes $p_1, p_2, ..., p_t$ and possitive exponents $f_1, f_2, ..., f_t$. Essentially, we have been able to relate questions about factorization in $\mathbb{Z}[\theta]$ to questions of factorization in $\mathbb{Z}$. The high level idea is to choose a set $\mathcal{A}$ of prime ideals in $\mathbb{Z}[\theta]$, called an "algebraic factor base" which we will use analogously to our factor base in the integers. The idea is to try an find pairs (a, b) for which the element $a + b\theta$ has a principal ideal $\langle a + b\theta\rangle$ that factors as the product of prime ideals which are in $\mathcal{A}$. We call this being smooth over $\mathcal{A}$.

Two technical problems remains - storing representations of prime ideals in $\mathbb{Z}[\theta]$ on a computer and determining whether an algebraic element of $\mathbb{Z}[\theta]$ is a square.

The former can be overcome by only comprising one's factor base of prime ideals $\mathfrak{p}$ which satisfy $N(\mathfrak{p}) = p$ for some prime p. One can show that these ideals, called "first degree prime ideals", are the only prime ideals apearing in the prime ideal factorization of a principal idea $\langle a + b\theta\rangle$ for coprime integers $a, b \in \mathbb{Z}$ in $\mathbb{Z}[\theta]$. What makes these types of ideals perfect for computing is that they are in natural bijection with the set of all pairs (r, p) where p is a prime and r satisfying $f(r) \equiv 0 \bmod p$. Furthermore, there is a easy to check condition for determining whether a first degree prime ideal occurs in the ideal factorization of $\langle a + b\theta\rangle$. Specifically, a first degree prime ideal with representation (r, p) occurs in the ideal factorization of an element $\langle a + b\theta\rangle$ if and only if $a \equiv -br \bmod p$. To summarize, finding an element $\langle a + b\theta\rangle$ that is smooth over an algebraic factor base of first degree prime ideals of $\mathbb{Z}[\theta]$ amounts to finding an element $a + b\theta$ such that the integer $N(a + b\theta)$ factors completely over the primes occurring in the (r, p) pairs corresponding to the first degree prime ideals in the algebraic factor base.

To determine whether an element β is a square in $\mathbb{Z}[\theta]$,

2.3.2 Algorithm

Algorithm 1 The general number field sieve to factorize an integer N

```
1: return $N$
```

2.3.3 Implementation

```
/*
Here will be the python code.
use xelatex -shell-escape report.tex to compile
*/
```

3 Shor's Quantum Algorithm

4 Conclusion

5 Theorems and Definitions

Definition 5.1.

Definition 5.2. A *field* F is a ring satifying the following extra conditions

1. If $0, 1$ are the respective additive and multiplicative identities of F, then $1 \neq 0$
2. If $a, b \in F$, then $a \cdot b = b \cdot a$
3. For all $a \in F \backslash \{0\}$ there exists $b \in F \backslash \{0\}$ such that $ab = ba = 1$

Theorem 5.3. (*Fundamental Theorem of Algebra*) Let $f \in \mathbb{C}[Z]$ be a non-constant polynomial. Then there is a $z \in \mathbb{C}$ with $f(z) = 0$.

Proof. Let C be the finite set of critical points of f. C is finite by elementary algebra. Remove from the codomain f(C) (and call the resulting open set B) and from the domain its inverse image (again finite) (and call the resulting open set A). Now you get an open map from A to B, which is also closed, because any polynomial is proper (inverse images of compact sets are compact). But B is connected and so f is surjective. $\square$

Definition 5.4. An *ideal* of a ring R is non-empty subset satisfying:

1. If $x, y \in I$, then $x - y \in I$
2. For all $r \in R$ and $x \in I$, both $rx, xr \in I$

Definition 5.5. An ideal $\mathfrak{p}$ of a ring R is *prime* if:

1. For all $a, b \in R$, if $ab \in \mathfrak{p}$, then $a \in \mathfrak{p}$ or $b \in \mathfrak{p}$
2. $\mathfrak{p}$ does not eqaul the whole ring R

Definition 5.6. A *ring* is a set R together with two binary operations, denoted

$$+ : R \times R \longrightarrow R$$

$$\cdot : R \times R \longrightarrow R$$

such that

1. $(a+b)+c = a+(b+c)$ and $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ for all $a, b, c \in R$ (associative law)
2. $a + b = b + a$ for all $a, b \in R$ (commutative law)
3. There exists elements $0, 1 \in R$ such that $a+0 = 0+a = a$ and $a \cdot 1 = 1 \cdot a = a$ for all $a \in R$ (additive and multiplicative identities)
4. For all $a \in R$, there exists $b \in R$ such that $a + b = 0$ (additive inverse)
5. $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$ and $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$ for all $a, b, c \in R$ (distributive law)

6 References