CENG 218
Design and Analysis of Algorithms

Izmir Institute of Technology

Lecture 15: P, NP, NP-complete

Classifying problem complexity

- Most of the algorithms we studied in this course are *polynomial-time algorithms*, i.e. $O(n^k)$ worst case running time.

- Not all problems can be solved in polynomial time. Some of them are more ‘difficult’.

Class P

P is the class of problems that are solvable in polynomial time.
These problems are also called *tractable*.

Examples:
- Searching in a list
- Sorting the elements of a list
- Finding the minimum spanning tree of a graph

Class NP

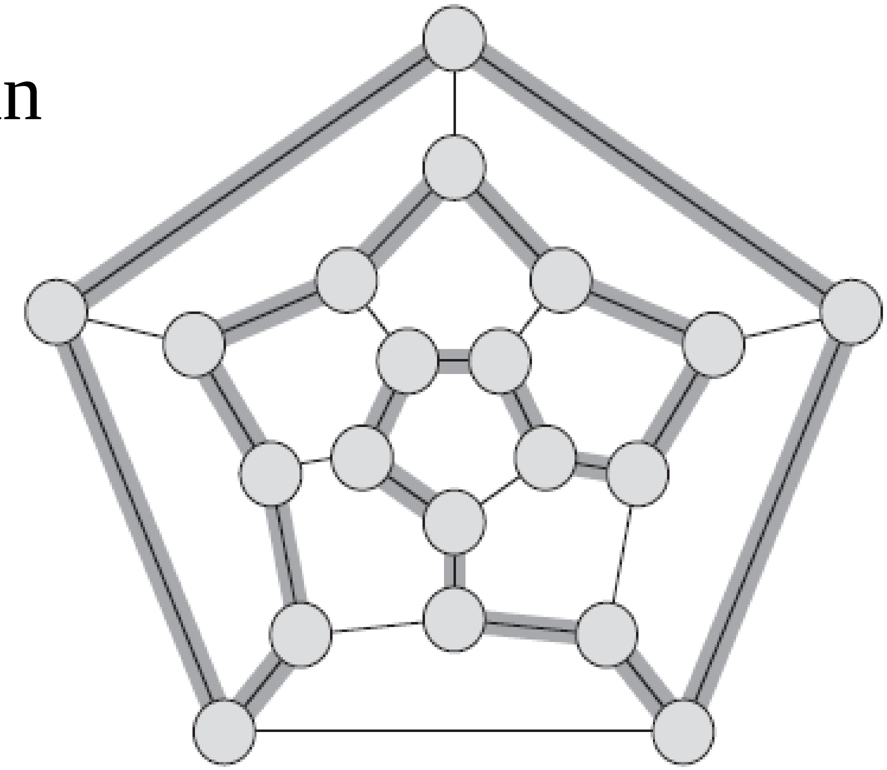
NP (*nondeterministic polynomial*) is the class of problems

- for which no polynomial-time algorithm has yet been discovered
- whose proposed solutions can be verified in polynomial time

i.e. Not solved in polynomial-time but a polynomial-time algorithm can check whether a proposed solution is correct or not.

Hamiltonian-cycle problem

A Hamiltonian cycle of an undirected graph *G* is a cycle that traverses each vertex in *G* exactly once.



We can define the ***hamiltonian-cycle problem***: ‘Does a graph *G* have a hamiltonian cycle?’

Hamiltonian-cycle problem

Solution to the ***hamiltonian-cycle problem*** was shown to be $\Omega(\sqrt{n}!)$ and that is $\Omega(2^{\sqrt{n}})$ which is not in polynomial time.

Verification algorithm for the Hamiltonian cycle:

When you are given a sequence of vertices,

i) Check every vertex in the graph is included.

ii) Check no vertex repeats.

iii) Each consecutive edge actually exists in the graph.

It takes $O(n^2)$ time to check if it is hamiltonian.

Therefore, this problem is in NP.

