**Quantum Computing: Foundations to Frontier** Fall 2018

Lecture 8

*Lecturer: Henry Yuen* *Scribes: Zhewei Sun, Faisal Qureshi*

# 1 Overview

In the last lecture, we have looked at the landscape of classical complexity. In particular, we learned about some of the following classical complexity classes:

$$P \subseteq NP \subseteq PSPACE \subseteq EXP$$

We then looked at the quantum complexity class $BQP$ which includes all problems solvable using a polynomial quantum algorithm with bounded error. We have related $BQP$ to classical complexity classes by showing $BQP \subseteq PSPACE$ and also provided some evidence that $NP \nsubseteq BQP$.

Today's lecture focuses on **quantum supremacy** and **near-term quantum computation**, where we are interested in whether quantum computers can outperform their classical counterparts in some computational task, and if so, whether there are useful problems that we can solve.

# 2 Quantum Supremacy and Near-Term Quantum Computation

In order to illustrate the advantages that quantum computers can bring, we want to find evidence that:

$$BPP \neq BQP$$

which shows that quantum algorithms can efficiently solve problems (i.e. in polynomial time) that classical (randomized) computers cannot.

## 2.1 Evidence from Previous Lectures

Here is perhaps some evidence that quantum computers can solve problems that classical computers cannot.

- Quantum Fourier Transform (QFT), Harrow-Hassidim-Lloyd (HHL) algorithms seem to accomplish classically hard things
- Shor's algorithm (Factoring is NP)
- Simulating quantum systems.

### 2.2 Quantum Simulations

**Problem:** Given a Hamiltonian $H = H_1 + ... + H_m$, an initial state $|\psi(0)\rangle = |0\rangle^{\otimes n}$, and a measurement $M$, what is $|\psi(t)\rangle = e^{-iHt}|\psi(0)\rangle$ and also what is $\langle\psi(t)|M|\psi(t)\rangle$?

$|\psi(t)\rangle$ is the final quantum state of our physical system solvable using the Schrödinger equation and $\langle\psi(t)|M|\psi(t)\rangle$ is the expectation of its measurement w.r.t. $M$. This expectation seems to be a hard quantity to estimate, as $M$ is an exponentially large matrix and $|\psi(t)\rangle$ is an exponentially long vector.

Quantum simulation is complete for BQP, meaning that if you can efficiently do quantum simulation, then you can also efficiently do any other quantum computation.

**Skeptics:** How can we verify that a quantum computer correctly simulates the quantum system?

For certain choices of $H$ and $M$, classical computers can tractably compute the answer to check against the QC results. But the skeptics can be annoying and claim your QC is nothing but a laptop running in the background.

For the most difficult choices of $H$ and $M$, as it is believed that $BQP$ is outside of $NP$, it doesn't appear that one can classically verify a general $BQP$ computation. Though adding interactivity between QC and classical computers can verify QC efficiently though cleverly designed protocols.

We're also in this primordial phase of having very noisy 50-100 qubit quantum machines, and it's not clear that we can do full-fledged quantum simulations in the near-term.

### 2.3 QFT, HLL

On the surface, these algorithms seem to be doing incredible things (solving exponentially large linear systems, applying the Fourier transform on exponentially long vectors), but there's an important caveat in that the inputs and outputs are encoded in quantum states. A skeptic would say: by themselves, they're not immediately useful, so they don't immediately imply, definitively, that quantum computers have an advantage over classical computers.

But what about, say, factoring?

### 2.4 Shor's algorithm

One nice thing about the factoring problem is that (a) we have a quantum algorithm that can efficiently factor, (b) the factoring problem is in $NP$, meaning that you can efficiently check the answer to the problem.

Two issues: one, is factoring *actually* that hard? There's no complexity theoretic evidence for it other than, "lots of people have thought about it and couldn't find a fast classical algorithm." Secondly, even if you believed factoring is hard for classical computers, we're not going to see large numbers being factored by Shor's algorithm any time soon, based on the current state of quantum computing hardware these days.

So is there anything we can do today to convince a skeptic that quantum computers are capable of something more than classical computers?

## 2.5 Goal of Quantum Supremacy

We wish to find some computational task $T$ such that:

(a) Runnable on a NISQ (Noisy Intermediate-Scale Quantum) machine.

(b) Verifiable on a classical computer, using supercomputing time (i.e. allow exponential time computation).

(c) Some complexity evidence that $T$ is hard for classical computers (e.g. $T$ is not solvable by classical randomized algorithms).

