Reducing Factoring to Order Finding

Note Title 4/23/2012

One of the most celebrated achievements in quantum algorithms was Peter Shor's algorithm to factor large numbers. There is no known polynomial time algorithm that can factor numbers on a classical computer despite a great deal of effort. Besides the fact that this is a fundamental mathematical problem, the hardness of factoring is the basis of certain cryptographic schemes, including RSA.

Factoring is defined as follows:

Input: integer N.
Output: $p_1 p_2 \cdots p_m$ and $\ell_1 \cdots \ell_m$
where the p_i's are prime and $N = \prod_{i=1}^{m} p_i^{\ell_i}$

Note that it is sufficient to find a non-trivial divisor n because then the algorithm can be applied recursively to n and N/n.

The size of the input for this problem is $O(\log N)$. A naive classical algorithm will take time $O(\sqrt{N}\ \text{polylog}(N))$. Sophisticated algorithms such as the Field Sieve method takes time $2^{O(\sqrt[3]{\log N})}$

The quantum algorithm for factoring makes use of a reduction from factoring to order finding. The reduction rests on number-theoretic arguments and was known independently of quantum computation.

Here is the definition for order finding:

Input: integers x + N such that $\gcd(x,N) = 1$.
Output: Smallest r such that $x^r \equiv 1 \bmod N$.

We first show that finding a non-trivial square root of 1 mod N is sufficient to factor:

Lemma: given composite N such that
$x^2 \equiv 1 \bmod N$ and $x \not\equiv \pm 1 \bmod N$
then we can factor N.

Proof $x^2 - 1 \equiv 0 \bmod N$.
$(x+1)(x-1)$ is a multiple of N

Since $x \not\equiv \pm 1 \bmod N$, neither $x+1$ or $x-1$ alone is a multiple of N.
Therefor $\gcd(x+1, N)$ and $\gcd(x-1, N)$ give non-trivial factors of N.

The algorithm for factoring will work as follows:

* Pick x at random from $\{2, \ldots, N-1\}$
* If $\gcd(x,N) \neq 1$ then $\gcd(x,N)$ is a non-trivial divisor of N + we're done
* Otherwise compute $r = \text{ord}(x)$
  If r is odd, start again.
  If r is even and $x^{r/2} \equiv -1 \bmod N$, start again
* Otherwise $x^{r/2}$ is a non-trivial sqrt of 1 $\Rightarrow$ use it to factor N

we know $x^{r/2} \not\equiv 1$ otherwise the order of x would be $r/2$

What is the probability that one iteration of the above algorithm will succeed?

$\mathbb{Z}_N^* = \{x \bmod N \mid \gcd(x, N) = 1\}$

$\mathbb{Z}_N^*$ is a group under multiplication.

elements in $\mathbb{Z}_N^* = \varphi(N)$ [the Euler function]

What is the probability that an element x randomly chosen from $\mathbb{Z}_N^*$ has even order and if so satisfies $x^{r/2} \not\equiv -1 \bmod N$? [We will show that if $N = p_1^{\alpha_1} \cdots p_m^{\alpha_m}$ then this prob $\geq (1 - 1/2^m)$]

Lets first look at the case $N = p^\alpha$ for prime p.

In this case $\varphi(N) = p^\alpha - p^{\alpha-1} = p^{\alpha-1}(p-1)$

note that only multiples of p have a divisor in common with p^α.

This necessarily means that $\varphi(N)$ is even, so we can express it as 2^d(odd #) for $d \geq 1$.

We will use a basic fact from number theory without a proof here:

The group $\mathbb{Z}_N^*$ for $N = p^\alpha$ has a generator g.

A generator is an element whose powers mod N are all the elements of $\mathbb{Z}_N^*$:

$\mathbb{Z}_N^* = \{g, g^2, g^3, \ldots, g^{\varphi(N)}\}$ (everything is mod N).

↳ note that this has to be $\equiv 1$ since multiplying by g again will bring us back to the beginning.

if $g^y \equiv 1 \bmod N$ then φ divides y.

if $x^y \equiv 1 \bmod N$ then order(x) divides y.