What problems are in NP?

All the problems in P can also be solved in this manner, so we have:

$$P \subseteq NP$$

What other problems are in NP?

- Euler tour. Is there a cycle that uses each edge exactly once? (P)
- Linear Programming (LP). Given a system of linear inequalities, find a solution. (P)

$$\begin{array}{rrrcr} 48x_0 & +16x_1 & +119x_2 & \leq & 88 \\ 5x_0 & + \ 4x_1 & + \ 35x_2 & \geq & 13 \\ 15x_0 & + \ 4x_1 & + \ 20x_2 & \geq & 23 \\ x_0 & , \ \ x_1 & , \ \ x_2 & \geq & 0 \end{array}$$

$$\begin{array}{rcl} x_0 & = & 1 \\ x_1 & = & 1 \\ x_2 & = & \frac{1}{5} \end{array}$$

variables are real numbers

What other problems are in NP?

- ILP. Given a system of linear inequalities, find a 0-1 solution. (NP).

$$\begin{array}{rcrcrcl} & & x_1 & + & x_2 & \geq & 1 \\ x_0 & & & + & x_2 & \geq & 1 \\ x_0 & + & x_1 & + & x_2 & \leq & 2 \end{array} \qquad \begin{array}{rcl} x_0 & = & 0 \\ x_1 & = & 1 \\ x_2 & = & 1 \end{array}$$

$\leftarrow$ variables are 0 or 1

- Satisfiability (SAT). Given a system of boolean equations, find a binary solution. (NP).

$$\begin{array}{rcrl} (x'_1 \text{ or } x'_2) & \text{and} & (x_0 \text{ or } x_2) & = \text{true} \\ (x_0 \text{ or } x_1) & \text{and} & (x_1 \text{ or } x'_2) & = \text{false} \\ (x_0 \text{ or } x_2) & \text{and} & (x'_0) & = \text{true} \end{array} \qquad \begin{array}{rl} x_0 & = \text{false} \\ x_1 & = \text{false} \\ x_2 & = \text{true} \end{array}$$

$\leftarrow$ variables are true or false

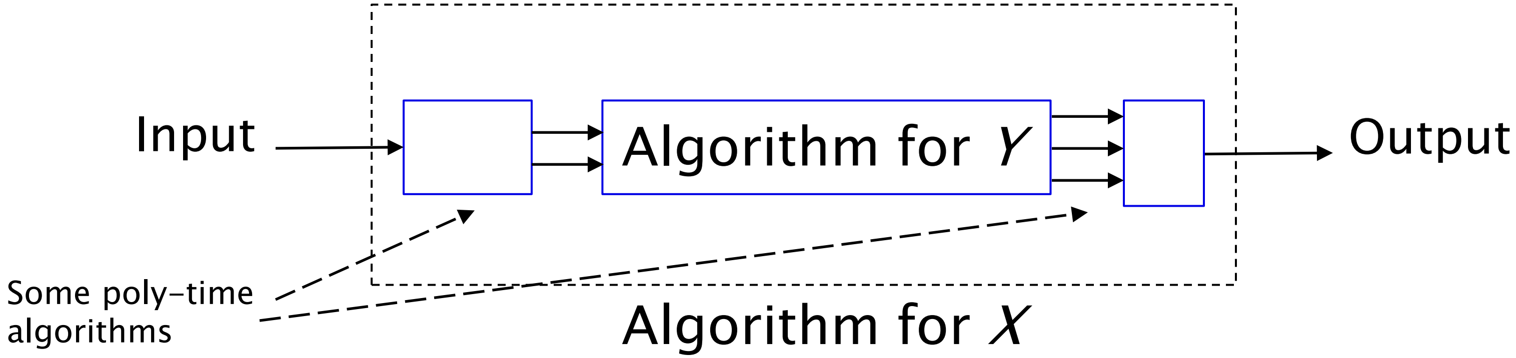
- Subset sum: Given a set of integers, is there a subset whose sum is zero? (NP) E.g. given set {-7, -3, -2, 5, 8}, the answer is yes since subset {-3, -2, 5} sums to zero. Brute-force solution: $O(2^n n)$, since there are 2^n subsets and, to check each subset takes at most n elements.

