A Tour to Computational Complexity

Theory Perspective: Classic and Quantum

Extended slides presented on NYC Quantum meet up 07/2020

Yong Yao NJ 2020

yaoyong99@gmail.com

Content

- Introduction
- Classic world
- Quantum world
- Here we are and where we go
- Human Intelligence

Introduction

- Turing machines and its universality
- Limitations
- P vs NP
- Applications
- Challenges from quantum world

Turing machine and its universality

- Profound outcomes of Hilbert program (*Entscheidungsproblem*)
  - Formalization and axiomatization of mathematics
  - Gödel's incompleteness theorems
  - Turing machine
  - Deepened greatly the understanding of our human's intelligence and reasoning

- University
  - Abstract computation models
    - Lambda calculus/Wang machine/Cellular automata/formal grammar
  - Recreation: Game of life (Conway), Video/card games
  - Math Diophantine sets = RE
  - Physics(?) MIP* = RE
  - Programming languages
    - General purpose languages (C, C++, Java etc), C++ templates/PLSQL

- Church-Turing thesis All computational models are Turing-equivalent
- Church–Turing–Deutsch principle Turing machine can simulate every physical process

Powerful but with limitations

- Power limitations
  - The halting problem is undecidable
  - There are uncomputable functions (Busy beaver problem)
  - No efficient algorithms found yet for NP-complete problems

- Efficiency
  - NP-complete problems — most amazing discovery in complexity theory
  - Strong Church-Turing thesis
    - All physical realized computational models are polynomial equivalent
    - Challenge from quantum computational models.
  - Completeness is universal
    - Most (natural defined) complexity classes have complete languages
- Limitations are universal
  - Undecidable everywhere
  - NP-complete problems everywhere

P vs NP

P vs NP is one of the most important open theory problems

Seven Clay millennium problems could be proved in one shot if you can prove P=NP

- What is feasible computation
  - Polynomial time P (with randomness BPP, with quantum BQP)
  - Why polynomial – explain later
- What is NP
  - Solution can be efficiently (polynomial time) verified
- NP-complete problems
  - SAT
  - TSP, Graph coloring
  - University

Intuition meaning of P vs NP

- Is it the same hard to find a proof as to check a proof?
- Are there "genius" creativities?

Aerospace engineering. Optimal mesh partitioning for finite elements.
Biology. Phylogeny reconstruction.
Chemical engineering. Heat exchanger network synthesis.
Chemistry. Protein folding.
Civil engineering. Equilibrium of urban traffic flow.
Economics. Computation of arbitrage in financial markets with friction.
Electrical engineering. VLSI layout.
Environmental engineering. Optimal placement of contaminant sensors.
Financial engineering. Minimum risk portfolio of given return.
Game theory. Nash equilibrium that maximizes social welfare.
Mathematics. Given integer a_1, ..., a_n, compute $\int_0^{2\pi} \cos(a_1\theta) \times \cos(a_2\theta) \times \cdots \times \cos(a_n\theta)\, d\theta$
Mechanical engineering. Structure of turbulence in sheared flows.
Medicine. Reconstructing 3d shape from biplane angiocardiogram.
Operations research. Traveling salesperson problem.
Physics. Partition function of 3d Ising model.
Politics. Shapley–Shubik voting power.
Recreation. Versions of Sudoku, Checkers, Minesweeper, Tetris, Rubik's Cube.
Statistics. Optimal experimental design.

P vs NP

- Structure of complexity classes
  - We have limit understanding of complexity classes hierarchy

    $$L \subseteq NL \subseteq P \subseteq NP \subseteq PSPACE \subseteq EXP \subseteq NEXP \subseteq EXPSPACE$$

    $L \subsetneq PSPACE$ and $P \subsetneq EXP$ but no proper inclusions in the above has been proved
  - NP-intermediate
    - [Ladner1975] There exist intermediate problems if P != NP
  - Isomorphism Conjecture → P != NP
    - All NP-complete problems are actually the same problem in different formats

- Resolving P vs NP is REALY hard
  - Logic, Relative, Combinatorics, Algebra, Algebraic geometry approaches
- Important concepts
  - Completeness, Interactive proof, Pseudorandom, Derandomization, One-way functions
- Interactive + randomness is powerful
  - The PCP Theorem and Hardness of Approximation
  - Zero-knowledge proofs

Applications

- SAT solvers
- Pseudorandom generator
- Derandomization algorithms
- Cryptography
- Machine learning
- Multiparty computation (MPC)
- Zero knowledge proofs

Challenges from quantum world

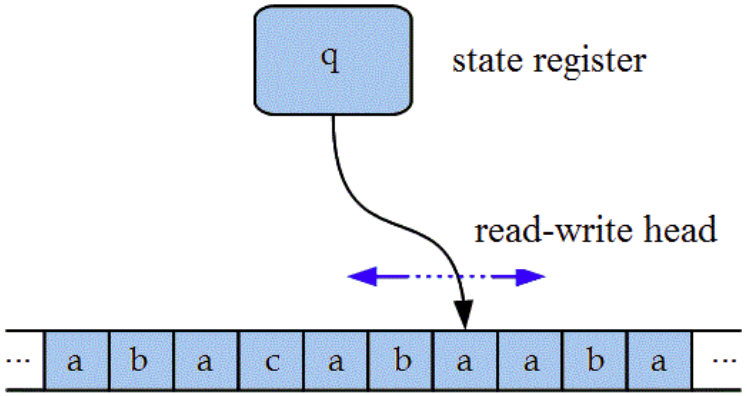
- Quantum world
  - Shor's factoring algorithm — factor n-bit integer in polynomial
  - Grover's algorithm — search n unordered items in square root of n
  - MIP* = RE

- Computational complexity meets physics
  - How much quantum physics can help?
    - [Deutch]Computation bases on physical process thus its limit is decided by physical laws.
    - Quantum computing models
    - Adding non-linearity into quantum theory can help (solve NPC problems efficiently)
    - What kind of physics theory can imply efficient algorithms for NPC problems?
  - Computational complexity to physics
    - Physics is NP hard — Extracting dynamical equations from experimental data is NP hard
    - Church–Turing–Deutsch principle — Turing machine can simulate every physical process
    - Emergentism
    - [Aaronson]P != NP as a principle of physics

Classic World

- Basic concepts
  - Models and reductions
  - Complexity classes and their properties
- Some key concepts
  - Isomorphism conjecture, NP-complete set and dense set
  - One-way functions, circuit complexity, and pseudorandomness
  - Interactive proof system, PCP theorem and Zero-Knowledge proof
  - The Unique Games Conjecture
  - Average-case complexity
  - Dealing with NP-hard problems
  - Phase transition in NP-Complete problem
- P vs NP
- Topics not covered

Turing Machine



Formal Definition

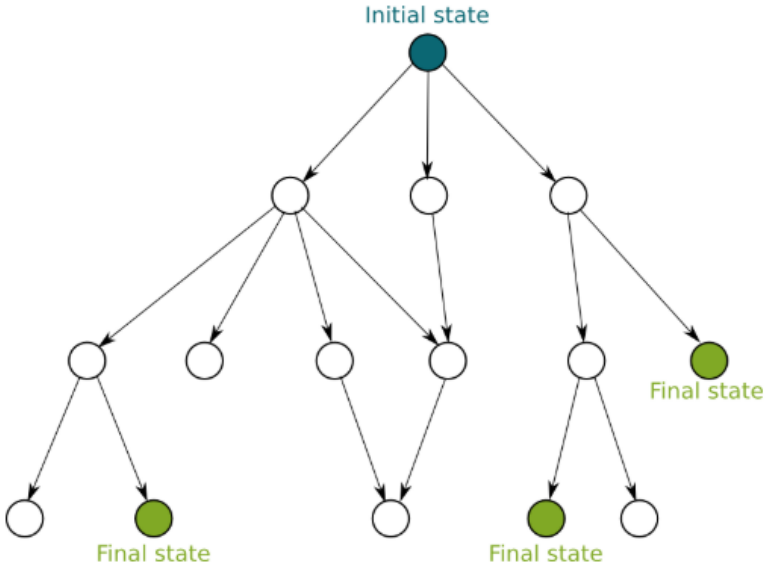
A Turing machine is formally defined as a 7-tuple $\langle Q, q_0, F, \Gamma, b, \Sigma, \delta \rangle$, where

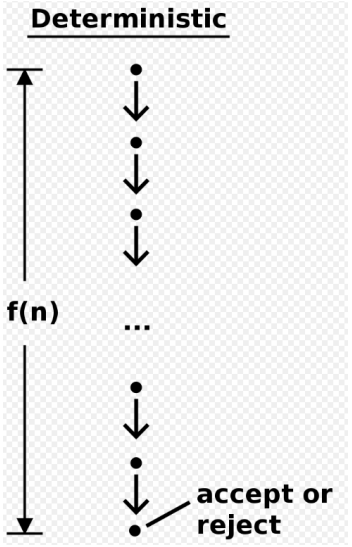
- Q is a finite, non-empty set of *states*
- $q_0 \in Q$ is the *initial state*
- $F \subset Q$ is the set of *accepting states*
- Γ is a finite, non-empty set of *tape symbols*
- $b \in \Gamma$ is the *blank symbol*
- $\Sigma \subset \Gamma \setminus \{b\}$ is a set of *input symbols*
- $\delta : (Q \setminus F) \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is the *transition function*, which is a partial function

