DD2445 COMPLEXITY THEORY

I

Lecture 7

Introduced circuit complexity in Lec 4

Hope: Fixed combinatorial objects
Easier to reason about than TMs
(But non-uniform circuit families can compute uncomputable things...)

Last 2 lectures:

Can prove strong lower bounds for <u>bounded-depth circuits</u>.

Didn't see in class, but can read in Arora-Barak
Can prove strong lower bounds for <u>monotone circuits</u> (AND- and OR-gates but no NOT-gates)

Compute <u>monotone functions</u>: flipping an input from 0 to 1 can never flip function from 1 to 0

But for <u>general circuits</u> next to nothing known

<u>Today:</u>

(1) Why do we believe that $NP \nsubseteq P/poly$

(2) Can we prove <u>anything</u> about TMs?

Yes: with more time, you can solve strictly more problems.

Want to prove:

If $NP \subseteq P/poly$, then the polynomial hierarchy collapses.

Recall

$\boxed{L \in P}$ if $\exists$ poly-time TM M s.t.

$$x \in L \iff M(x) = 1$$

$\boxed{L \in NP}$ if $\exists$ poly-time TM M
$\exists$ poly q

$$x \in L \iff \exists u \; M(x, u) = 1$$

u of length $= q(|x|)$

Complete problem
~~Fullständigt problem~~: CNFSAT

$$\{ \text{CNF formula } \varphi \mid \exists u \; \varphi(u) \text{ is TRUE} \}$$

$\boxed{\Sigma_2^P}$ $\exists$ poly-time TM M
$\exists$ poly q

$$x \in L \iff \exists u_1 \forall u_2 \; M(x, u_1, u_2) = 1$$

Example problem CIRCUITMINIMIZATION

$\langle C, k \rangle$ Is there a circuits C^* of size $\leq k$ s.t.
$\forall x \quad C(x) = C^*(x)$

Complete problem

$$\{ \psi \mid \psi = \exists u_1 \forall u_2 \, \varphi(u_1, u_2),$$

u_1, u_2 ~~of~~ sets of variables

φ CNF over $u_1 \cup u_2$,

ψ true $\}$

$\boxed{\Sigma_i^P}$ $\exists$ poly-time TM

$\exists$ poly q

$$x \in L \iff \exists u_1 \forall u_2 \exists u_3 \ldots Q_i u_i \, M(x, u_1, \ldots, u_i) = 1$$

Polynomial hierarchy

$$PH = \bigcup_{i=1}^{\infty} \Sigma_i^P$$

$$\Pi_i^P = co\Sigma_i^P$$

$$\Sigma_1^P = NP \quad \Pi_1^P = co\text{-}NP$$

Complete problem for ith level of the polynomial hierarchy

Σ_i SAT

$$\{ \psi \mid \psi = \exists u_1 \forall u_2 \exists u_3 \cdots Q_i u_i \; \varphi(u_1, u_2, \ldots, u_i) \\ \psi \text{ TRUE} \}$$

Proving this is a problem on pset 1.

The polynomial hierarchy "collapses to the ith level" if $PH = \Sigma_i^P$

THEOREM

If $\Sigma_i^P = \Pi_i^P$, then $PH = \Sigma_i^P$

Proving this theorem also on pset 1
(Arora - Barak proves similar statements, so read up on those and get inspired.)

Now we want to show

THEOREM [Karp - Lipton '80]

If $NP \subseteq P/poly$, then $PH = \Sigma_2^P$

Proof Sufficient to show $\Sigma_2^P = \Pi_2^P$
(by pset 1).

In fact, sufficient to show $\Pi_2^P \subseteq \Sigma_2^P$

Why? If $L \in \mathcal{C}$ then $\overline{L} \in co\mathcal{C}$ (by def.)
So take complement, use $co\mathcal{C} \subseteq \mathcal{C}$, and
take complement again.

Take Π_2^P-complete problem L

$\Pi_2 SAT =$

$$\left\{ \psi \mid \psi = \forall u \, \exists v \; \varphi(u,v) = 1 \atop u, v \in \{0,1\}^n, \; \varphi \text{ CNF} \right\}$$

Show $NP \subseteq P/poly \Longrightarrow \Pi_2 SAT \in \Sigma_2^P$

What does this mean?

Need to build TM M such that

$$\psi \text{ TRUE} \Longleftrightarrow \exists w \, \forall z \quad M(\psi, w, z) = 1$$

$$\psi = \forall u \, \exists v \;\; \varphi(u, v)$$

Given assignment $u \in \{0,1\}^n$, can plug in values for u-variables and simplify to obtain formula on v-variables only.

