Lecture 12.

1) Reduction of Factoring to Order finding.

2) Quantum algorithm for order finding.

3) Complexity of Shor's algorithm for factoring.

In this chapter we review Shor's famous algorithm for factoring. Classically there is no known classical polynomial method, i.e the classical methods are $O(\quad)$. Note however that there is no proof that a polynomial (classical method) of complexity $O((\ln N)^\alpha)$ does not exist!

As we will see Shor's quantum algorithm has a total complexity of $O((\ln N)^3 \ln\ln N)$. The strategy is as follows: first we reduce factoring to a probabilistic method for order finding in modular arithmetic. This goes back to Miller (?) ~1976. Then we will recognize that order finding is a particular case of period finding for the modular exponential function. We will be able to show that the "black box" representing the modular exponential can be realized with a polynomial number of gates. Combining these results with the QFT leads immediately to Shor's algorithm.

1) Reduction of Factoring to Order Finding.

Let N be an integer to be factored. We will suppose that

$$N = p_1^{e_1} p_2^{e_2} \cdots p_K^{e_K} \quad \text{with } p_i \neq 2 \text{ and } K \geq 2.$$

Indeed powers of 2 are easily recognized and extracted

(even numbers) and if $N = p^e$ there exist efficient classical methods to find p & e.

Consider the additive group $\left(\frac{\mathbb{Z}}{N\mathbb{Z}}, +\right)$ with elements $\{0, \ldots, N-1\}$ and modulo N addition.

[a] * Choose randomly with uniform probability $a \in \{2 \ldots N-1\}$ and compute

$$d = GCD(a, N).$$

This greatest common divisor can be computed by Euclid's algorithm in $\sim (\ln N)^3$ steps.

[b] * If $d > 1$: we have a factor of N. We keep this factor and start again at [a].

[c] * If $d = 1$ we find the "order of a mod N" i.e we find the smallest integer r such that :

$$a^r \equiv 1 \mod N.$$

Remark: For this step there is no known polynomial method and that is where Shor's algorithm fits,

[d] * Suppose r is odd : output failure and go back to [a].

[c] * Suppose r is even. Then

$$a^r - 1 = (a^{\frac{r}{2}} - 1)(a^{\frac{r}{2}} + 1).$$

Since N divides $a^r - 1$ we have three possibilities:

[c1] N divides $a^{\frac{r}{2}} - 1$. But this is impossible since we would have $a^{\frac{r}{2}} \equiv 1 \pmod N$ so r would not be the (smallest) order.

[c2] N divides $a^{\frac{r}{2}} + 1$. Then output failure and go back to [a].

[c3] N shares non-trivial factors with both $(a^{\frac{r}{2}} - 1)$ and $(a^{\frac{r}{2}} + 1)$. In other words

$$d_\pm = GCD\left(a^{\frac{r}{2}} \pm 1, N\right) \quad \text{is non trivial}$$

and we have two factors d_+ and d_- of N. This step can be done again in $(\ln N)^3$ steps thanks to Euclid's algorithm.

[f] * Go back to [a].

Summarizing we see that steps [a] → [e] enable to extract one or two factors for N as long as we do not output "failure" (and then choose some other $a \in \{2, \dots, N-1\}$).

What is the probability of success for one run? The answer is provided by the following Lemma proven by using the Chinese Remainder Theorem. For the proof we refer to the literature [e.g. Appendix in book of Chuang & Nielsen].

Lemma:

Let $N = p_1^{e_1} \dots p_k^{e_k}$; $p_i \neq 2$; $k \geq 2$. Choose a randomly uniformly in $\{2 \dots N-1\}$. Then

$$\text{Prob}\left\{ r \text{ is even } \& \; a^{r/2} \not\equiv -1 \mod N \right\} \geq \frac{1}{2}.$$

This is enough to ensure success by running the algorithm a large number of times (but finite with respect to $\log N$).

2) Quantum algorithm for order finding.

Given $a \in \{2, \dots N-1\}$ we want to find the smallest integer r such that

$$a^r \equiv 1 \pmod N .$$

We recognize that r is the period of the modular exponential:

$$f_a : \frac{\mathbb{Z}}{N\mathbb{Z}} \longrightarrow \frac{\mathbb{Z}}{N\mathbb{Z}}$$

