CS 331, Fall 2025

Lecture 24 (12/1)

Today: - Computability
- coNP
- NP-intermediate
- Polynomial hierarchy

Recap

Central topic of complexity theory:

Classify languages L by their computational complexity

- languages: subset of binary strings $\{0,1\}^*$
- computational complexity: cost to decide "is $x \in L$?"

Ex. Define language $SAT \subseteq \{0,1\}^*$

$x \in SAT$ if... $x = \langle \Phi \rangle$ s.t. Φ satisfiable

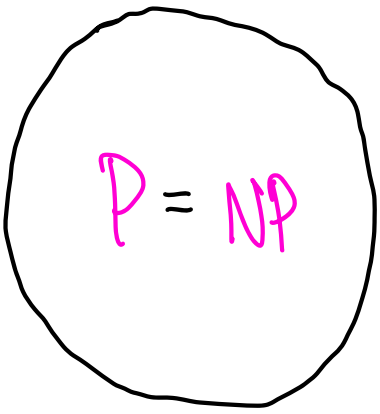
($\langle \Phi \rangle$: encode CNF formula)

$x \notin SAT$ if... $x = \langle \Phi \rangle$ s.t. Φ unsatisfiable

$x \neq \langle \Phi \rangle$ doesn't encode formula

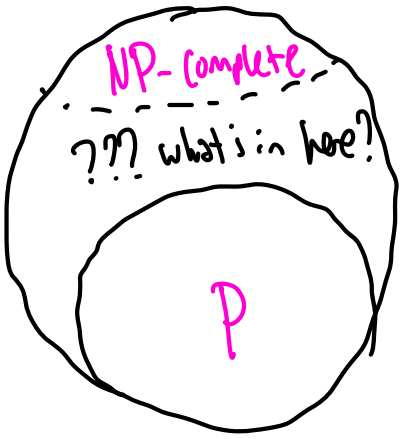
Exercise: how to show $L \in P$? NP? NP-hard?

The worlds we've discussed...



???
what's out here?

Algorithmica



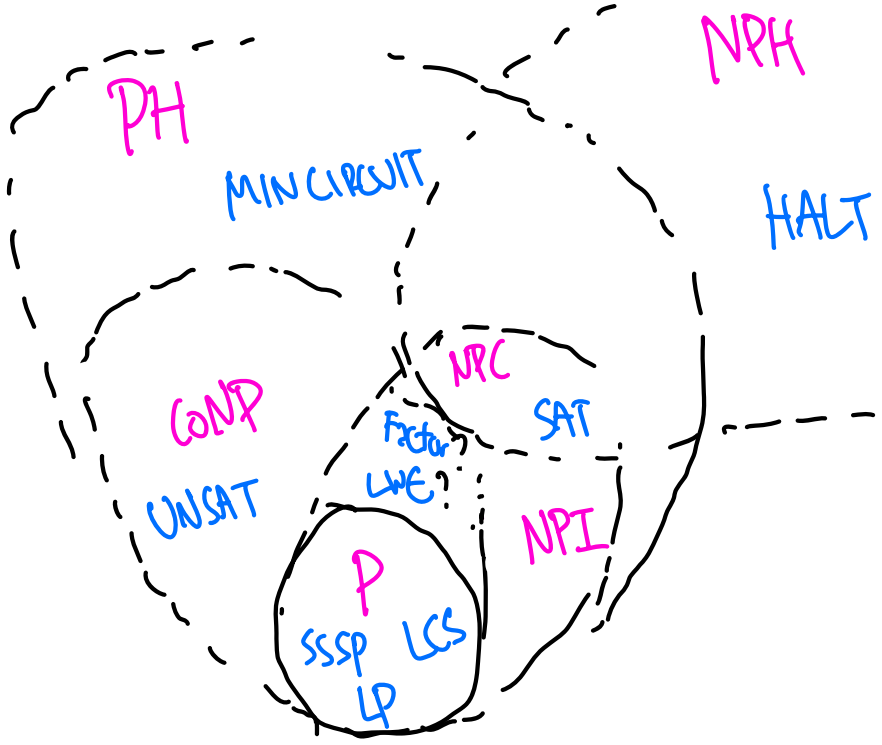
NP-hard \ NP
???

