Algorithms and Computation

(grad course)

Lecture 6: NP-Complete Problems

Instructor: Hossein Jowhari

jowhari@kntu.ac.ir

Department of Computer Science and Statistics
Faculty of Mathematics
K. N. Toosi University of Technology

Fall 2021

NP-Complete Problem: Definition

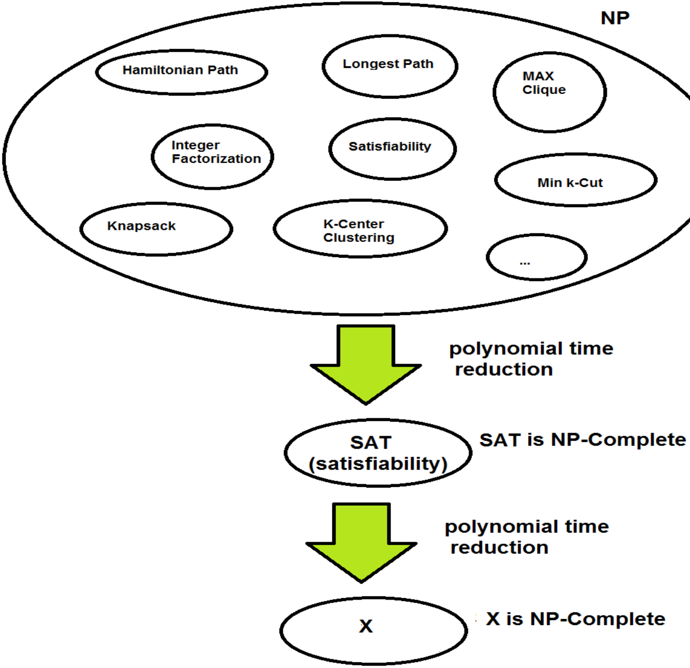
Definition: The problem X is NP-Complete if and only if

- $X \in$ NP
- All problems in NP are *polynomial-time reducible* to X.

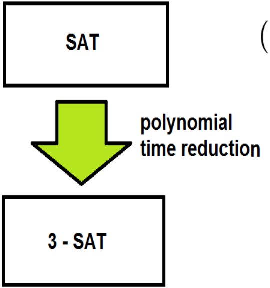
Cook/Levin Theorem: SAT is NP-Complete.

Alternative Definition: The problem X is NP-Complete if and only if

- $X \in$ NP
- There is an NP-Complete problem Y where Y is *polynomial-time reducible* to X. ($Y \leq_p X$)


NP
Hamiltonian Path
Longest Path
MAX Clique
Integer Factorization
Satisfiability
Min k-Cut
Knapsack
K-Center Clustering
...
polynomial time reduction
SAT (satisfiability)
SAT is NP-Complete
polynomial time reduction
X
X is NP-Complete

3-SAT is NP-Complete



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

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

Reduction Idea: Given a SAT formulae ϕ, we create a 3-SAT formulae ϕ' in polynomial time where ϕ' is satisfiable if and only if ϕ is satisfiable.

Challenge: We have clauses in ϕ with length < 3 or > 3.

Consider a clause $C = (x)$ with length 1. Here x is a literal.

$$\phi = \underbrace{(x)}_{C} \wedge (\ldots) \wedge \ldots \wedge$$

We add the new variables y and z and add replace C with $(x \vee y \vee z) \wedge (x \vee \overline{y} \vee \overline{z}) \wedge (x \vee \overline{y} \vee z) \wedge (x \vee y \vee \overline{z})$.

$$\phi' = (x \vee y \vee z) \wedge (x \vee \overline{y} \vee \overline{z}) \wedge (x \vee \overline{y} \vee z) \wedge (x \vee y \vee \overline{z}) \wedge (\ldots) \wedge \ldots \wedge$$

Now consider a clause $C = (x \vee y)$ with length 2. Similarly we replace C with $(x \vee y \vee z) \wedge (x \vee y \vee \overline{z})$.