For a doing some quantum computation, as the number of qubits increase, computational time for classical machines grow exponentially while it is only polynomial for QCs. We are currently at a sweet spot (∼50-100 qubits) where there is a tie in computation time. This allows us to verify QC results on classical supercomputers.

## 2.6 Random Circuit Sampling (RCS)

### 2.6.1 Procedure

Let there be $n$ qubits (say 50), $m$ gates (several hundred), and let $T \sim$ millions, do the following:

1. Pick Random circuit $C$ using all $m$ gates acting on any of the $n$ qubits
2. Run $C$ on your QC $T$ times, obtaining samples $x_1, x_2, ..., x_T \in \{0,1\}^n$. Let $\mathcal{D}_C$ be the underlying distribution of the samples (i.e. $x_i \sim \mathcal{D}_C$). Then the probability of drawing a particular sample $x$ is:
$$P_c(x) = |\langle x|C|0\rangle|^2$$
3. Run Verification on the collected samples: $\{x_i\}$

**Theorem 1** (Bouland, et al. [1]). *With high probability over $C$, the following holds: There is no efficient classical algorithm that samples from $\mathcal{D}_C$ unless the polynomial hierarchy (PH) collapses to the 3rd level.*

The statement "if $X$, then the polynomial hierarchy collapses" should be interpreted as (complexity-theoretic) evidence that $X$ is not true.

### 2.6.2 The polynomial hierarchy

The polynomial hierarchy is a hierarchy of complexity classes that generalize $P$ and $NP$. We can define it as

$$PH = \bigcup_k \Sigma_k$$

where $\Sigma_0 \subseteq \Sigma_1 \subseteq \Sigma_2 \subseteq \Sigma_3 \subseteq \cdots$

$$\Sigma_0 = P = \{f : \{0,1\}^* \to \{0,1\} : \forall x. f(x) \text{ is computable in } \mathit{poly}(|x|) \text{ time}\}$$

$$\Sigma_1 = NP = \{f : \exists \text{ efficient } g(x,y) \text{ s.t. } \forall x. f(x) = 1 \Leftrightarrow \exists y. g(x,y) = 1\}$$

$$\Sigma_2 = \{f : \exists \text{ efficient } g(x,y,z) \text{ s.t. } \forall x. f(x) = 1 \Leftrightarrow \exists y. \forall z. g(x,y,z) = 1\}$$

$$\Sigma_3 = \{f : \exists \text{ efficient } g(x,y,z,w) \text{ s.t. } \forall x. f(x) = 1 \Leftrightarrow \exists y. \forall z. \exists w. g(x,y,z,w) = 1\}$$

$$\vdots$$

The statement that the polynomial hierarchy collapses to the $k$-th level is simply the statement that $\Sigma_\ell = \Sigma_k$ for all $\ell \geq k$. For example, the statement that the polynomial hierarchy collapses to the 0-th level is equivalent to $P = NP$.

**Remark:** The theorem of [1] assumes exact sampling of the distribution $\mathcal{D}_C$ but does not prove the case for approximate sampling, which remains a conjecture.

**Conjecture:** There is no efficient classical algorithm that, on average over circuits $C$, samples **approximately** from $\mathcal{D}_C$ unless the polynomial hierarchy (PH) collapses to the 3rd level,

### 2.6.3 Verification

**Question:** How do we check that the $x_i$'s came from $\mathcal{D}_C$? We don't yet know of an efficient method of doing so, but here are some proposals.

**1st Proposal: Heavy Output Generation (HOG) [2]**

1. Collect $x_1, ..., x_T$ from QC.
2. Use supercomputer to compute median of $\{P_C(x)\}$, call this number $\alpha_C$.
3. Say that $x \in \{0,1\}^n$ is **heavy** if $P_C(x) > \alpha_C$.
   If at least $\frac{2}{3}$ of the samples are heavy, then accept, otherwise reject.