Nondeterministic Turing Machine

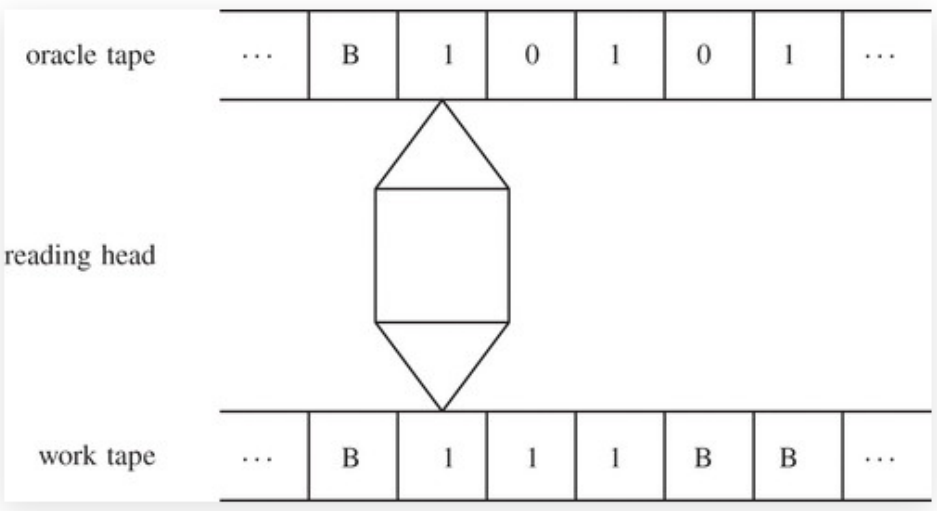
- Transition function

$$\delta: (Q/F) \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L,R\}}$$





Oracle Turing Machines



- Language accept by a Turing machine M

$$L(M) := \{x | M(x) \ halts\}$$

- With oracle A, when run Turing machine M we can check if $x \in A$ without cost. We can think oracle A as smart advisor

$$M^A$$

- Complement to a complexity class A

$$coA = \left\{ L | \overline{L} \ is \ in \ class \ A \right\}$$

Turing Machines

- Turing machine is a 7-tuple $\langle Q, q_0, F, \Gamma, b, \Sigma, \delta \rangle$

$$All\ 7\ elments\ are\ finite \Rightarrow we\ can\ enumerate\ all\ TMs$$

$$TM_1, TM_2, \cdots, TM_n, \cdots$$

$$\Rightarrow Universal\ TM\ exists (the\ enumerate\ procedure)$$

$$\Rightarrow By\ \text{diag}onization\ Halting\ problem\ is\ undecidable$$

$$\Rightarrow Rice\ Theorem\ Any\ non\ trival (not\ full\ or\ empty)\ set\ is\ undecidable$$

$$Example, in\ general\ we\ cannot\ decide\ which\ program\ is\ virus$$

- Rice Theorem
  - Every problem corresponds to a language $x \in L$?
  - Non trivial set is a set not empty or full set
  - Whether or not a program is in a given non trivial set is undecidable
  - For example, given any program, check its output is 2. This is undecidable
  - There cannot be anti-virus software

Reduction and Completeness

- Reduction
  - If we know solution to language B then we will also know solution to language A
  - B is “harder” then A

$$A \leq B$$

  - There are many kinds of reductions
  - Mapping reduction/many-one reduction: there is a computable function f: $\Sigma^* \to \Sigma^*$

$$w \in A \Leftrightarrow f(w) \in B$$

$$A \leq_m B$$

  - Polynomial-time reduction: f is polynomial
  - Turing reduction

- Completeness
  - A language A is completed for a complexity class C if any language in C can be reduced to A
  - A is the hardness problem in a complexity class C

Complexity classes

- Time complexity
  - Running steps

$$T_M(n) = \max\{t_M(x) | x \in \Sigma^*, |x| = n\}$$

- Space complexity
  - Spaces head scanned

$$DSPACE(s(n)) \subseteq DTIME(2^{s(n)})$$

- Space-time tradeoff
  - More space →less time
  - Lookup tables
  - Compressed vs uncompressed data
  - Bigger code -- Loop unrolling

$$L = SPACE(\log n)$$
$$NL = NSPACE(\log n)$$
$$P = TIME(n^{O(1)}) = \cup_{k \geq 1} TIME(n^k)$$
$$NP = NTIME(n^{O(1)}) = \cup_{k \geq 1} NTIME(n^k)$$
$$PSACE = SPACE(n^{O(1)}) = \cup_{k \geq 1} SPACE(n^k)$$
$$NPSPACE = NSPACE(n^{O(1)}) = \cup_{k \geq 1} NSPACE(n^k)$$
$$E = TIME(2^{O(n)}) = \cup_{k \geq 1} TIME(2^{kn})$$
$$NE = NTIME(2^{O(n)}) = \cup_{k \geq 1} NTIME(2^{kn})$$
$$EXP = TIME(2^{n^{O(1)}}) = \cup_{k \geq 1} TIME(2^{n^k})$$
$$NEXP = NTIME(2^{n^{O(1)}}) = \cup_{k \geq 1} NTIME(2^{n^{O(1)}})$$
$$EXPSPACE = SPACE(2^{n^{O(1)}}) = \cup_{k \geq 1} SPACE(2^{n^k})$$

Quasi polynomial

$$QP = TIME(2^{(\log n)^{O(1)}}) = \cup_{c \in N} TIME(2^{(\log n)^c})$$
$$SUBEXP = \cap_{\epsilon > 0} TIME(2^{n^\epsilon})$$

Polynomial Hierarchy (PH)

P is the set of polynomial solvable decision problems

$$\Delta_0 := \Sigma_0 := \Pi_0 := P$$

$$\Delta_{i+1} := P^{\Sigma_i}$$

$$\Sigma_{i+1} := NP^{\Sigma_i}$$

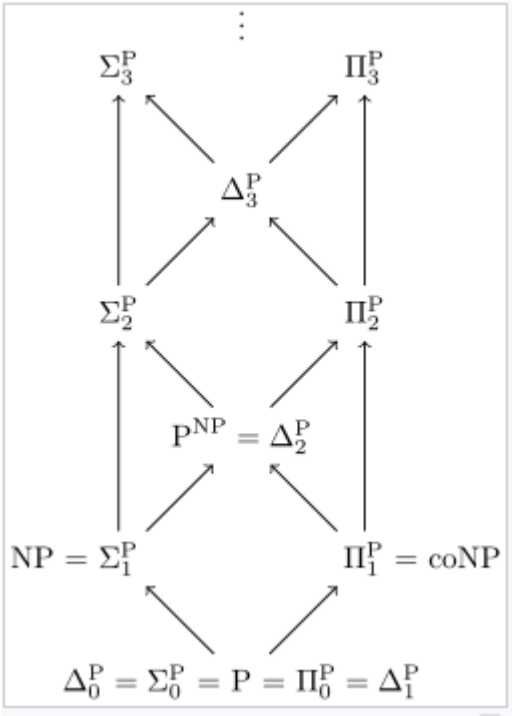
$$\Pi_{i+1} := coNP^{\Pi_i}$$

$$PH := \cup_i \Sigma_i := \cup_i \Pi_i$$

$$\Sigma_i := \{L \mid \exists y_1 \forall y_2 \cdots Q_i y_i R(x, y_1, y_2, \cdots, y_i)\},\ R \text{ is polynomial}$$

$$\Pi_i := \{L \mid \forall y_1 \exists y_2 \cdots Q_i y_i R(x, y_1, y_2, \cdots, y_i)\}$$

$$PH \subseteq PSPACE$$



PH collapses

$$If\ \Sigma_i = \Pi_i,\ for\ some\ i\ then\ PH = \Sigma_i$$

Conjecture: PH does not collapse

General Properties of Complexity Class

- Time vs Space

$$
\begin{aligned}
TIME(f(n)) &\subseteq SPACE(f(n)) \\
SPACE(f(n)) &\subseteq TIME(2^{f(n)}),\ f(n) \geq \log n \\
NTIME(f(n)) &\subseteq TIME(2^{f(n)}) \\
NSPACE(f(n)) &\subseteq SPACE(f^2(n)),\ f(n) \geq \log n,\ \textit{Savitch Theorem}
\end{aligned}
$$

