QWORLD

Quantum Summer School

Lecture 10: Shor’s Algorithm

Özlem Salehi Köken

Factoring

Given N = p.q, what are the integers p and q ?

Easy when

- N is prime
- N is even
- N is of the form x^y for some integers x and y

Factoring

Given N = p.q, what are the integers p and q ?

Easy when

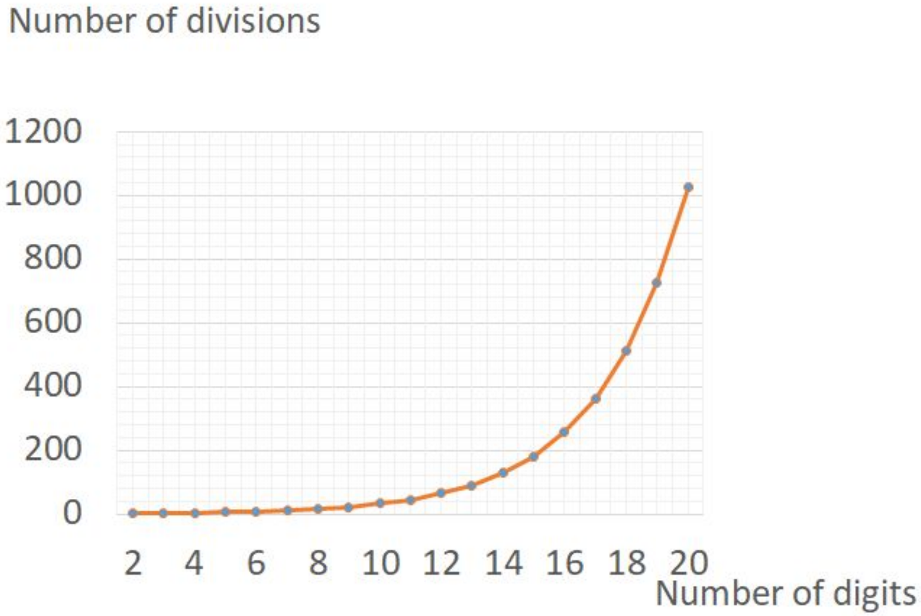
- N is prime
- N is even
- N is of the form x^y for some integers x and y

A Simple Algorithm

15: Try 2,3

143: Try 2,3,5,7,11

Approximately square root of N trials for integer N with logN qubits



Best Known Classical Algorithm

General number field sieve (1993)

$$\exp\left(\left(\sqrt[3]{\frac{64}{9}}+o(1)\right)(\ln n)^{\frac{1}{3}}(\ln\ln n)^{\frac{2}{3}}\right)=L_n\left[\frac{1}{3},\sqrt[3]{\frac{64}{9}}\right]$$

Any efficient (polynomial time) algorithm ?

Claim: If one can solve the order finding algorithm efficiently, then factoring can be solved efficiently as well.

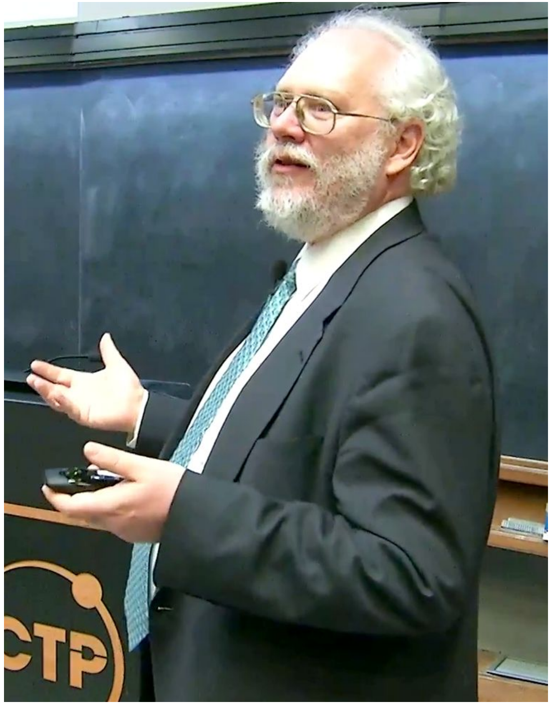
Reminder:

For positive integers x and N where $x < N$ with no common factors, order of x is the least positive integer r such that $x^r = 1 \pmod N$.

