CMPN302: Algorithms Design and Analysis

Lecture 10: NP-Completeness

Ahmed Hamdy

Computer Engineering Department

Cairo University

Fall 2017

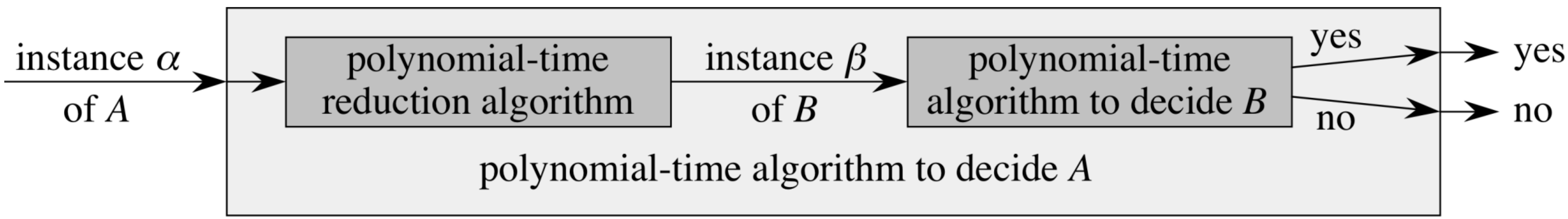
What is NP-complete?

- NP stands for "Non-deterministic Polynomial-time"
- All algorithms studied so far are *polynomial-time algorithms*, a.k.a $O(n^k)$
- Similar problems but P vs NP-complete:
  - – Shortest vs longest simple paths (not DAG)
  - – Euler tour vs Hamiltonian cycle
    - Euler tour: traverses each *edge* once, $O(E)$
    - Hamiltonian cycle: traverses each *vertex* once, NP-complete
  - – 2-CNF satisfiability ($(\overline{x_1} \lor x_2) \land (x_1 \lor x_3) \ldots$) vs. 3-CNF satisfiability ($(\overline{x_1} \lor x_2 \lor \overline{x_3}) \land (x_1 \lor x_3 \lor x_4) \ldots$)

Decision vs. optimization problems

- Optimization problem: solution achieves min/max
- Decision problem: solution is "yes" or "no"
- Decision problem related to (single-pair) shortest-path:
  - Does a path exist from u to v consisting of at most k edges?
- If an optimization problem is easy, its related decision problem is easy as well.
- If evidence exists that a decision problem is hard, means optimization problem is hard.

Reduction



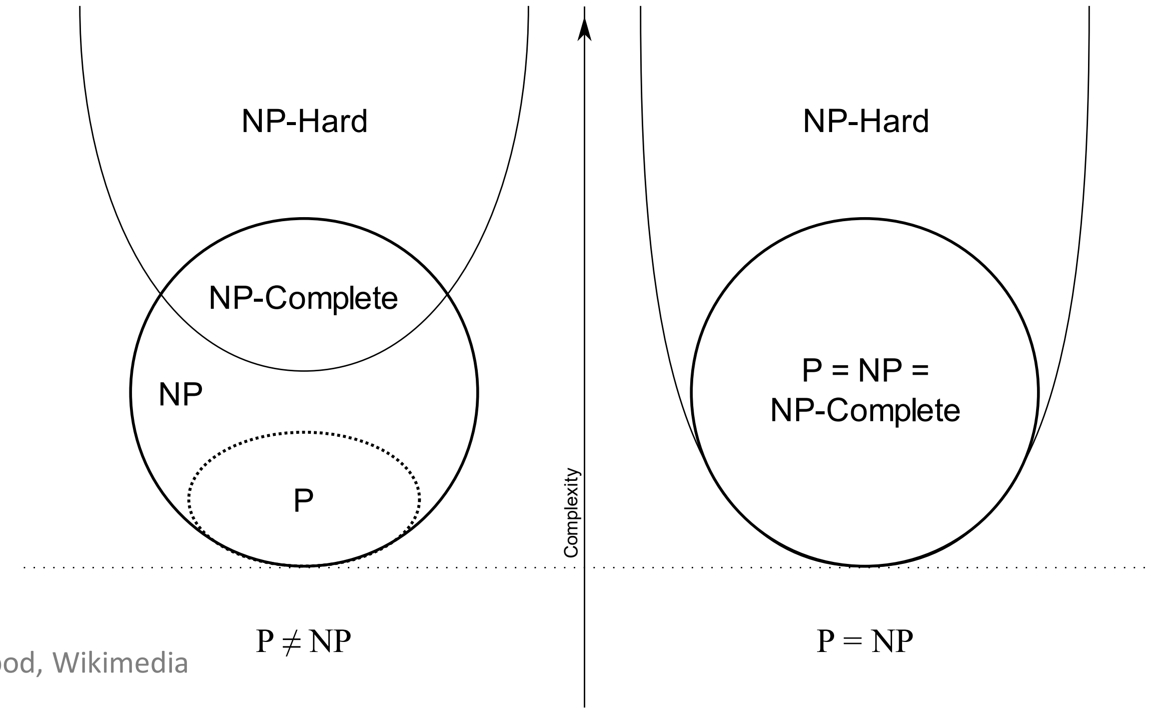
- Requirements:
  - Transformation takes polynomial time.
  - Answers are the same. Answer for A is “yes” if and only if the answer for B is also “yes.”
- If we have a problem to be classified, and another NP-complete one. Which to reduce to the other?

