CSE 421 Introduction to Algorithms

Lecture 22

NP-Completeness

Announcements

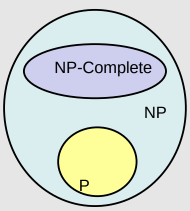
- Read Chapter 8
- Old final exams posted on course homepage

Algorithms vs. Lower bounds

- Algorithmic Theory
  - What we can compute
    - I can solve problem X with resources R
  - Proofs are almost always to give an algorithm that meets the resource bounds
- Lower bounds
  - How do we show that something can't be done?

Theory of NP Completeness

The Universe



Polynomial Time

- P: Class of problems that can be solved in polynomial time
  - Corresponds with problems that can be solved efficiently in practice
  - Right class to work with "theoretically"

Decision Problems

- Theory developed in terms of yes/no problems
  - Independent set
    - Given a graph G and an integer K, does G have an independent set of size at least K
  - Network Flow
    - Given a graph G with edge capacities, a source vertex s, and sink vertex t, and an integer K, does the graph have flow function with value at least K

Definition of P

Decision problems with polynomial time algorithms

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's algorithm	34, 39	34, 51
PRIMES	Is x prime?	Agrawal, Kayal, Saxena (2002)	53	51
EDIT-DISTANCE	Is the edit distance between x and y less than 5?	Dynamic programming	niether neither	acgggt ttttta
LSOLVE	Is there a vector x that satisfies Ax = b?	Gaussian elimination	$\begin{bmatrix} 0 & 1 & 1 \\ 2 & 4 & -2 \\ 0 & 3 & 15 \end{bmatrix}, \begin{bmatrix} 4 \\ 2 \\ 36 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

What is NP?

- Problems solvable in non-deterministic polynomial time

- Problems where "yes" instances have polynomial time checkable certificates

Non-deterministic Computation

- Non-deterministic finite automata
  - Multiple different next states
  - Accept a string if some set of choices get to an accept state

- Non-deterministic computer
  - Add a non-deterministic GOTO statement (choose between multiple statements)
  - Accept if some computation reaches an accept state

Certificate examples

- Independent set of size K
  - The Independent Set
- Satifisfiable formula
  - Truth assignment to the variables
- Hamiltonian Circuit Problem
  - A cycle including all of the vertices
- K-coloring a graph
  - Assignment of colors to the vertices

Certifiers and Certificates: 3-Satisfiability

SAT: Does a given CNF formula have a satisfying formula

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

Certifier: Check that each clause has at least one true literal,

instance s

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

certificate t

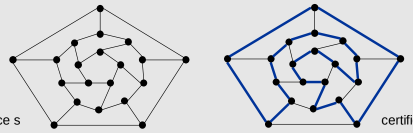
$$x_1 = 1,\ x_2 = 1,\ x_3 = 0,\ x_4 = 1$$

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.



Polynomial time reductions

- Y is Polynomial Time Reducible to X
  - Solve problem Y with a polynomial number of computation steps and a polynomial number of calls to a black box that solves X
  - Notations: $Y <_P X$

Composability Lemma

- If $X <_P Y$ and $Y <_P Z$ then $X <_P Z$

Lemmas

- Suppose $Y <_P X$. If X can be solved in polynomial time, then Y can be solved in polynomial time.
- Suppose $Y <_P X$. If Y cannot be solved in polynomial time, then X cannot be solved in polynomial time.