Now consider a clause $C = (x_1 \vee x_2 \vee x_3 \vee \ldots \vee x_k)$ with length $k > 3$.

We can say

$$C \equiv (x_1 \vee x_2 \vee z) \wedge (z \Leftrightarrow x_3 \vee \ldots \vee x_k)$$

On the other hand we have

$$(z \Leftrightarrow x_3 \vee \ldots \vee x_k) \equiv (z \Rightarrow x_3 \vee \ldots \vee x_k) \wedge (x_3 \vee \ldots \vee x_k \Rightarrow z)$$

$$\equiv (\overline{z} \vee x_3 \vee \ldots \vee x_k) \wedge (\overline{x_3 \vee \ldots \vee x_k} \vee z)$$

$$\equiv (\overline{z} \vee x_3 \vee \ldots \vee x_k) \wedge ((\overline{x_3} \wedge \ldots \wedge \overline{x_k}) \vee z)$$

$$\equiv (\overline{z} \vee x_3 \vee \ldots \vee x_k) \wedge ((\overline{x_3} \vee z) \wedge \ldots \wedge (\overline{x_k} \vee z))$$

Given the above procedure we have replaced a clause C containing k literals with 1 clause of length 3, $k-2$ clauses of length 2 and a clause with length $k-1$.

Let $T(k)$ be the number of clauses of length 3 created after continuing the transformation procedure. We have

$$T(k) = \begin{cases} 1 + 2(k-2) + T(k-1), & \text{if } k > 3 \\ 1, & \text{if } k = 3 \\ 2, & \text{if } k = 2 \\ 4, & \text{if } k = 1 \end{cases}$$

$$T(k) = O(k^2)$$

Observation: Let m be the length of the formulae ϕ. The size of the final 3-SAT formulae ϕ' after the transformation is $O(m^2)$.

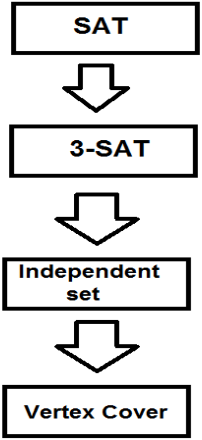
Conclusion: SAT $\leq_p$ 3-SAT.

Theorem: 3-SAT is NP-Complete.

From the previous lecture: 3-SAT $\leq_p$ Independent Set.

Conclusion: Independent Set is NP-Complete.

Conclusion: Vertex Cover is NP-Complete (since Independent Set $\leq_p$ Vertex Cover.)



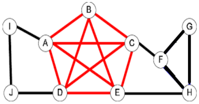
General Strategy for Proving NP-Completeness

Given a new problem X, to show that X is NP-Complete one general strategy is the following.

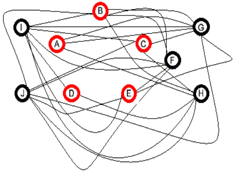
- Prove that $X \in NP$.
- Choose a problem Y that is known to be NP-Complete.
- Prove that $Y \leq_p X$.

MAX-Clique is NP-Complete

MAX-Clique problem: Given an undirected graph $G = (V, E)$ and number t, does G contain a complete graph K_t as a subgraph?



Graph G

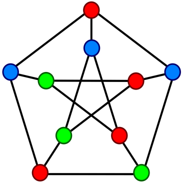


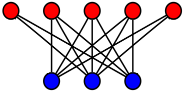
Complement of G

- MAX-Clique $\in NP$
- Independent Set $\leq_p$ MAX-Clique

Graph Coloring Problem

k-**Coloring problem**: Given an undirected graph $G = (V, E)$ and number k, can G be colored using k colors?

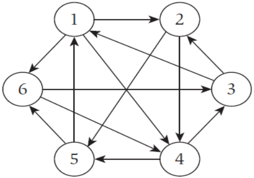




- 2-coloring $\in P$
- 3-SAT $\leq_p$ 3-coloring
- 3-coloring is NP-Complete

Hamiltonian-Cycle is NP-Complete

Hamiltonian-Cycle Problem: Given a directed graph $G = (V, E)$, does G has a cycle that visits every vertex in G exactly once?



