Week 6

Lecture 11 - Quantum Supremacy
11.1 Boson Sampling
11.2 IQP Problem
11.3 Google's pseudorandom circuits

Lecture 12 - Errors
12.1 Quantum errors: unitary and stochastic errors
12.2 Randomized Benchmarking
12.3 Purity

Lab 6
Quantum Supremacy and Errors

Quantum Supremacy

Physics 90045
Lecture 11

Determining supremacy?

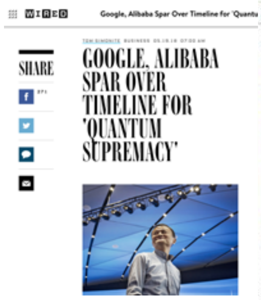
Gary Kasparov

vs

Deep Blue

On February 10, 1996, Deep Blue beat Kasparov under tournament regulations. In the subsequent 1997 rematch, Deep Blue won the series.

Quantum supremacy in the news



https://www.wired.com/story/google-alibaba-spar-over-timeline-for-quantum-supremacy/

https://www.newscientist.com/article/2176575-google-just-made-it-much-harder-to-build-a-serious-quantum-computer/

PHYC90045 Introduction to Quantum Computing

What is quantum supremacy?

Quantum supremacy is using a quantum computer to solve a problem which classical computers **practically** cannot.

PHYC90045 Introduction to Quantum Computing

Algorithms for quantum supremacy

The race is on to build a quantum computer which will achieve quantum supremacy. Implementing large scale factoring would demonstrate quantum supremacy, but that would require a very large (potentially millions of qubits) quantum computer. In the short term we will only have access to NISQ devices.

Noisy
Intermediate **S**cale (50-100 qubits)
Quantum devices

Three quantum algorithms that might be able to demonstrate quantum supremacy with 50-100 qubits:

- Boson Sampling
- Instantaneous Quantum Polynomial-Time circuits (IQP)
- Pseudorandom circuits

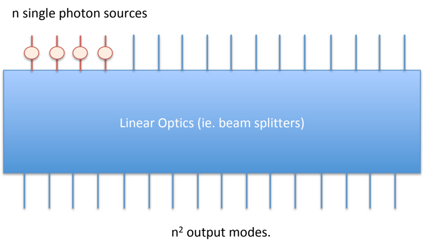
HOWTO quantum supremacy

Pick a problem which is:

- ❑ As easy as possible for a quantum computer
- ❑ As hard as possible for a classical computer to simulate

Boson Sampling

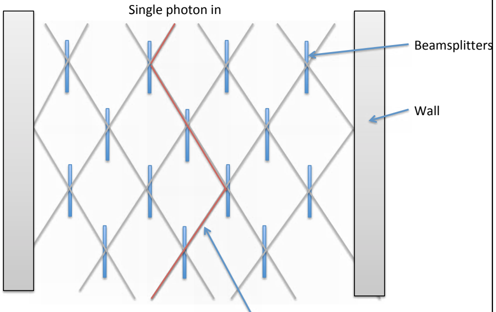
A little physics experiment…



You won't need to know the physics of this device.

Can a classical computer produce **samples** from the output which mimic the quantum device?

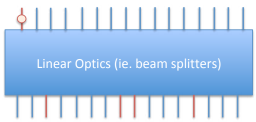
Quantum Pachinko



PHYC90045 Introduction to Quantum Computing

One single path

We can write out a matrix: If a single photon enters in mode x, what is the amplitude it will have at output y?



