Undergraduate Complexity Theory 15-455 @ CMU
TOC

Marcythm

October 30, 2022

1 Course Overview

resources; *efficient*; famous open problems; notations ($\langle X \rangle$).
Three types of problems: *decision* / *function* / *search*.

2 Turing Machines

language; *Turing machine*; *decider*; *computation trace*; **Church-Turing Thesis** (extended ver.)

3 Simulation and Turing Machine Variants

Standard model: 1-tape TM.
Variations: 1-way / alphabet / multitape / RAM; Boolean Circuits; TM operation tricks; **simulation**.

4 Time Complexity and Universal Turing Machines

(time) complexity class $\mathsf{TIME}(f(n))$; Speedup Theorem; P; *Universal TM*; ***diagonalization method***.

5 The Time Hierarchy Theorem

Time Hierarchy Theorem (more time, more power).
diagonalization method (important technique: simulation then do the opposite).
time constructible.

6 Problems in P

by THT $\exists$ more than P: EXP.
PATH, $\mathsf{2\text{-}COL}$, LCS, c-CLIQUE.

7 SAT

different forms: $\mathsf{CKT\text{-}SAT}$, $\mathsf{FORMULA\text{-}SAT}$ a.k.a. SAT, $\mathsf{CNF\text{-}SAT}$, k-SAT (descending on generality).
All NP-complete; $\mathsf{CKT\text{-}EVAL}$ not *parallelizable*; *formula*; time complexity about c-SAT.

8 NP

conjectures: $\mathsf{P} \neq \mathsf{NP}$;
ETH: $\exists \delta > 0 : \text{3-SAT} \notin \mathsf{TIME}((1+\delta)^n)$;
SETH: $\forall \delta > 0 : \exists k : \text{k-SAT} \notin \mathsf{TIME}((2-\delta)^n)$.
verifier-based def for NP. (In fact a trivial interactive proof system w/o randomness.)

9 Nondeterminism

nondeterminism (a non-realistic feature for computation models); NTIME; NP.
the equivalence between different views/defs about NP (nondeterminism / verifier).

10 Reductions

mapping reduction; $\mathsf{3COL} \leq_m^{\mathsf{P}} \mathsf{4COL} \leq_m^{\mathsf{P}} \mathsf{SAT} \leq_m^{\mathsf{P}} \mathsf{CIRCUIT\text{-}SAT} \leq_m^{\mathsf{P}} \mathsf{3SAT}$. ($\mathsf{3SAT} \leq_m^{\mathsf{P}} \mathsf{3COL}$ in lec12)
Turing reduction; $\mathsf{4CHROMA} \leq_T^{\mathsf{P}} \mathsf{SAT}$.
These reductions are *transitive*; P is closed under both reductions, but NP only under mapping reduction.

11 NP-Completeness and the Cook-Levin Theorem

NP-*hard*; NP-*complete*.
Cook-Levin Theorem: $\forall L \in \mathsf{NP} : L \leq_m^{\mathsf{P}} \mathsf{SAT}$. (proof idea: computation is local.)
Circuit-based proof of Cook-Levin.
Reading: *boolean circuit*, *circuit family*, *size/depth complexity*.

12 NP-Completeness Reductions

$\mathsf{3SAT} \leq_m^{\mathsf{P}} \mathsf{E3SAT} \leq_m^{\mathsf{P}} \mathsf{NAE\text{-}3SAT} \leq_m^{\mathsf{P}} \mathsf{3COL} \leq_m^{\mathsf{P}} \mathsf{INDSET}$; *CSP*; ***gadget***.
Reading: NP-complete problems, VERTEX-COVER, HAMPATH, UHAMPATH, SUBSET-SUM.

13 Search-to-Decision, Padding, Dichotomy Theorems

SAT is ***downward-self-reducible***; **search-to-decision reduction**.
padding (common technique: add meaningless contents to blow input size up);
Dichotomy Theorem: Every boolean CSP is either in P (2SAT, XOR-SAT, HornSAT) or NP-complete.
Dichtomy Conjecture: every CSP is either in P or NP-complete.

14 Ladner's Theorem and Mahaney's Theorem

Ladner's Theorem: $\mathsf{P} \neq \mathsf{NP} \implies \exists L \in \mathsf{NP} \backslash \mathsf{P}$ s.t. L is not NP-complete.

On class gives a weaker proof assuming ETH, with the idea basically the same: "water down" SAT to make it not so hard, thus get a valid algo for SAT which contradicts the assumption.

Mahaney's Theorem: $\mathsf{P} \neq \mathsf{NP} \implies$ *sparse* NP-complete language not exists.

