

Unit -5

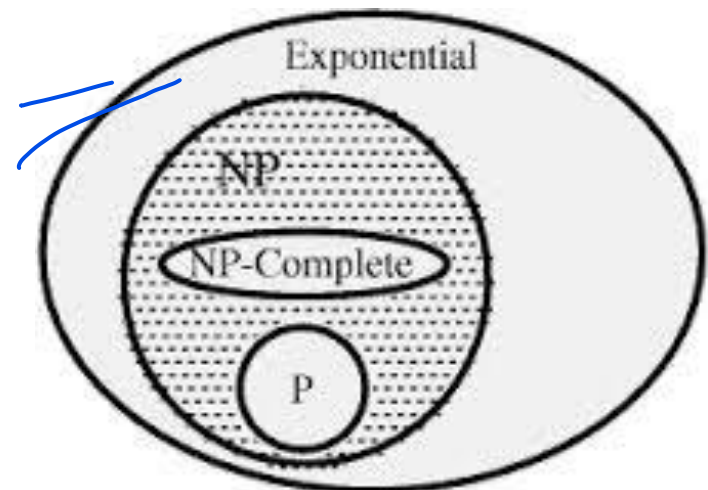
NP Complete problems

NP and P

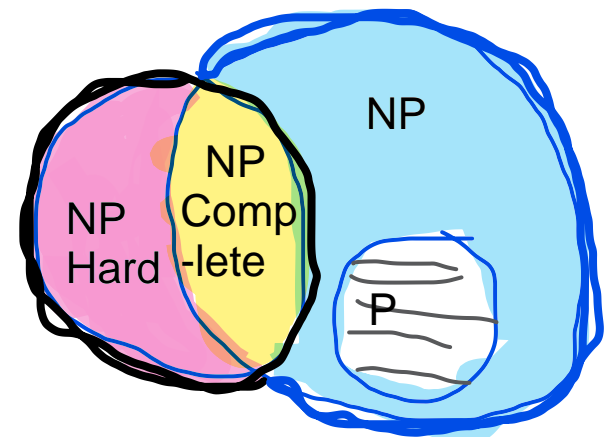
- **What is NP?**
- NP is the set of all decision problems (question with yes-or-no answer) for which the 'yes'-answers can be **verified** in polynomial time ($O(n^k)$ where n is the problem size, and k is a constant) by a deterministic Turing machine. Polynomial time is sometimes used as the definition of *fast* or *quickly*.
- **What is P?**
- P is the set of all decision problems which can be **solved** in polynomial time by a deterministic Turing machine. Since it can solve in polynomial time, it can also be verified in polynomial time. Therefore P is a subset of NP.

NP-Complete

- What is **NP-Complete**?
- A problem x that is in NP is also in NP-Complete if and only if every other problem in NP can be quickly (ie. in polynomial time) transformed into x . In other words:
 - x is in NP, and
 - Every problem in NP is reducible to x



NP-Hard



- What is NP-Hard?
- NP-Hard are problems that are at least as hard as the hardest problems in NP. Note that NP-Complete problems are also NP-hard. However not all NP-hard problems are NP (or even a decision problem), despite having 'NP' as a prefix. That is the NP in NP-hard does not mean 'non-deterministic polynomial time'. Yes this is confusing but its usage is entrenched and unlikely to change.

Polynomial time reductions

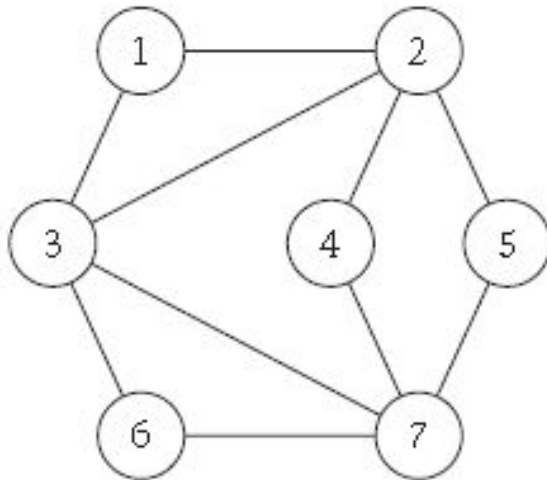
- “Problem X is at least as hard as problem Y .”
We will formalize this through the notion of reduction: we will show that a particular problem X is at least as hard as some other problem Y by arguing that, if we had a “black box” capable of solving X , then we could also solve Y . (In other words, X is powerful enough to let us solve Y .)