Pessiland

Complexity theory studies many worlds.

1) Time complexity beyond P & NP (this class)

2) Beyond time complexity (next class)



We'll explore more of this map today.

Computability (Part VIII, Section 5.1)

We will show a language $L \in NP\text{-hard} \setminus NP$.

Along way we prove: $\exists L \notin TIME(f(n))$

"uncomputable language"

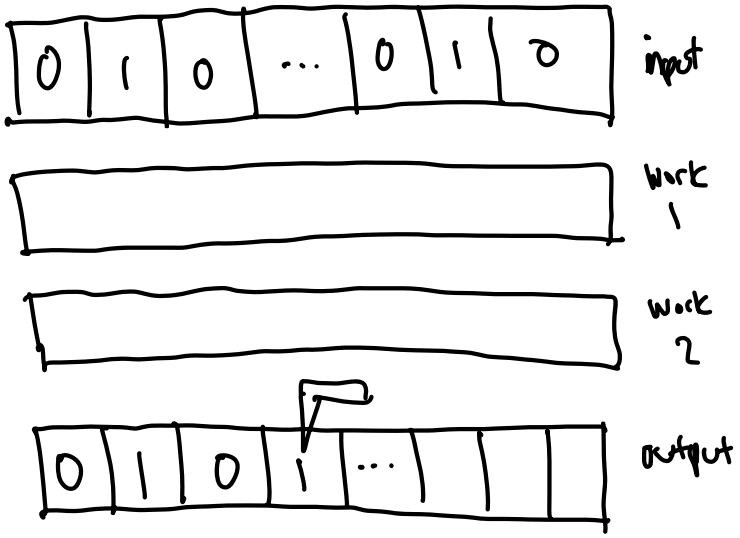
any f'n: n^2, $\exp(n)$, ...

What do we mean by $TIME(f(n))$?

Depends on computational model

- (Multi-tape) Turing machine?

  basic ops: read, write, move L/R1 tape



- Random access memory?

  GOTO: memory address XYZ

(Extended) Church-Turing thesis: TM suffices for simulating polytime computability in classical models of computation

$TIME(f(n))$: All L decidable using $f(n)$ steps of some TM A on length-n inputs

We lump multiple $f(n)$ together, e.g.

$TIME(O(n^2))$ = "quadratic time decidable"

$TIME(poly(n))$ = "polytime decidable" = P

How to establish $L \in NP\text{-hrd} \setminus NP$?

Idea 1: Trading nondeterminism for time

Lemma: $NP \subseteq TIME(exp(poly(n)))$

Proof: $\exists A(x, h)$ deciding $L \in NP$, $|h| = poly(|x|)$

Simulate nondeterminism yourself (brute force) ▨

$\Rightarrow$ there are limits to NP's power.

Idea 2: "code is data, data is code"

For every $\alpha \in \{0,1\}^*$ "data"

define assoc. algo (Turing machine instructions) A_α "code"

e.g. if α compiles to assembly $\Rightarrow$ A_α executes assembly

if α makes no sense as code $\Rightarrow$ A_α returns True

We assume <u>any algo</u> A can execute on <u>any input</u> x

Just as language L partitions $x \in \{0,1\}^*$

$x \in L$ vs. $x \notin L$

algo A partitions $x \in \{0,1\}^*$

$x \in Y(A)$ vs. $x \in N(A)$ vs. $x \in L(A)$

$A(x) = \text{True}$ $A(x) = \text{False}$ $A(x)$ loops indefinitely

Idea 3: diagonalization

Define a language

$$UC = \{ \alpha \in \{0,1\}^* \mid \alpha \notin Y(A_\alpha) \}$$

"feed A_α its own code"

Observe that

$\alpha \in UC \Rightarrow A_\alpha(\alpha) \neq$ True (★)

$\alpha \notin UC \Rightarrow A_\alpha(\alpha) =$ True (★★)

Claim: No also decides UC.