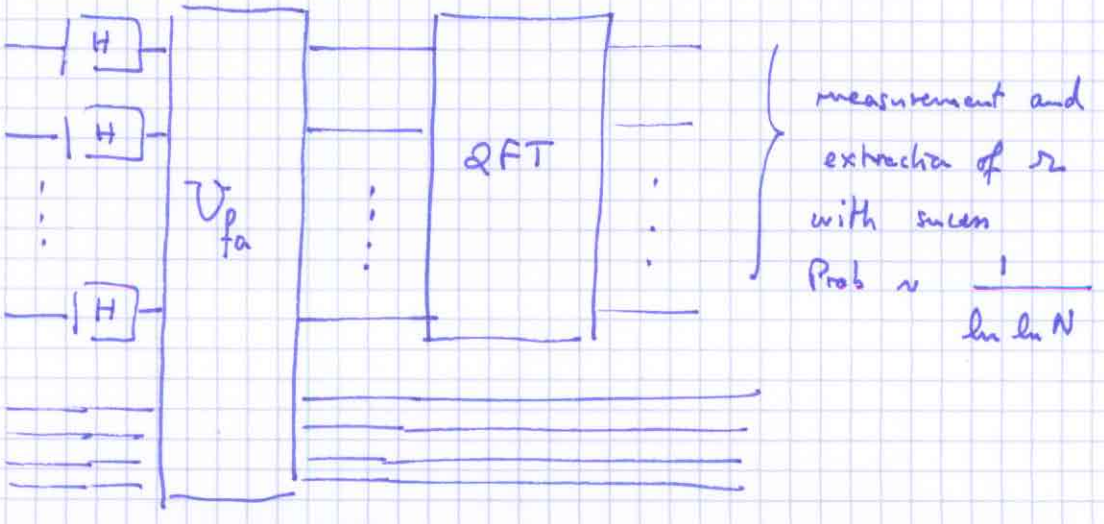
$$x \longmapsto f_a(x) = a^x \pmod N .$$

Indeed $f_a(x+r) = a^x a^r \bmod N = a^x \bmod N$ and r is the smallest such integer so it is the period of f.

Suppose now we are given a black box performing U_{f_a}:

$$U_{f_a} |x, 0\rangle = |x, a^x\rangle$$

Then we can simply use the algorithm for period finding which has circuit:



* We know that the QFT can be realized with $O((\ln N)^2)$ gates.

* We will show that U_{pa} can be realized with $O((\ln N)^3)$ gates.

Let $x = x_{m-1} 2^{m-1} + \dots + 2^0 \cdot x_0$ and set

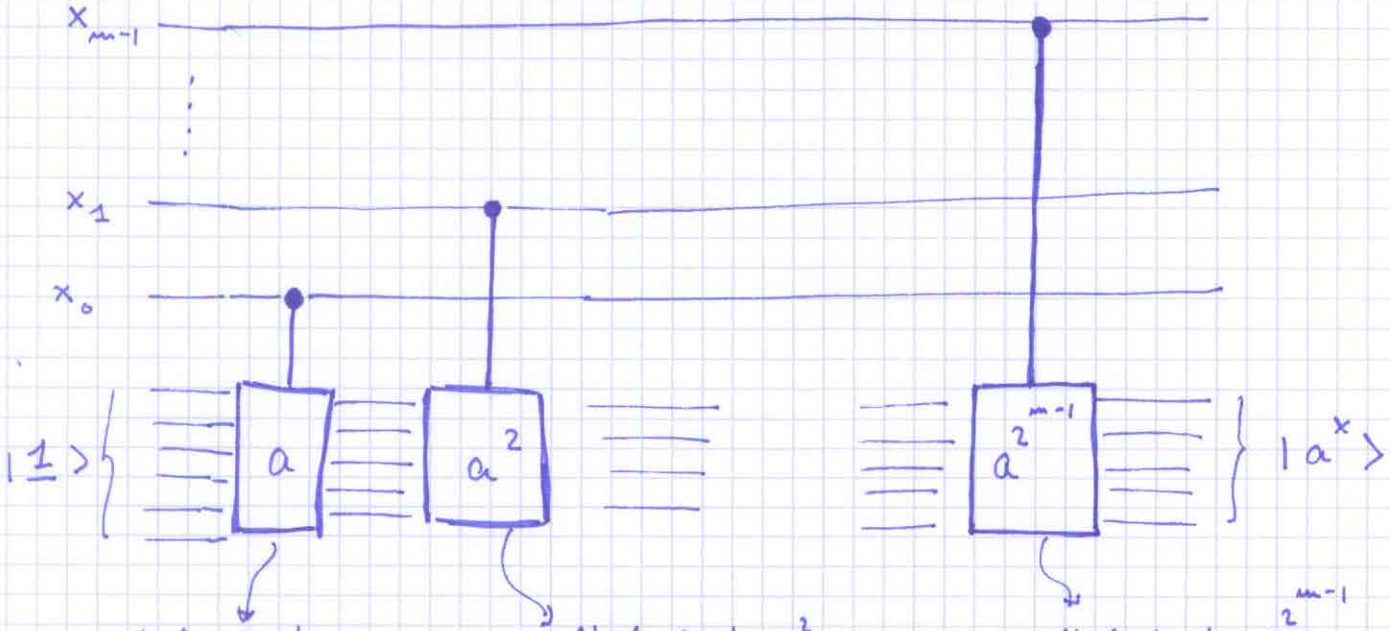
$M = 2^m$. Look at $a^x \bmod N$.

$$a^x \pmod N = \left(a^{2^{m-1}}\right)^{x_{m-1}} \dots \left(a^{2^0}\right)^{x_0} \pmod N.$$

So if we know all even powers of a, namely

$$a, a^2, \dots, a^{2^{m-1}}$$

we can use the circuit:



There are "controlled multiplications": multiply by a^{2^j} if $x_j = 1$.

