Lecture 25 – Quantum Advantage

(nee "Quantum Supremacy", a term coined by Preskill in 2012)

Quantum Advantage Experiment:

Getting a real-world quantum computer to do some task that is believed practically impossible for any classical computer.

→ any task: doesn't have to be useful/interesting

The absolutely greatest, most clear-cut Q.A. experiment would be...

Factoring RSA-2048 ?

No one can do it classically ($200k prize) & it's super-easy to verify correct answer.

But the engineers are nowhere close to being able to build the necessary large-scale quantum circuits for Shor...

Oct. 2019: Google team (... Boixo, ..., Martinis) announces success.

You might ask: what task did they do?

Task: Simulate a given random 53-qubit, ≈1000-gate quantum circuit!

"What?" you might say. Well, as we saw last lecture, there is some evidence that classically simulating the measurement outcomes of a large quantum circuit (given as input the circuit diagram) is classically intractable.

But in fact, getting a physically extant quantum device to simulate a given quantum circuit diagram is pretty darn hard, practically speaking, too!

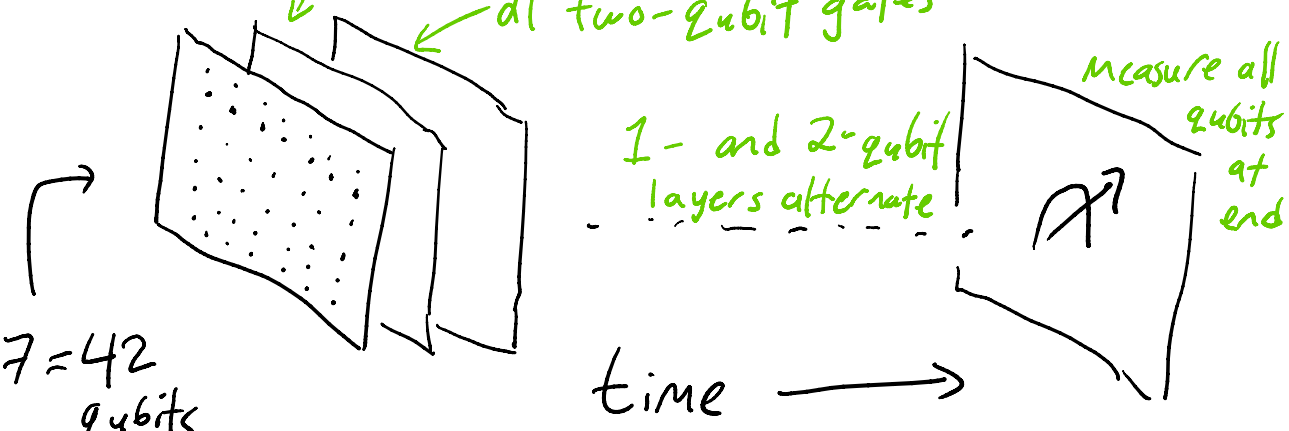
Let's discuss this issue first - why so hard to build quantum computers?

Practically very hard to shield a physical qubit from "noise" ← interaction with environment that ruins state & entanglement.

- Implementing the 0-qubit "do nothing" gate is hard!
- Implementing 1-qubit gates not much harder.
- Implementing 2-qubit gates: very hard.

Also: have to physically lay out qubits, typically on 2-d grid, and can only do 2-qubit gates on "adjacent" qubits.

Typical setup:



$6 \times 7 = 42$ qubits

(Google's 53 was $6 \times 9 = 54$ minus one, because one qubit broke!)

maybe 20 "layers" of gates

* Crucial to have high-quality qubits: no noise for entire time.

Classical Fault Tolerance?

Say each AND/OR/NOT gate fails (random output) independently with small probability $p > 0$.

Are you hosed, or can you still compute?

Thought about a lot in very early days...

Ultimately didn't matter: voltages on wires are very noise-robust! Today's computers *do* have failures, but like $p \ll 2^{-64}$.

But von Neumann *did* solve the prob. anyway:

von Neumann '52 (fleshed out by others): $\exists$ universal const. $p_c > 0$ such that $p < p_c \Rightarrow$ can make any m-gate circuit into a fault-tolerant version with $O(m \log m)$ gates.

(Also $\Omega(m \log m)$ is necessary. Exact value of p_c depends on gate set, noise model, fault-tolerance scheme.)

Could such a result be possible for quantum circuits too?

Yes! First step is error-correcting codes. Might seem difficult on quantum data due to No-Cloning, but it's possible! Shor did it first, you saw a hint of it on Test #1.

Next comes "fault-tolerance": computing on error-corrected data.

Quantum Fault-Tolerance Theorem: (Dorit Aharonov, Miki Ben-Or 1996)

$\exists$ universal const. $p_q > 0$ such that, if gate noise rate is $p < p_q$, can make any m-qubit quantum circuit fault-tolerant with $m \cdot \text{poly}(\log m)$ gates.

(Fawzi-Grospellier-Leverrier '18: $O(m)$ gates, assuming noise-free classical computing.)

↓

Their p_q: $\approx 10^{-6}$.

With latest, fanciest algs: $10^{-3} \ldots 10^{-2}$.

(depends on exact model details)

Best gates today have failure rate... $\approx 10^{-3}$.

(But... the fancy fault-tolerance algs. need, like, 100 physical qubits per logical qubit. So we can't... yet... do F.T.Q.C....)

