Shor's Factoring Algorithm

Anurag Pandey

Factoring Problem

- Problem: Given a large integer N (typically several hundred digits long), factorize n as a product of primes.
- We will assume that N = pq where p and q are large unknown primes. We must determine p and q
- Best known algo on a classical computer:

$$O\left(c\exp\left\lfloor(\lg N)^{1/3}(\lg\lg N)^{2/3}\right\rfloor\right)$$

Sub-exponential!!
(since the input size is logN)

- Can Quantum Computer solve it efficiently??

Quantum Computer

- Qubits

- **Qubit vs Classical Bit**

A qubit has a few similarities to a classical bit, but is overall very different. Like a bit, a qubit can have two possible values—normally a 0 or a 1. The difference is that whereas a bit *must be* either 0 or 1, a qubit *can be* 0, 1, or a superposition of both.

Representation:

The two states in which a qubit may be measured are known as basis states (or basis vectors). As is the tradition with any sort of quantum states, they are represented by Dirac—or "bra–ket"—notation. This means that the two computational basis states are conventionally written as |0> and |1> (pronounced "ket 0" and "ket 1").

- A pure qubit state is a linear superposition of the basis states. This means that the qubit can be represented as a linear combinationof |0> and |1> and :

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

  - where α and β are probability amplitudes and can in general both be complex numbers.
- Complex probability-amplitude and L-2 norm

$$|\mathbf{x}|_1 = \sum_{r=1}^{n} |x_r|.$$

$$|\mathbf{x}| = \sqrt{\sum_{k=1}^{n} |x_k|^2},$$

Superposition

- Superposition is essentially the ability of a quantum system to be in multiple states at the same time
- Due to superposition, a quantum computer can process a vast number of calculations simultaneously
  - we can store 10^{300} numbers on our 1,000 particles simultaneously
- Then, by performing various operations on the particles and on some auxiliary ones— perhaps hitting them with a sequence of laser pulses or radio waves— we can carry out an algorithm that transforms all 10^{300} numbers(each one a potential solution) at the same time

- If at the end of doing that, we could read out the particles' final quantum state accurately, we really would have a magic computer: it would be able to check 10^{300} possible solutions to a problem, and at the end we could quickly discern the right one

- Unfortunately, there is a catch. When the particles are measured (as is necessary to read out their final state), the rules of quantum mechanics dictate that the measurement will pick out just one of the 10^{300} possibilities at random and that all the others will then disappear. We would seem to be no better off than if we used a classical computer and tried out one randomly chosen possible solution—in either case, we end up knowing about only one such possible solution.

Speedups in Quantum?

- Can't pick a needle from the haystack
- [Grover] **Blackbox Quantum Algorithm:** Quadratic speed-up over a brute force search
- *the reason we get a quadratic speedup is that quantum mechanics is based on the L_2 norm rather than the L_1 norm.*
- **Classically**, if there are N solutions, only one of which is right, then after one query we have a 1/N probability of having guessed the right solution, after two queries we have a 2/N probability, after three queries a 3/N probability, and so on. Thus we need ~N queries to have a non-negligible (i.e. close to 1) probability of having guessed the right solution.
- But **quantumly**, we get to apply linear transformations to vectors of *amplitudes*, which are the square roots of probabilities. So the way to think about it is this: **after one query we have a 1/√N amplitude of having guessed the right solution**, after two queries we have a 2/√N amplitude, after three queries a 3/√N amplitude, and so on. So after **T queries, the amplitude of having guessed a right solution is T/√N,** and the probability is $|T/\sqrt{N}|^2 = T^2/N$. Hence the probability will be close to 1 after only $T \approx \sqrt{N}$ queries.

Further speedup?

- Happily, we still have tricks we can play to wring some advantage out of the quantum particles. Amplitudes can cancel out when positive ones combine with negative ones, a phenomenon called **destructive interference**. So a good quantum computer algorithm would ensure that **computational paths leading to a wrong answer would cancel** out in this way. It would also ensure that the paths leading to a correct answer would all have amplitudes with the same sign— which yields constructive interference and there-by boosts the probability of finding them when the particles are measured at the end.