  - We understand time better than space

- Blum's speedup theorem — We cannot define optimal algorithm uniformly
  - For any complexity measure(Blum measure) there is problem we can speedup arbitrarily
  - Blum complexity measure (most generic complexity measure, includes time and space)
  - Given any Blum complexity measure $(\varphi, \Phi),\ \varphi\ is\ Gödel\ numbering, \Phi\ is\ measurement$ and a total computational function (speed up) f, there exists a total computable predicate g such that for every algorithm i for g there exists algorithm j for g so that for almost all x, we have

$$f(x, \Phi_j(x)) \leq \Phi_i(x)$$

General Properties of Complexity Class

- The more resources the more power?
  - The functions can be used as boundaries of algorithms are called constructible
  - Time -- Yes
    - Time hierarchy theorem

$$TIME\left(o\left(\frac{f(n)}{\log f(n)}\right)\right) \subset TIME(f(n)),\ f(n) \geq n,\ constructible$$

$$TIME(f(n)) \subset TIME(f^2(n))$$

$$NTIME(f(n)) \subset NTIME(g(n)),\ f(n+1) = o(g(n)),\ g\ constructible$$

    - Consequences
      - **P** has arbitrary large exponent to solve: **P** does not collapse to **DTIME**(n^k) for any fixed k.
      - **P** is strictly contained in **EXPTIME**

$$P \subseteq TIME(2^n) \subset TIME(2^{2n}) \subseteq EXPTIME$$

    - Multiple Tapes

$$\delta: (Q/F) \times \Gamma^k \to Q \times (\Gamma \times \{L,R\})^k$$

$$Running\ in\ f(n)\ can\ be\ simulated\ by\ \text{single}\ tap\ in\ O(f^2(n))$$

General Properties of Complexity Class

- The more resources the more power?
  - Space -- Yes
    - Space hierarchy theorem

$$SPACE(o(f(n))) \subset SPACE(f(n)),\ f(n) \geq \log n,\ constructible$$

    - Consequences
      - **PSPACE** has arbitrary large exponent to solve: **PSPACE** does not collapse to **PSPACE**(n^k) for any fixed k.
      - **NL, PSPACE, and EXPSPACE**

$$NL \subseteq SPACE(\log^2 n) \subset PSPACE \subset EXPSPACE$$

    - Translation lemma

$$Let\ t_1, t_2, f\ are\ constructible,\ if\ TIME(t_1(n)) = TIME(t_2(n))\ then$$
$$TIME(t_1(f(n))) = TIME(t_2(f(n)))$$

General Properties of Complexity Class

- The more resources the more power ?
  - No
    - Gap Theorem For Blum complexity measure Φ, given any total recursive function g(n) ≥ n, there exists a total recursive function *t*(*n*) such that with respect to Φ the complexity classes with resources t(n) and g(t(n)) are the same.
    - For example DTIME(*t*(*n*)) = DTIME(*g*(*t*(*n*)))
    - t(n) may be very large or nonconstructible

- Complementation theorems

Let t and s are constructible and $s(n) \geq \log n$. Then following classes are closed under complementation

1. $TIME[t]$
2. $SPACE[s]$
3. $NSPACE[s]$

Known and Open

- By definition

$$L \subseteq NL \subseteq P \subseteq NP \subseteq PH \subseteq PSPACE = NPSPACE \subseteq EXP \subseteq NEXP \subseteq EXPSPACE$$

- By time hierarchy and space hierarchy

$$L \subset PSPACE$$
$$P \subset EXP$$
$$PSACE \subset EXPSPACE$$
$$P \subset PSPACE\ ?$$

- More

$$NSPACE(s(n)) = coSPACE(s(n))$$
$$PH \subseteq PP \subseteq P^{\#P} = P^{PP} \subseteq PSPACE$$

Known and Open

- Theorems

$$P = NP \Leftrightarrow P = PH$$

- [Ladner] If P!=NP then there exists NP-intermediate languages
- If there is NP-complete problem in $NP \cap coNP$ then NP=coNP=PH
- [Babai 2017] Graph isomorphism (GI) problem is in quasipolynmial

$$\exp\left(\log^{O(1)} n\right)$$

- [Håstad] If 3SAT can be approximated within $7/8 + \epsilon,\ \epsilon > 0$ then P=NP.
- [Schöning] There is randomized algorithm which solves 3SAT in $O((4/3)^n)$
- [Baker-Gill-Solovay] There exist A and B such that

$$P^A = NP^A$$
$$P^B \neq NP^B$$

Isomorphism Conjecture

- [Berman-Hartmanis]All NP-complete sets are polynomial isomorphic
  - There is polynomial mapping between any two NP-complete sets
  - This means there is essential just one NP-complete set
  - They showed all the then-known NP-complete sets are polynomial isomorphic to each other
- Isomorphism conjecture implies P!=NP
  - We have finite sets in P and NP-complete sets are infinite thus they cannot be isomorphic
- [Joseph-Young1985] Counter conjecture
  - The k-creative sets, NP-complete, for which no p-isomorphism to the standard NP-complete problems is known.
  - The Encrypted Complete Set Conjecture There is a one-way function and a standard complete set A such that A and the NP-complete set f(A) are not polynomial-time isomorphic
- For more restrictive reducibility we can prove isomorphism conjecture
- [Myhill's Theorem] There is essentially only one RE-complete set
  - Every RE-complete set has a 1-1 onto recursive, invertible map to every other RE-complete set.
- Isomorphic conjecture cannot be proven in realization way
  - There are A and B such that isomorphic conjecture is false relative to A but true relative to B.

Dense set and NP-complete set

Definition

Set A is dense if there exists an $\epsilon > 0$ such that for every n: $|A|_{\leq n} \geq 2^{n^\epsilon}$.
Set A is sparse if there exists a polynomial p such that for every n:
$|A|_{\leq n} \leq p(n)$.

- [Buhrman-Hitchcock] If PH is infinite then any NP-complete set is dense.
- All known NP-complete sets are dense
- [Mahaney] If any sparse language is NP-complete, then P=NP
- "Roughly speaking, NP-complete sets is hard because there are two many (exponential) elements in it to check"

One-way functions

- Easy to compute but hard to inverse

  $A\ function\ f:\{0,1\}^* \rightarrow \{0,1\}^*\ is\ polynomial,\ but\ for\ any\ polynomial\ randomized\ F$ $and\ any\ integer\ n>0,\ we\ have$

  $$Pr[f(F(x))=f(x)]<|x|^{-n}$$

- The existence of one-way function implies
  - P≠NP
  - No natural prove for P≠NP
  - Pseudorandom generators exist
  - If P = BPP then there is no one-way function
- Candidates for one-way functions
  - [Levin constructed]Universal one-way function which is one-way iff any one-way function exists.
  - Multiplication and factoring
  - Discrete exponential and logarithm
- Open problem Does P≠NP imply that one-way functions exist?

Pseudorandomness

- Computational Indistinguishable

$For\ ensemble\ of\ distribution\ \{X_n\}\ and\ \{Y_n\}\ over\ \{0,1\}^{l(n)}, l(n)\ is\ polynomial,\ there\ is$
$no\ efficient\ \text{algorithm}\ A\ such\ that\ for\ sufficient\ large\ n\ and\ any\ positive\ polynomial\ p(\cdot),\ we\ have$

$$|Pr[A(X_n)=1]-Pr[A(Y_n)=1]|>1/p(n)$$

- [Yao] A distribution is pseudorandom if it is computational indistinguishable from the uniform distribution.

- Pseudorandom generators
  - One-way function is hard to inverse which similar to that it is hard to guess output of random function
  - Convert hardness into pseudorandomness
  - Generate pseudorandom distribution based on short or no random bits

Derandomization

- Derandomization
  - Enumerate all values of the random bits in pseudorandom generators
  - Change probabilistic algorithms to deterministic algorithms
  - People found deterministic algorithms by derandomization
    - Primality test in this approach
    - [P.Laire] Smale 17th Problem Solving polynomial equations

> "Can a zero of n complex polynomial equations in n unknowns be found approximately, on the average, in polynomial time with a uniform algorithm?"
>
> — *S. Smale, 1998*

Probabilistic Turing Machines

- Transition function

$$\delta: (Q/F) \times \Gamma \times Q \times \Gamma \times \{L, R\} \rightarrow [0, 1]$$

$$p(accept\ w) = \sum_{c \in \{accept\ paths\}} p_c$$

- Can be viewed as flipping a coin at each step to decide next move

$$BPP = \left\{ \begin{array}{c} L \mid \forall x \in L, Prob[M(x) = 1] \geq 2/3 \\ \forall x \notin L, Prob[M(x) = 0] \geq 2/3 \end{array} \right\}$$

$$RP = \left\{ \begin{array}{c} L \mid \forall x \in L,\ Prob[M(x) = 1] \geq 2/3 \\ \forall x \notin L, Prob[M(x) = 0] = 1 \end{array} \right\}$$

$$ZPP = \left\{ \begin{array}{c} L \mid \forall x \in L, Prob[M(x) = 1] = 1 \\ \forall x \in L, Prob[M(x) = 0] = 1 \end{array} \right\}$$

$$RP \subseteq BPP, \quad RP \subseteq NP$$

$$ZPP = RP \cap coRP$$

- BPP is being viewed as efficient class
- Open problem P = BPP ?

Circuit complexity

An n-input, single-output boolean circuit is a dag with

- n sources (vertex with fan-in 0), and
- one sink (vertex with fan-out 0).

All non-source vertices are called gates and are labeled with

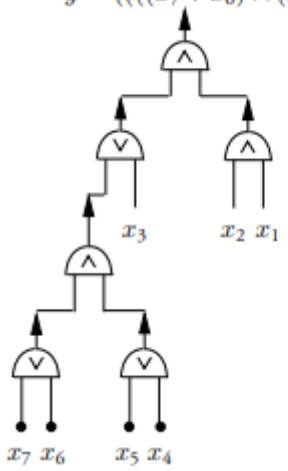
- $\vee, \wedge$ (vertex with fan-in 2), and
- $\neg$ (vertex with fan-in 1).

A Boolean circuit is monotone if it contains no $\neg$-gate.