Claim in more detail: If you have a procedure for finding the order of $x \; Mod \; N$, then picking some random x's and repeating the procedure for multiple times, you can factor N.

Shor’s Algorithm (1994)

Order finding can be solved efficiently using quantum computers!



https://www.youtube.com/watch?v=6qD9XEITpCE

Shor's Algorithm

Algorithm

- Pick x randomly in the range 1 to $N-1$, such that $gcd(x, N) = 1$.
- Use order finding algorithm to find order of $x \pmod N$, which will be denoted by r.
- If r is even, and $x^{r/2} \neq -1 \pmod N$, then compute $gcd(x^{r/2} - 1, N)$ and $gcd(x^{r/2} + 1, N)$.
- Test to see if one of these is a non-trivial factor. If so return, otherwise the algorithm fails. If that is the case, repeat.

Main algorithm

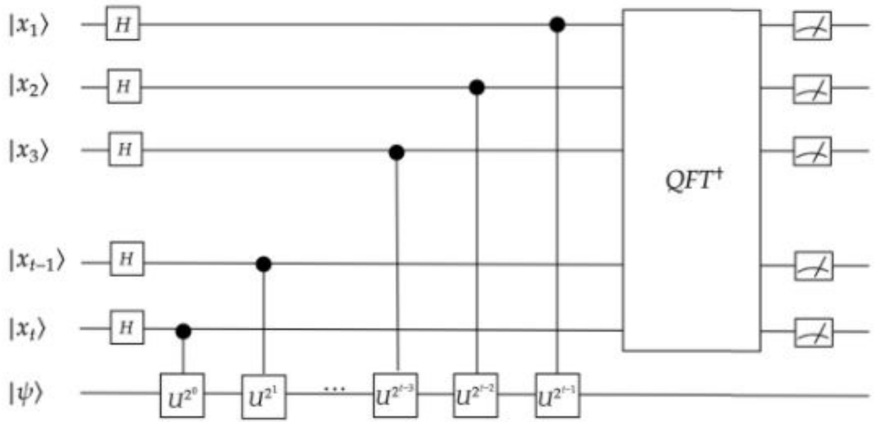
Remark: If r is even, and $x^{r/2} \neq -1 \pmod N$, then it can be proven that either $gcd(x^{r/2} - 1, N)$ or $gcd(x^{r/2} + 1, N)$ should be a factor of N. Such an r is found with probability greater than 1/2. You can check the two theorems at the end of the notebook.

Theorem 1 Suppose N is an L bit composite number and x is a non-trivial solution to the equation $x^2 = 1 \pmod N$ in the range $1 \leq x \leq N$, that is neither $x = 1 \pmod N$ nor $x = N - 1 = -1 \pmod N$. Then at least one of $gcd(x-1, N)$ and $gcd(x+1, N)$ is a non-trivial factor of N that can be computed using $O(L^3)$ operations.

Theorem 2 Suppose $N = {p_1}^{l_1} \dots {p_m}^{l_m}$ is the prime factorization of an odd composite positive integer. Let x be an integer uniformly chosen at random, such that $0 \leq x \leq N-1$ and x is co-prime to N. Let r be the order of $x \pmod N$. In such a case,

$$P(\text{r is even and } x^{r/2} \neq -1 \pmod N) > 1 - \frac{1}{2^{m-1}}.$$

Let’s go back to phase estimation



$\frac{1}{2^{t/2}}\sum_{k=0}^{2^t-1}|k\rangle|\psi\rangle$ into the state $\frac{1}{2^{t/2}}\sum_{k=0}^{2^t-1}|k\rangle U^k|\psi\rangle$

