FACTORING LARGE NUMBERS, A GREAT WAY TO SPEND A BIRTHDAY

LINDSEY R. BOSKO

I would like to acknowledge the assistance of Dr. Michael Singer. His guidance and feedback were instrumental in completing this project.

ABSTRACT. To keep private information secure, messages are encrypted and then decrypted using a number of different algorithms. One of the most famous encryption algorithms relies on the knowledge that factoring large numbers is difficult. We describe an attack on this algorithm, known as Pollard's Rho Method and relate it to a curiosity of probability known as the Birthday Problem.

1. Introduction

Example 1.1. Suppose two algebra students, Alice and Bob, are sending encrypted messages back and forth in the following manner. They agree to map the alphabet onto Z_{26} with $A \mapsto 0,\ B \mapsto 1,\ C \mapsto 2,\ \ldots,\ Z \mapsto 25$. Alice wishes to send the message '$FERMAT$' to Bob. She begins by converting each letter to its integer modulo 26. Thus,

$$F \mapsto 5,\ E \mapsto 4,\ R \mapsto 17,\ M \mapsto 12,\ A \mapsto 0,\ T \mapsto 19.$$

All students in the class choose primes p, q with $p \neq q$ and an encryption exponent, a such that $\gcd(a, m) = 1$ where $m = (p-1)(q-1)$. The class publishes a listing of $n = pq$ and a for each student. Alice uses the list to find Bob's information to be $(n = 31921, a = 47)$ and encrypts the message further.

Letter	$x \in \mathbb{Z}_{26}$	$x^a \mod n$
F	5	6599
E	4	28321
R	17	25589
M	12	31125
A	0	0
T	19	14944

THE AUTHOR WAS PARTIALLY SUPPORTED BY NSF GRANT CCR-0634123

Alice then sends Bob the six numbers in the last column of the table. Using a computer algebra system Bob finds b such that $ab = 1 \mod (p-1)(q-1)$. For this example, $b = 16783$. Bob then calculates the following modulo 31921:

y	$y^b \mod n$	Letter
6599	5	F
28321	4	E
25589	17	R
31125	12	M
0	0	A
14944	19	T

Thus, the encrypted message Bob receives converts back into Alice's original message of '$FERMAT$'. This introductory example depicts the Rivest-Shamir-Adleman (RSA), one of the most widely used public-key cryptosystems.

Exercise 1.2. Recall, p, q are primes, $n = pq$, and $m = (p-1)(q-1)$. a and b are chosen such that $ab = 1 \mod m$ and $\gcd(a, m) = 1$. Describe how the RSA works in general by encrypting and decrypting $x \in \mathbb{Z}_n$.

In our example, the factorization of $n = 31921$ was not known. If an intruder factors this into $n = (233)(137)$, then he could calculate $m = (232)(136) = 31552$ and determine b such that $ab = 1 \mod m$. The calculation of b can be done on any number of computer algebra systems. From here, Alice's intercepted, encrypted message could be raised to the power b, simplified modulo n and decrypted. Thus, the RSA relies on the known difficulty of factoring large numbers. Its secureness would fail if one could discover a way to factor large numbers efficiently, numbers much larger than $n = 31921$. With a and n public, one only need factor n into the product of p and q to easily calculate m. With this information now known, an intercepted message could be decrypted by the intruder.

Exercise 1.3. Using $p = 11$, $q = 5$, and $a = 27$ determine n, m, and b. Then encrypt message 'RHO' and decrypt to verify your work is correct.

2. Pollard's Rho Method

The factoring of numbers is a unique curiosity because it is a problem known to elementary aged school children and renowned mathematicians alike. Mainly, given a positive integer N, how do we find its prime divisors? Young children may recommend that one could simply test

all integers less than N until a divisor is found and this is certainly the way most of us first learn to factor. We can improve on this technique when we come to realize we need only test the integers up to $\lfloor\sqrt{N}\rfloor$ for divisors. Additionally, we strengthen our factoring techniques further by learning divisibility tests. We all know that the a number is divisible by three if the sum of its digits is divisible by three. Eventually, this may come to include the more obscure tests for divisibility by 7, 11, and perhaps 13. Even armed with this arsenal, a large enough number cannot be conquered by divisibility tests. For this note we explore a method for factoring beyond the naive guess and check.