Quantum Advantage Experiment Plan (Google 2019)

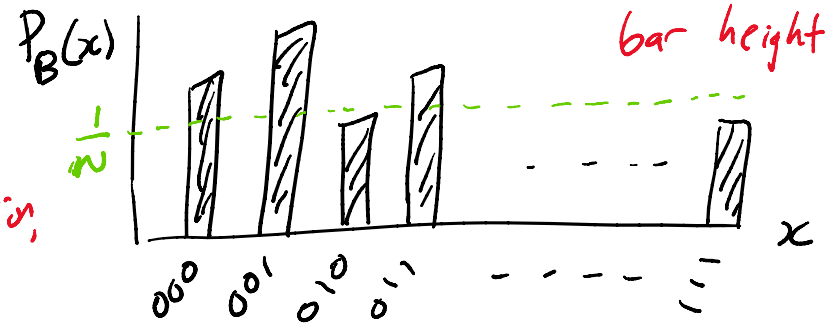
1. Draw (on paper) a random quantum circuit of aforementioned type, call it B. ("Blueprint") All inputs set to 0?, all outputs measured.
2. This determines a probability distribution P_B on outcomes in $\{0,1\}^n$. ($n = 53$, perhaps)
3. Declare task to be: "Create a machine whose output distribution is (close to) P_B."
4. (Try to) solve task by building Q, a physical implementation of B.
5. Hope Q's output distribution P_Q has $P_Q \approx P_B$.
6. Assert no classical alg. C running in less than centuries could have $P_C \approx P_B$.
7. Declare Quantum Advantage.

Note: Google expected P_Q would actually be pretty far from P_B — they just thought P_C would be even worse!

Let's talk about steps. First, what will P_B look like?

B random $\Rightarrow$ P_B is a randomly generated prob. dist on $\{0,1\}^n$. $N := 2^n$

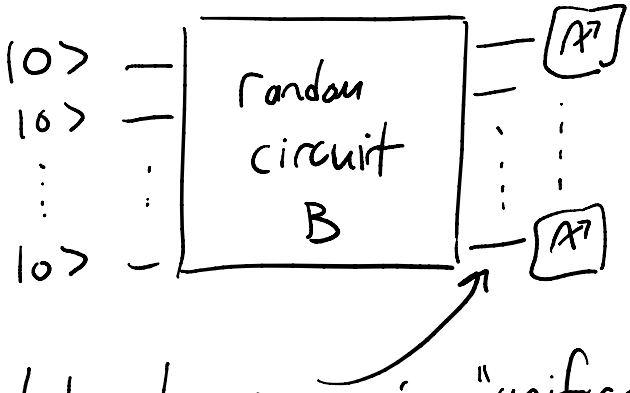
Will P_B be like the uniform distribution, $P_B(x) \approx \frac{1}{N}$ $\forall x$?



No!

Heuristic:

B $\approx$ "uniformly random unitary"

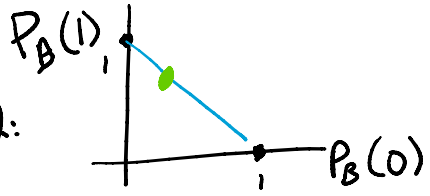


↳ If correct, state here is "uniformly rand" on unit-sphere, $\alpha_{00\cdots0}|00\cdots0\rangle + \cdots + \alpha_{11\cdots1}|11\cdots1\rangle$ such that $|\alpha_{00\cdots0}|^2 + \cdots + |\alpha_{11\cdots1}|^2 = 1$.

$P_B(00\cdots0)$ $P_B(11\cdots1)$

$P_B(x)$'s are "random #'s adding to 1".

N=2:



N=3:

Fact: If you sort the histogram of P_B, almost surely (over choice of B) it looks like ....

(Median $P_B(x)$ value is $\approx \frac{\ln 2}{N} \approx \frac{.7}{N}$.)

"Typical" $P_B(x)$ value: $\underset{x \sim P_B}{E}[P_B(x)] \approx \frac{2}{N}$.

How to test if $P_Q \approx P_B$?

P_Q: output distribution of actual quantum computer that's physically built

Google plan:
- Draw $\approx$ 1 million samples x from P_Q
- Compute $P_B(x)$ for each. (How?!?) (we'll come back to this)
- Hope average value is $\approx \frac{2}{N}$.
- Assert that for any classical C, average of $P_B(x)$ for 1 million draws of $x \sim C$ would be $\gtrapprox \frac{1}{N}$.

Google did this, and $\mathop{E}_{x \sim P_Q}[P_B(x)]$ was....

$$\frac{1.002}{N} \quad (!!)$$

Greater than $\frac{1}{N}$ "with statistical significance."

Heuristically comparable to...

"With prob. 10^{-3} do a correct draw from P_B, else output unif. random x."

(Q has $\approx 10^6$ gates, each has failure prob. $\approx 10^{-3}$...)

Quantum Advantage?

Wait: How did they compute $P_B(x)$? Isn't that, like, by definition supposed to be intractable?

They did a mix of things:

- try $n = 20, 30, 40$, where computing $P_B(x)$ with exponential time, poly space is feasible...
- try simpler kinds of semirandom circuits B (with $n=53$) where there's an efficient classical alg. for computing $P_B(x)$
- heuristic args. about estimated gate failure rates....

What about assertion that for large n, best classical C can just get $1/N$?

Well, C could just output random strings.

Then $\mathop{E}_{x \sim C}[P_B(x)] = \frac{1}{N}$.

Anything better?

No one knows. Task is similar to — but not the same as — the one shown "impossible unless polynomial hierarchy collapses".

- IBM claimed their supercomputer could compute all 2^{53} values of $P_B(x)$ in a few days... but they never did it...
- Several subsequent papers gave faster algs for computing $P_B(x)$ for similar-but-slightly-simpler circuit models B....