Design and Analysis of Algorithms

Lecture-45

Dharmendra Kumar (Associate Professor)
Department of Computer Science and Engineering
United College of Engineering and Research,
Prayagraj

NP-Completeness

Classes P and NP

- The class P consists of those problems that are solvable in polynomial time.
- More specifically, they are problems that can be solved in time $O(n^k)$ for some constant k, where n is the size of the input to the problem.
- The class NP consists of those problems that are “verifiable” in polynomial time.
- What do we mean by a problem being verifiable? If we were somehow given a “certificate” of a solution, then we could verify that the certificate is correct in time polynomial in the size of the input to the problem.

Classes P and NP

- For example, in the hamiltonian cycle problem, given a directed graph G(V,E), a certificate would be a sequence $< v_1, v_2, v_3, \ldots\ldots, v_n >$ of n vertices. We could easily check in polynomial time that $(v_i, v_{i+1}) \in E$ for $i = 1, 2, 3, \ldots\ldots\ldots, n-1$ and that $(v_n, v_1) \in E$ as well.

- Any problem in P is also in NP, since if a problem is in P then we can solve it in polynomial time without even being supplied a certificate.

NP-Hard and NP-complete(NPC)

NP-Hard: A problem L is said to be NP-hard if every problems belong in to NP is polynomial time reducible to L.

That is,

Problem L is NP-hard if for all problems $L' \in NP$,

$$L' \leq_p L.$$

That is, if we can solve L in polynomial time, then we can solve all NP problems in polynomial time.

NP-Complete

NP-Complete: Problem L is NP-complete if

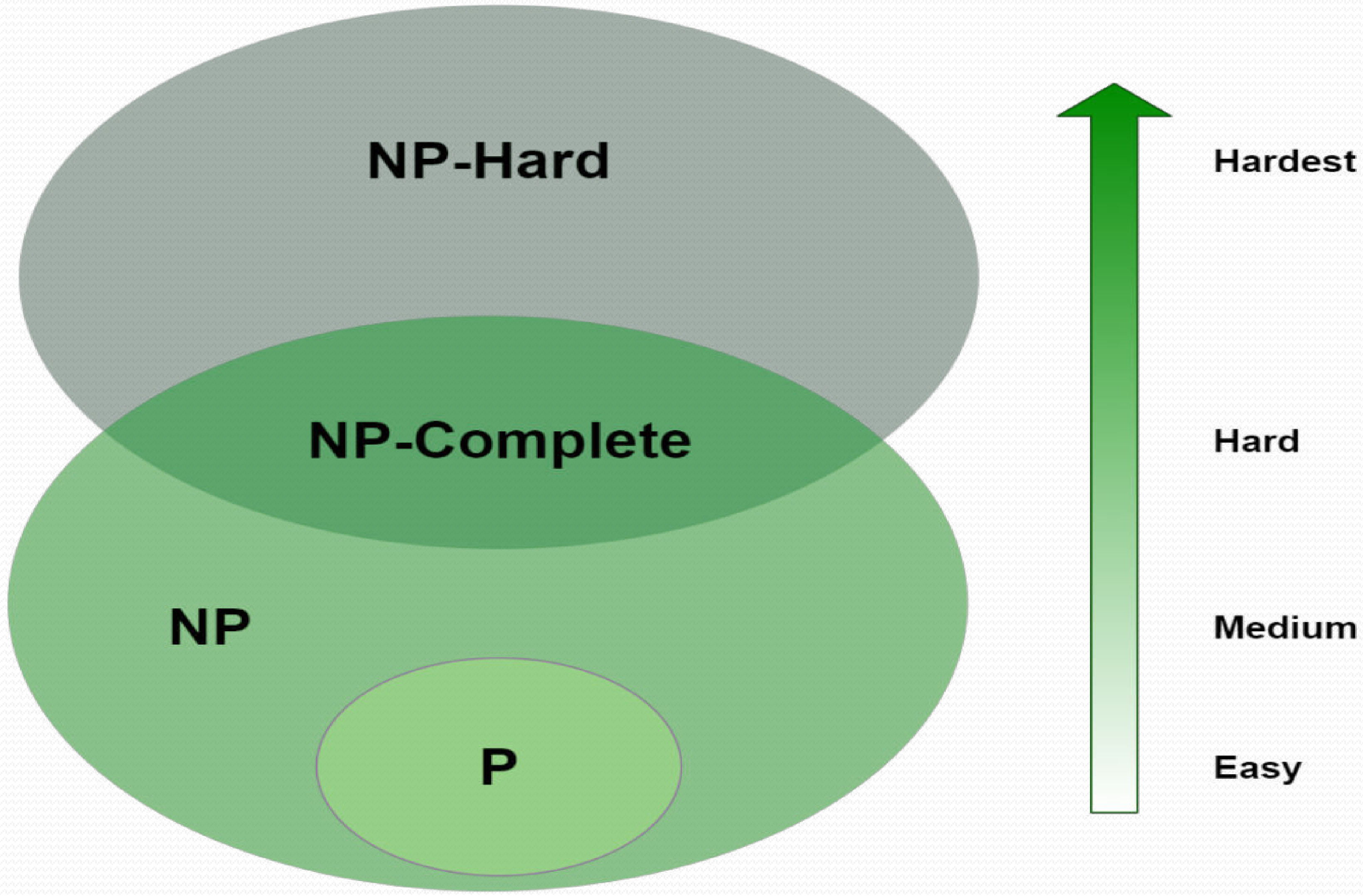
1. L ϵ NP and
2. L is NP-hard

NP-Complete problems:

- Boolean satisfiability problem (SAT)
- Hamiltonian cycle problem.
- Travelling salesman problem
- Vertex cover problem

Note: If any NP-complete problem is solvable in polynomial time, then every NP-Complete problem is also solvable in polynomial time.

