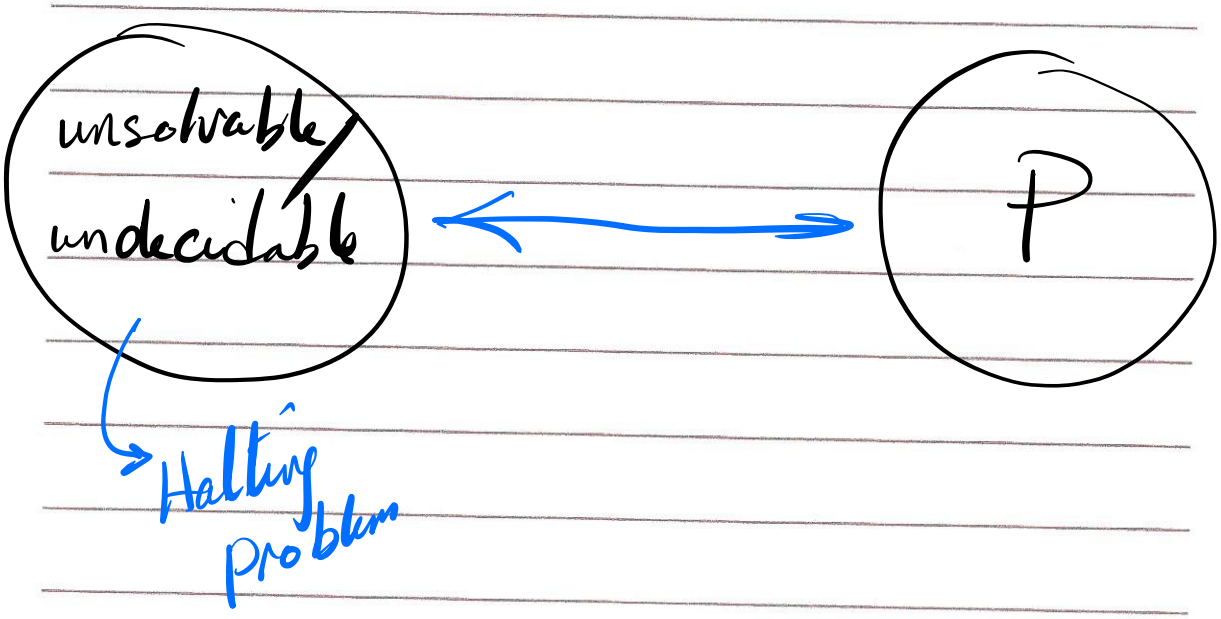
Computational Tractability



Plan: Explore the space of computationally hard problems to arrive at a mathematical characterization of a large class of them.

Technique: Compare relative difficulty of different problems.

Loose definition: If problem X is at least as hard as problem Y, it means that if we could solve X, we could also solve Y.

Formal definition:

$Y \leq_p X$ (Y is polynomial time reducible to X)

if Y can be solved using a polynomial number of standard computational steps plus a polynomial number of calls to a blackbox that solves X.

Suppose $Y \leq_p X$, if X can be solved in polynomial time, then Y can be solved in polynomial time.

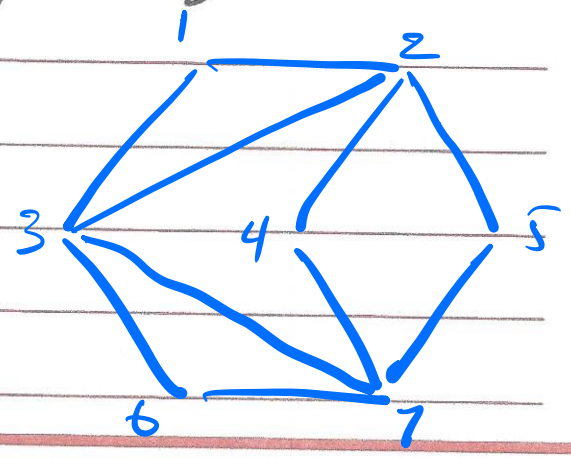
Suppose $Y \leq_p X$ if Y cannot be solved in polynomial time, then X cannot be solved in polyn. time.

