Intro to Theory of Computation

CS 332

LECTURE 22

Last time

- Measuring complexity
- Relationship between models
- Class **P**

Today

- Class **NP**
- The **P** vs **NP** question
- Polynomial-time reductions

Sofya Raskhodnikova

CS 332

Exercise

Consider the following algorithm A for PRIMES.

Given b, try to divide b by 2,3,…,$\sqrt{b}$.

If one of them divides b, accept; o.w. reject.

If n = input length, # of divisions A performs is

A. $\Theta(\sqrt{n})$

B. $\Theta(n)$

C. $2^{\Theta(n)}$

D. $2^{\Theta(\sqrt{n})}$

E. None of the above.

CS 332

Every CFL is in P

Recall: Chomsky Normal Form for CFGs

- Can have a rule $S \rightarrow \varepsilon$.
- All remaining rules are of the form

  $A \rightarrow BC$ $\qquad A, B, C \in V$

  $A \rightarrow a$ $\qquad a \in \Sigma$
- Cannot have S on the RHS of any rule.

Idea: use dynamic programming

- Solve smaller subproblems
- Record results in a table
- Construct solution for each subproblem from smaller solved instances

(On the board)

Difference in time complexity

At most ***polynomial*** difference between ***all reasonable*** deterministic models.

At most ***exponential*** difference between deterministic and nondeterministic models.

The class P

P is the class of languages decidable in polynomial time on a ***deterministic*** 1-tape TM:

$$\boldsymbol{P} = \bigcup_k TIME(n^k).$$

- The same class even if we substitute another reasonable deterministic model.
- Roughly the class of problems realistically solvable on a computer.

Examples of languages in P

Problem	Description	Algorithm	Yes	No
MULTIPLE	Is x a multiple of y?	Grade school division	51, 17	51, 16
RELPRIME	Are x and y relatively prime?	Euclid (300 BCE)	34, 39	34, 51
PRIMES	Is x prime?	AKS (2002)	53	51
all CFLs (e.g. the language of balanced parentheses and brackets)	Is the string in the given CFL? (e.g., is the string of parentheses and brackets balanced?)	Dynamic programming	Depends on the language; e.g. (([])[])	Depends on the language; e.g. ([)], (()
LSOLVE	Is there a vector x that satisfies Ax = b?	Gauss-Edmonds elimination	$\begin{bmatrix}0 & 1 & 1\\ 2 & 4 & -2\\ 0 & 3 & 15\end{bmatrix}, \begin{bmatrix}2\\ 4\\ 36\end{bmatrix}$	$\begin{bmatrix}1 & 0 & 0\\ 1 & 1 & 1\\ 0 & 1 & 1\end{bmatrix}, \begin{bmatrix}1\\ 1\\ 1\end{bmatrix}$

CS 332

Central ideas

- Poly-time as "feasible"
  - most natural problems either are easy (say in TIME(n^3)) or have no known poly-time algorithms

- P =languages that can be decided in poly-time
- NP = languages for which the membership in the language is easy to verify given a hint
- EXP = languages that can be decided in exponential time

- Poly-time Reductions: X is no harder than Y for poly-time TMs

CS 332

Class NP

- Verification algorithm intuition
  - Verifier views things from "managerial" viewpoint.
  - Verifier doesn't determine whether w $\in$ L on its own; rather, it checks with a proposed hint whether w $\in$ L.

- Algorithm V($\langle w, c \rangle$) is a **verifier** for language L if for every string w, **w $\in$ L** iff **there *exists* a string c such that V($\langle w, c \rangle$) accepts**.

  "certificate" or "witness"

- The running time of a verifier is measured only in terms of length of w. A **polynomial-time verifier** runs in time polynomial in |w| and has certificate c of length polynomial in w:
  i.e., |c|=O($|w|^k$) for some constant k.

CS 332

The class NP

NP is the class of languages that have polynomial-time verifiers.