Polynomial-time verification

- Given a solution, even for an NP-complete problem. It can be verified in polynomial-time.
- Problems with such property are in *complexity class NP*
- Examples:
  - Verifying a solution for 3CNF-satisfiability
  - Verifying a solution for Hamiltonian cycle

Complexity classes

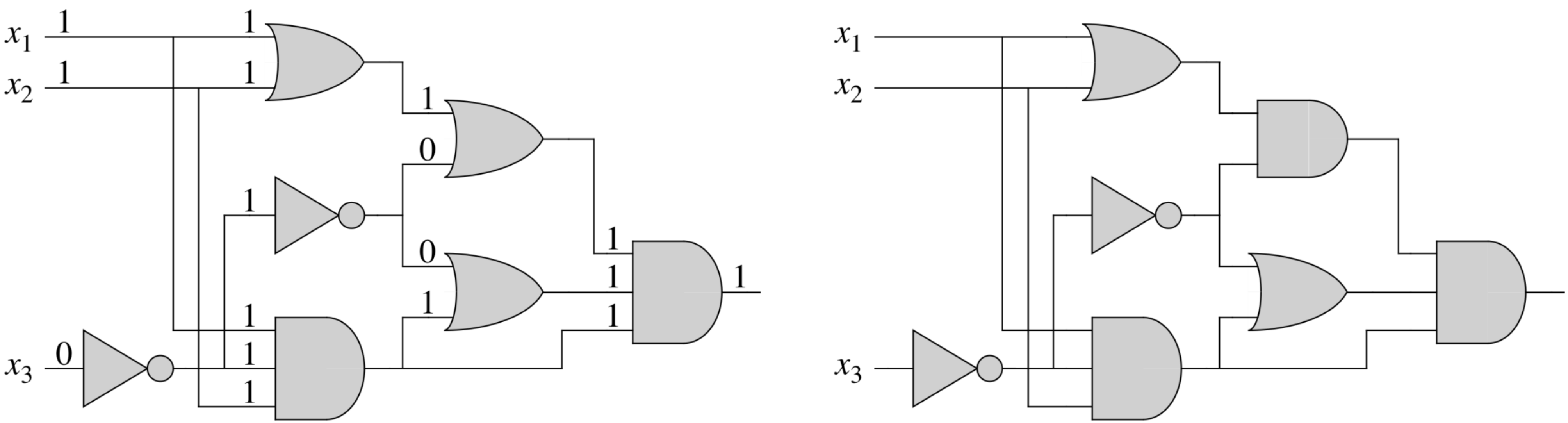
- P: class of decision problems that can be solved in polynomial time.
- NP: Class of decision problems which their solutions can be *verified* in polynomial time. $P \subseteq NP$.
- NP-hard: Class of decision problems which are at least as hard as the hardest problems in NP.
- NP-complete: Class of decision problems which contains the hardest problems in NP.