Independent Set

Def. In a graph $G = (V, E)$, we say that a set of nodes $S \subseteq V$ is "independent" if no two nodes in S are joined by an edge.

$\{1, 4, 5, 6\}$

$\{1\}, \{4, 3\}$



Independent set problem

- Find the largest independent set in graph G.

(optimization version)

- Given a graph G and a no. k does G contain an indep set of size at least k?

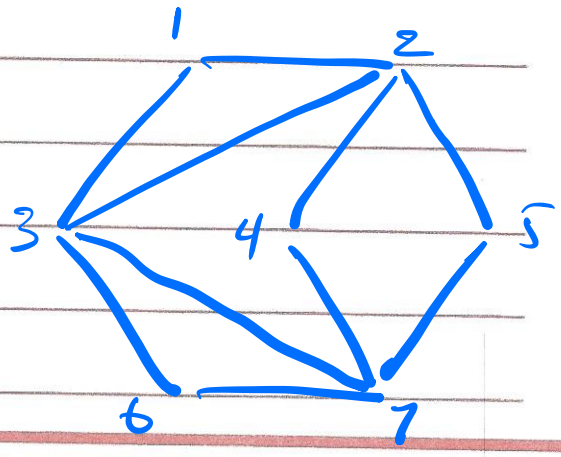
(decision version)

Vertex Cover

Def. Given a graph $G = (V, E)$, we say that a set of nodes $S \subseteq V$ is a vertex cover if every edge in E has at least one end in S.

$\{2, 3, 7\}$

$\{1, 2, 3, 4, 5, 6, 7\}$



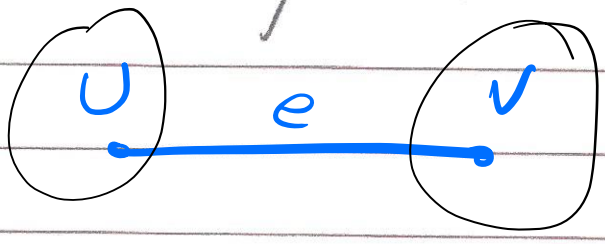
Vertex Cover problem

- Find the smallest vertex cover set in G. (opt. version)

- Given a graph G and a no. k, does G have a vertex cover of size at most k? (decision version)

FACT: Let $G=(V,E)$ be a graph, then S is an independent set if and only if its complement $V-S$ is a vertex cover set.

Proof: A) First suppose that S is an independent set



1- U is in S and V is not
⇒ V-S will have V and not U.

2- V is in S and U is not
⇒ V-S will have U and not V

3- Neither U nor V is in S
⇒ V-S will have both V & U

B- Suppose that V-S is a vertex Cover ....

