Lecturer: Tongyang, scribed by Shuo Zhou, Yexin Zhang
Date: October 25, 2023

Lecture 8

Shor's Algorithm

- Shor's algorithm(continued)

Finally, it comes to Shor's algorithm:

1 Shor's algorithm

1.1 Factorization(N)

Input: N (WLOG, N is composite).

Output: A non-trivial factoe of N.

1. If N is even, return factor 2 ;
2. If $N = p^\alpha$ for a prime $p \geq 3$ and $\alpha \geqslant 2$, compute the 2nd (square) root, 3rd, $\cdots$, $\lceil \log_2 N \rceil$ root, and return one of them being an integer;
3. Uniformly randomly choose x in $\{1, 2, \ldots, N-1\}$. If $\gcd(x, N) > 1$, then return factor $\gcd(x, N)$;
4. Use the order-finding subroutine to find the order r of x, modulo N;
5. If r is even and $x^{r/2} \neq -1 \pmod N$, compute $\gcd\left(x^{r/2} - 1, N\right)$ and $\gcd\left(x^{r/2+1}, N\right)$. If one of them > 1, return that. Otherwise, start over.

1.2 Observations

1. Primality testing, i.e., testing whether N is a prime or not, can be done in $\tilde{O}\left((\log N)^6\right)$ time on a classical computer. ($\tilde{O}$ omits poly-logarithmic factors, i.e., $\tilde{O}(f) = O(f \cdot \text{ poly } (\log f))$.)

   This is called the Agrawal-Kayal-Saxena (AKS) primality test, won Gödel Prize and Fulkerson Prize in 2006.

   We can regard AKS as a preprocessing step. Nevertheless, we can also run Shor's algorithm for poly$(\log N)$ rounds and return "prime" if it cannot find a factor.
2. Steps 1 and 2 has $O(\log n)$ iterations, and each root computation takes poly$(\log N)$ cost on a classical computer.

   In the rest of the algorithm, we can assume that N is an odd integer with more than one primer factor.
3. $\gcd(\mathrm{x}, \mathrm{N})$ can be computed classically using Euclid's algorithm. This takes cost $O\left(\log^2 N\right)$.

4. If $\gcd(x, N) = 1$ and r is the order of x mod N, and the condition in step 5 holds:

$$x^{r/2} \not\equiv 1 \pmod{N}, \quad x^{r/2} \not\equiv -1 \pmod{N},$$

then $N \nmid x^{r/2}+1, N \nmid x^{r/2}-1$ but $N \mid x^r - 1 = \left(x^{r/2}+1\right)\left(x^{r/2}-1\right)$, thus we have

$$\gcd\left(N, x^{r/2}-1\right) \neq 1, \quad \gcd\left(N, x^{r/2}+1\right) \neq 1.$$

Thus the factorization problem is solved.

If not (If $\gcd\left(N, x^{r/2}+1\right) = 1$): we failed.

1.3 Proof of correctness

In the remaining, we prove:

Theorem. Suppose $N = p_1^{\alpha_1} \cdots p_l^{\alpha_l}$ with different primes $p_1, \cdots, p_l, l \geqslant 2, \alpha_1, \cdots, \alpha_l \in \mathbb{N}^*$. Let x be chosen uniformly random from $\mathbb{Z}_N^* := \{x \in \mathbb{Z}_N \mid \gcd(x, N) = 1\}$, and let r be the order of x mod N. Then

$$\Pr\left[r \text{ is even and } x^{r/2} \neq -1 \pmod{N}\right] \geqslant 1 - \frac{1}{2^{l-1}}.$$

We first prove a lemma:

Lemma. Let p be an odd prime. Let 2^d be the largest power of 2 dividing $\varphi\left(p^\alpha\right)$, i.e., $2^d \| \varphi\left(p^\alpha\right)$: $2^d \nmid \varphi(p^\alpha)$ but $2^{d+1} \nmid \varphi(p^\alpha)$.

Then with probability exactly $\frac{1}{2}, 2^d$ divides the order mod p^α of a uniformly random chosen element of $\mathbb{Z}_{p^\alpha}^* := \left\{x \in \mathbb{Z}_{p^\alpha}^* \mid \gcd(x, p^\alpha) = 1\right\}$.

Proof of lemma.

It is known in elementary number theory that there exists primitive roots mod p^α, i.e., $\exists g \in \mathbb{Z}_{p^\alpha}^*$, s.t. $\left\{g, g^2, \cdots, g^{\varphi(p^\alpha)}\right\} = \mathbb{Z}_{p^\alpha}^*$. In other words, the order of g mod p^α is exactly $\varphi(p^\alpha)$.

Let r_k be the order of g^k modulo p^α and consider two cases:

1. k is odd. From $g^{kr_k} = (g^k)^{r_k} \equiv 1 \pmod{p^\alpha}$, we have $\varphi\left(p^\alpha\right) \mid kr_k$. As k is odd $\Rightarrow 2^d \mid r_k$.
2. k is even. Then $g^{k\varphi(p^\alpha)/2} = \left(g^{\varphi(p^\alpha)}\right)^{k/2} = 1^{k/2} = 1 \pmod{p^\alpha}$

   $\Rightarrow$ by the definition of r_k, $r_k \mid \varphi\left(p^\alpha\right)/2$. But $2^d \| \varphi(p^\alpha) \Rightarrow 2^d \nmid r_k$.