Proof: Suppose A_β decided UC. (β = code)

Is $\beta \in UC$? $A_\beta(\beta) \neq$ True by (★). Wrong!

Is $\beta \notin UC$? $A_\beta(\beta) =$ True by (★★). Wrong! ☠

We defined UC as: whatever A says, do the opposite.

Powerful idea...

- Cantor $|\mathbb{R}| \neq |\mathbb{N}|$
- Gödel's incompleteness theorems.
- Time hierarchy theorem

$$\mathrm{TIME}(O(n^2)) \subsetneq \mathrm{TIME}(O(n^3)) \subsetneq \cdots$$

Downside: objects constructed not very natural/explicit.

Luckily, more human-interpretable undecidable language:

$$\mathrm{HALT} = \left\{ (d, x) \in \{0,1\}^* \;\middle|\; x \notin L(A_d) \right\}$$

(A_d halts on x)

Claim: No algo decides HALT.

Proof: Say A decides HALT. Define A':

If $A(d,d) = \text{False} \Rightarrow$ return $d \in UC$

Else compute $A_d(d)$ to decide UC. $\Rightarrow\Leftarrow$

Corollary: HALT $\notin$ NP $\subseteq$ TIME(exp(poly(n)))

However, HALT is NP-hrd!

Let $L \in NP$. Wnt to decide $x \stackrel{?}{\in} L$

Design A as follows:

Let A' be non-deterministic algo for L

Keep looping all hints h, stop if

$A'(x, h) = YES$

Ask 

: does A halt on x?

So, $L \leq_p$ HALT. Karp reduction!

Unconditionally shows NP-hrd $\neq$ NP-complete.

co-NP (Part VIII, Section 5.2)

Define complement of language L:

$$\overline{L} = \{x \in \{0,1\}^* \mid x \notin L\} \text{ a.k.a. } co(L)$$

e.g. UNSAT:

$x \in$ UNSAT if... $x = \langle \Phi \rangle$ s.t. Φ unsatisfiable

$x \neq \langle \Phi \rangle$ doesn't encode formula

$x \notin$ UNSAT if... $x = \langle \Phi \rangle$ s.t. Φ satisfiable

coNP: all L s.t. $co(L) \in NP$

Equivalently: all L decidable by non-deterministic polytime

$A(x, h)$: if $x \in L$, $\forall h$ $A(x,h) =$ True

$|h| = poly(|x|)$ $x \notin L$, $\exists h$ $A(x,h) =$ False

Idea: use same hint, reverse output True vs. False

Example $x \in$ UNSAT hard to prove, easy to refute

Decide UNSAT(x, h):
  If x encodes CNF Φ & h encodes assignment:
    If Φ satisfied by assignment in h:
      return False
  return True

Claim: If $A \leq_p B$ then $A \leq_p co(B)$

Proof: $B \leq_p co(B)$

ask  & reverse output

(Warning: not true if $\leq_p$ w.r.t. Turing reduction)

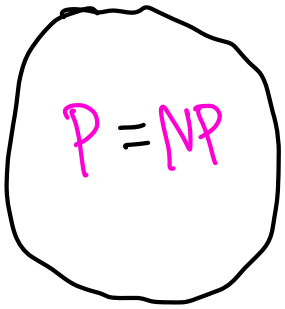
Claim: If $L \in$ NP-complete, $co(L) \in$ coNP-complete

Proof: For all $L' \in NP$,

$$co(L') \leq_p L' \leq_p L \leq_p co(L)$$

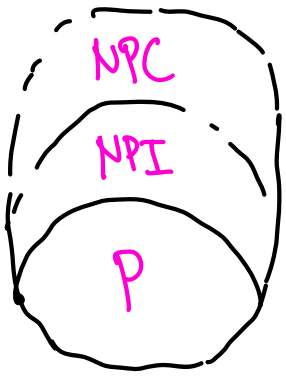
NP-intermediate (Part VIII, Section 5.2)

NP-intermediate: $NP \setminus (P \cup NP\text{-hard})$



Algorithmica

NP-intermediate $= \emptyset$



Pessiland

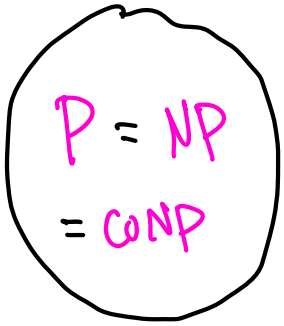
NP-intermediate $\neq \emptyset$!