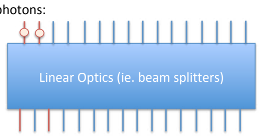
$$\begin{array}{cc} & \text{Input modes} \\ U = & \begin{bmatrix} \boxed{a_{11}} & a_{12} & a_{13} & \dots & a_{1n} \\ \boxed{a_{21}} & a_{22} & a_{23} & \dots & a_{2n} \\ \boxed{a_{31}} & a_{32} & a_{33} & \dots & a_{3n} \\ \boxed{\dots} & \dots & \dots & \dots & \dots \\ \boxed{a_{n1}} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix} \end{array} \quad \text{Amplitudes of output modes}$$

Unitary matrix

PHYC90045 Introduction to Quantum Computing

Many paths

If we have two input photons:



What is the amplitude associated with being detected in output modes 1 and 3?

$$\begin{array}{cc} & \text{Input modes} \\ U = & \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix} \end{array} \quad \text{Amplitudes of output modes}$$

Amplitude of photons in locations 1,3:

$$a_{11}a_{32} + a_{12}a_{31}$$

Many paths

In general take the submatrix corresponding to a particular input and output, and find its **permanent:**

$$U = \begin{bmatrix} \boxed{a_{11} \quad a_{12}} & a_{13} & \boxed{\cdots} & a_{1n} \\ a_{21} \quad a_{22} & a_{23} & \cdots & a_{2n} \\ \boxed{a_{31} \quad a_{32}} & a_{33} & \boxed{\cdots} & a_{3n} \\ \boxed{\cdots \quad \cdots} & \cdots & \boxed{\cdots} & \cdots \\ a_{n1} \quad a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix}$$

Submatrix defined by the input modes and output modes

The resulting amplitude is the **permanent** of the submatrix:

$$\mathrm{perm}(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma(i)}$$

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \quad \mathrm{perm}(A) = ad + bc$$

Same as determinant, but with no subtraction, all addition.

Complexity of finding permanent

Unlike determinants, finding a permanent of a matrix is a *surprisingly difficult* computational problem.

Finding the permanent is a #P complete problem.

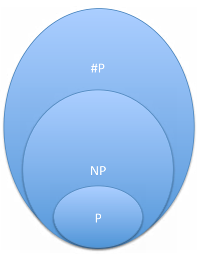
#P is the set of counting problems associated with decision problems in NP.

NP: Is there as a satisfying assignment of variables to this 3SAT problem?
#P: How many satisfying assignments of variables are there to this 3SAT problem?

NP: Is there a travelling salesman path with distance less than *d*?
#P: How many travelling salesman paths are there with a distance less than d.

Calculating amplitudes and probabilities for Boson sampling is a hard classical problem!

Some classical complexity classes



Informally:

P: Problems which can be solved in polynomial time

NP: Problems which can be checked in polynomial time (ie. they have an efficiently verifiable proof)

#P: Problems which count the number of solutions in NP

The polynomial hierarchy

Very quick introduction: Given an **oracle** in some complexity class which evaluates instantly, what problems can we now evaluate in polynomial time?

$$P^P = P$$

Polynomial time algorithm
With access to an oracle which can instantaneously evaluate functions in P

$$NP^P = NP$$

But a polynomial time algorithm with an NP oracle appears to be more powerful than both P and NP:

$$P^{NP}$$

We can recursively define complexity classes this way, with oracles which increase in strength at each level. This whole hierarchy is known as the Polynomial Hierarchy, **PH**.

If, at some level, providing the oracle didn't lead to a superset of problems, the polynomial hierarchy would "collapse". Computer scientists don't think this happens.

Sampling is also hard to simulate

Calculating amplitudes and probabilities for Boson sampling is a hard classical problem!

Calculate the **permanent** of the submatrix:

$$\text{perm}(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma(i)} \qquad \text{perm}\begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad + bc$$

We don't technically have to calculate the probabilities explicitly. Maybe we can *sample* from the probability distribution?

No – this would result in a collapse of the Polynomial Hierarchy.

Not proven, but like P=NP, computer scientists generally don't expect the polynomial hierarchy collapses.