- For which computational problems can we choreograph this sort of interference , using fewer steps than it would take to solve the problem classically? – Factoring! [Peter Shor]

Shor's Algorithm Idea

1. For a number N = pq, choose randomly a number x<N s.t GCD(x,N) = 1
2. Find the period r of $x^a \pmod N$ on a Quantum computer (using Quantum Fourier Transform)

- i.e. $x^r = 1 \pmod N$
- $(x^{r/2} - 1)(x^{r/2} + 1) = 1 \pmod N$

3. GCD($x^{r/2} - 1$,N) and GCD($x^{r/2} + 1$, N) gives the desired factor using Euclid's algo

❑ Quantum computer is relevant only for step 2.

- Other steps efficiently computable for classical computers too

- For $x \in R$ and $a \geq 0$, the value of $x^a \pmod n$ can also be determined in polynomial time and space using repeated squaring technique.

Example: To compute $x^{183} \bmod n$, first write 183 in binary as 10110111. Then

$$x^{183} = x^{128} x^{32} x^{16} x^4 x^2 x^1$$

where the powers $x^2, x^4, x^8, \ldots$ are found by successively squaring mod n, then multiplied together (mod n) two at a time only. This way if n has 100 digits, say, then intermediate computations have at most 200 digits.

$$x^r = 3^{14} = 3^{2^3+2^2+2^1} = 3^{2^3} \cdot 3^{2^2} \cdot 3^{2^1} = ((3^2)^2)^2 \cdot (3^2)^2 \cdot 3^2$$

- Euclid's Algo computes GCD(a,b) in O(log(max{a,b}))

Discrete Fourier Transform

The *Discrete Fourier Transform* of order q is the unitary matrix

$$U^*U = UU^* = I$$

$$U_q = \frac{1}{\sqrt{q}} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \zeta & \zeta^2 & \cdots & \zeta^{q-1} \\ 1 & \zeta^2 & \zeta^4 & \cdots & \zeta^{2(q-1)} \\ 1 & \zeta^3 & \zeta^6 & \cdots & \zeta^{3(q-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \zeta^{q-1} & \zeta^{2(q-1)} & \cdots & \zeta^{(q-1)^2} \end{pmatrix}$$

where $\zeta = e^{2\pi i/q}$.

If q is a product of small prime factors, then Uq can be factored as a product of a small number (polynomial in log(q)) of simpler unitary transformations, each representing the action of a quantum gate acting on only one or two qubits. (E.g. if $q = 2^l$ then only $l(l+1)/2$ such gates are necessary.)

Shor's algorithm

Given n, find $2n^2 < q < 3n^2$ such that q is a product of small prime factors. We'll suppose $q = 2^\ell$.

Construct a quantum computer with $q^2 = 2^{2\ell}$ qubits (plus additional qubits for 'workspace'). The base states are denoted

$$|a, b\rangle = |a\rangle|b\rangle$$

where a, b are binary vectors (i.e. vectors with entries 0,1) of length ℓ. Equivalently, a and b (called *registers 1 and 2*) are integers $< q$ written in binary.

At any time, the state of the system is given by

$$|\psi\rangle = \sum_{a=0}^{q-1} \sum_{b=0}^{q-1} c_{a,b}|a, b\rangle$$

where

$$c_{a,b} \in \mathbb{C}, \quad \sum_{a,b} |c_{a,b}|^2 = 1$$

and $|c_{a,b}|^2$ is the probability that a measurement of the system will find the state to be $|a, b\rangle$.

Example

- We simulate a quantum computer attempting to factor n= 55. This leads to q =2^{13}=8192.
- Let's fix x= 13. (This happens to have order r= 20.)

Step 1: Initial state.

