Quantum Computation (Preskill ch. 6)

Quantum Computation

Classical Circuits

* Universal Gates
* Circuit Complexity

Quantum Circuits

* Quantum Complexity
* Universal Quantum Gates

Review of Classical Circuit Theory

Think of a Computation as a function that maps *n* bits to *m* bits

$f: \{0,1\}^n \to \{0,1\}^m$

Maps *n* bits to *m* bits

A function with an *m* bit output is equivalent to *m* functions with a *one* bit output, so the basic task can be broken into *m* functions mapping *n* bits to *one* bit

There are 2^n possible inputs w/2 possible outputs, so a total of 2^{2^n} functions that map *n* bits to *one* bit

$f: \{0,1\}^n \to \{0,1\}$

2^{2^n} of these simple functions

Function evaluation ⟷ sequence of logic operations

Given a binary input $x = x_1 x_2 \ldots x_n$

⇨ separate in sets $\begin{cases} f(x) = 1 \\ f(x) = 0 \end{cases}$

Consider the input

$x^{(a)}: f(x^{(a)}) = 1$ ⇨ define $f^{(a)}(x) = \begin{cases} 1 & \text{for } x = x^{(a)} \\ 0 & \text{for } x \neq x^{(a)} \end{cases}$

one of the *m* simple functions

n of these

Given, for example, we implement $f^{(a)}$ w/logic operations

$x = \begin{cases} 111\ldots \to f(x) = x_1 \wedge x_2 \wedge x_3 \ldots \wedge x_n \\ 0110\ldots \to f(x) = (\neg x_1) \wedge x_2 \wedge x_3 \wedge (\neg x_4) \ldots \end{cases}$

Finally, given the $f^{(a)}(x)$'s we can implement the $f(x)$'s *as*

$$f(x) = f^{(1)}(x) \vee f^{(2)}(x) \vee \ldots \vee f^{(m)}(x)$$

Quantum Computation (Preskill ch. 6)

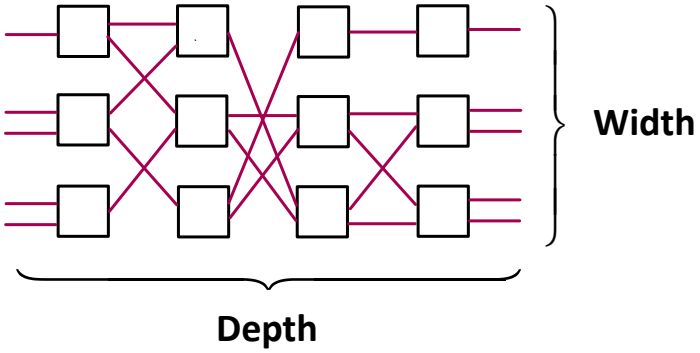
Circuit Complexity

(Pick a universal gate set)

Central Question: How hard is it to solve PROBLEM ?

* One measure is the size of the smallest circuit that solves it

Size = Width x Depth



Consider a circuit family $\{C_n\}$ that solves a decision problem $f: \{0,1\}^n \to \{0,1\}$

Examples

FACTORING

$$f(x,y) = \begin{cases} 1 & \text{if integer } x \text{ has divisor } < y \\ 0 & \text{otherwise} \end{cases}$$

HAMILTONIAN PATH

$$f(x,y) = \begin{cases} 1 & \text{if graph } x \text{ has Hamiltonian Path} \\ 0 & \text{otherwise} \end{cases}$$

We define:

Easy Problems: $size(C_n) \leq poly(n)$

Hard Problems: $size(C_n) > poly(n)$

This distinction allows us to define Complexity Classes, for example

Problem Class **P** = { Decision Problems solved by a polynomial-sized circuit }

Quantum Computation (Preskill ch. 6)

Consider a circuit family $\{C_n\}$ that solves a decision problem

$f: \{0,1\}^n \to \{0,1\}$

Examples

FACTORING

$$f(x,y) = \begin{cases} 1 & \text{if integer } x \text{ has divisor } < y \\ 0 & \text{otherwise} \end{cases}$$

