FACTORIZATION TECHNIQUES

ELVIS NUNEZ AND CHRIS SHAW

Abstract. The security of the RSA public key cryptosystem relies upon the computational difficulty of deriving the factors of a particular semiprime modulus. In this paper we briefly review the history of factorization methods and develop a stable of techniques that will allow an understanding of Dixon's Algorithm, the procedural basis upon which modern factorization methods such as the Quadratic Sieve and General Number Field Sieve algorithms rest.

1. Introduction

During this course we have proven unique factorization in $\mathbb{Z}$.

Theorem 1.1. *Given n, there exists a unique prime factorization up to order and multiplication by units.*

However, we have not deeply investigated methods for determining what these factors are for any given integer. We will begin with the most naïve implementation of a factorization method, and refine our toolkit from there.

2. Trial Division

Theorem 2.1. *There exists a divisor of n, a such that $1 > a \leq \sqrt{n}$.*

Definition 2.2. $\pi(n)$ denotes the number of primes less than or equal to n.

Proof. Suppose $b|n$ and $b \geq \sqrt(n)$ and $ab = n$. It follows $a = \frac{n}{b}$. Suppose $b = \sqrt{n}$, then $a = \frac{n}{\sqrt{n}} = \sqrt{n}$. Suppose $b > \sqrt{n}$, then $a < \sqrt{n}$. □

By theorem 2.1, we can find a factor of n by dividing n by the numbers in the range $(1, \sqrt{n}]$. By theorem 1.1, we know that we can express n as a product of primes, and by theorem 2.1 we know there is a factor of n less than $\sqrt{n}$. Therefore, we need not check every integer in the range $(1, \sqrt{n}]$, but need only check prime numbers in this range.

Example 2.3. Suppose we want to factor 679. We begin by first calculating $\sqrt{679} \approx 27$. We will try to divide 679 by the prime integers in the range $(1, 27] = \{2, 3, 5, 7, 11, 13, 17, 19, 23\}$. We find $2, 3, 5 \nmid 679$, but $7|679 \rightarrow 679 = 7 \cdot 97$. We check if 97 is prime; it is, so we are done.

Trial division on a modern computer is fairly quick for numbers up to about 10 digits in length. For example, an integer close to 10000000000, would have primes in the range $(1, 100000]$ and there are $\pi(100000) = 9592$

primes in this range. Assuming these primes have been stored in memory, it follows at most 9592 divisions occur while trying to factor such an integer, which could be done fairly quickly on a modern computer. However, this method is inefficient for large n. In particular, if n has 100 digits and is the product of 2 primes, we would have to check approximately 10^{47} primes in the possible range. Even on a modern computer, this could take years to compute. As we will explore in the coming sections, there are other methods of factoring numbers that do not rely on blindly testing integers.

3. Fermat's Algorithm

The heart of Fermat's algorithm lies in the following observation: If n can be written as a difference of squares such that $n = a^2 - b^2$, then $n = (a+b)(a-b)$. With this observation, we can define Fermat's Algorithm.

Definition 3.1. Fermat's Algorithm: Suppose we want to factor n. We will search for two numbers a, b such that $n = a^2 - b^2 = (a+b)(a-b)$. We begin by setting a equal to $\lceil \sqrt{n} \rceil$ and increase b until either $a^2 - b^2 = n$, in which case we have found factors of n, or if $a^2 - b^2 < n$, in which case we increment a by 1, and repeat the algorithm. Expressed differently, we will check if $a^2 - n$ is a square; if it is we are done, if not we will increment a by 1 until $a^2 - n$ is a square.

Proof. We present a proof that Fermat's algorithm will terminate. Let p, q be factors of n and n composite so that $n = p \cdot q$. Note that because n is odd, p, q must be odd. We can now write:

$$n = pq = \frac{1}{4}(2pq + 2pq) = \frac{1}{4}((p+q)^2 - (p-q)^2) = \left(\frac{p+q}{2}\right)^2 - \left(\frac{p-q}{2}\right)^2$$

Let $a = \left(\frac{p+q}{2}\right)^2$ and $b = \left(\frac{p-q}{2}\right)^2$. We now have $n = a^2 - b^2$. Note that because p, q are odd, $p \pm q$ is even, making a, b integers.