Claim: Indep. set $\leq_p$ vertex cover

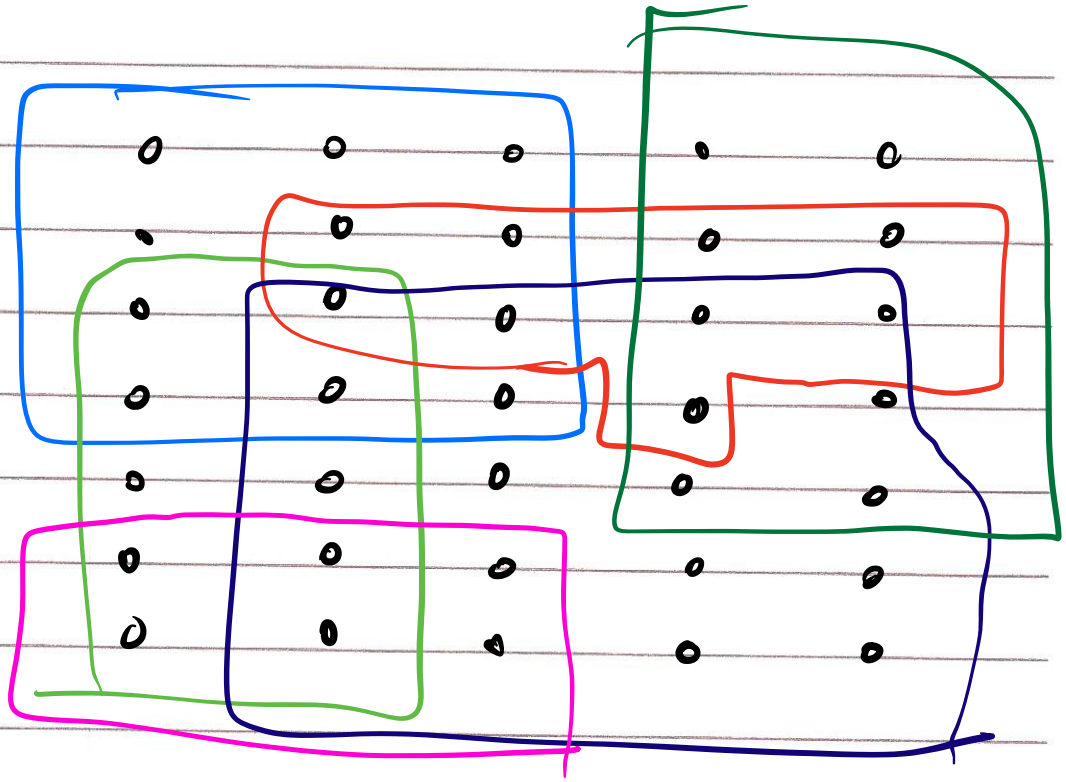
Proof: If we have a blackbox to solve vertex cover, we can decide if G has an independent set of size at least k, by asking the blackbox if G has a vertex cover of size at most $n-k$.

Claim: Vertex Cover $\leq_p$ Indep set

Proof: If we have a blackbox to solve independent set, we can decide if G has a vertex cover set of size at most k, by asking the blackbox if G has an indep set of size at least $n-k$.

Set Cover Problem

Given a set U of n elements, a collection $S_1, S_2, \ldots S_m$ of subsets of U, and a number k, does there exist a collection of at most k of these sets whose union is equal to all of U.



Claim: Vertex Cover $\leq_p$ Set Cover

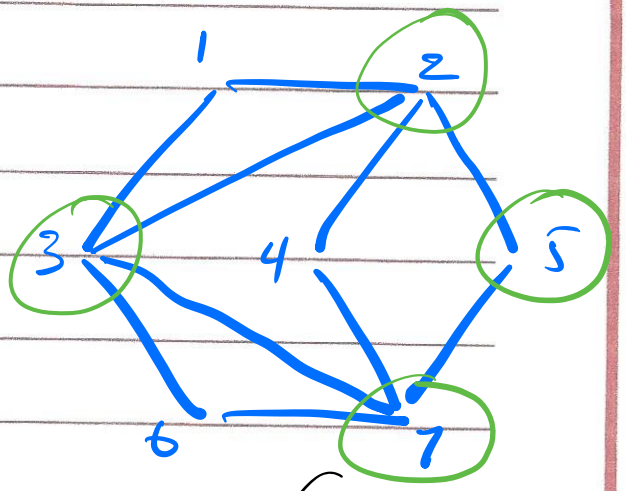
$S_1 = \{(1,2),(1,3)\}$

$S_2 = \{(1,2),(2,3),(2,4),(2,5)\}$

S_3

S_5

S_7



G, k

k

Need to show that G has a vertex cover of size k, iff the corresponding set cover instance has k sets whose union ~~is~~ contains ~~equivalent to~~ all edges in G.

Proof:

A) If I have a vertex cover set of size k in G, I can find a collection of k sets whose union contains all edges in G.

B) If I have k sets whose union contains all edges in G, I can find a vertex cover set of size k in G.