CS 332

Examples of languages in NP

- COMPOSITES = $\{\langle x \rangle \mid x = pq, \text{ for int } p, q > 1\}$
- certificate: integer $p > 1$ that divides x

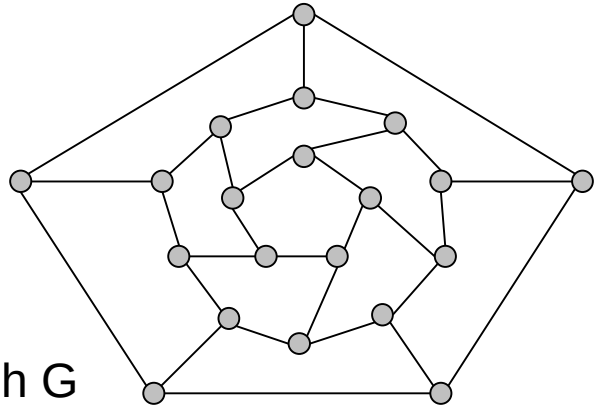
  such a certificate exists iff x is composite
- verifier

  V = `` On input $\langle x, p \rangle$, where x and p are integers:

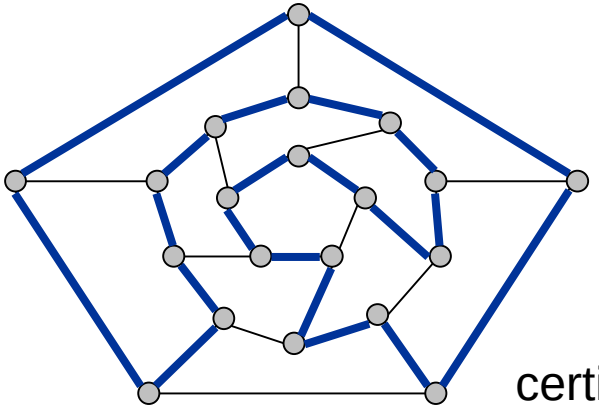
  1. **If $p \leq 1$ or $p \geq x$, reject.**
  2. **Else if x is a multiple of p, accept. O.w. reject.”**

Examples of languages in NP

- UHamCycle = $\{\langle G \rangle \mid G$ is an undirected graph that contains a cycle C that visits each node exactly once$\}$
- certificate C: Hamiltonian cycle (i.e., permutation of the nodes)



graph G



certificate C

CS 332

Examples of languages in NP

- UHamCycle = $\{\langle G \rangle \mid G$ is an undirected graph that contains a cycle C that visits each node exactly once$\}$
- certificate C: Hamiltonian cycle (i.e., permutation of the nodes)
- verifier **V = `` On input $\langle G, C \rangle$:**

  1. **Accept if**
  2. **each node of G appears in C exactly once**
  3. **there is an edge between every pair of adjacent nodes in C**
  4. **O.w. reject.''**

CS 332

Examples of languages in NP: SAT

- **Boolean variables:** variables that can take on values T/F (or 1/0)
- **Boolean operations:** $\vee$, $\wedge$, and $\neg$
- **Boolean formula:** expression with Boolean variables and ops

  Example: $(x_1 \vee \overline{x_2}) \wedge x_3$
- An **assignment** of 0s and 1s to the variables **satisfies** formula φ if it makes it evaluate to 1.
- φ is **satisfiable** if there exists an assignment that satisfies it.

$$\text{SAT} = \{\langle \varphi \rangle \mid \varphi \text{ is a satisfiable Boolean formula}\}.$$

Prove: SAT $\in$ **NP**.

CS 332

Classes P, NP, EXP

- **P. Languages for which there is a poly-time algorithm.**

  algorithm that runs in time $O(n^k)$ for some k

- **EXP. Languages for which there is an exponential-time algorithm.**

  algorithm that runs in time $O(2^{n^k})$ for some k

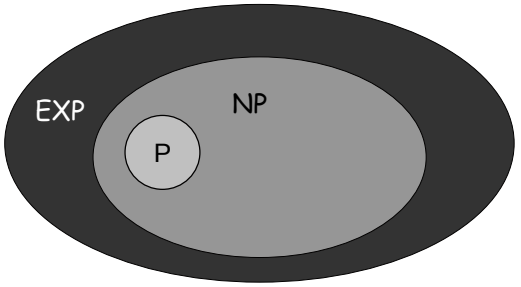
- **NP. Languages for which there is a poly-time verifier.**

- **Lemma.** $P \subseteq NP$.
- **Lemma.** $NP \subseteq EXP$.
- **Lemma.** A language L is in NP iff L can be decided by a polynomial-time nondeterministic TM.