**Remarks:**

- The intuition behind this test is that the set of heavy $x$'s constitute the majority of the probability density from $\mathcal{D}_C$, thus the majority of the samples generated by your QC should come from the set of heavy $x$'s.
- However, intuitively, it should be hard for an efficient classical algorithm to predict which strings $x$ will be heavy, given a circuit $C$. Thus if one verifies that the quantum machine was able to output lots of heavy strings, the quantum machine must've done "something right."
- With high probability over $C$, $\alpha_C \approx \frac{\ln 2}{2^n}$

**<u>2nd Proposal</u>: Cross-Entropy Verification [Google]**

- Given $x_1, ..., x_T$, use supercomputer to compute $P_C(x_i)$
- Compute the cross-entropy:
$$E = \frac{1}{T}\sum_i log(\frac{1}{P_C(x_i)})$$
- By the law of large numbers:
$$E \to \sum_x P_{device}(x) log(\frac{1}{P_C(x_i)}) = CE(P_{dev}, P_C)$$
- If $P_{dev} = P_C$, then:
$$CE(P_C, P_C) = n + \gamma + 1 \text{ w.h.p over C}$$
where $\gamma$ is the Euler's constant.
- Check if $E$ is close to $n + \gamma + 1$, if so accept, otherwise reject.

**Criticism:** There exist probability distributions that have $CE = n + \gamma + 1$ but are very different from each other. This procedure only works under a strong assumption on the underlying noise model, which may or may not be true in practice.

# 3 Quantum Approximate Optimization Algorithm (QAOA)

In the previous half of the lecture we discussed Quantum Supremacy experiments, which generally perform computationally useless tasks. In this half of the lecture we discuss doing something computationally useful with a near-term quantum computer. Our hope is that we can use these types of near-term noisy quantum machines to solve some optimization problems that are hard.

In this context we introduce the Quantum Approximate Optimization Algorithm (or QAOA), based on the ground breaking paper by Farhi, Goldstone and Gutmann (2014) [3].

The general idea is to develop a hybrid quantum-classical optimization scheme despicted below:

The hybrid algorithm iteratively solves optimization problems as follows.

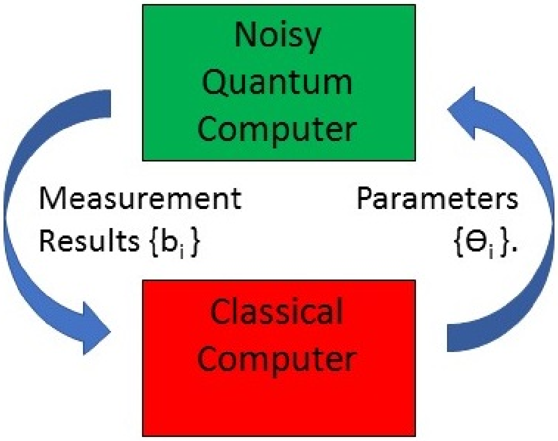


Figure 1: Hybrid Classical-Quantum Algorithm

- *Quantum Part* – A low depth quantum circuit (i.e. a circuit with not many layers) which is parametrized by $\{\theta_j\}$ runs a small quantum computation on a noisy QC. The output of the circuit is measured to obtain bits $\{b\}$.
- *Classical Part* – The classical computer uses the output $\{b_i\}$ to determine next set of parameters to try.

Hopefully, after not too many iterations we arrive art a pretty good solution $\{b_i\}$ to our optimization problem.

## 4 Example Problem: MAX-CUT

While QAOA can be used for any optimization problem we will restrict our discussion to the specific problem MAX-CUT, as described in [3].

Specifically, assume a graph $G = (V, E)$, with $n$ vertices and $m$ edges, where each vertex is represented by a qubit. A "cut" of the graph can be represented by $|x\rangle$ where $x \in \{0,1\}^n$ by letting $S_x = \{v : X_v = 1\}$, denote the set of all vertices on one side of a cut, and letting $\bar{S}$ denote its complement (i.e. vertices on the other side of the cut).

The MAX-CUT problem is then to maximize $E(S, \bar{S})$, i.e. the number of edges between $S$ and $\bar{S}$.

This is is equivalent to finding the minimum energy state (ground state) of the Hamiltonian:

$$H_G = \sum_{i \sim j} (I + \sigma_z(i) \cdot \sigma_z(j)) \tag{1}$$

$$\text{where } \sigma_z(i) = I \otimes \sigma_z \otimes I$$

is the $\sigma_Z$ operator acting only on the $i$-th qubit (and identity everywhere else). Recall that

$$\sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

Here's why. Let $|x\rangle$ be some assignment. Then:

$$\begin{aligned} \langle x|H|x\rangle &= \text{ the energy of } x \text{ with respect to } H \\ &= \sum_{(i,j)\in E} \langle x|(I + \sigma_z(i)\cdot\sigma_z(j))|x\rangle \\ &= \sum_{(i,j)\in E} \langle x||x\rangle + \langle x|\sigma_z(i)\cdot\sigma_z(j)|x\rangle \\ &= \sum_{(i,j)\in E} 1 + (-1)^{x_i + x_j} \\ &= \sum_{(i,j)\in E} 2\mathbb{1}[x_i = x_j] \\ &= 2(m - \#\text{of edges cut}) \end{aligned} \tag{2}$$

The minimum energy $|x\rangle$ maximizes the # of edges cut. The QAOA will help find an approximate solution to this.