HAMILTONIAN PATH

$$f(x,y) = \begin{cases} 1 & \text{if graph } x \text{ has Hamiltonian Path} \\ 0 & \text{otherwise} \end{cases}$$

We define:

Easy Problems: $size(C_n) \leq poly(n)$

Hard Problems: $size(C_n) > poly(n)$

This distinction allows us to define Complexity Classes, for example

Problem Class P = { Decision Problems solved by polynomial-sized circuit }

* Whether PROBLEM $\in P$ is independent of circuit design, universal gate set & other specifics

* Problems in P are special – they have structure that allows efficient computation

Note: The majority of functions $\notin P$

For example, if the output $f(x) \sim$ random we must compute $f(x)$ by lookup table with 2^n entries

Circuit that does lookup has exponential size

Special Class: One-Way Function

Problem Class NP = { PROBLEM is easy or hard, but the answer is easy to check }

Stands for "Non-deterministic Polynomial Time

Examples: FACTORING $\in$ NP

HAMILTONIAN PATH $\in$ NP

Note: Clearly $P \subseteq NP$, Conjecture that $P \neq NP$

Quantum Computation (Preskill ch. 6)

* Whether PROBLEM $\in P$ is independent of circuit design, universal gate set & other specifics

* Problems in P are special – they have structure that allows efficient computation

Note: The majority of functions $\notin P$

For example, if the output $f(x) \sim$ random we must compute $f(x)$ by lookup table with 2^n entries

Circuit that does lookup has exponential size

Special Class:

One-Way Function

Problem Class NP = { PROBLEM is easy or hard, but the answer is easy to check }

Stands for "Non-deterministic Polynomial Time

Examples: FACTORING $\in$ NP

HAMILTONIAN PATH $\in$ NP

Note: **Clearly** $P \subseteq NP$, **Conjecture** that $P \neq NP$

Special Problem: CIRCUIT-SAT $\in$ NP

Input = Circuit w/n gates, m input bits

Problem = is there an m-bit input w/output = 1

$$f(C) = \begin{cases} 1 & \text{if } \exists\, x^{(m)} \text{ so } C(x^{(m)}) = 1 \\ 0 & \text{otherwise} \end{cases}$$

Easy to check solution because if we have the input circuit C we can run it with the input $x^{(m)}$ and determine if it evaluates to 1.

Cooks Theorem: Every PROBLEM $\in$ NP is polynomially reducible to CIRCUIT-SAT

CIRCUIT-SAT
HAMILTONIAN PATH
} $\in$ NP- Complete

NPC $\neq$ NP

Quantum Computation (Preskill ch. 6)

Special Problem: CIRCUIT-SAT ∈ NP

Input = Circuit w/*n* gates, *m* input bits

Problem = is there an *m*-bit input w/output = 1

$$f(C) = \begin{cases} 1 & \text{if } \exists\, x^{(m)} \text{ s.t. } C(x^{(m)}) = 1 \\ 0 & \text{otherwise} \end{cases}$$

Easy to check solution because if we have the input circuit C we can run it with the input $x^{(m)}$ and determine if it evaluates to 1.

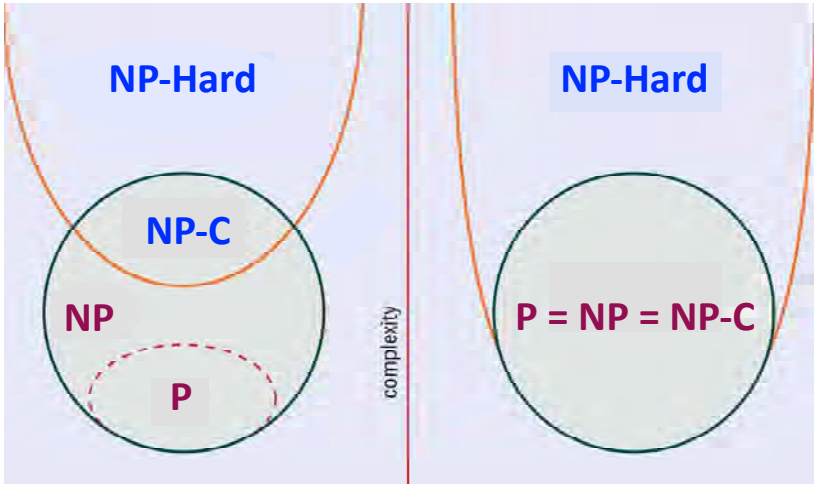
Cooks Theorem: Every PROBLEM ∈ NP is polynomially reducible to CIRCUIT-SAT