Let's go back to order finding

$\frac{1}{2^{t/2}} \sum_{k=0}^{2^t-1} |k\rangle|\psi\rangle$ into the state $\frac{1}{2^{t/2}} \sum_{k=0}^{2^t-1} |k\rangle U^k|\psi\rangle$

$U|y\rangle \rightarrow |xy \pmod N\rangle$ Operator we used in order finding

Conclusion: $$\frac{1}{2^{t/2}} \sum_{k=0}^{2^t-1} |k\rangle|1\rangle \rightarrow \frac{1}{2^{t/2}} \sum_{k=0}^{2^t-1} |k\rangle|x^k \pmod N\rangle.$$

$$\frac{1}{2^{t/2}} \sum_{k=0}^{2^t-1} |k\rangle |x^k \pmod N\rangle.$$

$$|x_0\rangle + |x_0 + r\rangle + |x_0 + 2r\rangle + |x_0 + 3r\rangle + \cdots$$

$$\Big\downarrow \text{ Inverse QFT}$$

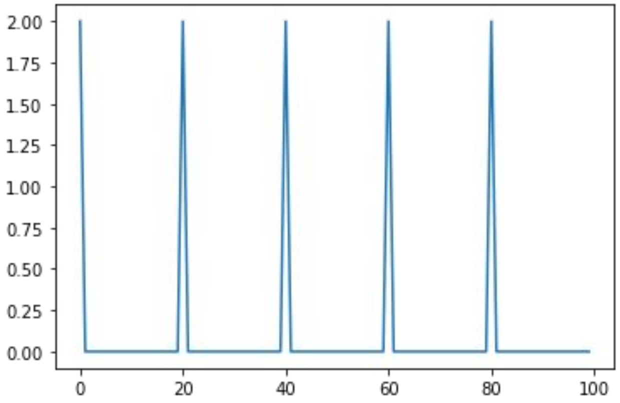
$$|0\rangle + |2^t/r\rangle + |2 \cdot 2^t/r\rangle + |3 \cdot 2^t/r\rangle + \ldots$$

When we measure the first register, we observe $s \cdot 2^t/r$ for some s. Hence by dividing with 2^t, we obtain an estimate for $\frac{s}{r}$, from which we extract r by continued fractions algorithm.

Intuition

Task 2

Create the following list in Python (1 0 0 0 0 1 0 0 0 0 ... 1 0 0 0 0) of length $N = 100$ where every 5'th value is a 1. Then compute its DFT using Python and visualize.



Maths

- Apply inverse QFT to the first register.

$$\sqrt{\frac{r}{2^t}} \sum_{k=0}^{2^t-1} \frac{1}{\sqrt{2^t}} \sum_{x=0}^{2^t/r-1} e^{-\frac{2\pi i(rx+x_0)k}{2^t}} |k\rangle$$

. Probability of observing a particular state $|k\rangle$ is given by $\frac{1}{r}\left|\frac{r}{2^t}\sum_{x=0}^{2^t/r-1} e^{-\frac{2\pi irxk}{2^t}}\right|^2$.

Probability peaks around the integer multiples of $2^t/r$ so that with probability (approximately) $\frac{1}{r}$, one of the states $|s \cdot 2^t/r\rangle$ is observed for $s = 0, \ldots, r-1$.

Summary

- We have an efficient (polynomial time) quantum algorithm for factoring integers.
- Tricky part is to implement order finding efficiently
  - QFT is efficient
  - Modular exponentiation can be done efficiently

Implementation on real hardware?

Factoring 21

Martín-López, Enrique; Enrique Martín-López; Anthony Laing; Thomas Lawson; Roberto Alvarez; Xiao-Qi Zhou; Jeremy L. O'Brien (12 October 2012). "Experimental realization of Shor's quantum factoring algorithm using qubit recycling".