Ph/CS 219A
Quantum Computation

Lecture 13. Black Box Model

The rest of this course concerns the power and potential applications of quantum computing.

We don't know how to prove from first principles that quantum computers are more powerful than classical computers (BQP strictly larger than BPP), but we believe it is true.

We'll discuss:

-- Efficient quantum algorithms that solve problems we don't know how to solve efficiently with a classical computer, e.g., factoring and simulation of quantum systems.

-- "Relativized speedups": The "black box model" = "oracle model" = "query model" in which we can prove a separation between classical and quantum computing (though the practical implications of these results are not entirely clear).

-- Polynomial speedups, such as $T_{quantum} \sim (T_{classical})^{1/2}$. Not relevant to BQP $\neq$ BPP, but nonetheless interesting.

Today we'll see an example of a relativized quantum speedup.

See Chapter 6 of the Lecture Notes. But I'll be organizing the material differently than in those notes.

Query Complexity

Consider “black box model” = “oracle model” = “query model.”

The box computes an a priori unknown function *f(x)*.
Given: a “promise” about *f*.
Find: *f*, or some property of *f*.

Query complexity: The minimal number of evaluations of *f(x)* (“queries”) needed to solve the problem.
How does this scale with n, the number of bits of *x*?
Don’t worry about the complexity of the task performed inside the box, or about the the hardness of the computation we need to do to solve the problem when the evaluations of *f(x)* are known.

The box is an opaque subroutine in our computation. We want to know: how many times do we need to call this subroutine?
The number of queries is at least a *lower bound* on the gate complexity.
This *might* provide insight into complexity outside the black box setting, if the box can be efficiently simulated, and our post-processing is also efficient.

We’ll distinguish between “classical” queries (bit strings) and “quantum queries” (superpositions of bit strings. In some cases, we’ll establish a large separation between classical and quantum query complexity.

Query Complexity

Function $f:\{0,1\}^n \to \{0,1\}$, Box $U_f:|x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle$,

Here *x* is *n* bits and *y* is one bit. In a classical query the input to *f* is a computational basis state. In a quantum query, the input to *f* may be a superposition of computational basis states.

Example: Deutsch's Problem (*n*=1).
Two classical queries complete characterize the function: {*f(0), f(1)*}.
But suppose we only want to know whether the function is *constant* or *balanced*.
That is, are *f(0)* and *f(1)* equal or not? Two queries are still needed in the classical setting.

But if we query in superposition, just one query is enough.

Phase Kickback Trick: prepare the second register properly, turning the oracle into a *phase oracle*.

Action of Pauli X on a qubit: $X^a|y\rangle = |y \oplus a\rangle$.

$$|\pm\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle \pm |1\rangle\right) \quad \Rightarrow \quad X|\pm\rangle = \pm|\pm\rangle \quad \Rightarrow \quad X^a|\pm\rangle = (-1)^a|\pm\rangle.$$

$$\Rightarrow \quad U_f:|x\rangle\otimes|-\rangle \to (-1)^{f(x)}|x\rangle\otimes|-\rangle.$$

The second register is unmodified by this query, so let's drop it:

$$\Rightarrow \quad U_f:|x\rangle \to (-1)^{f(x)}|x\rangle.$$

Deutsch and Deutsch-Jozsa Problems

The second register is unmodified by this query, so let's drop it: $\Rightarrow \quad U_f : |x\rangle \rightarrow (-1)^{f(x)} |x\rangle.$

Query in superposition: $U_f : \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \rightarrow \frac{1}{\sqrt{2}}\left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right).$

Now measure in the basis $|\pm\rangle$.
Outcome is $|+\rangle \rightarrow$ function *f* is constant.
Outcome is $|-\rangle \rightarrow$ function *f* is balanced.

If we insist that the measurement is in the computational basis $|0\rangle$, $|1\rangle$, precede measurement with a Hadamard transformation *H* (maps $|+\rangle$, $|-\rangle$ to $|0\rangle$, $|1\rangle$

Deutsch-Jozsa Problem. Now *f* maps n bits to 1 bit, and we are *promised* that f is either constant or balanced.