Proof idea: reduce SAT to L, the width of DSR's search tree is poly (restricted by L), thus $\mathsf{SAT} \in \mathsf{P}$.

15 coNP

coNP; P is closed under complement; UNSAT is coNP-complete.
$L \in \mathsf{NP} \cap \mathsf{coNP}$ has "good characterization".

16 Space Complexity

model for space complexity: individual work tape; space complexity class $\mathsf{SPACE}(f(n))$, L, PSPACE.
Savitch's Theorem: $\mathsf{ST\text{-}PATH} \in \mathsf{SPACE}(\log^2 n)$. (for generalized version, see lec17)
Space Hierarchy Theorem. (proof is similar to THT: diagonalization method)

17 Savitch's Theorem

Savitch's Theorem: $\forall f(n) \geq n : \mathsf{NSPACE}(f(n)) \subseteq \mathsf{SPACE}(f^2(n))$. (proof idea: "middle-first search")
Nondeterminism-based def of NL: $\mathsf{NL} = \mathsf{NSPACE}(\log n)$; $\mathsf{ST\text{-}PATH} \in \mathsf{NL}$.
Reading: $\mathsf{NL} \subseteq \mathsf{SPACE}(\log^2 n), \mathsf{NL} \subseteq \mathsf{P}$; Can extend Savitch's Theorem to $f(n) \geq \log n$.

18 NL-completeness and Logspace Reductions

$\forall f(n) \geq \log n : \mathsf{NSPACE}(f(n)) \subseteq \mathsf{TIME}(2^{O(f(n))}), \mathsf{NSPACE}(f(n)) \subseteq \mathsf{SPACE}(f(n)^2)$.
logspace reduction (to reason about logspace classes, so must be as weak as logspace): closure, transitivity.
(logspace algos are usually insane since they can use crazy time for saving space.)
$\mathsf{ST\text{-}PATH}$ is NL-complete.

19 From P-completeness to PSPACE-completeness

P-complete languages: $\mathsf{HornSAT}$, LP, $\mathsf{CKT\text{-}EVAL}$;
Empirically, polytime reduction implies logspace reduction. (also Cook-Levin's)
TQBF is PSPACE-complete. (proof idea: use quantifiers to reduce size from exp to poly)
(PSPACE's essence seems to be "games", like TQBF.)

20 The Immerman-Szelepcsényi Theorem

$\mathsf{NPSPACE} = \mathsf{coNPSPACE} = \mathsf{PSPACE}$ by Savitch's. What about scaling down to logspace?
Immerman-Szelepcsényi Theorem: $\overline{\mathsf{ST\text{-}PATH}} \in \mathsf{NL} (\mathsf{NL} = \mathsf{coNL})$.
Proof idea: one step each time from s to exploit the locality of "certificates".

21 Randomized Complexity: RP, coRP, and ZPP

new computation resource / power: randomness; can greatly reduce running time with error allowed.
randomized time complexity class (one-sided error): $\mathsf{RTIME}(f(n))$, RP, coRP.
zero-sided error: $\mathsf{ZPTIME}(f(n))$, ZPP.
$\mathsf{COMPOSITES}, \mathsf{PRIMES} \in RP$; $\mathsf{PRIMES} \in \mathsf{coRP}$;
$\mathsf{P} \subseteq \mathsf{RP} \subseteq \mathsf{NP}$, $\mathsf{P} \subseteq \mathsf{coRP} \subseteq \mathsf{coNP}$; $\mathsf{ZPP} = \mathsf{RP} \cap \mathsf{coRP}$.
Reading: amplification lemma.

22 BPP

two-sided error: $\mathsf{BPTIME}(f(n))$, BPP. (named by "bounded error probabilistic time")
$\mathsf{BPP} \subseteq \mathsf{PSPACE} \subseteq \mathsf{EXP}$;
$\mathsf{P} = \mathsf{NP} \implies \mathsf{P} = \mathsf{BPP}$;
$\mathsf{BPP} \subseteq \mathsf{P}/\text{poly}$. (by amplification, then union bound)
derandomization result: If $\mathsf{3SAT}$ requires $\mathsf{SIZE}(2^{\delta n})$ for some $\delta > 0$, then $\mathsf{P} = \mathsf{BPP}$. (worst-case hardness $\implies$ strong average-case hardness $\implies$ PRNG)
Reading: Read-Once Branching Programs, $EQ_{\text{ROBP}} \in \mathsf{BPP}$ (proof idea: arithmetization on $\mathbb{F}_p$.)