Polynomial time reductions

- Suppose we had a *black box* that could solve instances of a problem X ; if we write down the input for an instance of X , then in a single step, the *black box* will return the correct answer. We can now ask the following question:
- (*) Can arbitrary instances of problem Y be solved using a polynomial number of standard computational steps, plus a polynomial number of calls to a black box that solves problem X ?
- If the answer to this question is yes, then we write $Y \leq_P X$; we read this as “ Y is polynomial-time reducible to X ,” or “ X is at least as hard as Y (with respect to polynomial time).”

Independent set and vertex cover

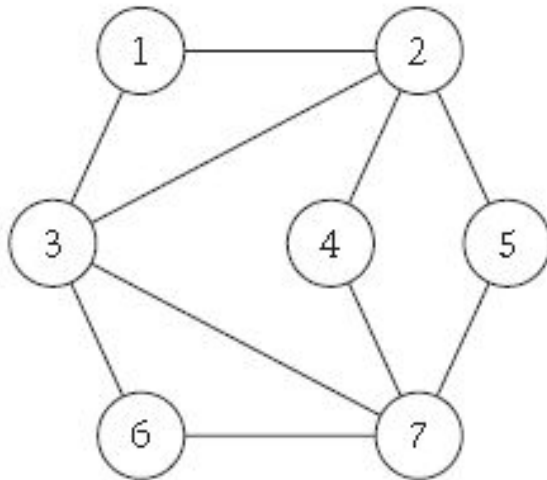
- in a graph $G = (V, E)$, we say a set of nodes
- $S \subseteq V$ is independent if no two nodes in S are joined by an edge.



Given a graph G and a number k , does G contain an independent set of size at least k ?

Independent set and vertex cover

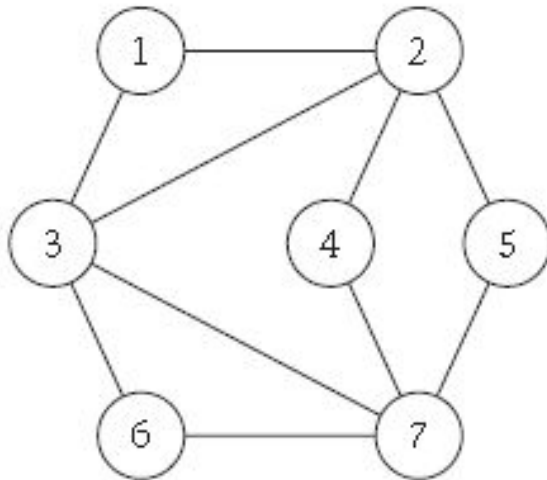
- *Vertex Cover.* Given a graph $G = (V, E)$, we say that a set of nodes $S \subseteq V$ is a vertex cover if every edge $e \in E$ has at least one end in S .



the set of nodes $\{1, 2, 6, 7\}$ is a vertex cover of size 4, while the set $\{2, 3, 7\}$ is a vertex cover of size 3.

Independent set and vertex cover

- We don't know how to solve either Independent Set or Vertex Cover in polynomial time; but what can we say about their relative difficulty?



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

$E = \dots\dots\dots$

Independent set $S = \{1, 4, 5, 6\}$

Vertex cover = $\{2, 3, 7\}$

Independent set and vertex cover

- 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.
- **Proof.**
- **First, suppose that S is an independent set. Consider an arbitrary edge $e = (u, v)$. Since S is independent, it cannot be the case that both u and v are in S ; so one of them must be in $V - S$. It follows that every edge has at least one end in $V - S$, and so $V - S$ is a vertex cover.**
- Conversely, suppose that $V - S$ is a vertex cover. Consider any two nodes u and v in S . If they were joined by edge e , then neither end of e would lie in $V - S$, contradicting our assumption that $V - S$ is a vertex cover. It follows
- that no two nodes in S are joined by an edge, and so S is an independent set.

Proof to determine that both of the problems are NP-Complete.
Vertex cover and Independent set problem with an example