A problem L is in **SIZE**$(S(n))$ if there exists an $S(n)$-size circuit family $\{C_n\}_{n \in \mathbf{N}}$ such that for each $x \in \{0,1\}^n$, $x \in L$ iff $C_n(x) = 1$.

Unlike in a uniform model, **SIZE**$(cS(n)) \neq$ **SIZE**$(S(n))$ generally. This follows from Circuit Hierarchy Theorem.

$y = ((((x_7 \vee x_6) \wedge (x_5 \vee x_4)) \vee x_3) \wedge (x_2 \wedge x_1))$



Circuit complexity

- Nonuniform : allows a different algorithm to be used for each input size.
- [Shannon] Most n-ary Boolean functions are hard of size

$$\frac{2^n}{n} - o\left(\frac{2^n}{n}\right)$$

- Arithmetic Circuits
  - gate functions are MOD functions
- Polynomial methods
  - represent the circuit (approximately or exactly) as a “nice” polynomial
- Meyer-Stockmeyer
  - There is a natural logic problem Π such that every Boolean circuit over B2 that decides every sentence over Π on all 360-bit instances requires size at least 10^{125}
- Set of language accepted by uniform circuit equals P
- Circuit complexity lower bounds imply universal derandomization

Circuit complexity

- P/poly: P-time TM using P-size advice

$$P_{/poly} = \cup_c SIZE(cn^c)$$

$$NP \nsubseteq P_{/poly} \Rightarrow P \neq NP$$

$$NP \subseteq P_{/poly} \Rightarrow PH = \Sigma_2^P$$

A language L is in NC^d if L can be decided by a uniform circuit family of $poly(n)$ size and $O(\log^d n)$ depth

$$NC = \cup_{d \geq 1} NC^d$$

AC^d extends NC^d by allowing unbounded fanins

$$AC = \cup_{d \geq 1} AC^d$$

$$AC = NC$$

- NC = Problems with efficient parallel algorithms
- P=NC iff there is P-complete language L in NC
- P-complete: use log-space reduction
- Monotone circuit evaluation is P-Complete
- Inside P

$$NC^1 \subseteq L \subseteq NL \subseteq NC^2 \subseteq \cdots \subseteq NC^j \subseteq \cdots \subseteq P$$

Interactive proof system

- Goldwasser, Micali, and Rackoff 1985
- Prover P with (unlimited) power
- Verifier V with limit (but efficient) power and using randomness (private)

$$A\ k(n)\ round\ interaction\ of\ P\ and\ V\ on\ input\ x\ and\ random\ r$$

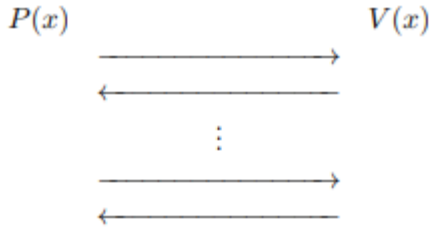
$$a_1 = V(x,r)$$

$$a_2 = P(x,a_1)$$

$$a_3 = V(x,a_1,a_2)$$

$$\vdots$$

$$a_{k(n)} = V(x,a_1,\cdots,a_{k(n)-1})$$



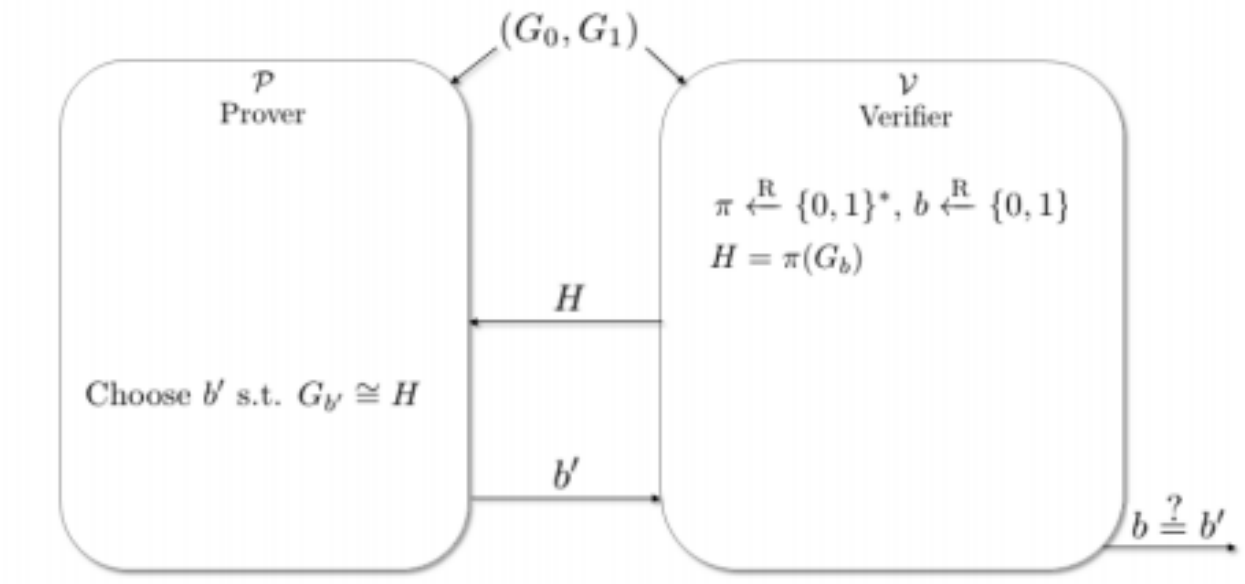
$$Completeness: \forall x \in L \exists P,\ Pr[\langle P,V\rangle(x) = 1] \geq 2/3$$

$$Soundness: \forall x \notin L \forall P,\ \ Pr[\langle P,V\rangle(x) = 1] \leq 1/3$$

$$IP = \cup_{c \geq 1} IP[n^c]$$

Interactive proof system

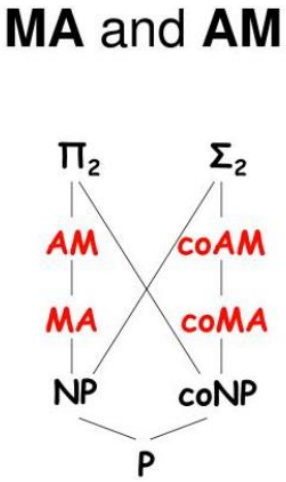
- GNI (graphic non-isomorphic) in IP



Interactive proof system

- Public coins
  - AM (Arthur-Merlin) and MA V sends all its random strings (polynomial) to V

$$IP[k] \subseteq AM[k+2], \quad k(n) \text{ is polynomial}$$
$$AM[n] = AM[2], \text{ constant } n \geq 2$$
$$AM := AM[2]$$



- Interaction + randomization gives more power
  - IP = PSPACE
  - MIP = NEXP Adding entanglement we have MIP*=RE
  - coNP in IP

- Public coins are at least as powerful as private coins

  IP = AM
- If Graphic Isomorphic (GI) is NP-complete then PH collapses to

  $\Sigma_2 = \Pi_2$
- Open problem

  $GNI \subseteq NP$?

Zero-Knowledge Proof

- P convinces V that x is in L but no more other information V gained
- View
  - all information V sees during the interactive with P
- Perfect zero-knowledge proof
  - view can be efficiently generated with same distribution
- Computational zero-knowledge (CZK)
  - view can be efficiently generated with distributions are computational indistinguishable.
- Statistical zero knowledge
- If one-way function exists then
  - CZK=IP=PSACE
  - Particularly every L in NP has CZK proof.

Zero-Knowledge Proof

Example of A Zero Knowledge Interactive Proof

1. Given G_1, G_2.
2. Do n rounds of the following:
   (a) The prover chooses a random permutation σ and computes $H = \sigma(G_2)$. Then he sends H to the verifier.
   (b) The verifier chooses a random $i \in \{1, 2\}$ and sends it to the prover.
   (c) The prover computes a permutation ρ such that $H = \rho(G_i)$:
   - If $i = 1$, then $\rho = \pi \circ \sigma$,
   - If $i = 2$, then $\rho = \sigma$.

   Then the prover sends ρ to the verifier.
   (d) The verifier checks that $H = \rho(G_i)$.
3. The verifier accepts the input if in all the rounds $H = \rho(G_i)$.

Secure Multi-Party Computation

- Secure multi-party computation (secure computation, multi-party computation (MPC), or privacy-preserving computation)

- [Yao 82]Millionaires' Problem
  - Two millionaires determine who is richer without knowing the other's wealth

- MPC (or SMC)
  - N parties each with private data want to compute a public function on all the private data without revealing their private data.

- Applications
  - electronic voting
  - electronic auctions

Probabilistically checkable proofs (PCP)

PCP(r(n),q(n)) Complexity class

1. Polynomial verifier V accessing r(n) random bits and q(n) bits of *proof* $P(x)$

2. Completeness

   $x \in L$, *then there is a proof* $P(x)$ *such that*

   $$Prob[V(x,r,P(x)) = 1] = 1$$

3. *Soundness*

   $x \notin L$, *then for all proofs* P *with error probablity* α

   $$Prob[V(x,r,P(x)) = 1] = \alpha < 1/2$$

$$P = PCP(0,0)$$

$$NP = PCP(0,poly) \qquad coNP = PCP(poly,0)$$

PCP Theorem $NP = PCP(\log n, O(1))$

We can verify any proof by just checking randomly 9 bits of the proof

Trade off between error probablity and bits of proof

0.76 $3bits$

0.32 $9bits$

PCP Theorem

- [1990s]Proofs could be transformed into probabilistically checkable ones with only a modest increase in their length($n \to n \log^{O(1)} n$ Dinur)
- Proof is highly structured (algebraic, graphic and more ?)