Either $f(x) = c \in \{0,1\}$, or $f(x) = 0$ for 2^{n-1} values of x, and $f(x) = 1$ for 2^{n-1} values of x.

Again, query the phase oracle in superposition:

$$U_f : H^{\otimes n}|000\ldots0\rangle = \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle \rightarrow \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}(-1)^{f(x)}|x\rangle.$$

Apply bitwise Hadamard $H^{\otimes n}$. If f is constant: $(-1)^c\left(\frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right)\right)^{\otimes n} \rightarrow (phase)|000\ldots00\rangle.$

Deutsch and Deutsch-Jozsa Problems

$$U_f : H^{\otimes n} |000\ldots0\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \rightarrow \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle.$$

Apply bitwise Hadamard $H^{\otimes n}$. If f is constant: $(-1)^c \left(\frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \right)^{\otimes n} \rightarrow (phase)|000\ldots00\rangle.$

If f is balanced: $\langle 000\ldots0 | H^{\otimes n} | state \rangle = \left(\langle + | \right)^{\otimes n} | state \rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} \langle y | \left(\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} | x \rangle \right) = \frac{1}{2^n} \sum_{x=0}^{2^n-1} (-1)^{f(x)} = 0$

(because (+1) occurs half the time and (-1) occurs half the time). Therefore, if we do the bitwise Hadamard, and then measure in the computational basis, we know the function is constant if the outcome is $|000 \ldots 0\rangle$, and we know the function is balanced if any other outcome is found. Just one quantum query suffices.

With classical queries only, we would need to query at least $2^{n-1} + 1$ times to be sure to solve the problem.

But with randomized classical queries you are likely to find a solution much more quickly. If the function is balanced and you query *k* times, you would find the same value of *f(x)* *k* times in a row with probability $1/2^{k-1}$. So if you query $k = 1 + \log_2(1/\varepsilon)$ times and declare "constant" only if you find the same value of *f(x)* each time, your probability of being wrong is ε.

Simon's Problem

A more interesting case: Exponential separation in query complexity between quantum queries and *randomized* classical queries.

$f:\{0,1\}^n \to \{0,1\}^n$. We are promised that f is a 2-to-1 function such that $f(x) = f(y)$ iff $x \oplus y \in \{0, a\}$.

Here a is an n-bit string. That is, $x_0 \oplus y_0 = a_0, \quad x_1 \oplus y_1 = a_1, \quad x_2 \oplus y_2 = a_2, \quad \ldots$

The problem is to find a.

How hard is this problem classically? If we wish, we can formulate it as a decision problem, by modifying the promise to say that *f* is either 1-to-1 or 2-to-1 with some "period" *a*. Then we may say Simon's problem is in NP^O. That is, we can easily verify the solution to the problem when provided with a classical witness and access to the oracle O. The witness is the string *a*, and it takes only two classical queries to check that *f(0) = f(a).*

But Simon's problem is not in BPP^O. We need a number of randomized queries exponential in *n* to find the solution with success probability ½ + positive constant.

Suppose we classically query *k* times, choosing a random value of *x* each time. If *f* is actually 2-to-1, we'll solve the problem only if we are so lucky that two of these values of *x* happen to get mapped to the same value of *f(x).* For any pair of values, we get lucky with probability $1/(2^n-1)$.

Simon's Problem

Suppose we classically query *k* times, choosing a random value of *x* each time. If *f* is actually 2-to-1, we'll solve the problem only if we are so lucky that two of these values of *x* happen to get mapped to the same value of *f(x).* For any pair of values, we get lucky with probability $1/(2^n-1)$.

With k randomized classical queries, we sample $\binom{k}{2}$ candidate pairs of input values of x.