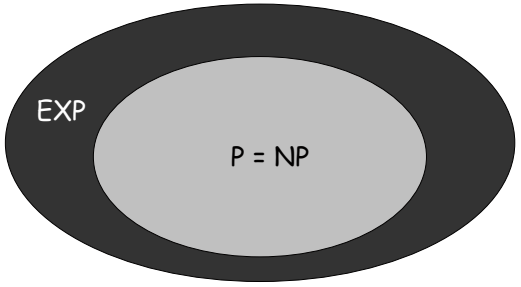
CS 332

P vs. NP

- Does P = NP? [Cook 1971, Edmonds, Levin, Yablonski, Gödel]
  - Is the decision problem as easy as the verification problem?
  - Clay $1 million prize.



If P ≠ NP



If P = NP

would break RSA cryptography
(and potentially collapse economy)

- If yes: Efficient algorithms for UHamPath, SAT, TSP, factoring
- If no: No efficient algorithms possible for these problems.

- Consensus opinion on P = NP? Probably no.

CS 332

Classify Problems

- **Desiderata:** classify problems according to those that can be solved in polynomial-time and those that cannot.
- Some problems *provably require exponential time* (Chapter 9):
  - Given a Turing machine, does it halt in at most *k* steps?
  - Given a board position in an *n*-by-*n* generalization of chess, can black guarantee a win?
- **Frustrating news:** huge number of fundamental problems have defied classification for decades.
- **Chapters 7.4-7.5 (NP-completeness):** Show that these fundamental problems are "computationally equivalent" and appear to be different manifestations of one **really hard** problem.

Polynomial-time reduction

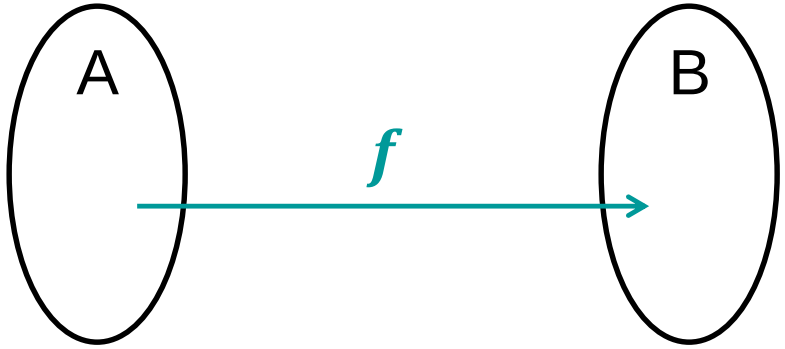
Given languages A and B,

$A \leq_p B$

if there is a ***poly-time*** computable function f,

such that for all strings w,

$w \in A$ iff $f(w) \in B$.



Polynomial-time reductions are the major tool we have to understand P and NP

Implication of poly-time reductions

Theorem. If $A \leq_p B$ and $B \in \mathbf{P}$ then $A \in \mathbf{P}$.

(So, if $A \leq_p B$ and $A \notin \mathbf{P}$ then $B \notin \mathbf{P}$.)

Theorem. If $A \leq_p B$ and $B \leq_p C$ then $A \leq_p C$.

(Poly-time reductions compose.)

CS 332

Basic reduction strategies

Basic reduction strategies

- Reduction by simple equivalence.
- Reduction from special case to general case.
- Reduction by encoding with gadgets.