We now show that that if we begin by letting $a = \lceil \sqrt{n} \rceil$ and increment by 1 until $a^2 - n$ is a square, the algorithm will eventually terminate. We first justify initializing $a = \lceil \sqrt{n} \rceil$. Suppose $a < \sqrt{n}$. It follows $n = a^2 - b^2 < (\sqrt{n})^2 - b^2 = n - b^2$ so $0 < -b^2$, which is false since the right hand side of this inequality is negative.

Now suppose $a \geq \frac{n+1}{2}$. We claim that if this occurs, then n is prime, contradicting our assumption that n is composite. If $a \geq \frac{n+1}{2}$, then $n \geq \left(\frac{n+1}{2}\right)^2 - b^2$ so $b^2 \geq \frac{n^2}{4} + \frac{n}{2} + \frac{1}{4} - n$ so $b \geq \frac{n-1}{2}$ It follows $a + b \geq \frac{n+1}{2} + \frac{n-1}{2} = n$. This is true only when $a + b = n$ and $a - b = 1$; or, in other words, when n is prime. □

Example 3.2. Suppose we want to factor 16463. We begin by first calculating $\lceil\sqrt{16463}\rceil = 129$ We intialize $a = 129$ and calculate:

a	a^2	$a^2 - n$	square?
129	16641	178	no
130	16900	437	no
131	17161	698	no
132	17424	961	yes

As seen in the table above, we find $a = 132$ and $b = 31$. Thus, $n = (132+31)(132-31) = 163 \cdot 101$. We check if these numbers are prime, they are, so we are done.

Fermat's algorithm can quickly find the factors of n if one of these factors is near $\sqrt{n}$. However, this algorithm can be worse than trial division when the factors of n are far from $\sqrt{n}$. The worst case occurs when n is prime. Fermat's Algorithm would require $n - \sqrt{n}$ trials to discover n is prime. By trial division, however, only $\sqrt{n}$ trials are necessary. It is interesting to note that because many modern cryptography systems rely on the difficulty of factoring numbers, large non-prime numbers are chosen so that their factors are not close to $\sqrt{n}$, otherwise they could be factored very quickly.

In the next section we will explore a more efficient method of finding numbers a, b such that $n = a^2 - b^2$ that does not require successively incrementing a until the conditions are met.

4. Kraitchik's Algorithm

Kraitchik's algorithm can be thought of as an improvement of Fermat's algorithm. Rather than searching for numbers a, b such that $n = a^2 - b^2$, we will search for random numbers a, b such that $a^2 \equiv b^2 \pmod{n}$. In other words, we will search for a, b such that $(a+b)(a-b)$ is a multiple of n.

Theorem 4.1. *Kraitchik's algorithm: Given $a^2 \equiv b^2 \pmod{n}$, it follows $n|(a-b)(a+b)$. This has trivial solutions when $a \equiv \pm b \pmod{n}$; these trivial solutions appear less than 50% of the time. In every other case, solving this congruence finds a nontrivial factor.*

When applying Kraitchik's method, we will only consider solutions to the congruence $a^2 \equiv b^2 \pmod{n}$ such that $a \not\equiv b \pmod{n}$. Without this restriction, when applying the Euclidean algorithm we would find that the $\gcd(n, a \pm b) = n$, which results in a trivial divisor of n.

In practice, Kraitchik's algorithm is used in conjunction with Fermat's algorithm. Recall Fermat's algorithm looks for a, b such that $b^2 = a^2 - n$. Kraitchik's algorithm multiplies the results from Fermat's algorithm, that is, $a_i^2 - n$ until a square is found. We illustrate this in the following example.