- Prepare the computer in initial state $|\psi\rangle = |0,0\rangle.$
- Then apply the quantum gate $R = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$

to each of the ℓ qubits in the first register; this leaves the computer in the state

$$|\psi\rangle = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle|0\rangle.$$

- For example for $q=2^2$ we have

$$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 & & & & & & \\ 1 & -1 & & & & & & \\ & & 1 & 1 & & & & \\ & & 1 & -1 & & & & \\ & & & & 1 & 1 & & \\ & & & & 1 & -1 & & \\ & & & & & & \ddots & \\ & & & & & & & \begin{matrix} 1 & 1 \\ 1 & -1 \end{matrix} \end{bmatrix}$$ (applies R to a_0)

$$\times\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & & 1 & & & & & \\ & 1 & & 1 & & & & \\ 1 & & -1 & & & & & \\ & 1 & & -1 & & & & \\ & & & & \ddots & & & \\ & & & & & 1 & & 1 \\ & & & & & & 1 & \\ & & & & & 1 & & -1 \\ & & & & & & 1 & \\ & & & & & & & -1 \end{bmatrix}$$ (applies R to a_1)

$$\times\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \frac{1}{2}(|00,00\rangle + |10,00\rangle + |01,00\rangle + |11,00\rangle)$$

where all vectors have length $q^2 = 16$ and all matrices are 16×16.

For our example: after step 1

$$|\psi\rangle = \frac{1}{\sqrt{8192}}\left(|0,0\rangle + |1,0\rangle + |2,0\rangle + \cdots + |8191,0\rangle\right)$$

Step 2: Apply modular exponentiation.

Fix a randomly chosen x between 1 and n.

Apply the reversible transformation

$$|a, 0\rangle \mapsto |a, x^a \bmod n\rangle$$

to the state of the quantum computer. This transforms the state $|\psi\rangle$ from

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle|0\rangle$$

to

$$\frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a\rangle|x^a \bmod n\rangle.$$

- Our example after Modular exponentiation

$$|\psi\rangle = \frac{1}{\sqrt{8192}} \Big(|0,1\rangle + |1,13\rangle + |2,13^2 \bmod 55\rangle + \cdots + |8191,13^{8191} \bmod 55\rangle \Big)$$

$$= \frac{1}{\sqrt{8192}} \Big(|0,1\rangle + |1,13\rangle + |2,4\rangle + \cdots + |8191,2\rangle \Big)$$

Step 3: Observe register 2.

Measure the second register only. We observe the second register to be in a base state $|k\rangle$ where k is some power of x mod n (and all powers of x mod n are equally likely to be observed).

This measurement projects the state $|\psi\rangle \in \mathbb{C}^{q^2}$ into the q-dimensional subspace spanned by all base states $|a, k\rangle$ for the fixed k whose value we have observed.

Thus the new state is

$$|\psi\rangle = \frac{1}{\sqrt{M}} \sum_{a \in A} |a, k\rangle$$

where A is the set of all $a < q$ such that x^a mod n is k and $M = |A|$. That is,

$$A = \{a_0, a_0+r, a_0+2r, \ldots, a_0+(M-1)r\}$$

where $M \approx \frac{q}{r} \gg 1$. Thus

$$|\psi\rangle = \frac{1}{\sqrt{M}} \sum_{d=0}^{M-1} |a_0+dr, k\rangle.$$

- For our example:-

All ten powers of x mod 55 are equally likely to be observed. Suppose we observe 28 as a power of x mod 55.

$$|\psi\rangle = \frac{1}{\sqrt{410}}\Big(|9,28\rangle + |29,28\rangle + |49,28\rangle + \cdots + |8189,28\rangle\Big)$$

Step 4: Discrete Fourier Transform of register 1.