CIRCUIT-SAT
HAMILTONIAN PATH
} ∈ NP- Complete

NPC ≠ NP

Complexity Hierarchy

- **Conjecture**: P ∈ NP
- ∃ Problems in NP that are neither P or NPC
- NPI: Problems of intermediate difficulty
- Conjecture: Factoring ∈ NPI



P ≠ NP P = NP

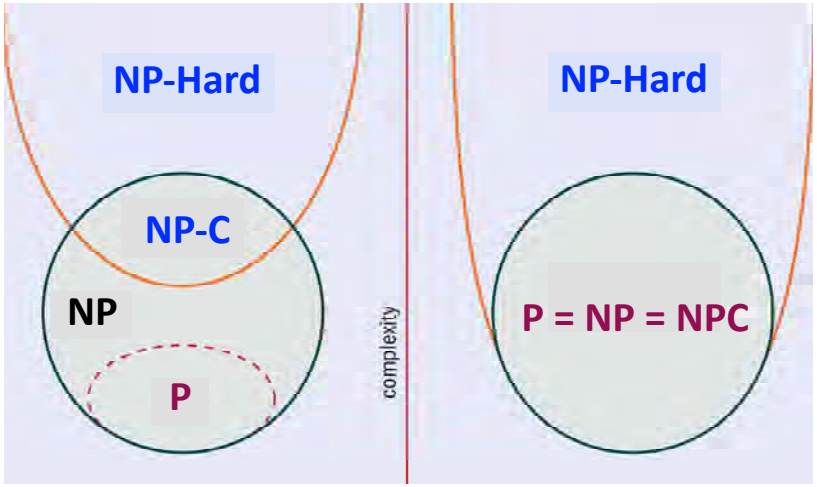
Takeaway Message

- Complexity theory is a rich field with many known complexity classes
- Many foundational conjectures remain unproven
- As we will see, switching to Quantum Circuits changes things

Quantum Computation (Preskill ch. 6)

Complexity Hierarchy

* Conjecture: P ∈ NP
* ∃ Problems in NP that are neither P or NPC
* NPI: Problems of intermediate difficulty
* Conjecture: Factoring ∈ NPI



P ≠ NP P = NP

Takeaway Message

* Complexity theory is a rich field with many known complexity classes
* Many foundational conjectures remain unproven
* As we will see, switching to Quantum Circuits changes things

Aside: Classical Reversible Computation

Motivation:

Quantum Computation = Unitary Transformation

⇓

Reversible !

Classical Reversible Comp: $f: \{0,1\}^n \to \{0,1\}^n$

Repackage $f: \{0,1\}^n \to \{0,1\}^m$ as reversible

$f: \{0,1\}^{n+m} \to \{0,1\}^{n+m}$

$f(x, 0^{(m)}) = (x; f(x))$

we separate $n + m$ qubit register into input and output so no information is lost

Note: Not all 1 & 2-bit gates are reversible, e. g., AND, OR, ERASE

Quantum Computation (Preskill ch. 6)

Quantum Circuits

Classical Computer = finite set of gates acting on bits

Quantum Computer = finite set of quantum gates acting on quantum bits

Quantum Computation:

$U: |000\ldots0\rangle \to |\psi\rangle$

- unitary composed of finite # of gates
- *n* qubit input
- output = outcome of Orthog. Measurement in basis $\{|0\rangle, |1\rangle\}^n$

Note:

* The Hilbert space of the Quantum Computer has a preferred decomposition into tensor producs of low dimensional spaces (qubits), respected by gates which act on only a few qubits at a time.

  - This helps establish notion of Quantum Complexity

* Decomposition into subsystems and local manipulations means gates act on qubits in a bounded region.