NP-Completeness

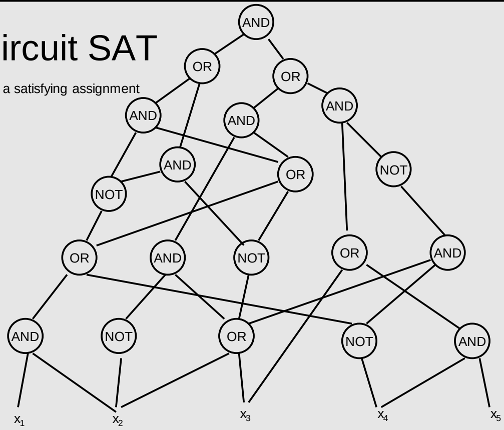
- A problem X is NP-complete if
  - X is in NP
  - For every Y in NP, $Y <_P X$
- X is a “hardest” problem in NP
- If X is NP-Complete, Z is in NP and $X <_P Z$
  - Then Z is NP-Complete

Cook’s Theorem

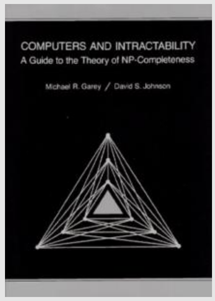
- The Circuit Satisfiability Problem is NP-Complete

Circuit SAT

Find a satisfying assignment



Garey and Johnson



History

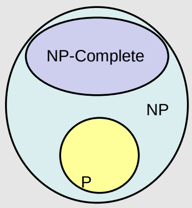

Jack Edmonds
- Identified NP

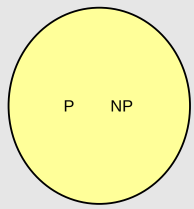

Steve Cook
- Cook’s Theorem – NP-Completeness


Dick Karp
- Identified the “standard” collection of NP-Complete Problems

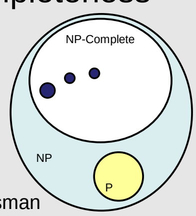

Leonid Levin
- Independent discovery of NP-Completeness in USSR

P vs. NP Question





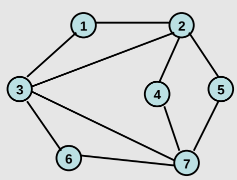
Populating the NP-Completeness Universe



- Circuit Sat $<_P$ 3-SAT
- 3-SAT $<_P$ Independent Set
- 3-SAT $<_P$ Vertex Cover
- Independent Set $<_P$ Clique
- 3-SAT $<_P$ Hamiltonian Circuit
- Hamiltonian Circuit $<_P$ Traveling Salesman
- 3-SAT $<_P$ Integer Linear Programming
- 3-SAT $<_P$ Graph Coloring
- 3-SAT $<_P$ Subset Sum
- Subset Sum $<_P$ Scheduling with Release times and deadlines

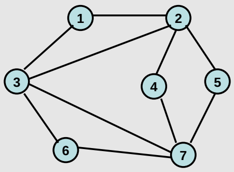
Sample Problems

- Independent Set
  - Graph G = (V, E), a subset S of the vertices is independent if there are no edges between vertices in S



Vertex Cover

- Vertex Cover
  - Graph G = (V, E), a subset S of the vertices is a vertex cover if every edge in E has at least one endpoint in S



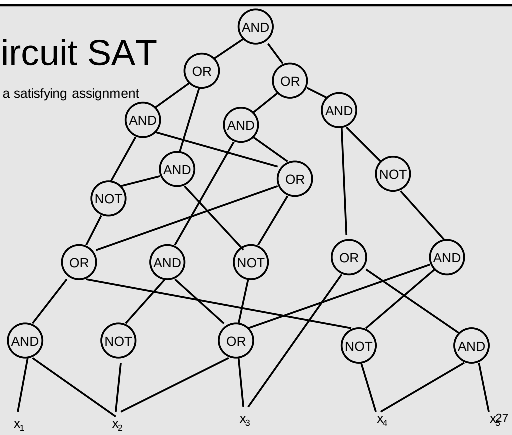
Cook's Theorem

- The Circuit Satisfiability Problem is NP-Complete

- Circuit Satisfiability
  - Given a boolean circuit, determine if there is an assignment of boolean values to the input to make the output true

Circuit SAT

Find a satisfying assignment



Proof of Cook's Theorem

- Reduce an arbitrary problem Y in NP to X
- Let A be a non-deterministic polynomial time algorithm for Y
- Convert A to a circuit, so that Y is a Yes instance iff and only if the circuit is satisfiable