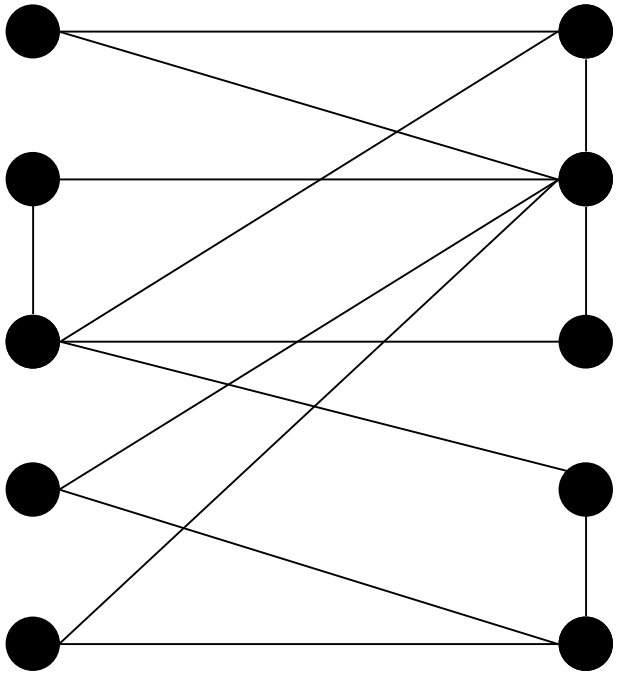
Independent Set

Given an undirected graph G, an **independent set** in G is a set of nodes, which includes at most one endpoint of every edge.

INDEPENDENT SET $= \{\langle G, k\rangle \mid G$ is an undirected graph which has an independent set with k nodes$\}$

- Is there an independent set of size ≥ 6?
  - Yes.

- Is there an independent set of size ≥ 7?
  - No.



CS 332

Vertex Cover

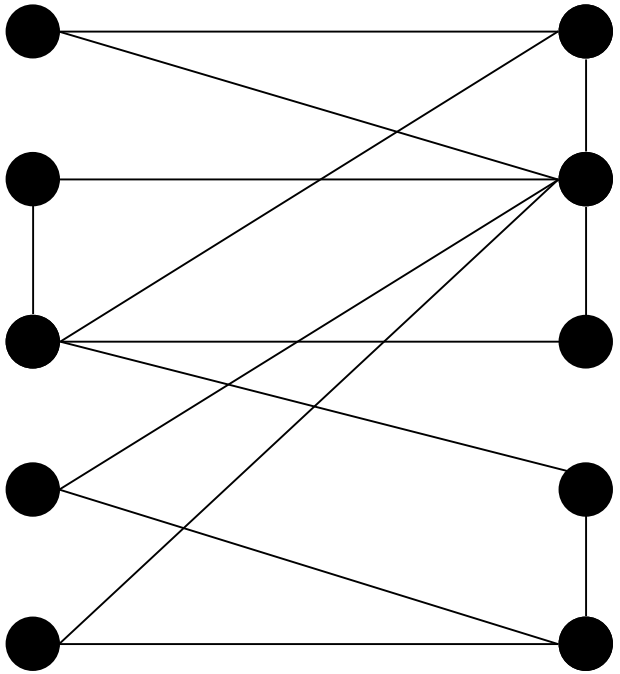
Given an undirected graph G, a **vertex cover** in G is a set of nodes, which includes at ***least*** one endpoint of every edge.

VERTEX COVER = $\{\langle G, k\rangle \mid G$ is an undirected graph which has a vertex cover with k nodes$\}$

- Is there vertex cover of size ≤ 4?
  - Yes.

vertex cover

- Is there a vertex cover of size ≤ 3?
  - No.



Independent Set and Vertex Cover

Claim. S is an independent set iff V – S is a vertex cover.

- $\Rightarrow$
  - Let S be any independent set.
  - Consider an arbitrary edge (u, v).
  - S is independent $\Rightarrow$ u $\notin$ S or v $\notin$ S $\Rightarrow$ u $\in$ V – S or v $\in$ V – S.
  - Thus, V – S covers (u, v).
- $\Leftarrow$
  - Let V – S be any vertex cover.
  - Consider two nodes u $\in$ S and v $\in$ S.
  - Then (u, v) $\notin$ E since V – S is a vertex cover.
  - Thus, no two nodes in S are joined by an edge $\Rightarrow$ S independent set. ▪

CS 332

INDEPENDENT SET reduces to VERTEX COVER

Theorem. INDEPENDENT-SET $\leq_p$ VERTEX-COVER.

Proof. "On input $\langle G, k \rangle$, where G is an undirected graph and k is an integer,

1. Output $\langle G, n - k \rangle$, where n is the number of nodes in G."

Correctness:

- G has an independent set of size k iff it has a vertex cover of size $n - k$.
- Reduction runs in linear time.

CS 332

Reduction from special case to general case

Basic reduction strategies

- Reduction by simple equivalence.
- Reduction from special case to general case.
- Reduction by encoding with gadgets.