- PCPs and approximations
  - 3SAT→MAX-3SAT based on PCP

$$polynomial\ mapping\ \phi \in 3SAT\ to\ \psi \in 3SAT\ such\ that$$
$$if\ OPT(\phi) = 1,\ then\ OPT(\psi) = 1$$
$$if\ OPT(\phi) < 1,\ then\ OPT(\psi) \le \alpha < 1,\ \alpha\ is\ universal\ constant$$

  - MAX-3SAT is APX-complete, APX: polynomial approximated to any constant
  - Format the problem in the way that there is a gap between YES and NO instances

The Unique Games Conjecture

Gap version of approximation problem $Gap_{c,s}$ *for* $0 < s < c$ *such that*

$$OPTI(I) \geq c$$

$$OPT(I) \leq s$$

Unique Game $\boldsymbol{U}(\mathrm{G(V,E)},[\mathrm{n}],\{\pi_e|\mathrm{e}\in\mathrm{E}\})$

1. $\mathrm{G(V,E)}$ is a directe graph with vertices as variables and edges as constraints.
2. $\pi_e : [n] \mapsto [n]$ *bijection*
3. *A labeling* $L : V \mapsto [n]$ *statisfies the contraints on dedge* $e = (v,w)$ *iff* $\pi_e(L(v)) = L(w)$

$$OPT(U) := \max_{L:V\mapsto[n]} \frac{1}{|E|}|\{e \in E | L \text{ statisfies } e\}|$$

unique means the labeling of one vertice of an edge decides the labeling of the other vertice.

Unique game problem find an (approximately) optimal labeling.

Unique Game Conjecture [*Khot*2002] *for every* $\epsilon, \delta < 0$, *there exists a contatnt* $n(\epsilon,\delta)$, *such that a given* $U(G(V,E),[n],\{\pi_e|e\in E\})$, *it is* NP-Hard *to distinguish between the following two cases*:

$$OPT(U) \geq 1 - \epsilon$$

$$OPT(U) \leq \delta$$

The Unique Games Conjecture

- UGC implied a lot of inapproximabilities
- the Max-Cut and Vertex Cover problems are optimal

Problem	Best Approx. Known	Inapprox. Known Under UGC	Best Inapprox. Known
Vertex Cover (VC)	2	$2-\varepsilon$	1.36
VC on k-uniform Hypergraphs, $k \geq 3$	k	$k-\varepsilon$	$k-1-\varepsilon$
MaxCut	α_{MC}	$\alpha_{\mathsf{MC}}-\varepsilon$	$\frac{17}{16}-\varepsilon$
Max-2SAT*	α_{LLZ}	$\alpha_{LLZ}-\varepsilon$	APX-hard*
Any CSP $\mathcal{C}$ with integrality gap $\alpha_{\mathcal{C}}$	$\alpha_{\mathcal{C}}$	$\alpha_{\mathcal{C}}-\varepsilon$	
Max-kCSP	$O(2^k/k)$	$\Omega(2^k/k)$	$2^{k-O(\sqrt{k})}$

Average-case complexity

- [Levin86]Formalized the setting

$Expetation\ definition\ E[t_A(x)]\ is\ not\ invariant\ when\ change\ machine\ model\ (P \to EXP)$

$Average\ polynomial\ runnting\ time{:}run\ p(n)\ and\ stop.\ Small\ probability\ do\ not\ know\ the\ answer$

$$AvgP := \left\{ \langle L, \mu \rangle : \Pr_{x \sim \mu} [t_A(x) \geq t] \leq \frac{p(|x|)}{t^c}, c > 0 \right\}$$

$$distNP := \{ \langle L, \mu \rangle : L \in NP, \mu\ is\ polynomial\ time\ computable \}$$

- Same hardness between average-case and worst-case
  - The shortest vector problem in a lattice
  - Computing the permanent of a matrix
  - Neither of these problems is believed to be NPC

- Generate hard instances by average-case complexity analysis
- Open problem
  - AvgP vs distNP is the analogues of P vs NP
  - Does worst-case hardness on NP imply distNP not in AvgP?
  - Is there an NPC problem with same worst-case and average-case hardness?

Dealing with NP-hard problems

- Brute Force
  - Traveling salespeople problems with more than ten thousand cities
  - SAT problems of about 100 variables.
- Parameterized Complexity
- Approximation
- Heuristics and Average-Case Complexity
  - In competition SAT solvers can often settle for a million variables

Dealing with NPC problems

- Exact algorithms for NP-hard problems
  - There is a wide variation in the worst case complexities of known exact algorithms
  - Classical complexity theory cannot explain the differences
  - Exponential time hypothesis (ETH)
    - k-SAT does not have sub-exponential algorithms for k > 2
  - Techniques
    - Dynamic programming across the subsets

TSP	O(n!) → O(2^n)
Graph Coloring	O(2.4422^n)

    - Pruning the search tree

3SAT	O(1.4963^n)
Independent Set	O(1.1844^n)
Bandwidth problem	O(n!) → O(10^n)

    - Preprocessing the data

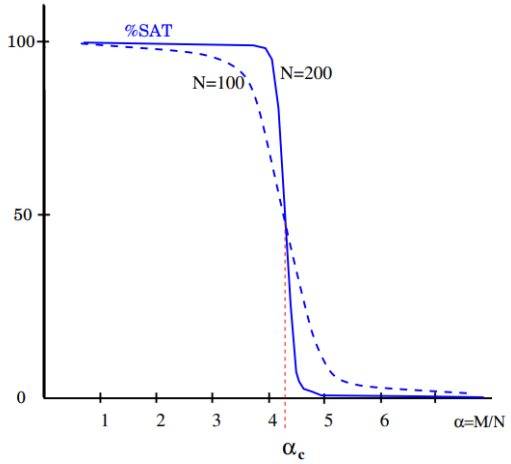
subset sum problem	O(2^n) → O(2^(n/2))

    - Local Search

3SAT	O(1.3302^n)

Phase transition

- Phase transition in practice



generically SAT for $\alpha < \alpha_c$

generically UNSAT for $\alpha > \alpha_c$

- A threshold theorem

$[Friedgut] For\ each\ k \geq 2,\ there\ is\ \alpha_c(n)\ such\ that\ for\ any\ \epsilon > 0$

$$\lim_{n\to\infty} Pr[F_k(n, m = \alpha n) = 1] = \begin{cases} 1 & if\ \alpha < (1-\epsilon)\alpha_c(n) \\ 0 & if\ \alpha > (1+\epsilon)\alpha_c(n) \end{cases}$$

$where\ F_k(n,m)\ is\ random\ kSAT\ with\ n\ variables\ and\ m\ clauses$

- [Ding-Sly-Sun2016]For random k-SAT and large k we have the threshold

$$r_k = 2^k \ln 2 - (1+\ln 2)/2 + o(1)$$

$when\ k \to \infty\ the\ error\ term\ o(1) \to 0$

P vs NP

- Three definitions of NP
  1. Nondeterministic Turing machine
  2. Proof system: given an input and a certificate efficiently verify it
  3. PCP theorem NP = PCP(logn, O(1))
- NP-completeness
  - University: found in almost all human intelligence fields.
  - Uniqueness: isomorphic conjecture
- P vs NP intuition meaning
  - Is it the same hard to find a proof as to checking the proof?
  - Is there really creativity? Creativity corresponds to oracles

P vs NP

- Why polynomial
  - Polynomials are the smallest class of functions that
    - contains the linear functions
    - closed under basic operations like addition, multiplication, and composition
  - Independency of the low-level details of the machine model corresponds to composite
  - Relativized Complexity Theory (RCT)
    - Based on Cobham's axiomatization of polynomial-time computation
    - It has standard axioms of arithmetic (Peano axioms), and an axiomatic definition of the class of functions that is supposed to correspond to the class P.
    - Functions definable by Cobham axioms without minimality
    - The (standard) models of RCT are exactly the classes P over all oracles O
    - A non-relativizing statement is precisely a statement independent of RCT. The "P vs NP" is independent of RCT
  - Axiomatic Approach to Algebrization
    - algebraic complexity theory
  - Any other reasons?

