The Pollard Rho factorization method

By Kwang Hyun Kim

1. Introduction

One of the two fundamental problems of computational elementary number theory is that of factorization : given a positive integer n, determine the complete factorization of n as a product of primes. (The other problem is primality testing.)

Pollard rho algorithm is a special-purpose integer factorization algorithm. It was invented by John Pollard in 1975. It is particularly effective at splitting composite numbers with small factors.

Given a relatively small known composite n, the Pollard rho algorithm seeks a non-trivial factor of n. The method finds these factors by implicitly traversing Z_p, where p is a prime factor of n, while traversing Z_n. If a random map $f : Z_n \to Z_n$ is used to iterate from a random seed s, creating the sequence $\{s, f(s), f(f(s), f(f(f(s))), \cdots\}$, on the average a cycle will form after $\sqrt{p}$ steps.

When a cycle in p has been found, we have $f^j(s) = f^k(s) \pmod p$, $j \neq k$. If these two iterates, which do not differ $\bmod p$, do differ $\bmod n$, then this difference is a multiple of p. In this case the greatest common divisor of $f^j(s) - f^k(s)$ and n yields a non-trivial divisor.

2. Quadratic Maps

In practice, a random map requires too many computations to be used. A quadratic map used in placed of a random map proves to be "random enough" to support the factorization method, while minimizing computations. We examine this assumption with the quadratic map $f(x) = x^2 + a$. The parameter a is termed the Pollard rho parameter and can have a significant effect on the mean cycle length. The choice $a = 0$ covers only the quadratic residues and is not "random enough" to be used. Other specific choices of a can be shown to generate non-ideal mean cycle lengths, such as $a = p - 2$. However, the effects of these choices are not easily quantified, and thus we do not attempt to systematically identify non-ideal values.

3. Random sequences

Suppose that you have a 20-sided fair die, with faces numbered from 1 to 20. You throw the die repeatedly to get a sequence $x_0, x_1, x_2, \cdots$ of integers in the range $1 \leqq x_i \leqq 20$. Eventually you'll get a value that's already occurred earlier in the sequence. That is, there is an integer k such that

$$x_0, x_1, \cdots x_{k-1} \text{ are all distinct, but } x_k = x_j \text{ for some } 1 \leqq j < k$$

How large is k, on average? In other words, how many throws should you expect to make before the first duplicate shows up?

It turns out that the mean value of k is 5.29. That is, on average we expect the first repeat to appear somewhere near x_5.

Now replace the 20-sided die by an n-sided die, and ask the same question : how many throws do you expect to make before the first repeat shows up? A general formula is :

$$\text{average value of } k = \sum_{j=1}^{n} \frac{n!}{n^j (n-j)!}$$

Here, $\frac{n!}{n^j (n-j)!}$ is the probability that the first j throws are all distinct. Also, a very good approximation to this formula is :

$$\text{average value of } k \sim \sqrt{\frac{n\pi}{2}} - \frac{1}{3}$$

Example 1 (The Birthday Paradox)

Let $n = 366$, the number of possible birthdays. Then the average value of k is around $\sqrt{366\pi/2} - 1/3$, or approximately 23.6. As a consequence, if there are 24 or more people in a room, it's likely that at least two people in the room share a birthday.

More precisely, if there are exactly 23 people in the room, the probability of a shared birthday is $1 - \frac{366!}{343! \times 366^{23}} \sim 0.506$, so is greater than 50%. For a room of 41 people the probability of a shared birthday is greater than 90%, while for 58 people the probability is at least 99%.

4. Pollard Rho factorization method

Suppose that n is a composite integer. We define a sequence $x_0, x_1, x_2, \cdots$ of integers x_i in the range $0 \leqq x_i < n$ recursively as follows : $x_0 = 0$, and for each $k \geqq 0$,

$$x_{k+1} = (x_k^2 + 1)\% n$$

For example, suppose that $n = 527$. Then the sequence starts

$$0, 1, 2, 5, 26, 150, 367, 305, 274, 243, 26, 150, 367, \cdots$$

Eventually the sequence must start to repeat. Now we need a leap of faith : Suppose that the length of the initial part of the sequence (the part before the first repeat) behaves in roughly the same way as for a random sequence. Then the time before the first repeat should be, on average, somewhere around $\sqrt{n\pi/2}$.

Now let p be the smallest prime divisor of n (so $p \leqq \sqrt{n}$), and define a sequence $y_0, y_1, \cdots$ of integers by $y_i = x_i \% p$. Then,

$$y_{k+1} \equiv x_{k+1} \equiv x_k^2 + 1 \equiv y_k^2 + 1 \pmod{p}$$

and hence $y_{k+1} = (y_k^2 + 1)\% p$, so the y_i obey a similar recurrence to the x_i, and by the same leap of faith we expect the sequence of y values to repeat after around $\sqrt{p\pi/2}$ steps. Note that since $p \leqq \sqrt{n}$, $\sqrt{p\pi/2} \leqq 1.26\sqrt[4]{n}$.