HOWTO quantum supremacy

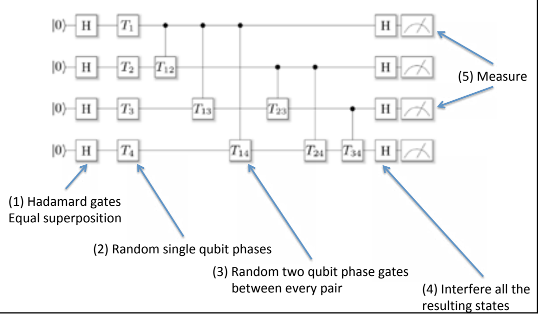
Boson Sampling is a problem which:

- ❑ "Easy" to implement using linear optics
- ❑ Hard for a classical computer to simulate – Polynomial Hierarchy would collapse

IQP Circuits
Instantaneous Quantum Polynomial-Time

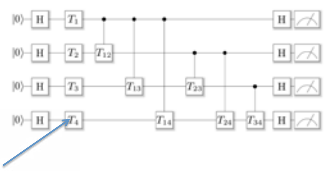
PHYC90045 Introduction to Quantum Computing

IQP Circuits



PHYC90045 Introduction to Quantum Computing

Random Phases



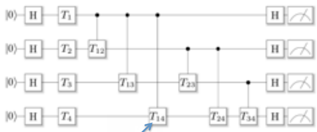
Each of these T_m gates Is a rotation (around z) by a multiple of $\pi/4$:

$$T_m = \cos\left(\frac{k_m \pi}{8}\right) I + i \sin\left(\frac{k_m \pi}{8}\right) Z_m$$

$$T_m = R_z\left(-\frac{k_m \pi}{4}\right)$$ on the m^{th} qubit

Where k_m is an integer chosen uniformly at random between 0 and 7. This is equivalent (up to a global phase) of applying a T gate k_m times.

Random Joint Phases



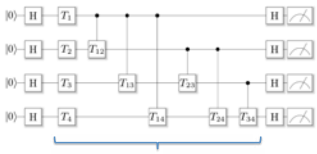
Each of these T_{mn} gates Is a joint phase rotation by a multiple of π/8:

$$T_{mn} = \cos\left(\frac{k_{mn}\pi}{8}\right) I + i \sin\left(\frac{k_{mn}\pi}{8}\right) Z_m Z_n$$

Where k_m is an integer chosen uniformly at random between 0 and 7.

In the lab we can implement a similar algorithm with controlled T_{mn} gates.

"Instantaneous"



All of these phase gates *commute*

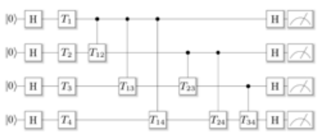
The order which you apply the single and two qubit phase gates doesn't matter. They commute with each other, so can be applied in any order.

Eg. $$ZT_2 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}$$

$$T_2 Z = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}$$

Diagonal gates commute

Collapse of the polynomial hierarchy



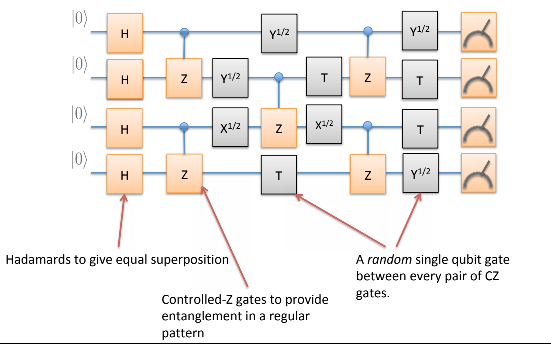
Aim: To sample from the output of this circuit. Easy for a quantum computer.

If this could be done efficiently using a classical computer, it would imply the collapse of the polynomial hierarchy (and so isn't expected to be possible).

Practically, classical simulations are limited to <50-70 qubits (for low error rates).