P vs NP

What we have tried without success

1. Relativization
   - [Baker-Gill-Solovay] no relativizable proof can solve P vs NP problems in either direction

     $$\exists A, B, \ such\ that\ P^A = NP^A \ and\ P^B = NP^B$$

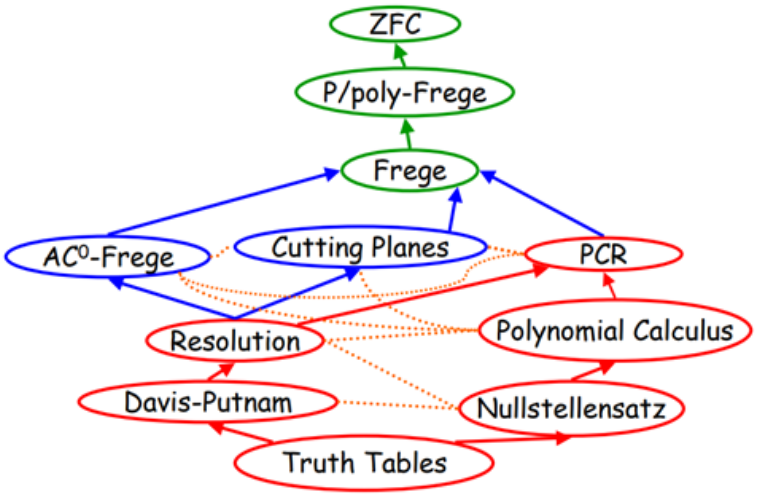
   - Relativization barrier
2. Circuit Complexity
   - If a NPC problem cannot be solved by relatively small circuits(AND,OR,NOT) then P!=NP
   - [Razborov1985] No small circuits for AND,OR (no NOT)
   - [Razborov-Rudich]Circuit complexity techniques follow into natural proof
   - Natural proofs barrier cannot solve P vs NP
3. Algebrization IP = PSPACE
   - Arithmetization replace AND,OR, or NOT by arithmetic operation over larger field F then Boolean ones. Boolean formula becomes a low-degree polynomial which has error-correcting properties
   - Algebrization Extend boolean oracle A into algebraic oracle $\tilde{A}$
   - Circumvent relativization and natural proof barriers
   - There exist oracle A, B such that $NP^A \subseteq P^{\tilde{A}},\ NP^B \nsubseteq P^B$
   - Algebrizaiton barrier

P vs NP

What we have tried without success

4. Proof Complexity[Cook-Reckhow]
   - Proof system $x \in L \Leftrightarrow P(x,\pi) = 1, P \text{ and proof } \pi \text{ is polynomial length in } |x|$
   - Tautology is dual to Satisfiability
   - Subsystems of Peano arithmetic
     - bounded arithmetic
     - to capture feasible/polynomial-time reasoning
   - Low-complexity proofs system generalize better and are often more constructive
   - Relate to P vs NP
     - [Cook-Reckhow]NP=coNP iff there exists a polynomial bounded propositional proof system.
     - No short proofs of tautology in an arbitrary proof system that would imply P != NP.
   - Proof systems
     - Frege proofs
     - resolution proofs

       no short proofs for pigeonhole principle
     - polynomial calculus
     - cutting planes
     - algebraic proof systems



P vs NP

Descriptive complexity

- Finite model theory: TM interpretation on finite structures
- Descriptive complexity: logic approach to computation
  - The type of logic need to express the languages in the complexity class
  - No explicit mention resource bound or computing model
  - How hard is it to **check** if input has property S?
  - How rich a language do we need to **express** property S?
  - Each logical system produces a set of queries expressible in it. The queries(when restricted to finite structures) correspond to the computational problems of traditional complexity theory

Definitions of Complexity Classes	
P	First-order logic with a least fixed point and linear order
NP	Existential second-order logic
coNP	Universal second-order logic (without existential second-order quantification)
PSAPCE	Second-order logic with a transitive closure
EXPTIME	Second-order logic with a least fixed point operator
PH	Languages expressed by second-order logic

P vs NP

Independent of set theory approaches

- First-generation
  - [Ben-Halevi] If P ≠ NP is independent of PA then SAT has algorithms very close to polynomial
  - [Hartmains-Hopcroft]There is a Turing machine M such that relative to oracle L(M), neither P=NP nor P!=NP is provable in ZF assuming ZF is consistent.
  - [DeMillo-Lipton]P!=NP is unprovable in ET (a weak logic theory)

- Second-generation
  - Based on bounded arithmetic theories
  - $[\mathrm{R}azborov] If\ there\ is\ a\ pseudrandom\ generator\ requiring\ circuits\ of\ size\ \Omega(2^{n^{\epsilon}}),$ $then\ NP \nsubseteq P_{/poly}\ is\ unprovable\ in\ theory\ S_2^2.$

P vs NP

Possible approaches ?

1. [Mulnuley-Sohoni]Algebraic geometry approach

> In essence, they defines a family of high-dimension polygons P_n based on group representations on certain algebraic varieties. Roughly speaking, for each n, if P_n contains an integral point, then any circuit family for the Hamiltonian path problem must have size at least $n^{\log n}$ on inputs of size n, which implies $\mathbf{P} \neq \mathbf{NP}$. Thus, to show that $\mathbf{P} \neq \mathbf{NP}$ it suffices to show that P_n contains an integral point for all n.

2. Ramsey Theory
   - Arrowing F→(G,H) is in $coNP^{NP}$ [Any two coloring F the is one G or the other color H]
   - Provide natural example of a problem complete for a higher level of the polynomial hierarchy
   - Ramsey numbers are guaranteed existence but hard to find
   - We know little about computational complexity of Ramsey numbers

3. Any relation to fast-growing hierarchy?

4. Reverse Mathematics
   - Understand P vs NP from logic and axiomiazation point of view

P vs NP

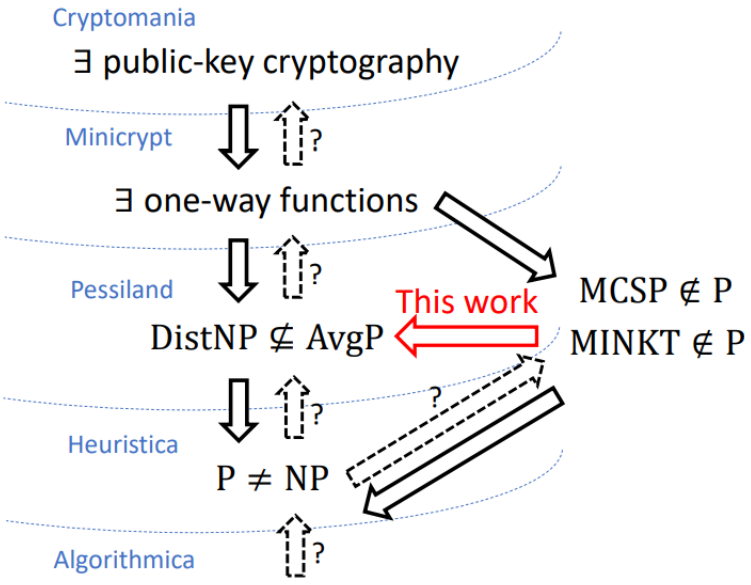
- What if P=NP (or NP in BPP)
  - [Gödel,1956]Formal proof of any theorem is easy and can be automated.
  - "We can solve all the Clay million-dollar problems at the same time".
  - Public-key cryptography becomes impossible.
  - Learning is easy.
  - Human creativities become trivial
    - design, composing music/literals, creating physical theories, etc.

- [Knuth]We can prove P=NP but cannot construct the efficient algorithms
  - There is non-constructive proof that some algorithms exist
    - *Robertson-Seymour graph minors theory*

- People believe P != NP with strong evidences

P vs NP

[1995]The five worlds of Russell Impagliazzo	
Algorithmica	N=NP with known algorithms
Heuritica	P!=NP but NP is easy on average
Pessiland	There are problems hard on average but no one-way functions. This means we cannot generate instances with known answers for NP-complete problems
Minicrypt	one-way functions exist but public-key cryptograph is impossible
Cryptomania	public-key cryptograph is possible. We are here, world 5?



More Topics

- Hartmanis-Stearns Conjecture (HSC)
  - Every real number computed by a real-time multi-tape Turing machine is either rational or transcendental.

$$HSC \Rightarrow \begin{cases} Multiplcation\ is\ super\ liear \\ TIME(n) \neq NTIME(n) \end{cases}$$

- Real-time Turing machines
- Communication complexity and Parallel complexity
- Busy Beaver Problem and Kolmogorov complexity
- [Smale] Complexity and real computation
- [Spielman-Teng]Smoothed analysis
  - hybrid of worst-case and average-case analysis

Milestones in Classic World

- Turing machines and its university
- NP-complete problem and its university
- Interactive proof system and PCP theorem

Quantum World

- Physics background
- Quantum circuit model and Quantum Turing machines
- Quantum Algorithms and Quantum Complexity Classes
- Physicists have more
- More Topics
- Personal views
- Skeptics of quantum computing

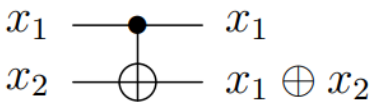
Physics background

- Physicists have “physical” understand about computing
  - Maxwell's Daemon[Landauer 1961][Penrose 1970][Bennett 1982]
    - Relation between irreversibility in computation and thermodynamics
  - [Bennett 1973][Toffoli 1980]
    - Universal reversible computation and equivalence to general computation
    - Reversible computation without dissipating any energy
  - 1st conference on Physics and Computation, MIT, 1981
    - [Feynman]”… if you want to make a simulation of Nature, you’d better make it quantum mechanical, … it doesn’t look so easy”
  - [Poplvskii1975]
    - Quantum systems cannot be simulated feasibly on classical computers due to superposition
  - [Deutsch]Thinking testing Many-Worlds Interpretation superposition power