NP-Complete problems

A decision problem D is NP-complete if it is in NP and it's as 'hard' as any problem in NP. That is

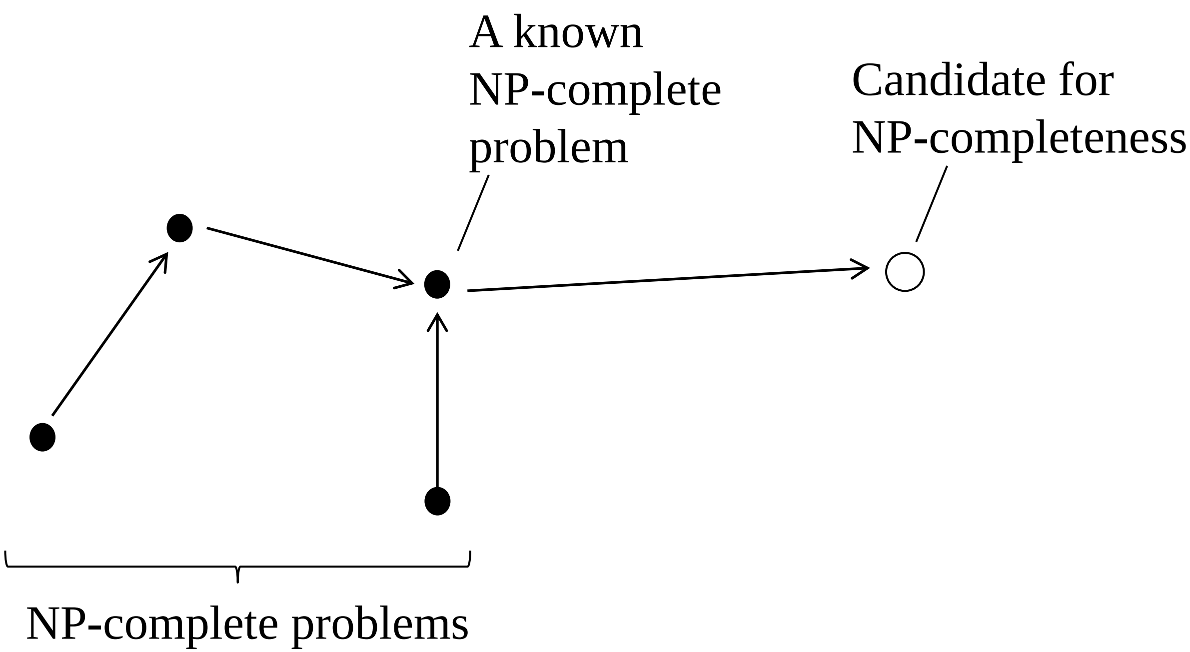
- D is in NP
- all problems in NP is polynomial-time reducible to D

Reduction: Problem X poly-time reduces to problem Y if X can be solved using the solution to Y plus some extra 'polynomial-time' work.



NP-Complete problems

Other NP-complete problems obtained through polynomial-time reductions from a known *NP*-complete problem.



NP-Complete problems

by Karp, Cook, Levin.