23 The Polynomial Hierarchy

$\mathsf{P} = \mathsf{NP} \implies \mathsf{NP} = \mathsf{coNP}$; quantifier-based def of PH; Σ_i-SAT is $\Sigma_i\mathsf{P}$-complete.
PH "*collapses*" to the i-th level if $\Sigma_i\mathsf{P} = \Pi_i\mathsf{P}$. (e.g., if $\mathsf{P} = \mathsf{NP}$ or $\mathsf{NP} = \mathsf{coNP}$)
Reading: *Alternating TM*, ATIME, ASPACE, AP, $\mathsf{APSPACE}$, AL.
$\forall f(n) \geq n : \mathsf{ATIME}(f(n)) \subseteq \mathsf{SPACE}(f(n)) \subseteq \mathsf{ATIME}(f^2(n))$; $\forall f(n) \geq \log n : \mathsf{ASPACE}(f(n)) = \mathsf{TIME}(2^{O(f(n))})$.
corollary: $\mathsf{AL} = \mathsf{P}$, $\mathsf{AP} = \mathsf{PSPACE}$, $\mathsf{APSPACE} = \mathsf{EXP}$.

24 Oracle TMs & P^{NP}

Oracle TM; $\mathsf{P}^{\mathsf{NP}} \subseteq \Sigma_2\mathsf{P}$ (and thus $\mathsf{P}^{\mathsf{NP}} = \mathsf{co}\text{-}\mathsf{P}^{\mathsf{NP}} \subseteq \mathsf{co}\text{-}\Sigma_2\mathsf{P} = \Pi_2\mathsf{P}$). In fact P^{NP} is just $\Delta_2\mathsf{P} = \Sigma_2\mathsf{P} \cap \Pi_2\mathsf{P}$.

25 Interactive Proofs: $\mathsf{IP} = \mathsf{PSPACE}$

A glimpse past the 80s from this lec; from mid 60s to the end of 80s (space/time/randomness) in prev.
Interactive Proof System (interaction + randomness): $\mathsf{IP}[k], \mathsf{IP} = \mathsf{IP}[\text{poly}(n)]$.
$\mathsf{NP} \subseteq \mathsf{IP}[1]$; $\mathsf{BPP} \subseteq \mathsf{IP}[0]$; $\mathsf{IP} \subseteq \mathsf{PSPACE}$.
The *two-sided error* def of IP can be *automatically* upgraded to *one-sided error*.
[Shamir '89] $\mathsf{TQBF} \in \mathsf{IP}$ is motivated by [LFKN '89] $\#\mathsf{3SAT} \in' \mathsf{IP}$. (proof idea: arithmetization on $\mathbb{F}_p$.)
Reading: IP is kind of a probabilistic analog of NP, like the probabilistic analog RP of P.

26 Beyond Worst-Case Analysis

Assume $\mathsf{P} \neq \mathsf{NP}$ in next lecs. (Thus $\mathsf{3SAT} \notin \mathsf{P}$.)

To solve $\mathsf{3SAT}$: 1. allow more time; 2. allow errors (correct for $\geq 99\%$ inputs); 3. approx algos (satisfies 90% clauses for every input). Dream is to show hardness for these 3 possibilities, just with $\mathsf{P} \neq \mathsf{NP}$.

27 Hardness within P

What is really efficient maybe not P, but $O(n\,\text{polylog}(n))$; For these the <u>model</u> matters! (RAM here)
SETH $\implies \forall \epsilon > 0 : \mathsf{LCS} \notin \mathsf{TIME}(n^{2-\epsilon})$. Similar results for $\mathsf{3SUM}$, APSP, $\mathsf{k}\text{-}\mathsf{CLIQUE}$, etc.
fine-grained complexity; care about the exact complexity of reductions, not just P.
some reductions between problems, with assumptions SETH / CNF-SETH.

28 Why is P vs. NP difficult?

History of P vs. NP, & some negative results: HALTS not computable; THT; both use diagonalization.
Baker-Gill-Solovay's Theorem: $\exists A, B : \mathsf{P}^A = \mathsf{NP}^A, \mathsf{P}^B \neq \mathsf{NP}^B$. (negative result about *proof tech*!)
Proof idea: $A = \mathsf{TQBF}$, construct B using diagonalization.

Approaches after BGS: In '80s theorists try to prove harder statements; In '88 Hästad shows ckt lower-bound for PARITY; In '94 Razborov shows limitations of "natural" proof strategy.

Assuming good PRNG exists, $\nexists$ "natural" proof that NP has no poly-size ckts.