Reduction Using Gadgets

- Given n Boolean variables $x_1 \dots x_n$, a clause is a disjunction of terms $t_1 \vee t_2 \vee \dots \vee t_\ell$ where $t_i \in \{x_1 \dots x_n, \bar{x}_1 \dots \bar{x}_n\}$

- A truth assignment for X is an assignment of values 0 or 1 to each x_i:

$(x_1 \vee \bar{x}_2)$

$x_1 = 0, x_2 = 0$ ✓

$x_1 = 1, x_2 = 0$ ✓

$x_1 = 0, x_2 = 1$ ✗

- An assignment satisfies a clause C if it causes C to evaluate to 1.

- An assignment satisfies a collection of clauses if

$C_1 \wedge C_2 \wedge \dots C_k$

evaluates to 1.

ex. $(x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_3) \wedge (x_2 \vee \bar{x}_3)$

$x_1 = 1, x_2 = 1, x_3 = 1$ ✗

$x_1 = 0, x_2 = 0, x_3 = 0$ ✓

$x_1 = 1, x_2 = 0, x_3 = 0$ ✓

Problem Statement: Given a set of clauses $C_1 \ldots C_k$ over a set of variables $X = \{x_1 \ldots x_n\}$ does there exist a satisfying truth assignment?

Satisfiability Problem (SAT) (general form)

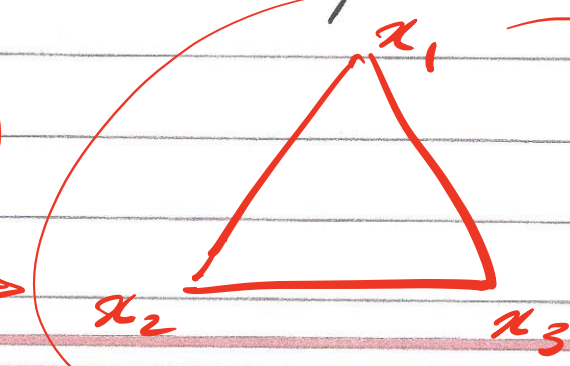
Problem statement: Given a set of clauses $C_1 \ldots C_k$ each of length 3 over a set of variables $X = \{x_1 \ldots x_n\}$ does there exist a satisfying truth assignment?

3-SAT

Claim: $3SAT \leq_p$ Independent Set

Plan: Given an instance of 3SAT with k clauses, build a graph G that has an indep. set of size k iff the 3SAT instance is satisfiable.