## 5 QAOA

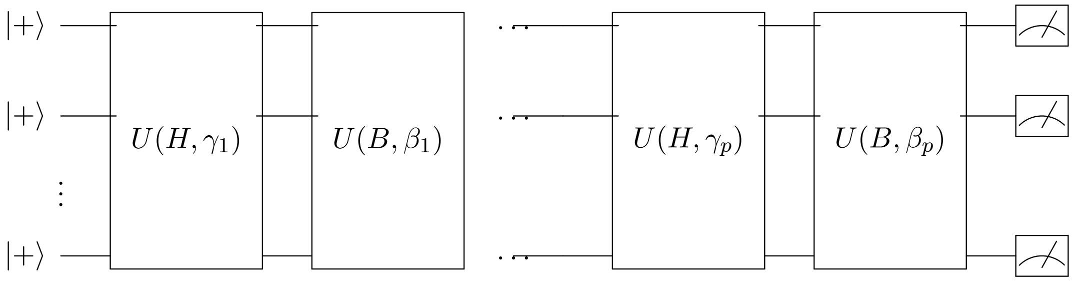
The algorithm is parametrized by an integer $p \geq$, and positive real numbers $\{\beta_i, \gamma_i\}_{i=1,..,p}$. In the paper these are referred to as angles, but they can also be interpreted as time durations.

$$U(H,\gamma) = e^{-i\gamma H} = \prod_{(a,b)\in E} e^{-i\gamma(I+\sigma_z(a)\sigma_z(b))} \tag{3}$$

$$U(B,\beta) = \prod_{j=1}^{n} e^{-i\beta\sigma_x(j)} \tag{4}$$

$$\text{where } \sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

The QAOA is just going to interleave $U(B,\beta)$ and $U(H,\gamma)$



where $|+\rangle = 1/\sqrt{2}(|0\rangle + |1\rangle)$.

Let $|\psi_{\beta,\gamma,p}\rangle = U(B,\beta_p)U(H,\gamma_p)\cdots U(B,\beta_1)U(H,\gamma_1)|s\rangle$, where$|s\rangle = |+\rangle^{\otimes n}$

### 5.1 Definition

Let $F_p(\beta,\gamma) = \langle\psi_{\beta,\gamma,p}|H|\psi_{\beta,\gamma,p}\rangle$.

Its not hard to show this is the same as expectation of $\langle x|H|x\rangle$ with respect to $H$ when $|x\rangle$ is sampled by measuring $|\psi_{\beta,\gamma,p}\rangle$ in the standard basis.

### 5.2 Objective

Want to find $(\gamma,\beta)$ so that $F_p(\gamma,\beta)$ is as small as possible.

$$\begin{aligned} &M_p = min_{\gamma,\beta}F_p(\gamma,\beta) \\ &M_p \leq M_{p-1} \end{aligned} \tag{5}$$

### 5.3 Theorem

$$\lim_{p\to\infty} M_p = \min_x \langle x|H|x\rangle \tag{6}$$

this algorithm will find MAX-CUT, in the limit. Of course, since we don't know QC's can solve NP-Hard problems like MAX-CUT in polynomial time then in the worst case this will also take QAOA exponential time, but we can hope for **<u>some</u>** quantum speedup, for some problems.

Which problems can we get quantum speedup on? We are not sure, but a lot of people are trying to play around with QC's to find out.

How well does this algorithm do for small $p$? For $p = 1$, QAOA can produce solutions that are within 0.69224 of the optimal MAX-CUT. There is a an open question as to what is the best approximation ratio for small $p > 1$.

# References

[1] Bouland, Adam and Fefferman, Bill and Nirkhe, Chinmay and Vazirani, Umesh, *On the complexity and verification of quantum random circuit sampling*, Nature Physics, page 1, 2018.

[2] Aaronson, Scott and Chen, Lijie, *Complexity-theoretic foundations of quantum supremacy experiments*, 32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia, 22:1-67, 2018.

[3] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, *A Quantum Approximate Optimization Algorithm*, arXiv preprint, arXiv:1411.4028, 2014.