Pseudorandom Circuits

The circuit



Square Root X and Y

In the previous slide, we simply have that

$$X^{1/2} = R_x\left(\frac{\pi}{2}\right)$$

and similarly,

$$Y^{1/2} = R_y\left(\frac{\pi}{2}\right)$$

Schedule of CZ

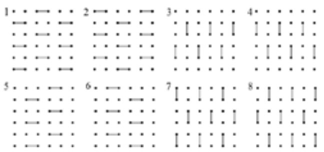
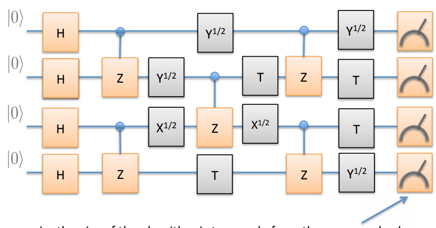


FIG. 6. Layouts of CZ gates in a 6×6 qubit lattice. It is currently not possible to perform two CZ gates simultaneously in two neighboring superconducting qubits [33, 34, 49, 52]. We iterate over these arrangements sequentially, from 1 to 8.

From Boixo et al, https://arxiv.org/pdf/1608.00263.pdf, 2016.

Sampling is hard



Once again, the aim of the algorithm is to sample from the measured values.

If this were possible to do efficently classically, it would imply a collapse of the polynomial hierarchy.

In practice, simulating ~45 qubits for this problem is hard (note: need high depth circuit)

What do you need for quantum supremacy?

- Many qubits (>~50)
- Large depth circuit (>~100)
- Entanglement, high T gate count
- Low error rates (<1%)

PHYC90045 Introduction to Quantum Computing

60 Qubit Simulations of Shor's algorithm

Using MPS, Aidan Dang wrote parallel code were able to do large scale simulations of Shor's algorithm.

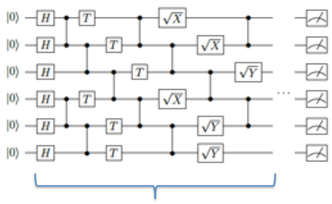
l	r	α	β	n_{node}	t_U	t_{meas}	t_{QFT}	t_{total}
16	28140	2	7035	2	1538	353	4290	6181
17	57516	2	14379	24	1694	406	4544	6644
20	479568	4	29973	216	4271	1496	20236	26003

Table 3.2: Further QCMPS benchmarks, this time across multiple nodes of a supercomputer. Each node has 24 cores and 64 GB of RAM. With n_{node} nodes, we simulated the three cases $l = 16$, $N = 56759$, $a = 2$; $l = 17$, $N = 124631$, $a = 2$; and also $l = 20$, $N = 961307$, $a = 5$.

Source: Dang, thesis, 2017

PHYC90045 Introduction to Quantum Computing

Simulating Google's Pseudo-Random Circuits



Depth of the circuit is equivalent to the number of timesteps

PHYC90045 Introduction to Quantum Computing

Importance of Depth

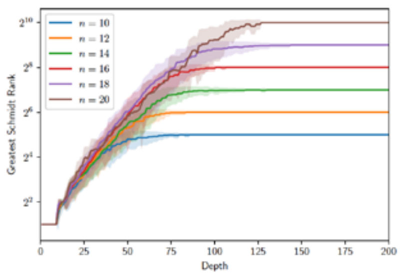
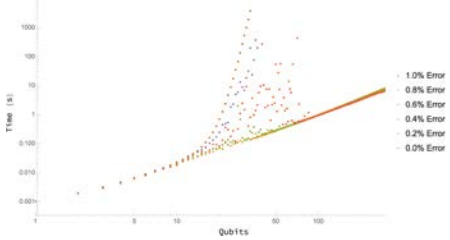


Figure 4.3: Mean maximum Schmidt ranks in the MPS as a function of depth. Sampled over 100 instances, the shading represents one standard deviation.

Source: Dang, thesis, 2017

Effects of Errors on IQP simulation runtime



What do you need for quantum supremacy?

- Many qubits (>~50)
- Large depth circuit (>~100)
- Entanglement, high T gate count
- Low error rates (<1%)

Week 6

Lecture 11 - Quantum Supremacy
11.1 Boson Sampling
11.2 IQP Problem
11.3 Google's pseudorandom circuits

Lecture 12 - Errors
12.1 Quantum errors: unitary and stochastic errors
12.2 Randomized Benchmarking
12.3 Tomography

Lab 6
Quantum Supremacy and Errors