Probability of getting lucky: $P_{\text{success}} \le \dfrac{\frac{1}{2}k(k-1)}{2^n-1} < \dfrac{k^2}{2^n}$. For $k = 2^{\frac{1}{2}n(1-\varepsilon)}$, $P_{\text{success}} < 2^{-\varepsilon n}$.

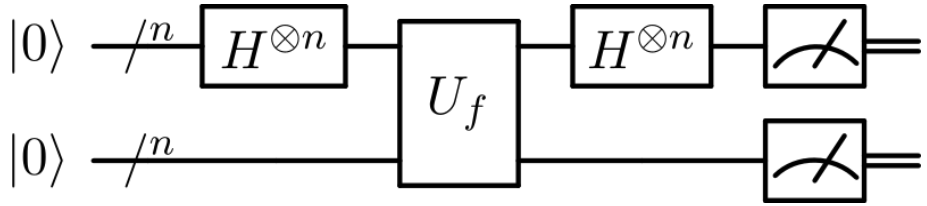
Even for exponentially many randomized classical queries, our success probability is exponentially small. Therefore, Simon's Problem is *not* in BPP^{O}.

But it *is* in BQP^{O}. That is, we can solve the problem efficiently if quantum queries are allowed. Thus $\text{BPP}^{\text{O}} \neq \text{BQP}^{\text{O}}$. In this sense, quantum computing is more powerful than randomized classical computing, in this black box setting.

What is the quantum algorithm?

Simon's Problem

What is the quantum algorithm? Query with the uniform superposition of all computational basis states, measure input register in the complementary basis.



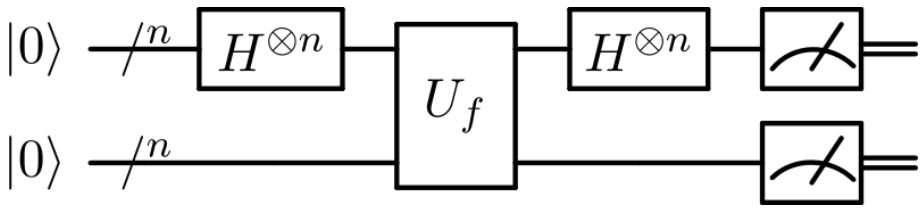
$\frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle\otimes|0\rangle \to \frac{1}{\sqrt{2^n}}\sum_{x=0}^{2^n-1}|x\rangle\otimes|f(x)\rangle$. Then trace out (or measure) the output register.

If the output is $f(x_0)$, the state of the input register becomes (if f is 2-to-1) $\frac{1}{\sqrt{2}}\left(|x_0\rangle+|x_0\oplus a\rangle\right)$.

Measuring now in the standard basis would not work; we would find either x_0 or a at random, and either way we learn nothing about a. Instead, we apply the bitwise Hadamard and then measure. Applying H to one qubit:

$$H|x\rangle=\frac{1}{\sqrt{2}}\left(|0\rangle+(-1)^x|1\rangle\right)=\frac{1}{\sqrt{2}}\sum_{y=0,1}(-1)^{xy}|y\rangle \quad\Rightarrow\quad H^{\otimes n}|x_{n-1}x_{n-2}\ldots x_1x_0\rangle=\bigotimes_{i=0}^{n-1}\left(\frac{1}{\sqrt{2}}\sum_{y=0,1}(-1)^{x_iy_i}|y_i\rangle\right)=\frac{1}{2^{n/2}}\sum_{y=0}^{2^n-1}(-1)^{x\cdot y}|y\rangle.$$

Here $x\cdot y$ denotes the bitwise inner product of n-bit strings: $x\cdot y=\sum_{i=0}^{n-1}x_iy_i \pmod 2$.



