Scribe Notes for *Algorithmic Number Theory*
Class 28—June 25, 1998

Scribes: Yizhong Wang, Jeremy Rotter, and Wen Wang

Abstract

In this our last class, we discuss random square factoring and the quadratic sieve.

1 Random Square Factoring

We want to factor an integer n, and we are only interested in the case where n has at least two distinct prime factors. If

$$x^2 \equiv y^2 \pmod{n},$$

then

$$n \mid (x^2 - y^2)$$

and therefore,

$$n \mid (x+y)(x-y).$$

If neither $x+y$ nor $x-y$ is a multiple of n, then $\gcd(x+y,n)$ or $\gcd(x-y,n)$ will give us a nontrivial factor of n. We know that if $x, y \in (\mathbb{Z}/(n))^*$ are randomly chosen such that they fulfill the above congruence, the probability that either $x+y$ or $x-y$ yields for us a nontrivial factor is greater than $1/2$. This makes the following strategy possible.

1.1 Random Square Factoring Strategy

Let B be a smoothness bound, and let $S = \{p_1, p_2, \ldots, p_t\}$ be the set of primes $\leq B$. We will call S the factor base.

1. Randomly select an integer a, where $1 \leq a \leq n-1$, and compute $b = a^2$ $(mod\ n)$. If b is not B-smooth, then discard the pair (a,b) and try again. Otherwise, store the pair in list R. Iterate until we have a set $R = \{(a_1,b_1), \ldots, (a_{t+1}, b_{t+1})\}$.

2. For each b_i, we have,

$$b_i = \prod_{j=1}^{t} p_j^{e_{ij}}$$

   where $e_{ij} > 0$. Map b_i to a t-tuple $\hat{b_i}$ by

$$\hat{b_i} = (e_{i1}\ mod\ 2, e_{i2}\ mod\ 2, \ldots, e_{it}\ mod\ 2),$$

   so $\hat{b_i}$ is an element of $(\mathbb{Z}/(2))^t$.

It is easy to see that $L = \{\hat{b_1}, \hat{b_2}, \ldots, \hat{b_{t+1}}\}$ is a linearly dependent set over $\mathbb{Z}/(2)$, and hence we can find a subset T of $\{1, 2, \ldots, t+1\}$ such that

$$\sum_{i \in T} \hat{b_i} \quad = \quad 0$$

in $(\mathbb{Z}/(2))^t$. Then $\prod_{i \in T} b_i$ is a square, and $\prod_{i \in T} a_i^2$ is a square as well. Let $x^2 = \prod_{i \in T} b_i$ and $y^2 = \prod_{i \in T} a_i^2$. Then we have

$$x^2 \equiv y^2 \ (mod\ n).$$

Now we can take $\gcd(x + y, n)$ or $\gcd(x - y, n)$ as a factor of n.

Example 1.1.

$$\begin{array}{rcl} n & = & 7 \cdot 11 \cdot 13 \cdot 17 \quad = \quad 17,017 \\ B & = & 5 \\ S & = & \{2, 3, 5\} \\ t & = & 3 \end{array}$$

$$\begin{array}{rclrclcl} a_1 & = & 10733 & b_1 & = & 9216 & = & 2^{10}3^2 \\ a_2 & = & 9832 & b_2 & = & 11664 & = & 2^4 3^6 \\ a_3 & = & 10909 & b_3 & = & 6400 & = & 2^8 5^2 \\ a_4 & = & 5010 & b_4 & = & 25 & = & 5^2 \end{array}$$

Clearly, $\hat{b_1} = \hat{b_2} = \hat{b_3} = \hat{b_4} = (0, 0, 0)$. So we may pick $T = \{3\}$. This gives us $x = a_3 = 10909$ and $y = \sqrt{b_3} = 80$. Now we can calculate the factor,

$$\begin{array}{rcl} \gcd(x - y, n) & = & \gcd(10909 - 80, 17017) \\ & = & 1547 \\ & = & 7 \cdot 13 \cdot 17. \end{array}$$

2 Quadratic Sieve

The quadratic sieve is simply a refinement of the random square method that makes b values smaller, hence increasing the probability that they are smooth.

Let $m = \lfloor \sqrt{n} \rfloor$. If we take $x = 0, \pm 1, \pm 2, \ldots$ to be small integers, then

$$(x + m)^2 - n \approx x^2 + 2xm,$$

which is small compared to n. Take

$$\begin{array}{rcl} a & = & x + m \\ b & = & (x + m)^2 - n \ = \ a^2 - n \end{array}$$

We will keep the pair (a, b) if b is p_t-smooth.

Notice that if p is a prime that divides b, then $a^2 \equiv n \pmod{p}$. Hence $\left(\frac{n}{p}\right) = 1$. Therefore, our factor base S need only contain primes for which n is a quadratic residue.

3 Parallelism

Both of the factoring algorithms discussed in this class spend the majority of their running time computing squares mod n and seeing if they are smooth. Once the squares have been found, both algorithms must reduce a linear system of binary numbers, which is not nearly as time consuming as the search for smooth b values. Because of this, both algorithms lend themselves nicely to parallelism, either on parallel supercomputers, or distributed on networks of machines. In fact, the quadratic sieve has been used to factor numbers on the World Wide Web. "Slave" computers all over the Web look for a values that generate a smooth b, and when they find one, they pass the (a, b) pair to the "master" computer, which collects R and then completes the algorithm.