In 1975 Pollard devised a 'novel factorization method' to find a proper divisor of an integer, provided one exists (see [2]). It is of interest to note that the computer algebra system, *Mathematica*, uses Pollard's rho method in its *FactorInteger* function. We describe Pollard's method here. Suppose $N > 0$ and $p\big|N$ where p is prime. We begin by randomly selecting with replacement from the set $S_1 = \{0, 1, 2, \ldots N - 1\}$ to form a sequence $x_1, x_2, x_3, \ldots$. Defining $\overline{x_i} = x_i \mod p$ our sequence is now $\overline{x_1}, \overline{x_2}, \overline{x_3}, \ldots$ with each $\overline{x_i} \in S_2 = \{0, 1, 2, \ldots p - 1\}$. Because the sets S_1, S_2 are finite, eventually there will be a repeated integer in both sequences. We expect to achieve a repeat in the the sequence $(\overline{x_i})_{i=1}^{\infty}$ prior to $(x_i)_{i=1}^{\infty}$. Say, $\overline{x_i} = \overline{x_j}$ for some $i \neq j$. Then, $x_i \equiv x_j \mod p$ and by definition, $p\big|(x_i - x_j)$ if $x_i \neq x_j$. Thus, $gcd(|x_i - x_j|, N) \neq 1$. As long as $gcd(|x_i - x_j|, N) \neq N$, we have found a proper divisor for N, namely the gcd of $|x_i - x_j|$ and N. It is vital to note that in executing this algorithm, we do not need to know the value of p. We only need to calculate $gcd(|x_i - x_j|, N)$ and know that there exists a prime divisor of N.

A question may arise from the description above. Mainly, what is the best way in which to choose random numbers from the set $\{0, 1, 2, \ldots, N - 1\}$? Pollard proposes using a function $f : \mathbb{Z}_N \to \mathbb{Z}_N$, choosing an initial element $x_0 \in \{0, 1, 2, \ldots, N - 1\}$, and then recursively defining $x_{i+1} = f(x_i)$. With a proper choice of f, the sequence generated will be pseudo-random, which is sufficient for the algorithm.

Example 2.1. Given $N = 82123$ is not prime, we use Pollard's rho-method with the function $f(x) = x^2 + 1 \mod N$, $f(x_i) = x_{i+1}$, and $x_0 = 631$ to determine a prime factor of N.

i	$x_i \mod N$
0	631
1	69670
2	28986
3	69907
4	13166
5	64027
6	40816
7	80802
8	20459
9	71874
10	6685

We can determine $gcd(|x_i - x_j|, N)$ for $0 \leq x_i < x_j \leq 10$. The first time we achieve a result other than 1 is for $i = 3$ and $j = 10$. In this instance,

$$gcd(|x_3 - x_{10}|, N) = gcd(63222, 82123) = 41.$$

Thus, we have a prime divisor, $p = 41$, which allows us to completely factor $N = 82123 = 41 \cdot 2003$.

Exercise 2.2. Determine the $gcd(|x_4 - x_{11}|, N)$, $gcd(|x_5 - x_{12}|, N)$, and $gcd(|x_6 - x_{13}|, N)$. What do you notice? Why might this be occurring?

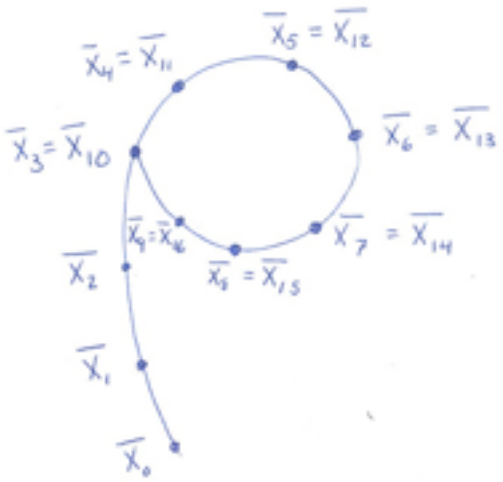
In the example above, we wisely chose not to display the work involved in checking $gcd(|x_i - x_j|, N)$ for $0 \leq x_i < x_j \leq 10$. The actual work involves checking j gcd's for each $j \geq 1$ until a gcd greater than 1 is achieved. For the above example, this meant calculating over 45 different gcd's where operation counts for determining gcd's are $O((\log_2 uv)^2)$ with u, v as the bit sizes of the two inputs. Luckily for us, there exists a more efficient way to implement Pollard's rho-method that eliminates many of these costly gcd calculations.