Apply the Discrete Fourier Transform U_q to the first register. This transforms the state from

$$\frac{1}{\sqrt{M}} \sum_{d=0}^{M-1} |a_0 + dr, k\rangle$$

to

$$|\psi\rangle = \frac{1}{\sqrt{qM}} \sum_{c=0}^{q-1} \sum_{d=0}^{M-1} \exp(2\pi i \frac{c(a_0+dr)}{q}) |c, k\rangle$$

$$= \sum_{c=0}^{q-1} \frac{e^{2\pi i c a_0/q}}{\sqrt{qM}} \sum_{d=0}^{M-1} \exp(2\pi i \frac{cdr}{q}) |c, k\rangle$$

$$= \sum_{c=0}^{q-1} \frac{e^{2\pi i c a_0/q}}{\sqrt{qM}} \left(\sum_{d=0}^{M-1} \zeta^d \right) |c, k\rangle$$

where $\zeta = e^{2\pi i c r/q}$.

For Our Example:

$$|\psi\rangle = \sum_{c=0}^{8191} \frac{e^{2\pi i \cdot 9c/8192}}{\sqrt{3358720}} \left(\sum_{d=0}^{409} \zeta^d \right) |c, 28\rangle$$

where $\zeta = e^{2\pi i \cdot 20c/8192}$.

Step 5: Measure register 1.

Measure register 1. We observe register 1 to be in state $|c\rangle$ with probability

$$Pr(c) = \frac{1}{qM} \left| \sum_{d=0}^{M-1} \zeta^d \right|^2$$

where $\zeta = e^{2\pi i \frac{cr}{q}}$.

If $\frac{cr}{q}$ is not *very close* to an integer, then powers of ζ very nearly cancel out ('destructive interference') and such states $|c\rangle$ are extremely unlikely to be observed. Note that

$$\sum_{d=0}^{M-1} \zeta^d = \frac{1-\zeta^M}{1-\zeta}$$

is small in this case.

But if

$$\frac{cr}{q} \approx d$$

where d is an integer, then $\zeta \approx 1$ and

$$Pr(c) \approx \frac{M^2}{qM} = \frac{M}{q}$$

is much larger. Thus the observed probability distribution of c is concentrated around values such that

$$\frac{c}{q} \approx \frac{d}{r}$$

where d is an integer.

For Our Example:

The probability of observing register 1 to be in state $|c\rangle$ is

$$Pr(c) = \frac{1}{3358720} \left| \sum_{d=0}^{409} \zeta^d \right|^2$$

Let's say we observe register 1 to be in state $|4915\rangle$. (This happens with probability 4.4%.)

Step 6: Continued Fraction Convergents

For the observed value of c, we use a classical computer to find fractions d/r very close to c/q, hoping that this will give us the true order r of x mod n.

For this we use the method of continued fractions, computing the convergents d_1/r_1 to c/q for which the denominator $r < n$. Noting that all the fractions

$$\frac{d_1}{r_1}, \frac{2d_1}{2r_1}, \frac{3d_1}{3r_1}, \ldots$$

are close to c/q, it is reasnoable to try small multiples of r_1 as possible values of r. Odlyzko (1996) suggests trying

$$r_1,\ 2r_1,\ 3r_1, \ldots, \lfloor \log(n)^{1+\epsilon} \rfloor r_1$$

as possible values for r, checking whether x^r mod n gives 1 in each case, and repeating the experiment as often as necessary ($O(1)$ times on average, compared with $O(\log \log n)$ trials on average if multiples of r_1 are not considered).

Continued Fractions

- A real number α can be approximated by a set of positive integers $a_0, a_1, \ldots, a_n$ as

$$CF_n(\alpha) = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\cdots + \cfrac{1}{a_n}}}} = \frac{P_n}{Q_n}, \quad \text{where } P_n \text{ and } Q_n \text{ are integers.}$$

Example: If we decided to approximate π to four decimal places, we would have

$$\begin{aligned}
\pi &= 3.1415\ldots \\
&= 3 + \frac{1415}{10000} \\
&= 3 + \frac{1}{\frac{10000}{1415}} \\
&= 3 + \frac{1}{7 + \frac{95}{1415}} \\
&= 3 + \frac{1}{7 + \frac{1}{\frac{1415}{95}}} \\
&= 3 + \frac{1}{7 + \frac{1}{14 + \frac{85}{95}}} \\
&\approx 3 + \frac{1}{7 + \frac{1}{14}} \\
&= \frac{311}{99}
\end{aligned}$$

