# CS 401

## NP and NP-Complete

Xiaorui Sun

# Stuff

Homework 5 due May 6

- Helpful for final exam preparation

Teaching evaluation

- Extra 1% score for all the students if overall response rate >= 80%
- Additional to the final score cut
- May improve the final grade (if overall response rate >= 80%)

Final exam review this Thursday

# Decision Problems

Decision problem

- X is a set of strings.
- Instance: string s.
- Algorithm A solves problem X: A(s) = `yes` iff s $\in$ X.

Polynomial time Algorithm A runs in poly-time if for every string s, A(s) terminates in at most p(|s|) "steps", where p(·) is some polynomial.

length of s

P: Decision problems for which there is a poly-time algorithm.

# NP

Certification algorithm intuition

- Certifier views things from "managerial" viewpoint.
- Certifier doesn't determine whether $s \in X$ on its own; rather, it checks a proposed proof t that $s \in X$.

Def Algorithm C(s, t) is a certifier for problem X if for every string s, $s \in X$ iff there exists a string t such that C(s, t) = `yes`.

"certificate" or "witness"

NP Decision problems for which there exists a poly-time certifier.

C(s, t) is a poly-time algorithm and $|t| \leq p(|s|)$ for some polynomial $p(\cdot)$.

Remark NP stands for nondeterministic polynomial-time.

# Certifiers and Certificates: Composite

COMPOSITES. Given an integer s, is s composite?

Certificate. A nontrivial factor t of s. Note that such a certificate exists iff s is composite. Moreover $|t| \leq |s|$.

Certifier.

```
boolean C(s, t) {
   if (t ≤ 1 or t ≥ s)
      return false
   else if (s is a multiple of t)
      return true
   else
      return false
}
```

Instance. s = 437,669.

Certificate. t = 541 or 809. ← $437{,}669 = 541 \times 809$

Conclusion. COMPOSITES is in NP.

# Certifiers and Certificates: Hamiltonian Cycle

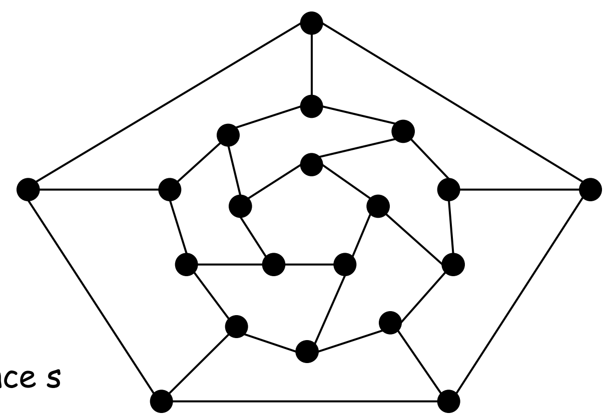
HAM-CYCLE. Given an undirected graph G = (V, E), does there exist a simple cycle C that visits every node?

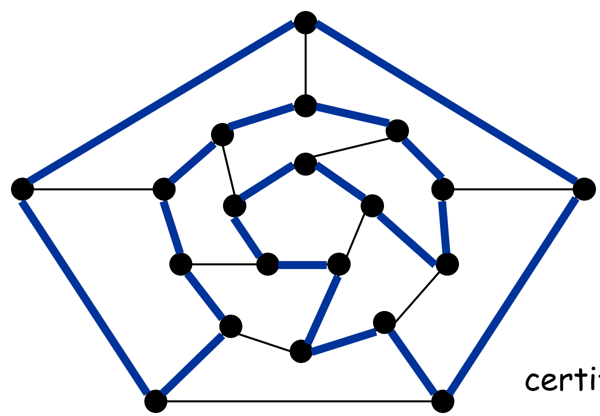
Certificate. A permutation of the n nodes.

Certifier. Check that the permutation contains each node in V exactly once, and that there is an edge between each pair of adjacent nodes in the permutation.