In the example above, the smallest prime factor of n is 17, and the sequence y_i starts :

$$0, 1, 2, 5, 9, 14, 10, 16, 2, 5, 9, 14, 10, \cdots$$

Here is the key observation : even though we don't know what p is, we can detect when the sequence of y_i starts to repeat. For any indices j and k,

$$y_j = y_k \;\Rightarrow\; x_j \% p = x_k \% p \;\Rightarrow\; p \mid (x_k - x_j)$$

Since p divides n, we then have

$$p \mid \gcd(x_k - x_j, n).$$

In particular, it follows that

$$y_j = y_k \;\Rightarrow\; \gcd(x_k - x_j, n) \neq 1$$

and then provided that $x_k \neq x_j$, $\gcd(x_k - x_j, n)$ must be a proper (that is, not equal to n) non-trivial (that is, not equal to 1) factor of n.

5. Naive Pollard-Rho

The above gives a naive factorization algorithm : suppose that n is the integer that you want to factor. Then

1. Compute the sequence $x_0, x_1, \cdots$ as above.
2. For each x_k computed, calculate

$$g = \gcd(x_k - x_j, n) \text{ for all } 0 \leq j < k .$$

Eventually you should find valued of j and k for which $g \neq 1$, and then with luck $\gcd(x_k - x_j, n)$ is a proper factor of n.

We expect the first successful value of k to be of the same order of magnitude as $\sqrt{n\pi/2}$. In the example above, with $n = 527$, we find that $\gcd(x_5 - x_4, 527) = 31$, hence obtain a factorization $527 = 31 \times 17$. (In this case the sequence $(x_i)_{i \geq 0}$ repeats sooner modulo 31 than modulo 17, so we end up finding the bigger prime factor first.) Unfortunately, this naive algorithm is not practical :

1. It requires storage space proportional to $\sqrt{p}$ to hold the x_k values.
2. More seriously, the number of gcd computations required will be around $k^2/2$, where k is the first integer for which $\gcd(x_k - x_j, n)$ is non-trivial for some j. If, as we expect, k is on the order of $\sqrt{p}$, then $k^2/2$ is of the order of p. This makes the naive version of the algorithm above no better than trial division, which finds the smallest prime factor p of n in time roughly proportional to p.

To make the Pollard Rho method practical, we need one more idea.

Theorem

Suppose that $y_j = y_k$ for some $0 \leq j < k$. Let m be the smallest positive multiple of $k - j$ for which $m \geq j$. Then $m \leq k$ and $y_m = y_{2m}$. Hence $\gcd(x_{2m} - x_m, n) \neq 1$ is a non-trivial factor of n.

Proof.

If $j = 0$ then $m = k - j = k$ and if $j > 0$ then since

$$m = j + (-j \,\%\, (k-j)) < j + (k-j) = k ,$$

we have $m \leq k$. Now, show that $y_m = y_{2m}$. First, note that

$$y_j = y_k \;\Rightarrow\; y_{j+1} = y_{k+1}$$

So, we deduce from induction that $y_m = y_{j+(m-j)} = y_{k+(m-j)} = y_{m+(k-j)}$ and hence

$$y_m = y_{m+(k-j)t}$$

for any nonnegative integers t. We apply this with $t = m/(k-j)$. ■

6. The Pollard Rho algorithm

Suppose that n is a composite integer that you would like to factorize. Set $x_0 = 0$. Now for each $k \geqq 1$,

1. Compute x_k from x_{k-1} by the formula

$$x_k = (x_{k-1}^2 + 1)\,\%\, n .$$

2. Compute x_{2k} from x_{2k-2} by the formula

$$x_{2k} = \left(\left(x_{2k-2}^2 + 1\right)\%\, n\right)^2 + 1\Big)\,\%\, n .$$

3. Compute $g = \gcd(x_{2k} - x_k, n)$. If $g \neq 1$ then stop : g is a nontrivial factor of n. Otherwise, continue with the next k.

When the algorithm terminates, in step 3 above, g will be a nontrivial factor of n. Continuing with the example above, $n = 527$, we get the following results :

k	x_k	x_{2k}	$\gcd(x_{2k} - x_k, 527)$
0	0	0	527
1	1	2	1
2	2	26	1
3	5	367	1
4	26	274	31

-Table 1-

Thus the Pollard-Rho method has discovered the proper factor 31 of n.

7. When Pollard Rho fails

The algorithm above must terminate eventually (although making precise predictions about when this will happen is difficult). But it's possible for the algorithm to fail : if $x_{2k} = x_k$ the first time the gcd is nontrivial, then the returned factor is simply n. (This will always happen, for example, if n is prime to begin with.)