For Our Example:

$$\frac{c}{q} = \frac{4915}{8192} = \cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{1638}}}}$$

Convergents:

$$\frac{1}{1} = 1$$

$$\frac{1}{1+\frac{1}{1}} = \frac{1}{2}$$

$$\frac{1}{1+\cfrac{1}{1+\frac{1}{2}}} = \frac{3}{5}$$

$$\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{1638}}}} = \frac{4915}{8192}$$

We stop before the denominator exceeds $n = 55$:

$$r_1 = 5$$

Possible values for r are multiples of $r_1 = 5$:

a	$13^a \bmod 55$
5	43
10	34
15	32
20	1

Evidently $r = 20$. Now

r has to be even – else repeat !

$$y = 13^{10} \bmod 55 \;=\; 34$$

and the factors of $n = 55$ are

$$p = \gcd(y+1, n) = \gcd(35, 55) = 5;$$
$$q = \gcd(y-1, n) = \gcd(33, 55) = 11.$$

- This succeeds in factoring n 25% of the time; the remaining 75% of the time we obtain the trivial factors 1 and n

Remarks:

- The factoring problem is neither known nor believed to be NP-complete. To create his algorithm, Shor exploited certain mathematical properties of composite numbers and their factors that are particularly well suited to producing the kind of constructive and destructive interference that a quantum computer can thrive on. The NP-complete problems do not seem to share those special properties.

- Only few other algo with exponential speedup

- Fro structurd problem and not structureless– for structure less, brute force search- Quadratic Speedup