3. Floyd's Cycle Detection Method

Before describing the new method we inspect Pollard's method further by expanding upon Example 2.1. Recall $N = 82123$ and $p = 41$.

i	$x_i \mod N$	$\overline{x_i} = x_i \mod p$	i	$x_i \mod N$	$\overline{x_i} = x_i \mod p$
0	631	16	10	6685	2
1	69670	11	11	14314	5
2	28986	40	12	75835	26
3	69907	2	13	37782	21
4	13166	5	14	17539	32
5	64027	26	15	65887	0
6	40816	21	16	74990	1
7	80802	32	17	45553	2
8	20459	0	18	73969	5
9	71874	1	19	50210	26

From our previous work, the smallest such t, l for which $gcd(|x_t - x_{t+l}|, N) \neq 1$ occurs when $t = 3$ and $l = 7$. Notice that $\overline{x_j} = \overline{x_{j+l}}$ for $j \geq 3$. Thus, the sequence, $(\overline{x_i})_{i=1}^{\infty}$ cycles every 7 iterations starting with $\overline{x_3}$. The cycling occurs as a result of $\overline{x_3} = \overline{x_{10}}$, which implies that $x_3 = x_{10} \mod p$. So, $f(x_3) = f(x_{10}) \mod p$. Thus, $\overline{x_4} = \overline{x_{11}}$, $\overline{x_5} = \overline{x_{12}}$, and so on.



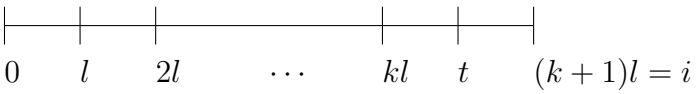
A clever depiction of this sequence forms the shape of the lowercase Greek letter rho, ρ. Because of this cycling we also will have $gcd(|x_j - x_{j+l}|, N) \neq 1$ for each $j \geq 3$. In executing Pollard's rho-method with the goal of efficiency, we want to find the first indication of this cycling without checking each gcd.

To achieve this, we employ Floyd's cycle detection method. We first describe an overview of the method using a common analogy. Suppose there exists a sequence which eventually lapses into a cycle of some finite length. To find this cycle, we send a tortoise and a hare through the sequence. Assume the tortoise moves through each element, one

at a time in order, and the hare starts at the same point, but moves twice as fast as the tortoise. We compare the positions of the animals after each move. The hare will be in cycle first, and may go through it several times, but, eventually, both will be in the cycle. Since the cycle has finite length, they will meet. This should occur before the tortoise's makes a full revolution in the cycle.

Mathematically, if $f : \mathbb{Z}_N \to \mathbb{Z}_N$ and $x_0 = y_0 \in \mathbb{Z}_N$ then define $x_{i+1} = f(x_i)$ and $y_{i+1} = f(f(y_i))$. Here, the x_i's mark the tortoise's path and the y_j's mark the hare's. We identify t and l as the smallest such integers for which $x_{t+l} = x_t \mod p$. We have already seen that $x_{j+l} = x_j \mod p$ for $j \geq t$. So, there are many instances for which we obtain equality modulo p. We claim that the first time $x_i = y_i \mod p$ occurs when $i \leq t + l$.

Proof of Claim. Suppose $y_i = x_i$. Then $x_i = x_{2i}$ and we have $i \geq t$ since t was defined to be the smallest such integer. Also, $l \big| (2i - i) = i$ and there exists $k \in \mathbb{Z}$ such that $(k + 1)l = i$. We depict this as:



Then, $i = (k + 1)l = t + (-t \mod l) \leq t + l$ if $t > 0$. If $t = 0$, then $i = l$. Thus, we have detected the existence of the cycle before completing more than one loop. □

Now, let us rework Example 2.1 with this new technique.

Example 3.1. Recall that $N = 82123$ and it originally took checking over 45 different *gcd*'s before finding a nontrivial divisor, p, of N.

i	$x_i \mod N$	$y_i \mod N$	$gcd(\mid x_i - y_i \mid, N)$
0	631	631	–
1	69670		
2	28986		
3	69907		
4	13166		
5	64027		
6	40816		
7	80802		

Exercise 3.2. Complete the last two columns and prove that after checking just seven *gcd*'s, we determine a prime divisor for $N = 82123$. Note that this is an improvement over the initial execution of the method without Floyd's cycle detection.

4. The Birthday Problem

To determine an operation count for Pollard's rho-method we examine a curiosity of probability known as the Birthday Problem. Suppose we question n people, then the Birthday Problem asks how likely is it that two people share the same birthday? We eliminate February 29 and assume all of the other 365 birthdays are equally likely. A generalization of the Birthday Problem involves selecting elements from the set $S = \{1, 2, 3, \ldots, n\}$ sequentially with replacement until two are the same. How long do we expect this experiment to take?