* Now this uses $O(m)$ controlled multiplications. Each multiplication can be done in $O(m^2)$ steps (or gates).

* Then the modular exponential

$$U_{f_a}(|x\rangle \otimes |\underline{1}\rangle) = |x\rangle \otimes |a^x \bmod M\rangle$$

can be realized by a circuit of size $O(m^3) = O((\ln M)^3)$.

* Now remember that in period finding with $n < N$ we choose to work with m bits where $2^m = M \sim N^2$. So the total size of this circuit is $O((\ln N)^3)$ again.

3) Shor's Algorithm and its complexity.

Here we summerize the resulting algorithm.

input: an odd number N with at least two distinct prime factors.

output: a non trivial factor of N.

1. Choose randomly uniformly $a \in \{2, \dots N-1\}$.
2. Compute $d = GCD(a, N)$ by Euclid's algorithm. If $d > 1$ output the factor d.
3. If $d = 1$ compute the order of a mod N (i.e $a^{\hat{}} = 1 \bmod N$) by using Shor's quantum circuit. Work with m qbits where $M = 2^m \sim N^2$.

4. Check that the output of Shor's circuit i.e the number r satisfies $a^r \equiv 1 \bmod N$. If not output "failure".

5. If $a^r \equiv 1 \bmod N$ then check if r is odd or if $a^{r/2} \equiv -1 \bmod N$. Output "failure" if this happens.

6. Otherwise (no "failure") compute $\mathrm{GCD}(a^{r/2} \pm 1, N)$ by Euclid's Algorithm.

What is the probability of succes? The period finding works with prob $\sim \frac{1}{\ln \ln N}$ as we saw in the previous chapter (point 4).

Moreover (point 5) does not happen with prob $\geq \frac{1}{2}$. So the probability of succes of one run is $O\left(\frac{1}{\ln \ln N}\right)$. By making $O(\ln \ln N)$ runs we can make this probability close to 1.

What is the total complexity?

First, for one run we have: (point 2) $\sim O((\ln N)^3)$. Euclid

(point 3) $\sim \underbrace{O((\ln N)^3)}_{\text{Shor : modular exponential}} + \underbrace{O((\ln N)^2)}_{\text{QFT}}$

U_{f_a}

(point 4) $\sim O((\ln N)^2)$

(point 5) $\sim O((\ln N)^2)$ } (multiplications).

(point 6) $\sim O((\ln N)^3)$ Euclid

So for one num we have $O((\ln N)^3)$ steps in total.

Since we need $O(\ln \ln N)$ nums the total time needed to find one non trivial factor is $\sim O((\ln N)^3 \ln \ln N)$.

Appendix: Main idea of RSA cryptosystem.

Alice wants to send a message to Bob. Bob generates a public key that Alice uses to encrypt the message. Only Bob knows how to "invert" the public key in order to decode.

* Bob takes two large primes p & q. He keeps them secret. Bob computes $N = pq$; $\varphi(N) = (p-1)(q-1)$ [Euler function = nb of coprimes with N]. He selects $e < \varphi(N)$ randomly such that $GCD(e, \varphi(N)) = 1$.

* Bob reveals (N, e) to everybody. This is the public key.

* Alice has a message (e.g string of bits) and converts it to a number $a < N$. She checks that $GCD(a, N) = 1$; in fact this is very likely.

  Now Alice encodes her message by doing

$$b = a^e \pmod N .$$

  The number b is the encoded message sent to Bob.

* How does Bob decode? Since he knows p & q he knows $\varphi(N) = (p-1)(q-1)$. Then Using Euclid's algorithm he find d s.t

$$e\,d = 1 \mod \varphi(N).$$

This d exits for sure because $GCD(e, \varphi(N)) = 1$.

Then he computes

$$b^d = a^{ed} = a^{1 + k\varphi(N)} = a \pmod N$$

Here we have used $a^{\varphi(N)} = 1 \pmod N$. This is Euler's theorem and follows from $GCD(a, N) = 1$.

So the decoded message is $b^d = a \pmod N$.

* Remark 1. The inverse d is Bob's secret! In order for somebody else to compute $\varphi(N)$ must be known. And this requires p & q the factors of N. So attack from Eve is to factor N into $p \cdot q$.

* Remark 2. Another possible attack from Eve is to find the order of a mod N : $a^r = 1 \bmod N$ for the smallest integer r. Indeed once she knows r she computes $\tilde{d}$ s.t $\tilde{d} \cdot e = 1 \bmod r$. and then she can decode because

$$b^{\tilde{d}} = a^{\tilde{d}e} = a^{1 + kr} = a. \pmod N.$$

If Eve had a quantum computer she would perform both kind of attacks in time $O((\ln N)^3 \ln\ln N) = O[(\text{length of message in bits})^3 \ldots]$