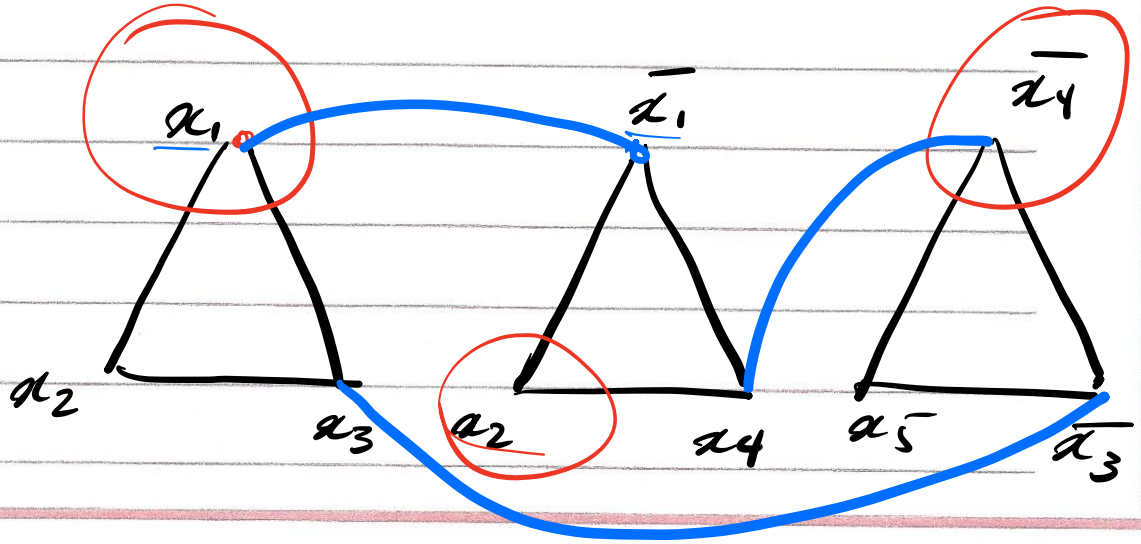
$(x_1 \vee x_2 \vee x_3)$



ex. $C_1 = (x_1 \vee x_2 \vee x_3)$

$C_2 = (\bar{x}_1 \vee x_2 \vee x_4)$

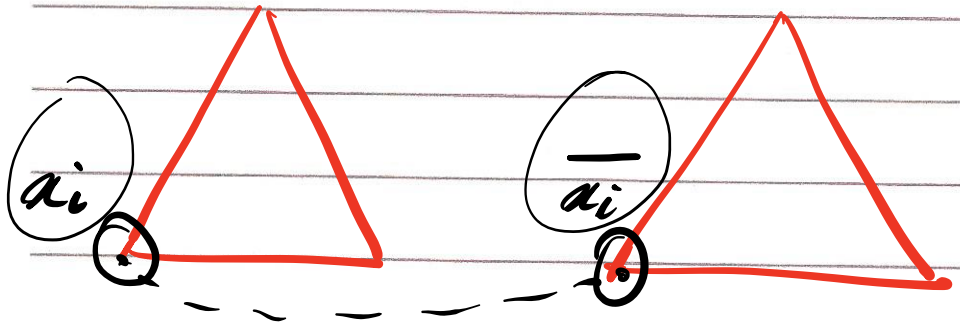
$C_3 = (\bar{x}_4 \vee x_5 \vee \bar{x}_3)$



Claim: The 3-SAT instance is satisfiable iff the graph G has an independent set of size k.

Proof: A) If the 3-SAT instance is satisfiable, then there is at least one node labeled per triangle that evaluates to 1.

Let S be a set containing one such node from each triangle



B) Suppose G has an independent set S of size at least k.

if x_i appears as a label in S then set x_i to 1

if $\bar{x}_i$ appears as a label in S then set x_i to 0

if neither x_i nor $\bar{x}_i$ appear as a label in S, then set x_i to 0 or 1

Efficient Certification

To show efficient certification:

1- Polynomial length certificate

2- Polynomial time certifier

Efficient certification

3-SAT Certificate t is an assignment of truth values to variables (x_i)

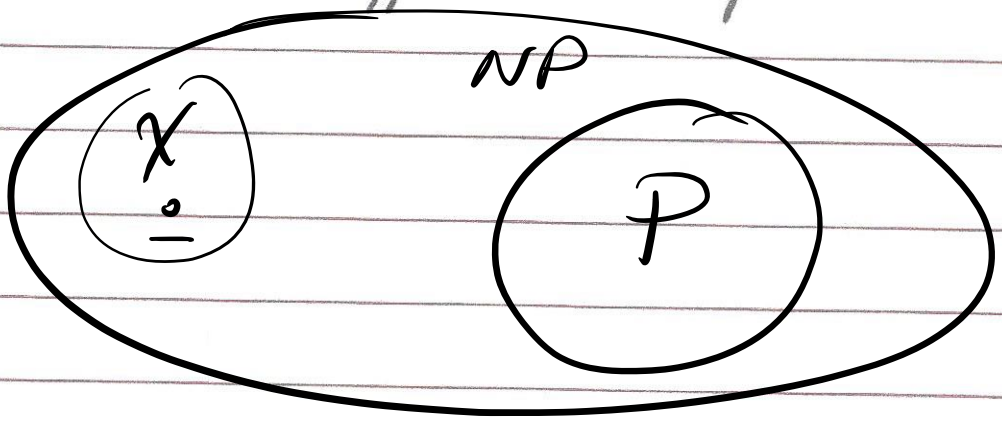
Certifier: evaluate the clauses. If all of them evaluate to 1 then it answers yes.