We analyze this problem by defining A_i as the outcome that we need to make exactly i selections. The expected value for which we are searching is then:

$$\begin{aligned}
\sum_{i=1}^{\infty} P(A_i)i &= P(A_1) + 2P(A_2) + 3P(A_3) + \cdots \\
&= P(A_1) + P(A_2) + P(A_3) + \cdots \\
&\qquad + P(A_2) + P(A_3) + \cdots \\
&\qquad\qquad + P(A_3) + \cdots \\
&\qquad\qquad\qquad + \cdots \\
&= \textit{probability 1 or more are selections are made} \\
&\qquad + \textit{probability 2 or more are selections are made} \\
&\qquad + \textit{probability 3 or more are selections are made} \\
&\qquad + \cdots \\
&= \sum_{i=0}^{\infty} \textit{probability i+1 or more selections are made}
\end{aligned}$$

Note that the probability that $i+1$ or more selections are made is the same as the probability that i selections are made and no two are the same. So, the probability that i selections are made and no two are the same is:

$$\begin{aligned}
&\frac{\left|\{(a_1, a_2, \ldots, a_i) : a_j \in S,\ a_j \neq a_k \textit{ for } 1 \leq j < k \leq i\}\right|}{\left|\{(a_1, a_2, \ldots, a_i) : a_j \in S\}\right|} \\
&= \frac{n(n-1)(n-2)\cdots(n-i+1)}{\underbrace{n \cdot n \cdot \ldots \cdot n}_{\text{n times}}}
\end{aligned}$$

$$= 1\left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)\cdots\left(1-\frac{i-1}{n}\right) \tag{4.1}$$

As $1+x \leq e^x$ by its Taylor series expansion, then $1-x \leq e^{-x}$ and we have

$$\begin{aligned} Eq.\,(4.1) &\leq e^{\frac{-1}{n}} e^{\frac{-2}{n}} \cdots e^{\frac{-(i-1)}{n}} \\ &= exp\left(-\sum_{j=0}^{i-1}\frac{j}{n}\right) = exp\left(\frac{-1}{n}\sum_{j=0}^{i-1} j\right) \\ &= exp\left(\frac{-1}{n}\left(\frac{1}{2}(i-1)\,i\right)\right) = exp\left(\frac{-i(i-1)}{2n}\right) \end{aligned}$$

Suppose we want to find which i satisfy the inequality $exp\left(\frac{-i(i-1)}{2n}\right) \leq \frac{1}{2}$. This is equivalent to $\frac{-i(i-1)}{2n} \leq -\ln 2$. Thus,

$$\frac{i(i-1)}{2n} \geq \ln 2 \Rightarrow i^2 - i \geq 2n(\ln 2) \Rightarrow i^2 - i - 2n(\ln 2) \geq 0$$

By the quadratic formula, equality is achieved when $i = \frac{1}{2}\left(1 \pm \sqrt{1+8n(\ln 2)}\right)$. So, $i^2 - i - 2n(\ln 2) \geq 0$ when $i \geq \frac{1}{2}\left(1+\sqrt{1+8n(\ln 2)}\right)$. Here, we can use $n = 365$ to conclude that after asking 23 people, we expect with over 50% probability that two people will have the same birthday.

Return to our expected value, which we know to be bounded above by

$$\begin{aligned} \sum_{i=0}^{\infty} exp\left(\frac{-i(i-1)}{2n}\right) &\leq 1 + \sum_{i=1}^{\infty} exp\left(\frac{-(i-1)(i-1)}{2n}\right) \\ &= 1 + \sum_{j=0}^{\infty} exp\left(\frac{-j^2}{2n}\right) \\ &\leq 2 + \int_0^{\infty} exp\left(\frac{-x^2}{2n}\right) dx \qquad Let\ t = \frac{x}{\sqrt{2n}}. \\ &= 2 + \sqrt{2n}\int_0^{\infty} exp\left(-t^2\right) dt \\ &= 2 + \sqrt{2n}\frac{\sqrt{\pi}}{2} = 2 + \sqrt{\frac{n\pi}{2}} \end{aligned}$$

where the integral is solved

Thus, our expected value is of $\mathcal{O}(\sqrt{n})$. The birthday problem, with $n = 365$, results in an expected value of no more than 26 people. Relating this to Pollard's rho-method, where we are choosing with replacement from the set $\{0, 1, 2, \ldots p-1\}$, we expect to have to test $\mathcal{O}(\sqrt{p})$ values before two are the same modulo p. Since $p \leq \sqrt{N}$, we can say the algorithm has $\mathcal{O}(\sqrt[4]{N})$ As a result, we expect to have a $gcd > 1$ and, in turn, a divisor of N.

References

[1] Frank H. Mathis, A Generalized Birthday Problem, *SIAM Reivew*, Vol. 33, No. 2, (June 1991), 265-270.

[2] J.M Pollard, Kangaroos, A Monte Carlo Method for Factorization, BIT 15 (1975), 331-334.