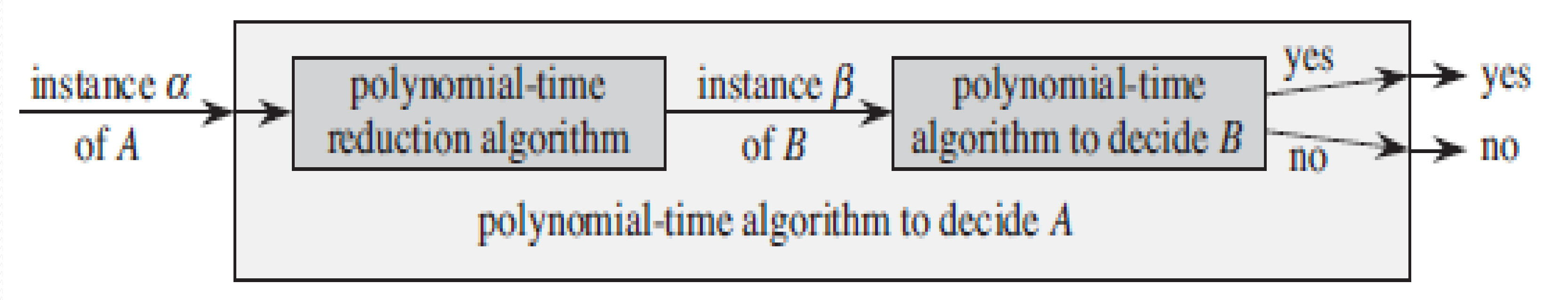
Relationship between P, NP, NP-hard and NPC



Polynomial-time *Reduction Algorithm*

- Consider a decision problem A, which we would like to solve in polynomial time.
- We call the input to a particular problem an ***instance of that problem.***
- Suppose that we already know how to solve a different decision problem B in polynomial time.
- Finally, suppose that we have a procedure that transforms any instance α of A into some instance β of B with the following characteristics:
  - The transformation takes polynomial time.
  - The answers are the same. That is, the answer for α is "yes" if and only if the answer for β is also "yes."
- We call such a procedure a **polynomial-time *reduction algorithm.***

Polynomial-time *Reduction Algorithm(cont.)*


instance α of A
polynomial-time reduction algorithm
instance β of B
polynomial-time algorithm to decide B
yes
no
yes
no
polynomial-time algorithm to decide A

Polynomial-time reduction algorithm provides us a way to solve problem A in polynomial time:

1. Given an instance α of problem A, use a polynomial-time reduction algorithm to transform it to an instance β of problem B.
2. Run the polynomial-time decision algorithm for B on the instance β.
3. Use the answer for β as the answer for α.

NP-complete

Lemma

If L is a language such that $L' \leq_P L$ for some $L' \in NPC$, then L is NP-hard. If, in addition, $L \in NP$, then $L \in NPC$.

Note: The hamiltonian cycle problem is NP-complete.

Note: Satisfiability of boolean formulas in 3-conjunctive normal form is NP-complete.

Note: The vertex-cover problem is NP-complete.

NP-complete

Theorem: Show that the traveling-salesman problem is NP-complete.

Proof:

We first show that TSP belongs to NP.

Given an instance of the problem, we use as a certificate the sequence of n vertices in the tour. The verification algorithm checks that this sequence contains each vertex exactly once, sums up the edge costs, and checks whether the sum is at most k. This process can certainly be done in polynomial time.

NP-complete

To prove that TSP is NP-hard, we show that HAM-CYCLE$\leq_p$TSP.

Let G=(V,E) be an instance of HAM-CYCLE. We construct an instance of TSP as follows.

We form the complete graph G'=(V,E'), where

E'={(i,j) ! i,j ∈ V and i≠j} and

we define the cost function c by:-

$c(i,j) = 0$ if $(i,j) \in E$

$= 1$ if $(i,j) \notin E$

The instance of TSP is then < G',c,0 >, which we can easily create in polynomial time.

NP-complete

We now show that graph G has a hamiltonian cycle if and only if graph G' has a tour of cost at most 0.

Suppose that graph G has a hamiltonian cycle h. Each edge in h belongs to E and thus has cost 0 in G'. Thus, h is a tour in G' with cost 0. Conversely, suppose that graph G' has a tour h' of cost at most 0. Since the costs of the edges in E' are 0 and 1, the cost of tour h' is exactly 0 and each edge on the tour must have cost 0. Therefore, h' contains only edges in E. We conclude that h' is a hamiltonian cycle in graph G.

AKTU Examination Questions

1. Write and explain the algorithm to solve vertex cover problem using approximation algorithm.
2. Explain NP-complete and NP-Hard.
3. Explain Randomized algorithm in brief.
4. What is an approximation algorithm? What is meant by P(n) approximation algorithms? Discuss approximation algorithm for Travelling Salesman Problem.
5. Define NP-Hard and NP- complete problems. What are the steps involved in proving a problem NP-complete? Specify the problems already proved to be NP-complete.

AKTU Examination Questions

6. What are approximation algorithms? What is meant by P(n) approximation algorithms?
7. Define NP, NP hard and NP Complete Problems. Prove that Travelling Salesman Problem is NP-Complete.
8. What is the application of Fast Fourier Transform (FFT)? Also write the recursive algorithm for FFT.
9. Discuss the problem classes P, NP and NP –complete with class relationship.
10. Explain Approximation and Randomized algorithms.
11. What do you mean by polynomial time reduction?
12. Explain applications of FFT.