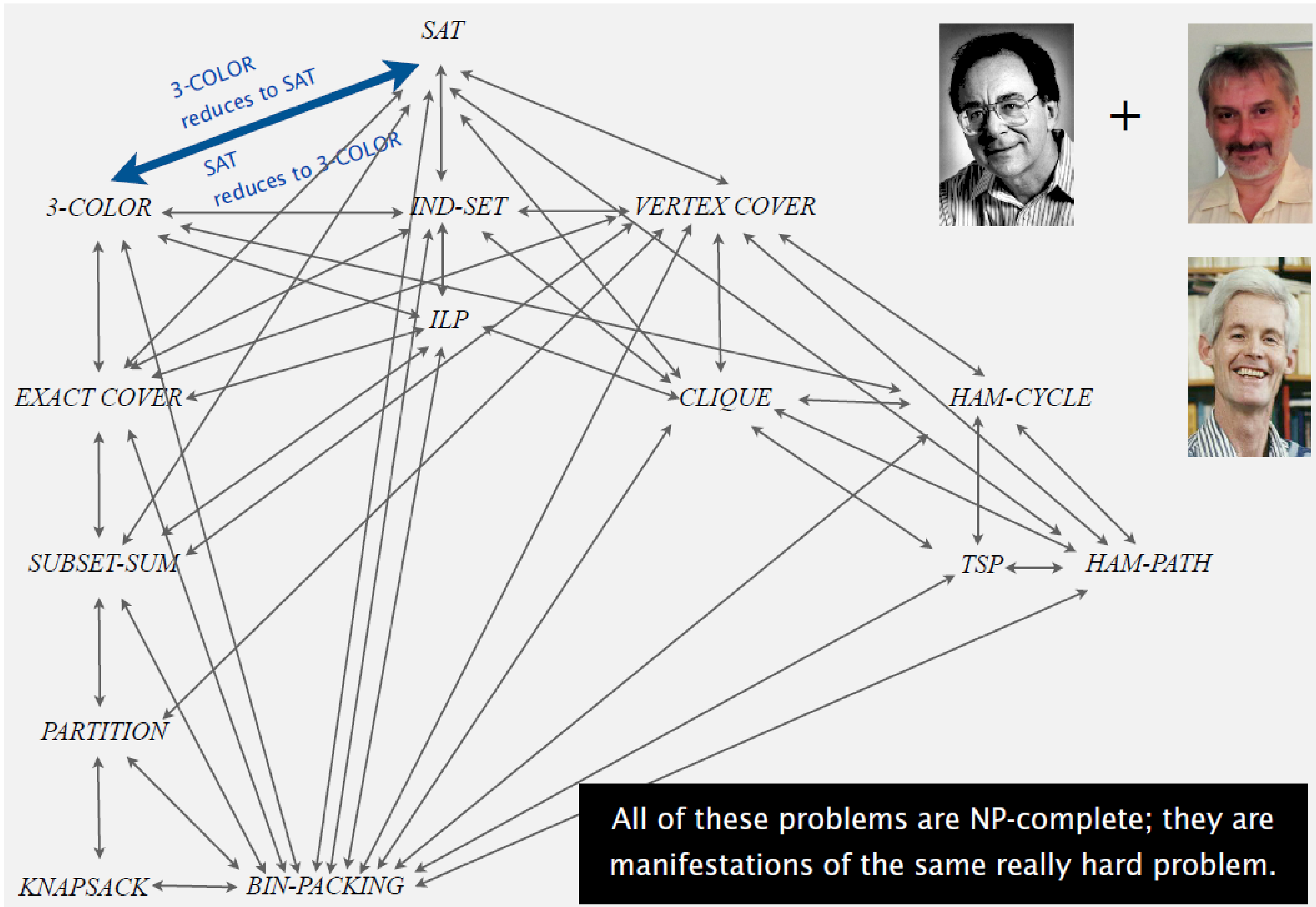


Figure source: Algorithms, 4th edition by Sedgewick and Wayne

P = NP ?

- P = NP would imply that all NP-complete problems, could be solved in polynomial time.
- If a polynomial-time algorithm for just one NP-complete problem is discovered, then every problem in NP can be solved in polynomial time, i.e., P = NP.
- Most researchers believe that P $\neq$ NP , i.e. P is a proper subset of NP.
- However, it is not yet proven that NP-complete problems are intractable.

The End

Textbook Section 34.1, 34.2.