Denote this formula $\varphi(u, \cdot)$

Is $\varphi(u, \cdot)$ satisfiable?

This is a problem in NP.

We are assuming $NP \subseteq P/poly$

Hence $\exists$ poly-size circuit C_n that given (description of) φ and assignment u computes whether $\varphi(u, \cdot)$ is satisfiable or not

$$C_n(\varphi, u) = 1 \iff \exists v \in \{0,1\}^n \; \varphi(u, v) = 1$$

Now reduce decision to search

Run C_{n-1} on $\varphi, u, v_1 = 0$ et cetera

Yields multi-output circuits $\{C'_n\}_{n \in \mathbb{N}}$ of size poly in ψ such that for every u, C'_n computes good assignments to v-variables if such an assignment exists.

That is

$$\exists v \in \{0,1\}^n \ \varphi(u,v) = 1 \iff \varphi(u, C_n'(\varphi, u)) = 1$$

The circuit has poly size ($q(n)$, say).

Then it can be described with roughly $(q(n))^2$ bits, say

(Describe for every gate type and inputs)

But how can we find such a circuit?

Make a lucky NP-guess, and then verify

So build a TM M to

1. Guess description w of C_n'
2. Verify for ~~all~~ any $u \in \{0,1\}^n$ that $\varphi(u, C_n'(\varphi, u)) = 1$ (+ verify that w describes a legal circuit)

Such a TM runs in poly-time.

We claim

$$\psi \text{ TRUE} \iff \exists w \ \forall u \ (M(\psi, w, u) = 1)$$

If ψ TRUE, we can guess ~~circuit~~ description w and circuit will always compute good assignment

If ψ FALSE, there is u for which no good v-assignment exists $\Rightarrow$ guess will fail $\Rightarrow$ TM M rejects.

This shows that

$\Pi_2 SAT \in \Sigma_1^P$

Hence $\Pi_2^P \subseteq \Sigma_2^P$ and $\Sigma_2^P = \Pi_2^P$, QED $\boxtimes$

MORAL OF THIS STORY?

The proof we did uses completely elementary math
But still not so trivial...

Complexity theory is deep and conceptually hard stuff (at least for me)

Suggested approach

1. Skim chapter <u>before</u> lecture (but don't get stuck!)
2. Attend lecture
3. Afterwards (and ASAP), read lecture notes and chapter again, and more carefully.

<u>So far</u> Computational model (Turing machine)

Cplx class P - efficiently solvable (decision) problems

Cplx class NP - efficiently verifiable (decision) problems

NP-complete problems SAT

coNP

And above... polynomial hierarchy

EXP, NEXP

Boolean circuits as a way too understand TMs (Not terribly successful, but useful for many other reasons)

Next collection of topics on the agenda

- Is it true that more time makes it possible to solve more problems?
- Are there complexity classes between P and NP?
- Diagonalization
- Oracles (Next lecture)

How to prove that two cplx classes are different?

Find a language in one class that is not in the other

Every language $L \in \mathcal{C}$ decided by some TM M_L that runs within resource bound specified by $\mathcal{C}$

Separate $\mathcal{C}_1$ and $\mathcal{C}_2$ by finding TM M running within resource bounds specified by $\mathcal{C}_1$ that differs from every TM in $\mathcal{C}_2$ on at least one input

Then

$$L = \{ x \mid M(x) = 1 \}$$

is a language separating $\mathcal{C}_1$ and $\mathcal{C}_2$

$$L \in \mathcal{C}_1 \setminus \mathcal{C}_2$$

Essentially only known tool to do this:

DIAGONALIZATION

XI

DIAGONALIZATION

Recall: Turing machine specified by
- finite alphabet Γ (symbols)
- finite set of possible states Q
- transition function (program) mapping
  $Q \times$ {read symbols on tapes} to
  $Q \times$ {written symbols on tapes} $\times$ {head movements}

Can agree on some encoding of TMs as (finite) binary strings. Let's use encoding such that

(a) exists "stop marker", and padding with more bits after stop marker has no effect but encodes same machine.

(b) "syntax error" encoding identified with trivial TM that immediately halts and rejects, say.