Example 4.2. Suppose we want to factor $n = 5349$. We begin by initializing $a = \lceil \sqrt{5349} \rceil = 74$. We apply Fermat's algorithm:

a	$a^2 - n$	$a^2 - n$ factored
75	276	$2^2 \cdot 3 \cdot 23$
76	427	$7 \cdot 61$
77	580	$2^2 \cdot 5 \cdot 29$
78	735	$3 \cdot 5 \cdot 7^2$
79	892	$2^2 \cdot 223$
81	1212	$2^2 \cdot 3 \cdot 101$
82	1375	$5^3 \cdot 11$
83	1540	$2^2 \cdot 5 \cdot 7 \cdot 11$
84	1707	$3 \cdot 569$
85	1876	$2^2 \cdot 7 \cdot 67$
86	2047	$23 \cdot 89$
87	2220	$2^2 \cdot 3 \cdot 5 \cdot 37$
88	2395	$5 \cdot 479$
89	2572	$2^2 \cdot 643$
90	2751	$3 \cdot 7 \cdot 131$
91	2932	$2^2 \cdot 733$
92	3115	$5 \cdot 7 \cdot 89$
93	3300	$2^2 \cdot 3 \cdot 5^2 \cdot 11$

Note we have omitted values for which $a^2 - n$ is prime. After computing several values using Fermat's algorithm, Kraitchik's algorithm looks for possible combinations of $a_i^2 - n$ such that the result is a square. An easy way of doing this is by looking at the factored form of $a_i^2 - n$ and ensuring the exponents are even when the product is taken. In our example, consider $a_{78} \cdot a_{82} \cdot a_{93}$ we take the product to obtain $a = 78 \cdot 82 \cdot 93 = 594828$. We calculate $b = (3 \cdot 5 \cdot 7^2)(5^3 \cdot 11)(2^2 \cdot 3 \cdot 5^2 \cdot 11) = 2^2 \cdot 3^2 \cdot 5^6 \cdot 7^2 \cdot 11^2$ which is a square since all exponents are even and can be expressed as $(2 \cdot 3 \cdot 5^3 \cdot 7 \cdot 11)^2$; thus, $b = 6930$. We know use the Euclidean algorithm to calculate $\gcd(n, a + b) = \gcd(n, 594828 + 6930) = 3$. Thus, we find 3 is a nontrivial factor of n. We can then calculate the second factor by dividing n by 3: $\frac{n}{3} = 1783$. We check if these are prime; they are, so we are done.

As shown by the previous example, Kraitchik's algorithm still relies on trial and error. A more systematic approach to finding a, b was found by D.H Lehmer and R.E. Powers which utilizes continued fractions. The advent of powerful computing led John Brillhart and Michael Morrison to develop the continued fraction algorithm (CFRAC) in the 1970s. This algorithm can be thought of as the forerunner of the factoring techniques we will explore in the next sections.

5. Dixon's Algorithm

Before discussing Dixon's algorithm, we present a few definitions.

Definition 5.1. Smooth number: We say n is B-smooth if the largest prime divisor of n is less than B. For example $4959 = 3^2 \cdot 19 \cdot 29$; therefore, we can say 4959 is 29-smooth. B need not necessarily be prime, as long as the largest divisor of n is less than B. Thus, we can also say 4959 is 50-smooth.

Definition 5.2. Factor base: A set of primes bounded by some bound B. For example, $\{2, 3, 5, 7, 11\}$ is a factor base bounded by 11. We will denote factor bases with the letter P and subscript denoting the bound B. Thus, the example given will be expressed as P_{11}.

Definition 5.3. Factor vector: We call the vector $v(n)$ a factor vector if each index in v denotes its corresponding prime power where each index i corresponds to the ith prime number. For example, $2520 = 2^3 \cdot 3^2 \cdot 5 \cdot 7$. Thus, $v(2520) = \{3, 2, 1, 1\}$.

Recall the Fermat and Kraitchik algorithms search for a, b such that $a^2 \equiv b^2 \pmod{n}$. Dixon's algorithm does not search for a, b such that $a^2 - n$ is a square, but rather searches for some $a^2 - n$ that has only small factors. We describe Dixon's algorithm below.