Conclusion. HAM-CYCLE is in NP.

instance s





certificate t

# 3-Satisfiability

Literal: A Boolean variable or its negation. $x_i$ or $\overline{x_i}$

Clause: A disjunction of literals. $C_j = x_1 \vee \overline{x_2} \vee x_3$

Conjunctive normal form: A propositional formula $\Phi$ that is the conjunction of clauses. $\Phi = C_1 \wedge C_2 \wedge C_3 \wedge C_4$

SAT: Given CNF formula $\Phi$, does it have a satisfying truth assignment?

3-SAT: SAT where each clause contains exactly 3 literals.

Ex: $\left(\overline{x_1} \vee x_2 \vee x_3\right) \wedge \left(x_1 \vee \overline{x_2} \vee x_3\right) \wedge \left(x_2 \vee x_3\right) \wedge \left(\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}\right)$

Yes: $x_1$ = true, $x_2$ = true $x_3$ = false.

# Certifiers and Certificates: 3-Satisfiability

SAT. Given a CNF formula $\Phi$, is there a satisfying assignment?

Certificate. An assignment of truth values to the n boolean variables.

Certifier. Check that each clause in $\Phi$ has at least one true literal.

Ex.

$$\left( \overline{x_1} \vee x_2 \vee x_3 \right) \wedge \left( x_1 \vee \overline{x_2} \vee x_3 \right) \wedge \left( x_1 \vee x_2 \vee x_4 \right) \wedge \left( \overline{x_1} \vee \overline{x_3} \vee \overline{x_4} \right)$$

instance s

$$x_1 = 1,\ x_2 = 1,\ x_3 = 0,\ x_4 = 1$$

certificate t

Conclusion. SAT is in NP.

# P and NP

P: Decision problems for which there is a poly-time algorithm.

NP: Decision problems for which there is a poly-time certifier.

Claim P $\subseteq$ NP.

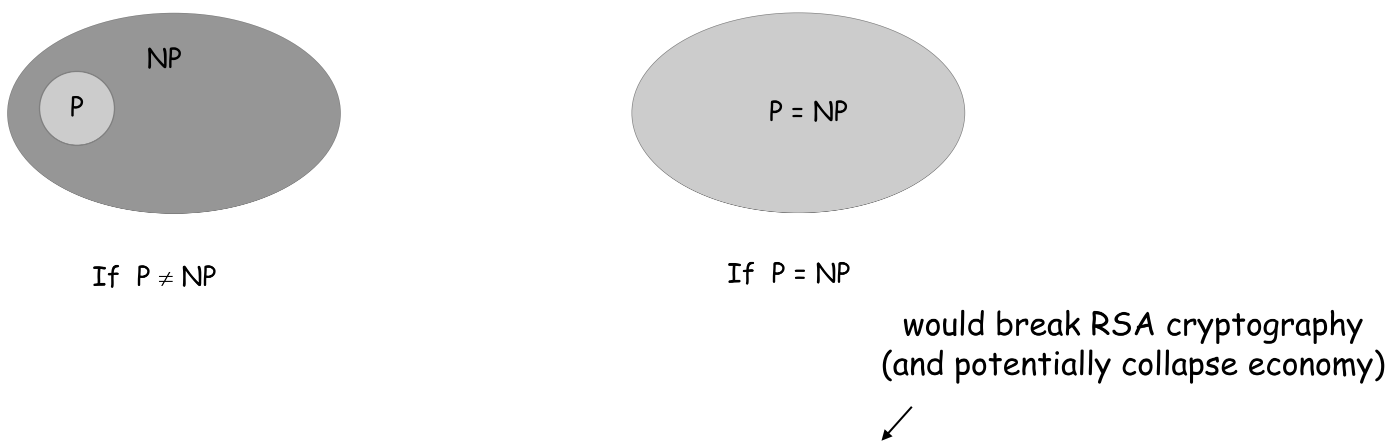
Pf. Consider any problem X in P.

- By definition, there exists a poly-time algorithm A(s) that solves X.
- Certificate: t = empty string, certifier C(s, t) = A(s).

# The Main Question: P Versus NP

Does P = NP? [Cook 1971, Edmonds, Levin, Yablonski, Gödel]

- Is the decision problem as easy as the certification problem?
- Clay $1 million prize.