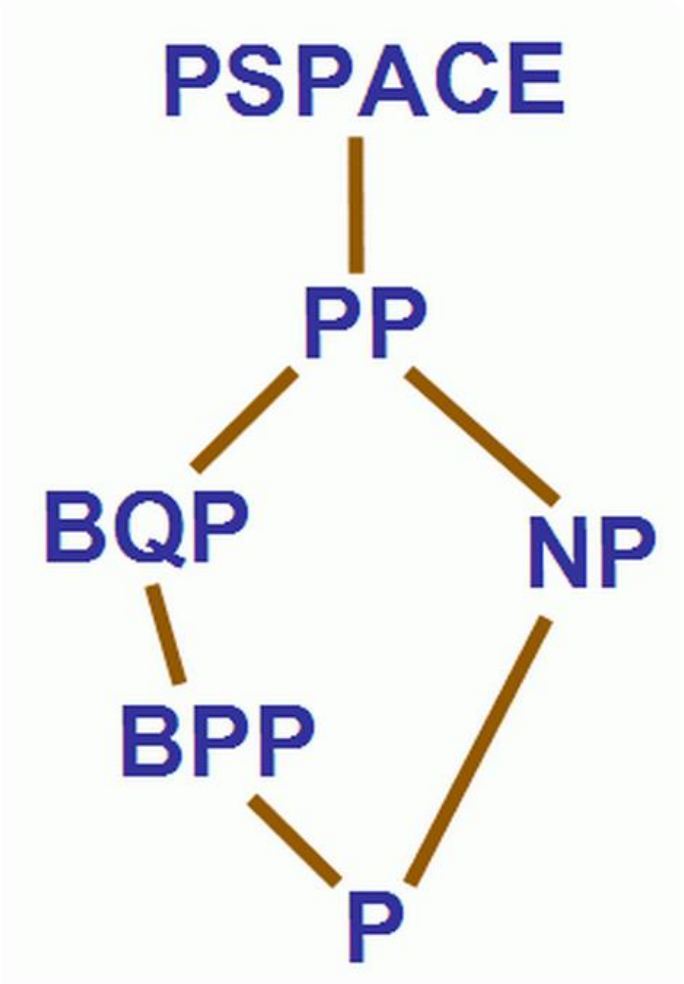
BQP

- Bounded-Error Quantum Polynomial-Time
- A language *L* is in **BQP** if and only if there exists a polynomial-time uniform family of quantum circuits $\{Q_n : n \in \mathbb{N}\}$,such that
  - For all $n \in \mathbb{N}$, Q_n takes *n* qubits as input and outputs 1 bit
  - For all *x* in *L*, $\Pr(Q_{|x|}(x) = 1) \geq \frac{2}{3}$
  - For all *x* not in *L*, $\Pr(Q_{|x|}(x) = 0) \geq \frac{2}{3}$

Relation to Classical Complexity Classes

- **BPP $\subseteq$ BQP**:
  - any time you were gonna flip a coin, you just apply a Hadamard gate instead
- **BQP $\subseteq$ EXP:**
  - quantum computers can provide *at most* an exponential advantage over classical computers.
  - because if you allow exponential slowdown, then a classical computer can just simulate the whole evolution of the state vector!
- **BQP $\subseteq$ PSPACE (**Bernstein and Vazirani)
- **BQP $\subseteq$ PP (**Adleman, DeMarrais, and Huang)
- **BPP $\neq$ BQP??**
  - whether quantum computing is more powerful than classical : Shor's Algorithm
- **NP $\not\subset$ BQP??**
  - **Even if P=NP, then also NP $\not\subset$ BQP??**

Currently-known inclusions



References:

- Shor's Algorithm or Factorizing Large Integers G. Eric Moorhouse, UW Math
- Quantum Computing Since Democritus [Scott Aaronson]
- http://en.wikipedia.org/wiki/Quantum_computer
- http://www.scottaaronson.com/blog/?p=208
- https://uwaterloo.ca/institute-for-quantum-computing/quantum-computing-101

EXTRA SLIDES

Sub-Exponential

- A problem is said to be sub-exponential time solvable if it can be solved in running times whose logarithms grow smaller than any given polynomial.
- i.e. a problem is in sub-exponential time if for every ε > 0 there exists an algorithm which solves the problem in time O($2^{n\hat{}\varepsilon}$). The set of all such problems is the complexity class **SUBEXP** which can be defined in terms of **DTIME** as follows.

$$\mathrm{SUBEXP} = \bigcap_{\varepsilon>0} \mathrm{DTIME}\left(2^{n^{\varepsilon}}\right)$$

What is required to build a quantum computer?

- Simply put: we need qubits that behave the way we want them to. These qubits could be made of photons, atoms, electrons, molecules or perhaps something else. Scientists at IQC are researching a large array of them as potential bases for quantum computers. But qubits are notoriously tricky to manipulate, since any disturbance causes them to fall out of their quantum state (or "decohere"). Decoherence is the Achilles heel of quantum computing, but it is not insurmountable. The field of quantum error correction examines how to stave off decoherence and combat other errors. Every day, researchers at IQC and around the world are discovering new ways to make qubits cooperate.

- decoherence (unwanted interaction between a quantum computer and its environment, which introduces errors). In particular, the bounds on what it is mathematically possible to program a computer to do would apply even if physicists managed to build a quantum computer with no decoherence at all

https://uwaterloo.ca/institute-for-quantum-computing/quantum-computing-101

So when will there be a real quantum computer?

- It depends on your definition. There are quantum computers already, but not of sufficient power to replace classical computers. A team of researchers from IQC and MIT hold the current world record for the most number of qubits used in an experiment . While practical quantum technologies are already emerging — including highly effective sensors, actuators and other devices — a true quantum computer that outperforms a classical computer is still years away. Theorists are continually figuring out better ways to overcome decoherence, while experimentalists are gaining more and more control over the quantum world through various technologies and instruments. The pioneering work being done today is paving the way for the coming quantum era.

https://uwaterloo.ca/institute-for-quantum-computing/quantum-computing-101