When this happens, one can restart the Pollard-Rho method using either a different starting value - for example, $x_0 = 2$ instead of $x_0 = 0$, or a different iteration - for example, $f(x) = x^2 + 2$ or $f(x) = x^3 + 1$ instead of $f(x) = x^2 + 1$. A common practice is to use the iteration $f(x) = x^2 + c$ for a randomly chosen c in the range $0 \leqq c \leqq n$. However, the values $c = 0$ and $c = n - 2$ should be avoided.

Example 2

When $n = 1241$ we get the following results :

k	x_k	x_{2k}	$\gcd(x_{2k} - x_k, 1241)$
0	0	0	1241
1	1	2	1
2	2	26	1
3	5	401	1
4	26	801	1
5	677	26	1
6	401	401	1241

-Table 2-

Thus Pollard Rho fails. However, running Pollard Rho with the iteration $x_{k+1} = (x_k^2 + 1) \% n$ replaced with $x_{k+1} = (x_k^2 + 2) \% n$ quickly produces the factor 17 of 1241 .

Example 3

-Step 1

$$x_0 = 2 \quad \text{and} \quad f(x) = x^2 + 1$$

-Step 2

$$x_1 = f(x) \equiv 5 \pmod{8051},$$
$$x_2 = f(5) \equiv 26 \pmod{8051},$$
$$x_3 = f(26) \equiv 677 \pmod{8051},$$
$$x_4 = f(677) \equiv 7474 \pmod{8051},$$
$$x_5 = f(7474) \equiv 2839 \pmod{8051},$$
$$x_6 = f(2839) \equiv 871 \pmod{8051}$$

-Step 3

$$(x_2 - x_1, 8051) = (21, 8051) = 1,$$
$$(x_4 - x_2, 8051) = (7448, 8051) = 1,$$
$$(x_6 - x_3, 8051) = (194, 8051) = 97$$

Therefore, 97 is a divisor of 8051.

In practice failure appears to be rare, and to become rarer as n gets larger : of the 929565 composite numbers n in the range $10^6 \leqq n < 2 \times 10^6$, there are just 4 integers for which Pollard Rho fails to find a factor with either of the iterations $f(x) = x^2 + 1$ or $f(x) = x^2 + 2$. (both with starting value 0). And for each of these 4 integers, Pollard Rho with the iteration $f(x) = x^2 + 3$ succeeds.

8. Efficiency of Pollard-Rho

For large n, the expected running time of this algorithm is roughly proportional to $\sqrt{p}$, where p is the smallest prime factor of n. Note that this running time depends on p, rather than depending directly on n. Thus Pollard Rho can be used to find small factors (up to 10^{20}, say) of integers with hundreds of digits or more.

The cost of the greatest common divisor computations can be significantly reduced by performing several gcd operations in one : one can compute the product of the $x_{2k} - x_k$ terms, modulo n, for a few thousand (say) successive values of k, and then compute the greatest common divisor of the product with n. If the gcd is not equal to 1 then one can backtrack to find the first k for which $\gcd(x_{2k} - x_k, n) \neq 1$.

9. Application

This algorithm can be applied the factorization of Fermat numbers and Mersenne numbers. See the following table :

k	p_k	M_k	M_k / E_k
5	641	16	0.45
6	274,177	855	1.46
7	59,649,589,127,497,217	2.67×10^8	1.24
8	1,238,926,361,552,897	2.29×10^7	0.95
9	2,424,833	420	0.51
10	45,592,577	1,521	0.56
11	319,489	112	0.65
12	114,689	30	0.38
13	2,710,954,639,361	38,896	0.13

-Table 3-

Where p_k is the least prime factor p_k of Fermat numbers $F_k = 2^{2^k} + 1$, E_k is the expected number of multiplications $(\mathrm{mod}\, F_k)$ to find the least prime factor p_k of F_k and the formula is

$$E_k = (k+3)(\pi p_k/8)^{1/2}(3/\ln 4 + 1)/(2^{k+2} - 1)^{1/2}$$

and M_k is the number of multiplications $(\mathrm{mod}\, F_k)$ required to find it (by the algorithm just described).

The application of more than 100 trials Rabin's probabilistic algorithm lead us to suspect that the cofactor

$$q_8 = F_8/p_8 =$$
$$93,461,639,715,357,977,769,163,558,199,606,$$
$$896,584,051,237,541,638,188,580,280,321$$

was prime. Professor H. C. Williams kindly proved the primality of q_8. Thus we have the complete factorization of the eighth Fermat number F_8.

10. Reference

1. Web

2. R.P. Brent and J.M. Pollard, *Factorization of the eighth Fermat number*, Mathematics of Computation 36 (1981), 627-630.

3. Edlyn Teske, *On random walks for Pollard's Rho method*, Mathematics of Computation Volume 70, Number 234 (2000), 809-825