Courtesy of Behnam Esfahbod, Wikimedia

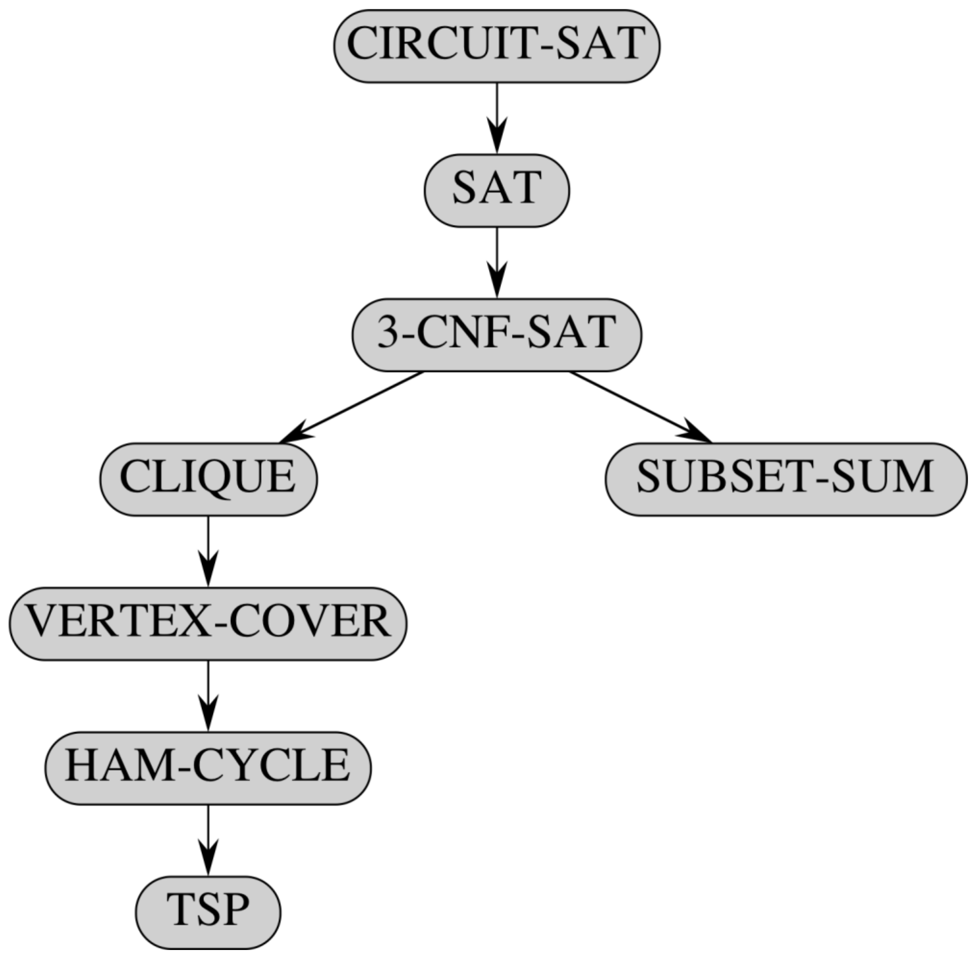
Circuit Satisfiability

- First problem proven to be NP-complete
- Will be used to prove all other problems to be NP-complete by reducibility



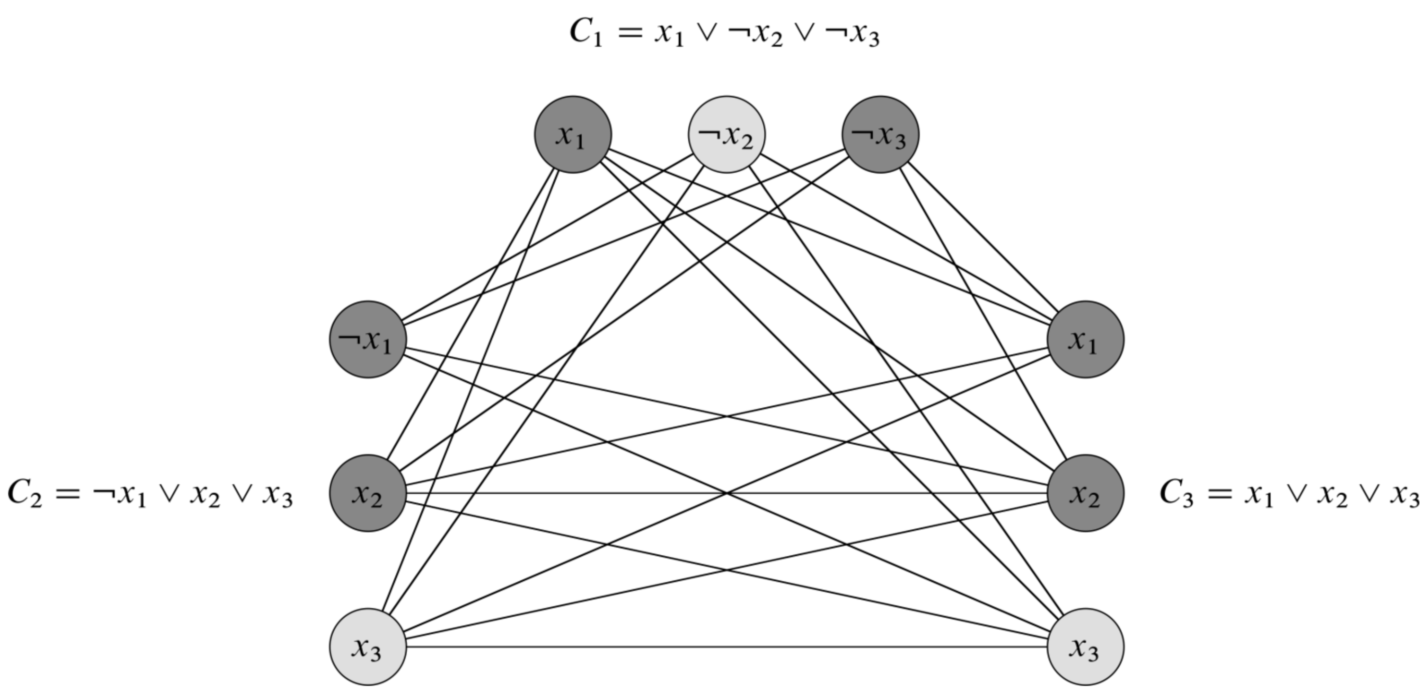
(a) (b)