If yes: Efficient algorithms for 3-COLOR, TSP, FACTOR, SAT, …

If no: No efficient algorithms possible for 3-COLOR, TSP, SAT, …

Consensus opinion on P = NP? Probably no.

# Summary

P: Decision problems for which there is a poly-time algorithm.

NP: Decision problems for which there is a poly-time certifier.

Claim P $\subseteq$ NP

Open question: Does P = NP? [Cook 1971, Edmonds, Levin, Yablonski, Gödel]

- Is the decision problem as easy as the certification problem?

would break RSA cryptography
(and potentially collapse economy)

If yes: Efficient algorithms for 3-COLOR, TSP, FACTOR, SAT, …

If no: No efficient algorithms possible for 3-COLOR, TSP, SAT, …

# NP-Complete

# NP Completeness

Complexity Theorists Approach: We don't know how to prove that some problems in NP, like Independent Set, are hard. So, let's find hardest problems in NP.

NP-complete: A problem Y in NP with the property that for every problem X in NP, $X \leq_p Y$.

Motivations:

If $P \neq NP$, then every NP-Complete problems is not in P. So, we shouldn't try to design polytime algorithms

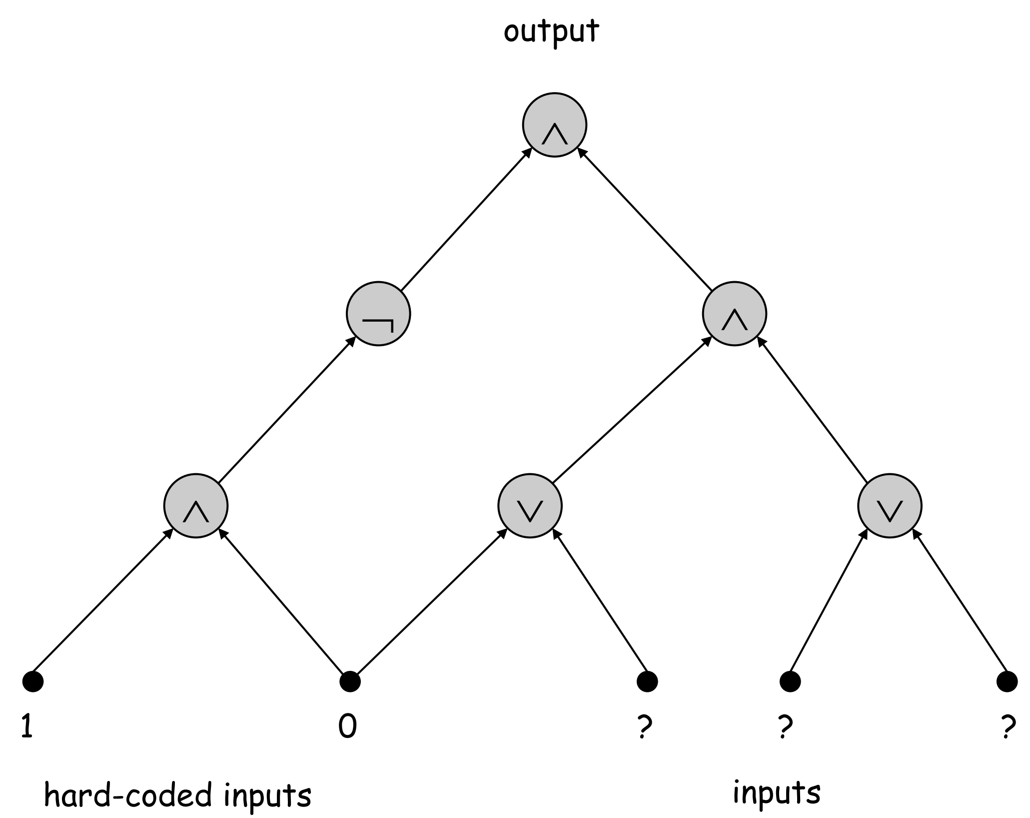
To show $P = NP$, it is enough to design a polynomial time algorithm for just one NP-complete problem.