Quantum circuit model

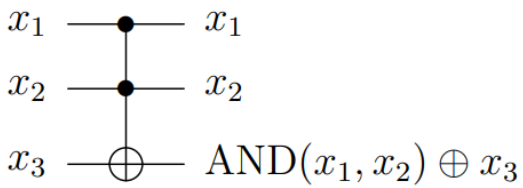
- Mimic (logically) classic circuit model (AND, OR, NOT)

- CNOT (controlled-NOT) gate



input	output
$\|00\rangle$	$\|00\rangle$
$\|01\rangle$	$\|01\rangle$
$\|10\rangle$	$\|11\rangle$
$\|11\rangle$	$\|10\rangle$

Toffoli gate (CCNOT)



input	output
$\|000\rangle$	$\|000\rangle$
$\|001\rangle$	$\|001\rangle$
$\|010\rangle$	$\|010\rangle$
$\|011\rangle$	$\|011\rangle$
$\|100\rangle$	$\|100\rangle$
$\|101\rangle$	$\|101\rangle$
$\|110\rangle$	$\|111\rangle$
$\|111\rangle$	$\|110\rangle$

- Hadamard gate create equal superposition of all basis states

$$H = H_1 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$H_m = H_1 \otimes H_{m-1} = \frac{1}{\sqrt{2}} \begin{pmatrix} H_{m-1} & H_{m-1} \\ H_{m-1} & -H_{m-1} \end{pmatrix} = H^{\otimes m}$$

$$(H_m)_{i,j} = \frac{1}{2^{\frac{n}{2}}} (-1)^{i \cdot j}$$

Quantum circuit model

- Preparation Prepare input state

$$|i\rangle,\ usually\ |0^n\rangle$$

- Evolution Gates are unitary thus have matrix representation

$$U = U_T U_{T-1} \cdots U_2 U_1$$

- Measurement

$$Pr[|k\rangle] = |\langle k|U|i\rangle|^2$$

- Accuracy of Quantum Gates

$$E(U,V) = \max_{|\varphi\rangle} \| (U-V)|\varphi\rangle \|$$

- [Kitaev-Solovay]Universal gate sets are efficiently simulate each other

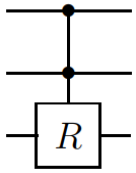
A set of unitary operators U_i *generate a dense subste of* $SU(d)$. *Then any matrix* $U \in SU(d)$ *can be approximated to within* ϵ *by* $O\left(\log^c\left(\frac{1}{\epsilon}\right)\right)$ *elements of* U_i *and* U_i^{-1}, c *is* constant

- Thus we can define quantum complexity classes since universal gate sets are equivalent

Quantum circuit model

- Universal quantum gates
  - Any unitary gate on n-qubits can be decomposed as a product of 1- and 2-qubit gates
  - Decomposed blindly requires 4^n gates
  - Any generic 2-quit gate is universal
  - Generic means for all but a set of measure zero of 4 x 4 unitary matrices

- Deutsch gate controlled-controlled-R



$$R = -iR_x(\theta),\ rotation\ about\ x\ axis\ and\ \theta\ is\ not\ rational\ multiple\ of\ \pi$$

- Toffoli + Hadamard is universal reversible gate
- XOR and arbitrary 1-qubit gate form a universal gate set

Quantum Turing machines

- [Bernstein-Vazirani 1997]Mimic classic Turing machine definition

$$\delta: (Q/F) \times \Gamma \to \tilde{C}^{Q \times \Gamma \times \{L,R\}}$$

where $\tilde{C}$ *is complex number which can be computed to* exponential *precision in polynomial time*

- [Lance] Reformed in matrix multiplication

Configuration	*consists contents of taps, position of header and current state*
Transition matrix	$T(c,s)$
Deterministic	*entries of* T *are* 0 *or* 1
Nondeterministic	*entris of* T *are in* $[0,1]$ *and* $\sum_j T_{i,j} = 1$ *for all row* i
Quantum	*entries of* T *are comlex number and* $\sum_j \lvert T\rvert^2_{i,j} = 1$, *for all row* i
	T *is unitary*
Prob$[accept\ x]$	$T^n(initial, final)$, n *is the running time*

Quantum Turing machines

- If we allow arbitrary real/complex number we can do anything by encoding the answer to the number such as Chaitin constant

- We should only allow efficiently computable numbers, i.e., compute i-th bit in polynomial time of i

$$[Solovay, Yao\ etc] \qquad \left\{0, \pm 1, \pm \frac{3}{5}, \pm \frac{4}{5}\right\}$$

$$[Kitaev\ 1997] \qquad \left\{0, \pm 1, \pm \frac{1}{\sqrt{2}}\right\}$$

- [Yao1993]Uniform quantum circuits are polynomial equivalent to quantum Turing machines

$$O(T^2 \log^{O(1)} \epsilon)$$

Quantum Algorithms

- Physicists show power of superposition
  - Basic idea
    - Use Hadamard gate to get superposition of all basic states
    - Implement the gate corresponding to the function we want compute
    - Then we have S = {f(x)| for all x}
    - The challenge is to find a transform that transfers S into {f(x)|x is what we are interested in}
  - Deutsch–Jozsa problem (in BPP) find f is constant or balanced in O(1)

$$f: \{0,1\}^n \to \{0,1\},\ f\ is\ either\ \text{constant}\ or\ balanced\ (half\ 0\ and\ half\ 1)$$

- Computer scientists found "harder" problems
  - Bernstein–Vazirani problem and Simon problem
- Peter Shor's breakthrough $O(N^2 \log N \log\log N)\ vs\ O(e^{cN^{1/3}(\log N)^{2/3}})$
- Grove's surprise find unstructured N items in $O(\sqrt{N})$
- "Artificial" Quantum random walk
- [Tang 2019] Gave a classical algorithm which is only polynomial slower than its quantum version(Kerenidis and Prakash's algorithm in QML)
- http://quantumalgorithmzoo.org/

Quantum Complexity Classes

Classic	relationship	Quantum
BPP	⊆	BQP
NP	⊆	QMA
IP	=	QIP
RE	=	MIP*

Problem	Lower Bound	Speedup
OR	Θ(√N)	√
AND	Θ(√N)	√
PARITY	N/2	2
MAJORITY	Θ(N)	1

QMA-complete Problems
1. The k-local Hamiltonian problems, k>1
2. Density matrix consistency problem
3. Quantum clique problem

Results and Open Problems

$$P \subseteq BPP \subseteq BQP \subseteq MQA \subseteq PP \subseteq PSPACE$$

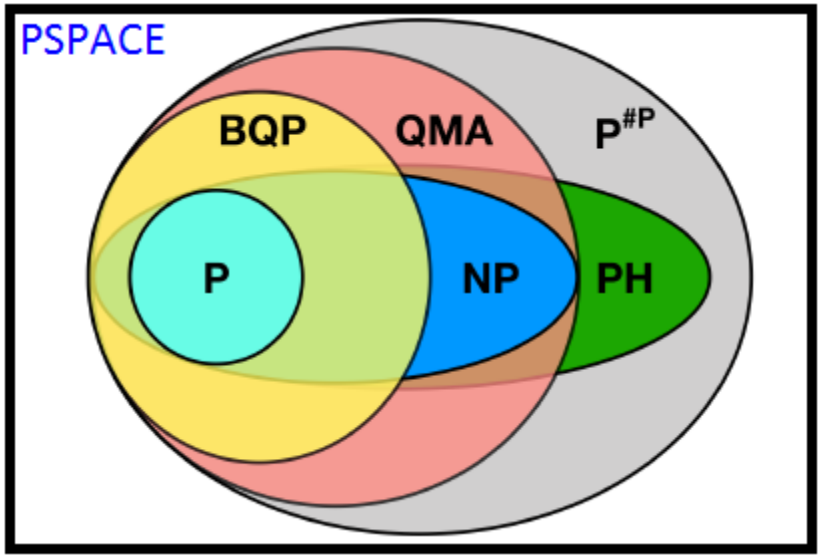
$$QSPACE(f(n)) \subseteq SPACE(f^2(n))$$

$$P \; vs \; NP \; \rightarrow \; NP \; vs \; BQP$$

$$NP \; vs \; BQP, \; \exists A[BQP^A \nsubseteq PH^A], \; a.e. A[NP^A \nsubseteq BQP^A]$$

- It is hard to prove quantum computer is more powerful (will separate P and PSPACE)
- BQP could find a solution which is hard to verify (Group non-membership is in QMA that is not known to be in NP)
- Lower bound results give us hint to find exponential speedup quantum algorithms
- Without entanglement (product state thus 2n parameters) we cannot have super polynomial speedup
- We could not be able to quantify entanglement (see MIP*=RE)

Quantum Complexity Classes



Physicists have more

- Adiabatic algorithms

  1. *Intial* H_0 *its Hamiltonian is easy and in its ground state*
  2. *Final* H_1 *its ground state corresponds to the solution*
  3. *Evovle slow enough*

  $$H_s = (1-s)H_0 + sH_1,\ O\left(\frac{1}{g_{\min}}\right),\ g_{\min}\ \textit{is the}\ \text{minimum sprectral gap of}\ H(s)$$

- Hypercomputation — Solve problems which are not Turing computable
  - Oracle-Machines
  - Infinite state Turing machines
- Add nonlinear into QM
  - Assuming arbitrary nonlinear gates and no error, PSPACE exactly characterizes the power of nonlinear quantum mechanics
  - QM is linear at least to 17 decimal precision
- Closed time-like curves — Time travel
- Anthropic — Parallel universities
  - Kill yourself if in one try you have find the answer is NO
- Soap bubbles procedure
- Protein folding