Indep set Certificate t is a set of nodes of size at least k in G.

Certifier: check each edge to make sure no edges have both ends in the set
check size of the set $\geq k$
no repeating nodes

decision

Class NP is the set of all problems for which there exists an efficient certifier



$NP \stackrel{?}{=} P$ we don't know!

if $X \in NP$ and for all $Y \in NP$ $Y \leq_p X$, then X is the hardest problem in NP.

3-SAT has been proven to be the hardest problem in NP

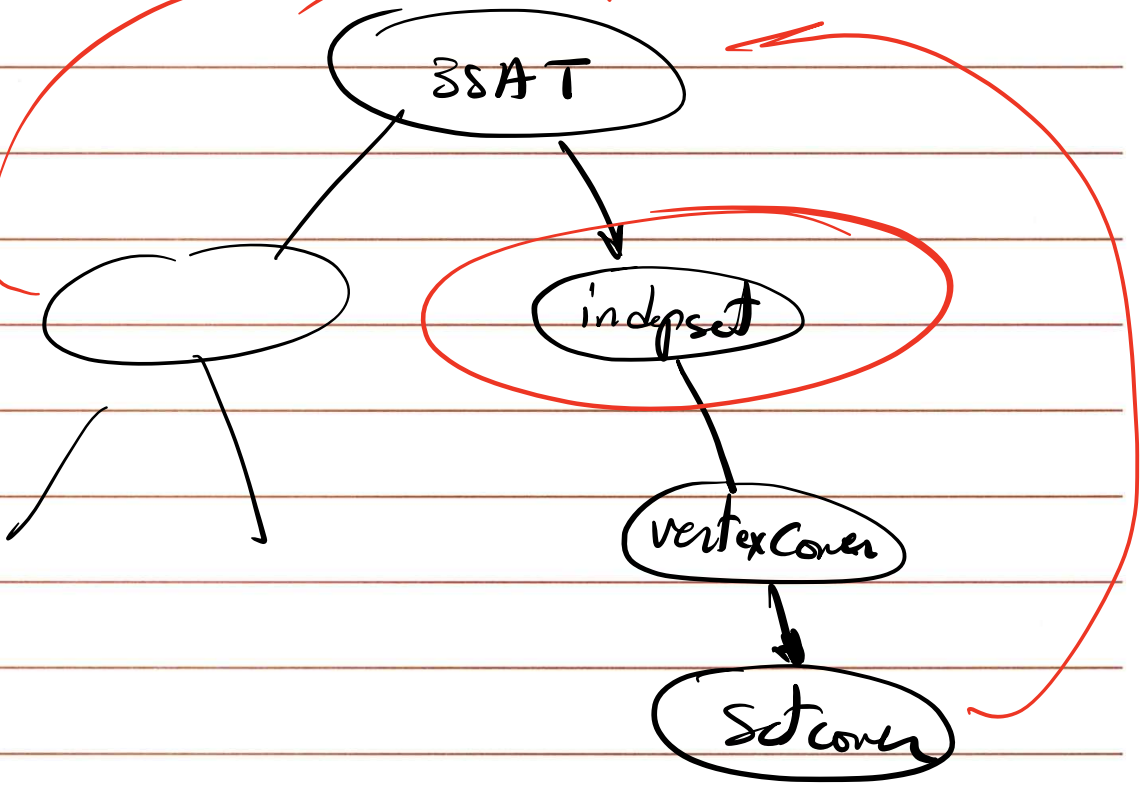
Such a problem is called NP-complete

Transitivity

if $Z \leq_p Y$ and $Y \leq_p X$

then $Z \leq_p X$

3SAT $\leq_p$ indep set $\leq_p$ vertex cover $\leq_p$ Set Cover



Basic Strategy to prove a problem X is NP-complete

1- Prove $X \in NP$

2- Choose a problem Y that is known to be NP-complete

3- Prove $Y \leq_p X$ ←

NP-hard is the class of problems that are at least as hard as NP-Complete problems.