# The "First" NP-Complete Problem

CIRCUIT-SAT. Given a combinational circuit built out of AND, OR, and NOT gates, is there a way to set the circuit inputs so that the output is 1?

Theorem. CIRCUIT-SAT is NP-complete. [Cook 1971, Levin 1973]

yes: 1 0 1




Read KT 8.4 for the proof

# Establishing NP-Completeness

Remark. Once we establish first "natural" NP-complete problem, others fall like dominoes.

Recipe to establish NP-completeness of problem Y.

- Step 1. Show that Y is in NP.
- Step 2. Choose a known NP-complete problem X.
- Step 3. Prove that $X \le_p Y$.

Justification. If X is an NP-complete problem, and Y is a problem in NP with the property that $X \le_P Y$ then Y is NP-complete.

Pf. Let W be any problem in NP. Then $W \underset{\text{by definition of NP-complete}}{\le_P} X \underset{\text{by assumption}}{\le_P} Y$.

- By transitivity, $W \le_P Y$.
- Hence Y is NP-complete. ▪

# 3-SAT is NP-Complete

Theorem. 3-SAT is NP-complete.

We know CIRCUIT-SAT is NP complete.

To show 3-SAT is NP-complete, what do we need to prove?

CIRCUIT-SAT $\leq_P$ 3-SAT or 3-SAT $\leq_P$ CIRCUIT-SAT?

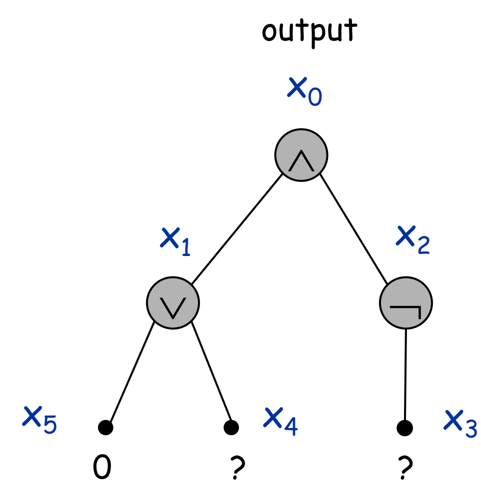
Answer: CIRCUIT-SAT $\leq_P$ 3-SAT

# 3-SAT is NP-Complete

Theorem. 3-SAT is NP-complete.

Pf. Suffices to show that CIRCUIT-SAT $\leq_P$ 3-SAT since 3-SAT is in NP.

