Analysis and Design of Algorithms

NUS
National University of Singapore

Algorithms
CS3230

Tutorial

Week 13

NP class

Yes

No instance

X : any decision problem

I : any (input) instance of X

Efficient certifier for X :

A <u>polynomial time</u> algorithm A with output {yes,no}

- **Input** : (I, s)

Proposed solution

- **Behavior**: There is a polynomial function p such that I is yes-instance of X **if and only if** <u>there exists</u> a string s with $|s| \leq p(|I|)$ such that A outputs yes on input (I, s).

NP class

Definition (**NP**):

The set of all <u>decision</u> problems which have **efficient certifier**.

NP : “Non-deterministic polynomial time”

Definition (**P**):

The set of all decision problems which have **efficient** (poly-time) algorithm.

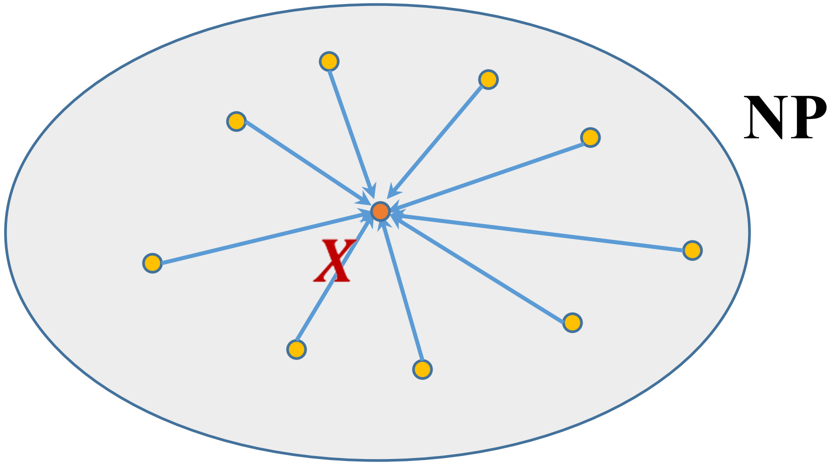
Is there any Relation between **P** and **NP** ?

NP-complete

If X is not known to be in **NP**, then we say X is just **NP-hard**

- A problem X in **NP** class is **NP-complete** if for every $A \in$ **NP**

$$A \leq_P X$$



Circuit Satisfiability $\leq_{\boldsymbol{P}}$ CNF-SAT $\leq_{\boldsymbol{P}}$ 3-SAT

So **3-SAT** is **NP**-complete

Question 1

NUS National University of Singapore

Which of the following imply P=NP?

1. There is a problem in P that is also in NP-COMPLETE.
2. There is a problem in P that is also in NP.
3. There is a problem in NP that is also in NP-HARD.

Question 2

L is an NP problem. One day, Mr Oh proves that it is impossible to find an algorithm to solve L in polynomial time. After this big event, which of the following classes of problems are now **known** to be **not solvable** in polynomial time?

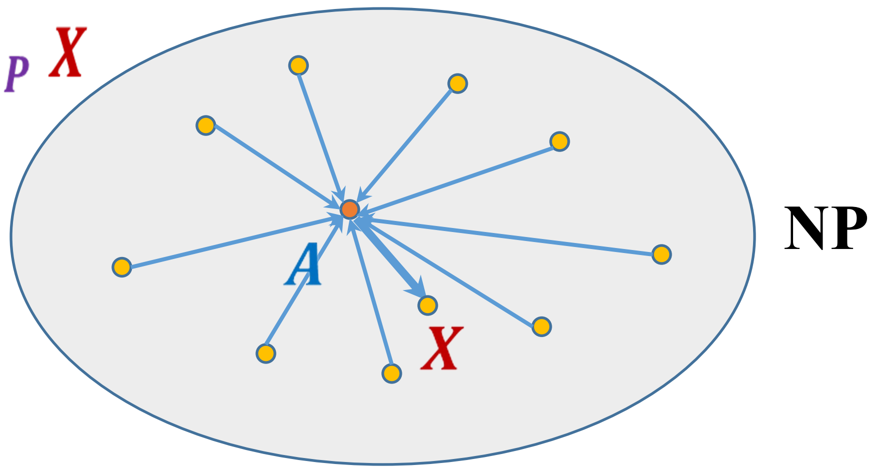
---

1. All NP-HARD problems.
2. All NP problems
3. All NP-COMPLETE problems.

How to show a problem to be NP-complete ?