Simon's Problem

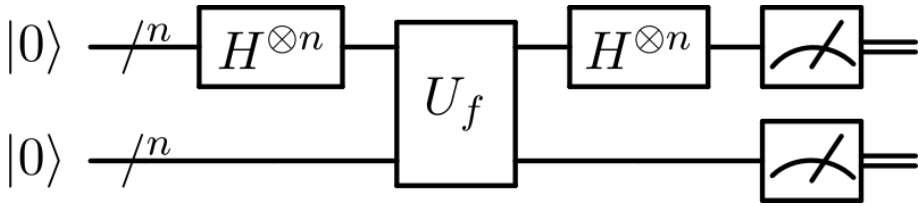
$$H^{\otimes n}\,|\,x\rangle = \frac{1}{2^{n/2}}\sum_{y=0}^{2^n-1}(-1)^{x\cdot y}\,|\,y\rangle \quad \Rightarrow \quad H^{\otimes n}\,\frac{1}{\sqrt{2}}\left(|\,x_0\rangle + |\,x_0 \oplus a\rangle\right) = \frac{1}{2^{(n+1)/2}}\sum_{y=0}^{2^n-1}\left((-1)^{x_0\cdot y} + (-1)^{(x_0\oplus a)\cdot y}\right)|\,y\rangle$$

$$= \frac{1}{2^{(n+1)/2}}\sum_{y=0}^{2^n-1}(-1)^{x_0\cdot y}\left(1+(-1)^{a\cdot y}\right)|\,y\rangle = \frac{1}{2^{(n-1)/2}}\sum_{y:\,a\cdot y=0}(-1)^{x_0\cdot y}\,|\,y\rangle.$$

Measuring in the standard basis samples uniformly from values of *y* that are "orthogonal" to *a* in the bitwise inner product. (Note that the value of x_0 has no effect on the probability distribution of outcomes.)

We run this sampling procedure multiple times. Once we have found *n*-1 values of *y* which are linearly independent, *a* is uniquely determined, and it is an easy (binary) linear algebra problem to find *a*.

Furthermore, after O(*n*) trials, with high probability we successfully find *n*-1 linearly independent values of *y*. Before *n*-1 linearly independent values have been found, the probability of finding a value linearly independent of previously found values is at least one half. So after *m* trials, the probability of failing to find *n*-1 linearly independent values is smaller than the probability of failing to find heads at least *n*-1 times when tossing an unbiased coin *m* times (unlikely for *m* = (large enough constant) $\times$ n).



Simon's Problem

$$H^{\otimes n}\,|\,x\rangle = \frac{1}{2^{n/2}}\sum_{y=0}^{2^n-1}(-1)^{x\cdot y}\,|\,y\rangle \quad \Rightarrow \quad H^{\otimes n}\,\frac{1}{\sqrt{2}}\left(|\,x_0\rangle + |\,x_0 \oplus a\rangle\right) = \frac{1}{2^{(n+1)/2}}\sum_{y=0}^{2^n-1}\left((-1)^{x_0\cdot y} + (-1)^{(x_0\oplus a)\cdot y}\right)|\,y\rangle$$

$$= \frac{1}{2^{(n+1)/2}}\sum_{y=0}^{2^n-1}(-1)^{x_0\cdot y}\left(1+(-1)^{a\cdot y}\right)|\,y\rangle = \frac{1}{2^{(n-1)/2}}\sum_{y:\,a\cdot y=0}(-1)^{x_0\cdot y}\,|\,y\rangle.$$

So we see that Simon's problem can be solved with O(n) quantum queries, but requires a number of randomized classical queries exponential in n: $\mathrm{BPP}^{\mathrm{O}} \neq \mathrm{BQP}^{\mathrm{O}}$, where O is Simon's oracle.

It is a remarkable result, but does it have any practical applications? Not that we know of. To find such an application we would want an instantiation of Simon's oracle such that the problem is still hard even when the structure of the function (what is "inside the black box") is known (and ideally for that particular function it would become a problem whose solution people really care about).

Because it has no obvious real-world applications, Simon's 1994 paper was rejected. But the problem and its solution inspired another black box problem for which a separation between quantum and classical query complexity can be established, which turns out to have practical applications. We'll talk about that next time.