Reducible problems



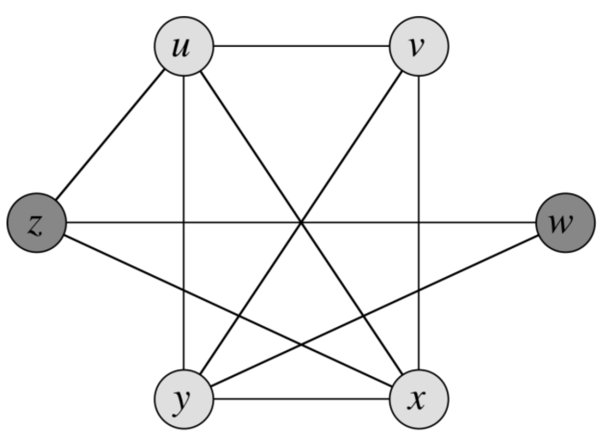
Clique problem

- Find a subset of vertices of a graph with maximum size such that all pairs of these vertices are connected by an edge.
- As a decision problem, ask whether a clique of size k exists in the graph.

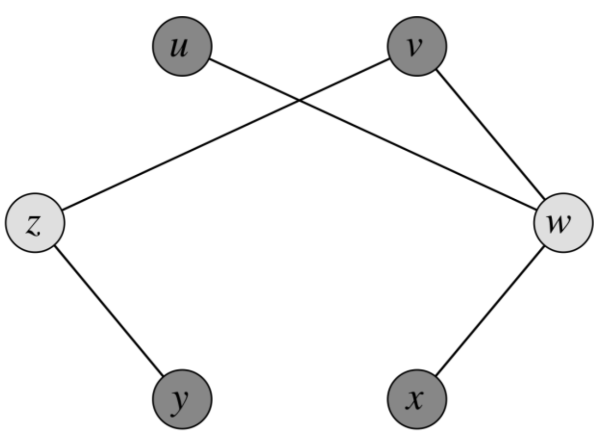


Vertex-cover problem

- Find a subset of vertices of a graph with minimum size such that every vertex “covers” its incident edge.
- As a decision problem, ask whether a vertex-cover with size k exists in the graph.



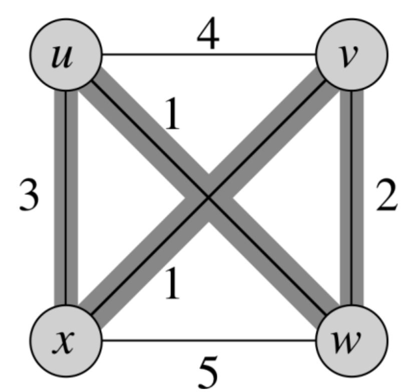
(a)



(b)

TSP

- Traveling Salesman Problem
- Find the minimum cost (sum of edges) to visit n-cities, each city visited exactly once and finishing with the first city.



To prove that TSP is NP-hard, we show that HAM-CYCLE $\leq_P$ TSP. Let $G = (V, E)$ be an instance of HAM-CYCLE. We construct an instance of TSP as follows. We form the complete graph $G' = (V, E')$, where $E' = \{(i, j) : i, j \in V$ and $i \neq j\}$, and we define the cost function c by

$$c(i, j) = \begin{cases} 0 & \text{if } (i, j) \in E , \\ 1 & \text{if } (i, j) \notin E . \end{cases}$$

Subset-sum problem

- Find a subset of integers picked from a set of positive integers such that their sum equals to a given value t.

		x_1	x_2	x_3	C_1	C_2	C_3	C_4
v_1	=	1	0	0	1	0	0	1
v_1'	=	1	0	0	0	1	1	0
v_2	=	0	1	0	0	0	0	1
v_2'	=	0	1	0	1	1	1	0
v_3	=	0	0	1	0	0	1	1
v_3'	=	0	0	1	1	1	0	0
s_1	=	0	0	0	1	0	0	0
s_1'	=	0	0	0	2	0	0	0
s_2	=	0	0	0	0	1	0	0
s_2'	=	0	0	0	0	2	0	0
s_3	=	0	0	0	0	0	1	0
s_3'	=	0	0	0	0	0	2	0
s_4	=	0	0	0	0	0	0	1
s_4'	=	0	0	0	0	0	0	2
t	=	1	1	1	4	4	4	4