More Topics

- Topological quantum computing
- Quantum version PCP
- Quantum machine learning
- Quantum error correction code
- Quantum information theory

Skeptics of quantum computing

- Impossible to build large scale quantum computer
  - Levin
    - Quantum computing need to manipulate $2^n (n \sim 100 - 1000)$ numbers
    - Too many numbers, more than the number of atoms in the universe
    - Some numbers are forced to exponential small due to probability sum to 1
    - No physics theory until now can be valid to more than 100 decimal precision
  - Preskill-Kalai
    - Noisy intermediate-scale quantum (NISQ) computers (at most 500 qubits)
    - Model noise: noise levels cannot be sufficiently reduced

- Quantum computer is not significantly more powerful than classic one
  - Most likely NP-complete problems cannot be solved efficiently by quantum computers
  - [Gasarch P vs NP poll 3]
    - Oded Goldreich
    - Eric Allender
    - Peter Shor
    - Doron Zeilberger

One personal guess

$$P \neq PSPACE$$

$$Due\ to\ NP\ vs\ BQP\ is\ undecidable$$

- Quantum to classic
  - No(?) result in classic computational complexity theory coming from quantum computing
  - No(?)complexity class characterized by entanglement (finite quantity)
- Main classes are between P and PSPACE

Sandwitch

$$P \subseteq BPP \subseteq BQP \subseteq MQA \subseteq PH \subseteq PP \subseteq P^{\#P} = P^{PP} \subseteq PSPACE$$

$$P \subseteq \quad NP \quad \subseteq PH \subseteq PP \subseteq P^{\#P} = P^{PP} \subseteq PSPACE$$

- NP vs BQP could be undecidable (any such pair of intermediate classes will work)
  - Entanglement is necessary to achieve quantum super polynomial speedup
  - From MIP*= RE it could be the case that entanglement cannot be quantified
  - If P=SPACE then the pair (NP and BQP) are equal thus decidable
- Could be wrong
  - Result in MIP*=RE may not be related to decidability of NP vs BQP
  - If it works what is the reason that P != PSPACE?

One more personal …

- Understanding number helps us understand Nature

- There are only 4 types numbers
  - We can do adding, subtracting, multiplying and dividing
  - Reals, Complexes, Quaternions, and Octonions
  - Four division algebras over the reals

- Can we get new computational complexity classes by using different type of numbers (entries)?

	Reals		**Complexes**	**Quaternions**	**Octonions**
Physics	Classic		QM	Relativity Theory	Standard Model
Entry in transform matrix	{0,1}	[0,1]	**Complexes**	**?**	**?**
Computing	DTM	NTM	QTM	**?**	**?**

Here we are

- We believe Turing model has captured what computing is
- NP-complete problems are everywhere
- Are NP-complete set essentially the same? May be not
- Interactive + randomness is powerful (PCP theorem)
- Randomness itself may not help P=BPP?
- We need new methods to resolve P vs NP

Here we are

- Computational resources
  - Time
  - Space
  - Randomness
  - Oracle query
  - Interactive/communication round
  - Entanglement
    - can entanglement be used to define complexity classes?
    - May be not. In MIP*=RE it looks like we may not be able to quantify entanglement
  - What else can we use as an measurement to define complexity classes?
- P vs NP is deep into the way human reasoning
- If P = NP then we can efficiently automatic human reasoning

Here we are

- Undecidable domain and NP-complete domain
  - By counting there are uncountable languages but countable TMs
  - Thus almost every language is undecidable
  - We found undecidable problems in every where

- NP-complete domain
  - By counting most Boolean functions are hard $\frac{2^n}{n} - o\left(\frac{2^n}{n}\right)$
  - Until now we CANNOT find any Boolean function which is hard

Here we are

- Quantum physics may help (but may not too much)
- Topological quantum computing may help build robust QC
- There are not many QC algorithms
- It is hard to build quantum computer
- It is fruitful when physics meets computational complexity theory

Where will we go

- Can we find more structures of NP-complete classes?
  - Polynomial isomorphic
  - All known NP-complete sets are dense
  - There is big variety in exact or approximation algorithms for NP-complete problems
  - Will this help us find some new structures in NP-complete classes?
- Refer to PCP, can we find more algebraic aspect in P vs NP?
- Deep learning and NP-complete algorithms (exact and approximation)
- Find more derandomization algorithms

Where will we go

- Find more quantum algorithms better than classic ones
  - NP intermediate problems
  - Find quantum polynomial algorithms for GNI
- Can QC help resolve main classic complexity problems?
- Computational complexity to physics
  - As a physical principle that NP-complete problems are intractable will it cause really meaning or impact to physical theories?
- What physical principles can be used in computational complexity theory?
  - Increase in entropy principle
  - Conservation of energy
  - Inverse of quantum physics
- Confusion coming from physical realization and mathematical truth

Human Intelligence

- In mathematics
  - *Elementary* Euclidean geometry (under Tarski's axioms) is consistent and complete
    - [Chou-Gao-Zhang 93]Generate readable proofs for geometry theorems automatically
      Butterfly theorem, Simson's theorem, Desargues' theorem and Feuerbach's theorem
  - Automated generate
    - Questions (geometry)
    - Short proof (projective geometry)
    - Conjecture-making
  - Mathematics AID systems
    - Maple
    - Mathematica
    - GAP
  - Representation of mathematical knowledge and store in large databases
    - Online Encyclopedia of Integer Sequences (16,000 queries every day)
  - Automated Theory Formation
    - The AutoGraphiX Program
    - The HR Program (group, graph and number theory)

Human Intelligence

- Deep learning
  - Alpha Go beats the best Go players
  - Deep learning is being used or explored in almost every industry field
    - Human face recognition
    - Autopilot deep learning
    - Human voice generator
  - The relationship between human intelligence and the domains deep learning can do well
- Denis Shiryaev on youtube (using AI)

Human Intelligence

- Finite vs infinity and discrete vs continuous
- Infinity causes troubles
  - Zeno's paradoxes
  - Calculation
    - "Made anything out of infinite"
    - [Riemann]We can sum to any number by rearranging a conditionally convergent series
    - [Cantor] Definition of infinity and set theory saved mathematicians
    - A set is infinite iff it has a proper subset which is one-one mapping to itself
    - There are more real numbers than natural numbers
    - There are same number of points in the whole plan as a line
  - Resolving the troubles helps us understand the world deeper
    - We can handle infinity to some degree (integral)
    - We have set theory of mathematics base which almost every mathematician is satisfied
  - "Paradox" -- Almost sure but can hardly find one
    - By counting almost every Boolean function is hard but we have not found anyone yet
    - By counting almost every language is undecidable but we found undecidable languages
    - The diagonalization language is not RE

Human Intelligence

- Infinity causes troubles
  - In Zeno's paradoxes and calculus the infinity is about the objects which we study
  - Infinity in our human reasoning
    - ✓ Infinity is inherently inside the definition of complexity classes (big O)
    - ✓ Infinity small is inside the definition of interactive proof systems (neglect probability)
    - ✓ Main results from structure complexity theory come from diagonalization

Are halting problem and P vs NP the troubles that infinity causes to the way we do reasoning?

Human Intelligence

- Is hour human intelligence emergence?
  - Emergentism
    - ✓ System properties which are 'irreducible'
    - ✓ Consciousness is an emergent property of our brains
  - By brute force we can find proofs for all decidable problems
  - Almost of all proofs found this way are not human readable
    - ✓ Plan geometry is complete and has automatic proof systems
    - ✓ Can we generate human readable proof for plan geometry
    - ✓ Four-Color theorem proof
    - ✓ Conway's Lost Cosmological Theorem (Zeilberger using Maple to prove it)

$$1, 11, 21, 1211, 111221, \cdots$$

$$JHC(a_1^{m_1} a_2^{m_2} \cdots a_r^{m_r}) = m_1 a_1 m_2 a_2 \cdots m_r a_r$$

$$\frac{len(C_{i+1})}{len(C_i)} \to \lambda = 1.003577\ldots$$

References

1. A Status Report on the P versus NP Question Eric Allender 2009
2. The Status of the P versus NP Problem, Lance Fortnow 2009
3. A. Wigderson. P, NP and mathematics - a computational complexity perspective. 2007
4. R. Impagliazzo. Computational complexity since 1980. 2005
5. The P versus NP Problem Stephen Cook 2000
6. The importance of the P versus NP question Stephen Cook 2003
7. C. H. Papadimitriou. NP-completeness: A retrospective. 1997
8. The history and status of the P versus NP question M.Siper 1992
9. https://www.win.tue.nl/~gwoegi/P-versus-NP.htm
10. NP-completeness columns
11. NP-complete Problems and Physical Reality Scott Aaronson
12. Is P Versus NP Formally Independent? Scott Aaronson
13. Exact algorithms for NP-hard problems
14. Quantum Computational Complexity John Watrous 2008
15. https://www.ias.edu/ideas/2011/kennedy-continuum-hypothesis
16. https://plato.stanford.edu/entries/computational-complexity/
17. https://cacm.acm.org/magazines/2019/1/233526-the-church-turing-thesis/fulltext

Thank You!

yaoyong99@gmail.com