Let X be a problem which we wish to show to be **NP**-complete

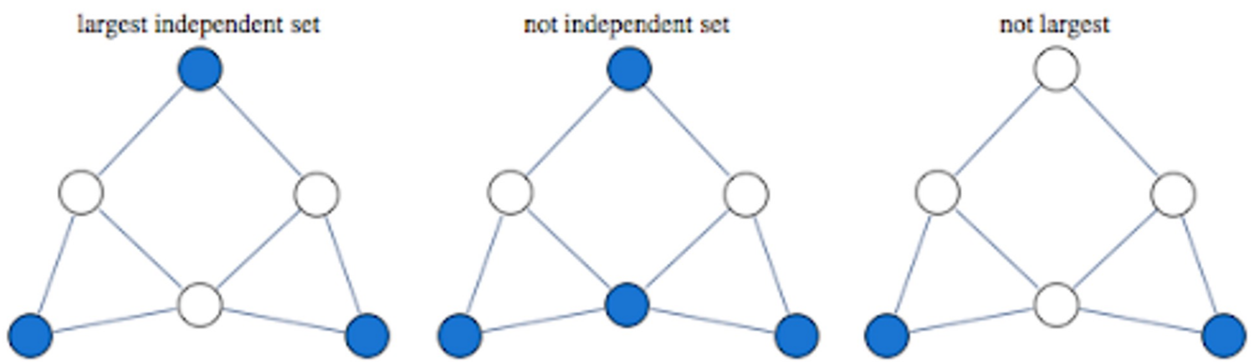
1. Show that $X \in$ **NP**
2. Pick a problem A which is already known to be **NP**-complete
3. Show that $A \leq_P X$



INDEPENDENT-SET Problem

Given a graph *G = (V, E)*, **independent set** is a subset of vertices ***V*** such that no two vertices in the graph is connected by an edge

INDEPENDENT-SET (IS) problem: Given a graph *G* and integer *k*, is there an independent set of size at least *k*?



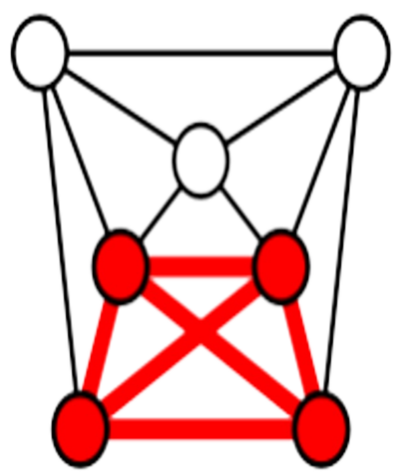
IS of size 3

IS of size 2

CLIQUE Problem

A set of vertices U of a graph G is a **clique** if every pair of vertices in U has an edge in G. Intuitively, a subgraph of G is a complete graph.

CLIQUE problem: Given a graph G and integer k, is there a clique of size at least k?



A graph with
clique of size 4

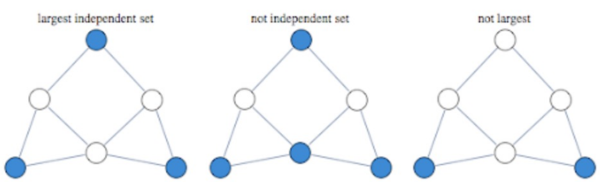
Question 3

Show that the CLIQUE problem is NP-complete. (Try a reduction from Independent Set)

INDEPENDENT-SET Problem

Given a graph *G = (V, E)*, **independent set** is a subset of vertices ***V*** such that no two vertices in the graph is connected by an edge

INDEPENDENT-SET (IS) problem: Given a graph *G* and integer *k*, is there an independent set of size at least *k*?

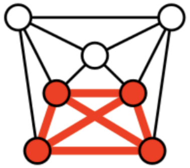


IS of size 3 IS of size 2

CLIQUE Problem

A set of vertices *U* of a graph *G* is a **clique** if every pair of vertices in *U* has an edge in *G*. Intuitively, a subgraph of G is a complete graph.

CLIQUE problem: Given a graph G and integer k, is there a clique of size at least k?



A graph with
clique of size 4