Ladner's theorem: uses diagonalization

What about explicit candidates $L \in$ NP-intermediate?

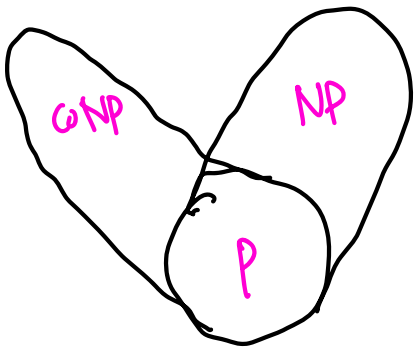
Fact: If $NP \neq coNP$,

any $L \in (NP \cap coNP) \setminus P$ is NP-intermediate.



World 1

World 2



World 3

What worlds correspond to...

- P vs. NP?
- NP vs. coNP?

What is in $(NP \cap coNP) \setminus P$?

Try 1: $Primes = \{\langle p \rangle \mid p \in \mathbb{N} \text{ is prime}\}$

(obvious) $Primes \in coNP$

(Pratt, 1975) $Primes \in NP$ "primality certificate"

(AKS, 2004) $Primes \in P$ ...

Try 2: $Factoring = \{\langle p, t \rangle \mid s|p \text{ for } 2 \leq s \leq t\}$

(obvious) $Factoring \in NP$

(Pratt, 1975) $Factoring \in coNP$

Proof: give prime factorization of p,
check all factors prime & $> t$

(Shor, 1994) $Factoring \in BQP$ (quantum??)

Many interesting problems in $NP \cap coNP$ Parity Games, LWE
... often lead to nontrivial algos! quasipoly time (2017) quantum algos?

Polynomial hierarchy (Part VIII, Section 52)

Motivation: If $P = NP$, also have $P = coNP$

What else can we conclude $= P$?

Example

$$\text{MinCircuit} = \{\langle C \rangle \mid C \text{ is "minimal" circuit for output}\}$$

Is $\text{MinCircuit} \in NP$?

... unclear how to certify w/o enumerating smaller circuits

$\text{MinCircuit} \in coNP$?

... given refuting smaller circuit, how to verify output?

Not obvious: "higher level" of non-determinism needed

$$\langle C \rangle \in \text{MinCircuit} \iff \forall |C'| < |C|, \exists x \text{ s.t. } C'(x) \neq C(x)$$

(not computing the same output)

Turns out MinCircuit $\in \Pi_2 P$

$\Pi_2 P$: all L decidable by non-deterministic polytime

$A(x, h_1, h_2)$: if $x \in L$: $\forall\, h_1 \in \{0,1\}^*$ $A(x, h_1, h_2)$
$\exists\, h_2 \in \{0,1\}^*$ = True

$|h_1|$, $|h_2|$ = poly($|x|$)

$x \notin L$: $\exists\, h_1 \in \{0,1\}^*$ $A(x, h_1, h_2)$
$\forall\, h_2 \in \{0,1\}^*$ = False

e.g. MinCircuit uses h_1 = smaller circuit, h_2 = wrong input

$\Sigma_2 P$: all L s.t. $co(L) \in \Pi_2 P$

PH: the polynomial hierarchy

$$P \cup (NP \cup coNP) \cup (\Sigma_2 P \cup \Pi_2 P) \cup \dots$$

Fact: If $P = NP$, then $P = PH$ (collapse to Level 0)

Other possible worlds: $PH = NP = coNP$, (collapse to Level 1)

$PH = \Sigma_2 P = \Pi_2 P$, (collapse to Level 2)

... ...