* It is suspected, but not proven, that the power of Q. C. derives from this decomposition:

> n qubits -> 2^n dimensional $\mathcal{H}$ resource grows $\sim 2^n$

* Unitaries form a continuum, but we restrict to discrete gate sets. This is necessary for Fault Tolerance

* Quantum Gates could be Superoperators, and readout could be POVM's

However:

we can simulate | Superoperators as unitaries
POVM's as Orthog. Meas | in larger $\mathcal{H}$

Our simpler conceptualization is general

Quantum Computation (Preskill ch. 6)

* It is suspected, but not proven, that the power of Q. C. derives from this decomposition:

n qubits -> 2^n dimensional $\mathcal{H}$ resource grows $\sim 2^n$

* Unitaries form a continuum, but we restrict to discrete gate sets. This is necessary for Fault Tolerance

* Quantum Gates could be Superoperators, and readout could be POVM's

However:

we can simulate | Superoperators as unitaries
POVM's as Orthog. Meas | in larger $\mathcal{H}$

Our simpler conceptualization is general

* Final readout could be collective or in a basis ≠ the standard logical basis →

  Unitary maps to standard basis $\{|0\rangle, |1\rangle\}^n$ with overhead included in complexity

* We could do measurements during computation, then condition later steps on the outcomes. But one can show the same results can be achieved by measuring at the end of the computation

  - In practice measurement during computation is essential for error correction

Note: None of the above changes notion of complexity

Quantum Computation (Preskill ch. 6)

* Final readout could be collective or in a basis ≠ the standard logical basis →

  Unitary maps to standard basis $\{|0\rangle, |1\rangle\}^n$ with overhead included in complexity

* We could do measurements during computation, then condition later steps on the outcomes. But one can show the same results can be achieved by measuring at the end of the computation

  - In practice measurement during computation is essential for error correction

Note: None of the above changes notion of complexity

At this point we are left with 3 main issues

(1) Universality: we must be able to implement the most general unitary $\in SU(2^n)$

↑ group of unitaries in $\mathcal{H}$, $\dim \mathcal{H} = 2^n$

→ Circuit of chosen gates must approx. any $U \in SU(2^n)$

(2) Quantum Complexity:

New class **BQP** [1]

↑ Decision problems solved w/high prob. by poly-sized quantum circuits

(3) Accuracy: BQP is defined assuming perfect gates. What happens if circuit elements do not have exponential accuracy?

Can show noisy gates are OK:

T - gate circuit requires error prob. $\propto 1/T$

[1] BQP = Bounded-error Quantum Polynomial time

Quantum Computation (Preskill ch. 6)

At this point we are left with 3 main issues

(1) Universality: we must be able to implement the most general unitary $\in SU(2^n)$

↑ group of unitaries in $\mathcal{H}$, $\dim \mathcal{H} = 2^n$

→ Circuit of chosen gates must approx. any $U \in SU(2^n)$

(2) Quantum Complexity: New class **BQP**

Decision problems solved w/high prob. by poly-sized quantum circuits

(3) Accuracy: BQP is defined assuming perfect gates. What happens if circuit elements do not have exponential accuracy?

Can show noisy gates are OK: T - gate circuit requires error prob. $\propto 1/T$

[1] BQP = Bounded-error Quantum Polynomial time

Note on Quantum Complexity:

A QC can simulate a probabilistic classical computer (most general class)

→ $BPP^{2)} \subseteq BQP$

Open Question: Is $BPP \neq BQP$? Seems reasonable, as a prob. C.C. cannot easily simulate QM in a 2^n – dimensional Hilbert space.

If so, a QC will negate the Strong Church-Turing Thesis which holds that any physically reasonable model of computation can be simulated on a probabilistic classical computer with only polynomial slowdown.

[2] BPP = Bounded-error Probabilistic Polynomial time

Quantum Computation (Preskill ch. 6)

Definition:

Let $U = e^{iH_j dt}$ be generic (H_j is the generator of U)

$\exists n \in N_0$ so U^n comes arbitrarily close to $U(\alpha) = e^{i\alpha H_j}$

($U(\alpha)$ is reachable by powers U^n)

Seems extraordinarily cumbersome! Why do it that way?