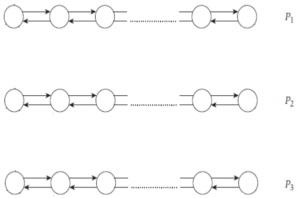
Observation: Hamiltonian-Cycle $\in NP$. Given a cycle in G it is straightforward to check if it is Hamiltonian or not in polynomial time.

Lemma: 3-SAT $\leq_p$ Hamiltonian-Cycle

Given a 3-SAT formulae ϕ we create a directed graph G_ϕ in polynomial time such that G_ϕ has a Hamiltonian cycle if and only if ϕ is satisfiable.

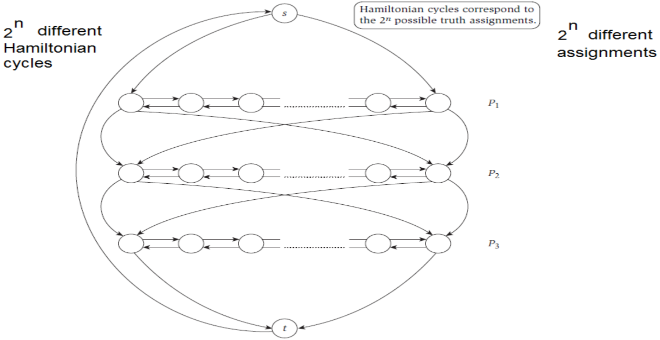
First step: Suppose the given formulae ϕ is defined on n variables and has k clauses. We first create a directed graph G on $(3k+3)n+2$ nodes as follows.

- Corresponding to each variable x_i, we put a bi-directed path P_i in G. Each path P_i has $3k+3$ nodes.



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

- Now we add two nodes s and t and connect the endpoints of the paths as shown in the figure.



Observation: Each Hamiltonian cycle in G corresponds to a different assignment of the variables $x_1, \ldots, x_n$.

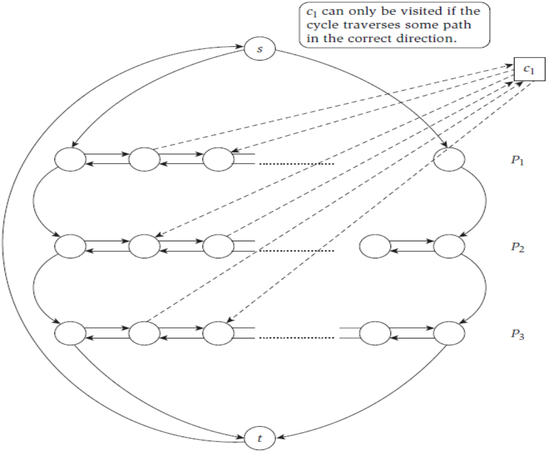
If we enter the path P_i from the left end, we interpret it as $x_i = True$, otherwise if we enter the path P_i from the right end, we interpret it as $x_i = False$.

- Now we need to impose the constraints defined by the clauses.

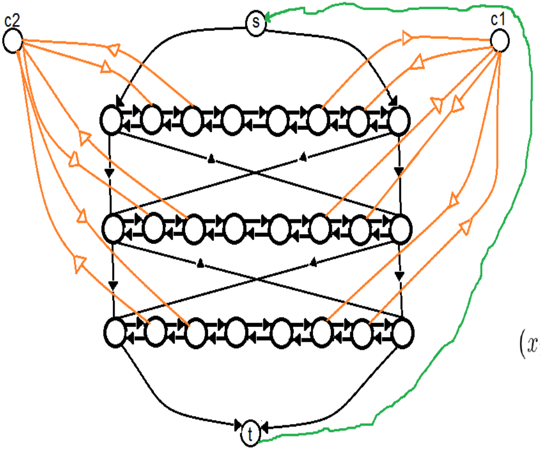
  Consider a clause like $C_1 = (x_1 \vee \overline{x_2} \vee x_3)$. To satisfy this clause, one way would be to have $x_1 = True$. The other ways is having $x_2 = False$ or $x_3 = True$.