Then

1. Every string $x \in \{0,1\}^*$ represents a TM M_x. Given $i \in \mathbb{N}$ write M_i to denote Turing machine encoded by i written in binary

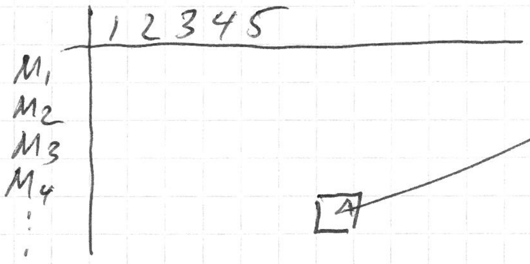
2. Every TM M is represented by infinitely many strings / infinitely many integers

3. This representation is efficient in that given x, we can simulate M_x on the universal Turing machine $\mathcal{U}$ with at most a logarithmic overhead.

Write table with rows and columns indexed by integers.

Interpret: Rows $\Leftrightarrow$ TMs

Columns $\Leftrightarrow$ inputs



(i,j) contains $M_i(j)$ output of TM M_i on input j (in binary)

~~Con~~struct TM by walking diagonally downwards to the left, making sure at least one mismatch per row $\Rightarrow$ Contradiction; such TM can't exist.

TIME HIERARCHY THEOREM

If f, g time-constructible functions satisfying $f(n) \log f(n) = o(g(n))$, then

$$\mathrm{DTIME}(f(n)) \subsetneq \mathrm{DTIME}(g(n))$$

$\subsetneq$ strict subset containment. Also $\subset$ $\subsetneq$

Time-constructible?

Technical condition which we won't go into

All "natural" functions $f(n) \geq n$ that you can think of are time-constructible

E.g. $f(n) = n \log n$, $f(n) = n^2$, $f(n) = 2^n$ etc

Will prove:

TIME HIERARCHY THEOREM, VANILLA VERSION

$$\mathrm{DTIME}(n) \subsetneq \mathrm{DTIME}(n^{1.5})$$

Proof Let D be following TM:

On input x, run universal TM $\mathcal{U}$ for $|x|^{1.4}$ steps to simulate execution of M_x on x

If $\mathcal{U}$ halts with output $b \in \{0,1\}$,

output opposite answer $1 - b$

Else output 0.

How set time bound for TM? E.g.

- compute $|x|$
- then compute $|x|^{1.4}$ & store in counter
- then fill dedicated worktape with special marker symbol, until counter decreased to 0.
- Now move back to start of work tape, start simulation, and at every step move right on "timer tape"
- abort if ever see non-marker symbol on timer tape

D decides some language, namely $L_D = \{ x \mid D(x) = 1 \}$

D runs in time $\sim n^{1.4}$ by construction (log factor for simulation does not change this)

Hence $L_D \in DTIME(n^{1.5})$ (by some margin).

We claim $L_D \notin DTIME(n)$

For contradiction, assume $\exists M$ that on any x runs in $\leq c|x|$ steps (for some fixed c) and outputs $D(x)$.

M can be simulated on any x in time $O(|x| \log |x|)$ $\leq c' \cdot |x| \log |x|$ for some c' by $\mathcal{U}$.

Fix large enough N s.t. $n^{1.4}$ is larger than this if $n \geq N$.

Pick some x of length $\geq N$ s.t.
$M_x = M$ (possible by 2 above)

Then – on input x, D will simulate M on x
- M will have time to terminate and output $M(x)$
- By def of D we have $D(x) = 1 - M(x) \neq M(x)$
- But M decides L_D by assumption, so $M(x) = D(x)$

Contradiction. Hence no such M exists, QED $\square$

There is also a time hierarchy theorem for nondeterministic computation

NONDETERMINISTIC TIME HIERARCHY THEOREM
If f, g are time-constructible functions, satisfying $f(n+1) = o(g(n))$, then

$$NTIME(f(n)) \subsetneq NTIME(g(n))$$

Proof more subtle. Will skip this.

Most problems studied $\boxed{\text{in NP}}$ are known either to be in P or to be NP-complete.
So can it be that every problem in NP is either in P or NP-complete?
(Results of that flavour known as DICHOTOMY THEOREMS.)

<u>Answer</u> If $P = NP$, then yes (trivially).
If $P \neq NP$, then no, in very strong sense.

LADNER'S THEOREM

If $P \neq NP$, then exists strict infinite hierarchy of complexity classes between P and NP

Will prove vanilla version of this either in class or on pset 2.

Today

$NP \subseteq P/poly \Rightarrow PH$ collapses

Time hierarchy theorem
Proved by diagonalization

Next: "Oracle Turing machines"
Limits of diagonalization technique