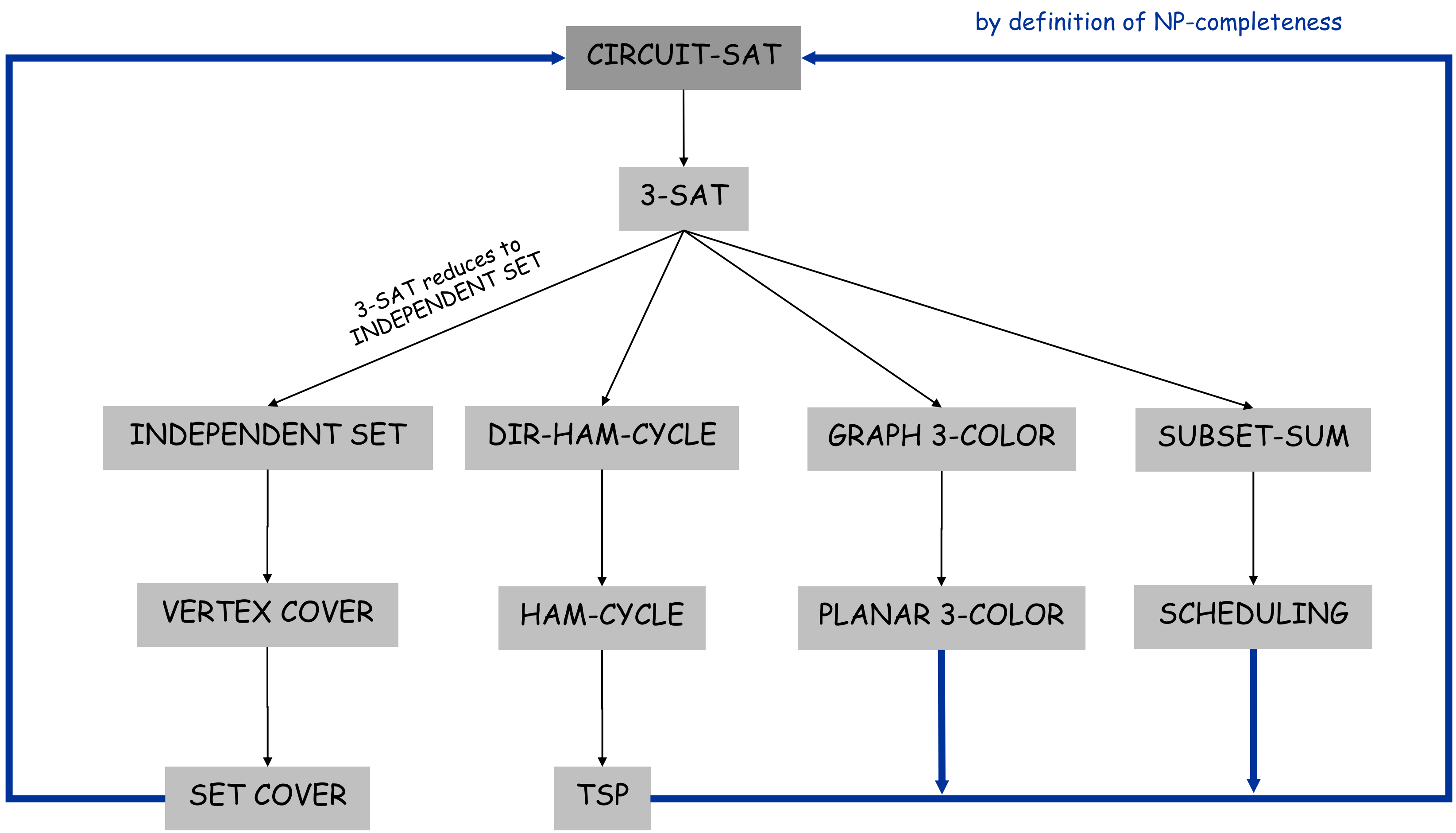
- Let K be any circuit.
- Create a 3-SAT variable $x_i$ for each circuit element i.
- Make circuit compute correct values at each node:
  - $x_2 = \neg x_3 \Rightarrow$ add 2 clauses: $x_2 \vee x_3$ , $\overline{x_2} \vee \overline{x_3}$
  - $x_1 = x_4 \vee x_5 \Rightarrow$ add 3 clauses: $x_1 \vee \overline{x_4}$, $x_1 \vee \overline{x_5}$ , $\overline{x_1} \vee x_4 \vee x_5$
  - $x_0 = x_1 \wedge x_2 \Rightarrow$ add 3 clauses: $\overline{x_0} \vee x_1$, $\overline{x_0} \vee x_2$, $x_0 \vee \overline{x_1} \vee \overline{x_2}$




- Hard-coded input values and output value.
  - $x_5 = 0 \Rightarrow$ add 1 clause: $\overline{x_5}$
  - $x_0 = 1 \Rightarrow$ add 1 clause: $x_0$
- Final step: turn clauses of length < 3 into clauses of length exactly 3. ▪

# NP-Completeness

Observation. All problems below are NP-complete and polynomial reduce to one another!



# Some NP-Complete Problems

Six basic genres of NP-complete problems and paradigmatic examples.

- Packing problems: SET-PACKING, INDEPENDENT SET.
- Covering problems: SET-COVER, VERTEX-COVER.
- Constraint satisfaction problems: SAT, 3-SAT.
- Sequencing problems: HAMILTONIAN-CYCLE, TSP.
- Partitioning problems: 3D-MATCHING 3-COLOR.
- Numerical problems: SUBSET-SUM, KNAPSACK.

Practice. Most NP problems are either known to be in P or NP-complete.

Notable exceptions. Factoring, graph isomorphism.

Determine if a composite number c has a factor $\leq$ k