  In the language of the graph that we constructed, having $x_1 = True$ means the Hamiltonian cycle should enter the path P_1 from the left end. Similarly $x_2 = False$ means the Hamiltonian cycle should go over path P_2 from right to left.

  To impose the constraint defined by the clause C_1, we add a node c_1 and put edges between c_1 and the path as follows.



There are three ways to have c_1 in the Hamiltonian cycle. One way is to traverse P_1 from left to right and meet c_1 on the way. The second way is to traverse P_2 from right to left and meet c_1 on the way. The third ways is to traverse P_3 from left to right and meet c_1 on the way.

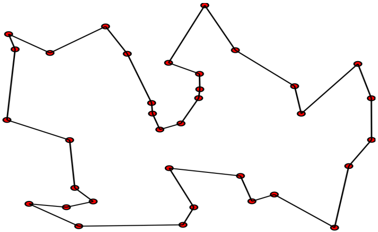


c1 c2

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

Traveling Salesman Problem is NP-Complete

TSP: Given a set of n cities C and distance function $d : C \times C \to \mathbb{R}^+$ and a number t, is there a tour of the cities that visits each city exactly once and its length is less than t?



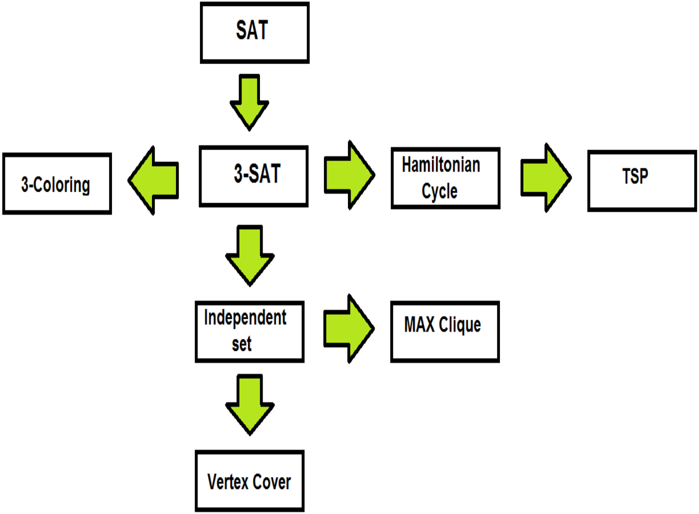
- TSP $\in NP$
- Hamiltonian Cycle $\leq_p$ TSP

Given a directed graph $G = (V, E)$, we define an instance of the Traveling Salesman Problem as follows. For each vertex $v_i \in V$, we add a city c_i. We define $d(c_i, c_j) = 1$ if $(i, j) \in E$ otherwise we let $d(c_i, c_j) = 2$.

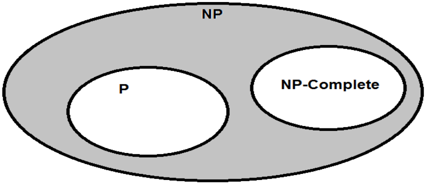
Claim: There is a tour of the cities with length n if and only if G has a Hamiltonian cycle.

Observation: Note that the distance function above is asymmetric. One can show that the TSP problem remains NP-Complete even if the distance function is symmetric and satisfy the triangle inequality. (Show that Hamiltonian cycle problem is NP-Complete for undirected graphs.)

Reduction Map



Possible state of the set NP



Could there be sets in the gray area? We have examples of problems that are in NP and do not have polynomial time algorithms. At the same time, if they are shown to be NP-Complete, it would mean $NP \subseteq DTIME(2^{o(n)})$ which is unlikely. Some examples of this kind include:

- Integer Factorization
- Graph Isomorphism