In summary. $\mathbb{Z}_{p^2}^*$ may be partitioned into two sets of equal size: those which may be written as g^k with k odd, for which $2^d \mid r_k$, and those which may be written as g^k with k even, for which $2^d \nmid r$. Thus with probability $1/2$ the integer 2^d divides the order r of a randomly chosen element of $Z_{p^\alpha}^*$, and with probability $1/2$ it does not. The lemma is established.

Corollary. Let x be a uniformly random chosen element of $\mathbb{Z}_{p^\alpha}^*$. Then for any nonegative integer $d_x = 0, 1, \ldots$, the probability that 2^{d_x} is the largest pover of 2 dividing the order of x mod p^α is $\leqslant 1/2$.

Proof. The lemma is $\Pr[d_x \geq d] = \frac{1}{2}$. Or $\Pr[d_x \leq d-1] = \frac{1}{2}$.

So $\Pr[d_x = 0], \Pr[d_x = 1], \cdots \leq \frac{1}{2}$.

Proof of theorem. Note that choosing x uniformly at random from $\mathbb{Z}_N^*$ is equivalent to choosing x_j independently and uniformly at random from $\mathbb{Z}^*_{p_j^{\alpha_j}}$, and requiring that $x \equiv x_j \left(\bmod p_j^{\alpha_j}\right)$ for each $j \in [l]$.

To prove the theorem. it is equivaled to prove:

$$\Pr\left[r \text{ is odd or } x^{r/2} \equiv -1 \pmod N\right] \leqslant \frac{1}{2^{l-1}}. \qquad (*)$$

Let r_j be the order of x_j modulo $p_j^{\alpha_j}$. Let $2^{d_j} \| r_j$ (the largest power of 2 that divides r_j). And let $2^d \| r$.

1. If r is odd, because $r_j \mid r$ for $j \in [l]$, it implies that all r_j are odd, hence $d_j = d = 0 \quad \forall j \in [l]$.
2. If $x^{r/2} \equiv -1 \pmod N, N \left| x^{r/2} + 1 \Rightarrow p_j^{\alpha_j} \right| x^{r/2} + 1 \quad \forall j \in [l]$.

   This implies that $r_j \nmid r/2$. Otherwise $p_j^{\alpha_j} \mid x^{r/2} - 1 \Rightarrow p_j^{\alpha_j} \mid 2$, contradicts with the fact that $p_j \geq 3, \alpha_j \geq 1$.

   However, $r_j \mid r \; \forall j \in [l]$, hence $d_j = d \quad \forall j \in [l]$.

Therefore: When the event in (*) holds, all d_j must take the same value for all $j \in [l]$:

$$\begin{aligned}\Pr[d_1 = \cdots = d_l] &= \sum_{i=0}^{\infty} \Pr[d_1 = i] \prod_{j=2}^{\infty} \Pr[d_j = i] \\ &\leq \sum_{i=0}^{\infty} \Pr[d_1 = i] \cdot \frac{1}{2^{l-1}} = \frac{1}{2^{l-1}}.\end{aligned}$$

□

Remark 1. Shor's algorithm can factorize integers with constant probability in poly $(\log N)$ time on quantun computer. The best-known classical algorithm takes time

$$\exp\left[\left(\sqrt[3]{\frac{64}{9}} + o(1)\right)(\log n)^{1/3}(\log\log n)^{2/3}\right].$$

Shor's algorithm gives a superpolynomial quentum speedup.

Remark 2. Shor's algorithm has many extensions.

Example 1. Computing discrete logarithms

Problem: Given $g \in \mathbb{Z}_p$ and $a \in \mathbb{Z}_p$ where g is a primitive root. Find x so that $g^x \equiv a \pmod p$ (i.e., $x = \log_g a$)

Interesting fact: Historically, Peter W. Shor first found an efficient quantum algorithm for the discrete logarithm problem, and then found the factorization algorithm.

Reference.

Childs and van Dam. Quantum algorithms for algebraic problens. Rev. Mod. Physics 2010, arxiv: 0812.0380.

Example 2. Deeper in number theory.

Can solve the Pell's equation

- Input: $d \in \mathbb{N}$, d is not a square;
- Solve: find nontrivial solution of $x^2 - dy^2 = 1$.

Denote the smallest nontrivial solution of $x^2 - dy^2 = 1$ as (x_1, y_1). All the solutions can be written as $x_n + y_n\sqrt{d} = \left(x_1 + y_1\sqrt{d}\right)^n$ for $n \in \mathbb{N}$. There exists an algorithm for finding (x_1, y_1) in time poly $(\log d)$ introduced by Hallgren.

Reference.

Hallgren. Polynomial-time quantum algorithms for Pell's equation and the prinpical ideal problem. JACM 2007, earlier version at STOC 2002.

Eisentraeger. Hallgren, Kitaev, Song, A quantum algorithm for computing the unit group of an arbitrary degree number field. STOC 2014.