Definition 5.4. Dixon's algorithm: Suppose we are trying to factor n and n is not prime. We will start by choosing some bound B and create a factor base P_B. We will then look for integers a such that $a^2 \pmod{n}$ is B-smooth. By the definition of smooth numbers and factor bases, if $a^2 \pmod{n}$ is B-smooth, it follows

$$\prod_{p_i \in P} p_i^{x_i} \equiv a^2 \pmod{n}$$

For some appropriately chosen exponents x_i. By the properties of modular arithmetic:

$$a^2 \equiv \prod_{p_i \in P} p_i^{x_i} \pmod{n}$$

We will repeat this process many times (typically slightly larger than the cardinality of P), recording each factor vector from the exponents x_i on each iteration. With these vectors we can construct a matrix and reduce all exponents modulo 2 and then draw from linear algebra methods to find a product such that the exponents are all even (this is similar to the Kraitchik example above). We are essentially looking for a set of vectors that are linearly dependent over $\mathbb{Z}/\mathbb{Z}_2$. To find such a product, we can use Gaussian elimination. Mathematically, we can express this as

$$a_1^2 \cdot a_2^2 \cdots a_k^2 \equiv \prod_{p_i \in P} p_i^{a_{i,1} + \cdots + a_{i,k}} \pmod{n}$$

where the exponents $a_{i,1} + \cdots + a_{i,k}$ are even. This gives us our previous congruence of the form $a^2 \equiv b^2 \pmod{n}$ where

$$a^2 = (a_1 \cdots a_k)^2$$

and

$$b^2 = \prod_{p_i \in P} p_i^{a_{i,1}+\cdots+a_{i,k}}$$

As discussed before, we proceed to find the $\gcd(n, a \pm b)$ to find the factor of n. If this is a nontrivial factor, we are done, otherwise we try again with a different combination of vectors from the one chosen above.

We illustrate Dixon's algorithm with the following example.

Example 5.5. Suppose we want to factor $n = 84923$. We will use the bound $B = 7$. Thus our factor base P is $P = \{2, 3, 5, 7\}$. We will then randomly test numbers between $\lceil\sqrt{84923}\rceil = 292$ and 84923 whose squares modulo n are 7-smooth. We now randomly test numbers between $\lceil\sqrt{84923}\rceil = 292$ and 84923 whose squares modulo n are 7-smooth. Suppose we find the following random number:

$$19337^2 \pmod{84923} \equiv 3600 = 2^4 \cdot 3^2 \cdot 5^2$$

In this case, our result is already a square: $3600 = 60^2$ meaning we have found $a = 19337$ and $b = 60$ whose squares are congruent mod n, and can continue as in Kraitchik's Method. Suppose we do not find a 7-smooth square right away, and instead find the following random numbers:

$$\begin{aligned} 1965^2 \pmod{84923} &\equiv 39690 = 2 \cdot 3^4 \cdot 5 \cdot 7^2 \\ 8954^2 \pmod{84923} &\equiv 6804 = 2^2 \cdot 3^5 \cdot 7 \\ 24524^2 \pmod{84923} &\equiv 1890 = 2 \cdot 3^3 \cdot 5 \cdot 7 \end{aligned}$$

We then construct the following matrix that corresponds to the factor vectors of our results:

$$\begin{bmatrix} 1 & 4 & 1 & 2 \\ 2 & 5 & 0 & 1 \\ 1 & 3 & 1 & 1 \end{bmatrix}$$

We then take this matrix and reduce modulo 2:

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

We now want to try to find a set of vectors that are linearly dependent in $\mathbb{Z}/\mathbb{Z}_2$. It is clear in this example that multiplying all three numbers will result in a number with even coefficients. In more complicated problems, methods such as Gaussian elimination can be employed.

$$39690 \cdot 6804 \cdot 1890 = 2^4 \cdot 3^{12} \cdot 5^2 \cdot 7^4$$

which is a square.

We now have a relationship in the form $a^2 \equiv b^2 \pmod{84923}$ where

$$a = 1965 \cdot 8954 \cdot 24524 = 431490215640 \equiv 19406 \pmod{84923}$$

$$b = \sqrt{39690 \cdot 6804 \cdot 1890} = 714420 \equiv 35036 \pmod{84923}$$

And finally, just like in Kraitchik's Method, we can use the Euclidean algorithm to compute:

$$\gcd(84923, 19406 + 35036) = 163$$

$$\gcd(84923, 19406 - 35036) = 521$$

We check if these are prime; they are, so we are done.

6. Refernces

(1) David M. Bressoud, *Factorization and Primality Testing*, Springer, 1989.