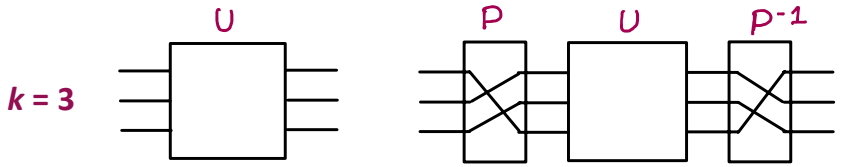
Answer: This is necessary for Fault Tolerant Operation

$\{U^n, n \in N_0\}$ is a set of measure zero → any "noise takes us to an invalid state that can be detected and corrected.

This is not enough! What else can we do?

(2) Switching leads

k qubits → $(2^k)!$ permutations $U' = P U P^{-1}$



This is not enough! What else can we do?

Aside: Consider a d - dimensional Hilbert space $\mathcal{H}$.

→ { Operators ($d \times d$ matrices) are vectors $\in d^2$ dim. Hilbert space $\mathcal{H}'$ w/a scalar product defined as

$$(m_i | m_j) = \mathrm{Tr}[m_i^+ m_i]$$

$\exists$ orthonormal basis $\{|A_1\rangle, \ldots |A_{d^2}\rangle\}$ in $\mathcal{H}'$

$$\langle A_i | A_j \rangle = \delta_{ij}$$

(3) Completing the Lie Algebra

Assume access to a set of Hamiltonians

$$\{H_0, H_1, \ldots H_n\}, \; n \leq (\dim \mathcal{H})^2$$

Trotter Formulae: for dt -> 0

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} = e^{-i(\alpha H_j + \beta H_k)dt}$$ (Lin. Comb. of H_j, H_k)

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} e^{i\alpha H_j dt} e^{i\beta H_k dt} = e^{-[\alpha H_j, \beta H_k] dt^2}$$

(NL. Comb. of H_j, H_k)

Quantum Computation (Preskill ch. 6)

Aside: Consider a d - dimensional Hilbert space $\mathcal{H}$.

➡ { Operators ($d \times d$ matrices) are vectors $\in d^2$ dim. vector space $\mathcal{H}'$ w/a scalar product defined as

$$(m_i | m_j) = \mathrm{Tr}[m_i^\dagger m_i]$$

$\exists$ orthonormal basis $\{|A_1\rangle, \ldots |A_{d^2}\rangle\}$ in $\mathcal{H}'$

$$(A_i | A_j) = \delta_{ij}$$

(3) Completing the Lie Algebra

Assume access to a set of Hamiltonians

$$\{H_0, H_1, \ldots H_n\},\ n \in (\dim \mathcal{H})^2$$

Trotter formulae: for dt -> 0

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} = e^{-i(\alpha H_j + \beta H_k)dt}$$

$$e^{-i\alpha H_j dt} e^{-i\beta H_k dt} e^{i\alpha H_j dt} e^{i\beta H_k dt} = e^{-[\alpha H_j, \beta H_k] dt^2}$$

* From the Set $\{H_0, H_1, \ldots H_n\}$ we can "simulate" new Hamiltonians using the Trotter formulae
* If a new Hamiltonian is linearly independent we add it to the set.
* Continue until the Set has $d^2 = (\dim \mathcal{H})^2$ linearly independent members (Lie Algebra complete)*)

➡ Set is a basis in $d^2 \times d^2$ matrix space
Allows to simulate any $H(t)$ & implement any U

Examples:

$d = 2$ ➡ $\{|A_i\rangle\} = \{I, \sigma_x, \sigma_y, \sigma_z\}$ ⬅ set of $2^2 = 4$ 2 x 2 matrices

$d = 4$ ➡ $\{|A_i\rangle\}$ ⬅ set of $d^2 = 16$ 4 x 4 matrices

Example: (single qubit control)

Let $d = 2$, initial set $\{\alpha\sigma_x, \beta\sigma_y\}$ (generic)

$[\sigma_x, \sigma_y] = i\sigma_z$ ➡ we can simulate $i\delta\sigma_z$

Set $[I, \alpha\sigma_x, \beta\sigma_y]$ sufficient for control

Quantum Computation (Preskill ch. 6)

Deutsch's Gate

(First generic gate, Reaches any $U \in SU(8)$)