So let's take a random element x of $\mathbb{Z}_N^*$: $X \equiv g^k \bmod N$.

Suppose the order of x is r: $X^r \equiv g^{kr} \equiv 1 \bmod N$.

Since g is a generator it must be the case that $\varphi(N)$ divides kr evenly. Moreover r is the smallest number such that $\varphi(N)$ divides kr evenly.

Since $\varphi(N) = 2^d(\text{odd \#}) \Rightarrow kr = 2^d(\text{odd \#})$

So we can divide the elements of $\mathbb{Z}_N^*$ into two groups

* { k odd: 2^d divides r ; k even: 2^d does not divide r } randomly chosen x falls in each category with probability $1/2$.

Remember all this was for the special case that $N = p^\alpha$. In order to address more general N, we need to use the

Chinese Remainder Theorem

Let $N = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_m^{\alpha_m}$

For any sequence $(x_1, \ldots, x_m)$ s.t. $0 \le x_j \le p_j^{\alpha_j} - 1$ there is exactly one $x \in \{0, \ldots, N-1\}$ such that

$$x_i \equiv x \bmod p_i^{\alpha_i} \qquad i \in \{1, \ldots, m\}$$

This gives us an alternative way of generating a random element of $\mathbb{Z}_N$:

for $i = 1, \ldots, m$ pick x_i independently at random from $\{0, \ldots, p_i^{\alpha_i} - 1\}$

Let x be the unique solution to $x_i \equiv x \bmod p_i^{\alpha_i}$

Now back to the original problem:

When we pick x at random from $\mathbb{Z}_N^*$, what is the probability that $\text{order}(x) = r$ is even and $x^{r/2} \equiv -1 \bmod N$.

We will bound the probability that r is odd or (r is even and $x^{r/2} \equiv -1 \bmod N$).

Let $r_i = \text{order}(x_i) \bmod p_i^{\alpha_i}$

(1) Will show that if r is odd then all the r_i are odd
Since the x_i's are chosen indepently and each r_i is odd w.p. $\leq 1/2$ (due to #) the prob r is odd $\leq 1/2^m$

(2) Will show that if r is even and $x^{r/2} \equiv -1 \bmod N$ then all the r_i can be expressed as $2^d(\text{odd \#})$ for some fixed d. Again the prob that a particular $r_i = 2^d(\text{odd \#})$ is at most $1/2$ $\Rightarrow$ prob $x^{r/2} \equiv -1 \bmod N$ is $\leq 1/2^m$.

The probability that either event happens is $\leq 1/2^{m-1}$
The probability that we pick a good x is $(1 - 1/2^{m-1})$

(1) $x^r \equiv 1 \bmod N \Rightarrow x^r \equiv 1 \bmod p_i^{\alpha_i}$

$x_i \equiv x \bmod p_i^{\alpha_i} \Rightarrow x_i^r \equiv x^r \bmod p_i^{\alpha_i}$

$\Big\}$ $x_i^r \equiv 1 \bmod p_i^{\alpha_i}$

This means that $\text{order}(x_i) \bmod p_i^{\alpha_i}$ (i.e. r_i) divides r. $\therefore$ If r is odd then all the r_i are odd.

(2) $x^{r/2} \equiv -1 \mod N$ (& r is even)

$\Downarrow$

$x^{r/2} \equiv -1 \mod p_i^{\alpha_i} \quad \Rightarrow \quad x_i^{r/2} \equiv -1 \mod p_i^{\alpha_i}$

r_i divides r but r_i does not divide $r/2$.
So if $r = 2^d$ (odd #) then $r_i = 2^d$ (odd #) for all i.

Note that the probability that a randomly chosen x is "good" (i.e. has even order and $x^{r/2} \neq -1 \mod N$) is at least $1 - 2/2^m$. If $m=1$ (i.e. $N = p^\alpha$ for prime p) this is a terrible bound. However we could use a classical algorithm to check for this before the algorithm begins.