Unit-VI

UNIT VI: Introduction to NP-Hard and NP-Complete problems: Basic concepts of non deterministic algorithms, Definitions of NP-Hard and NP-Complete classes, Modular Arithmetic.

Applications: Performance evaluation in the dynamic systems.

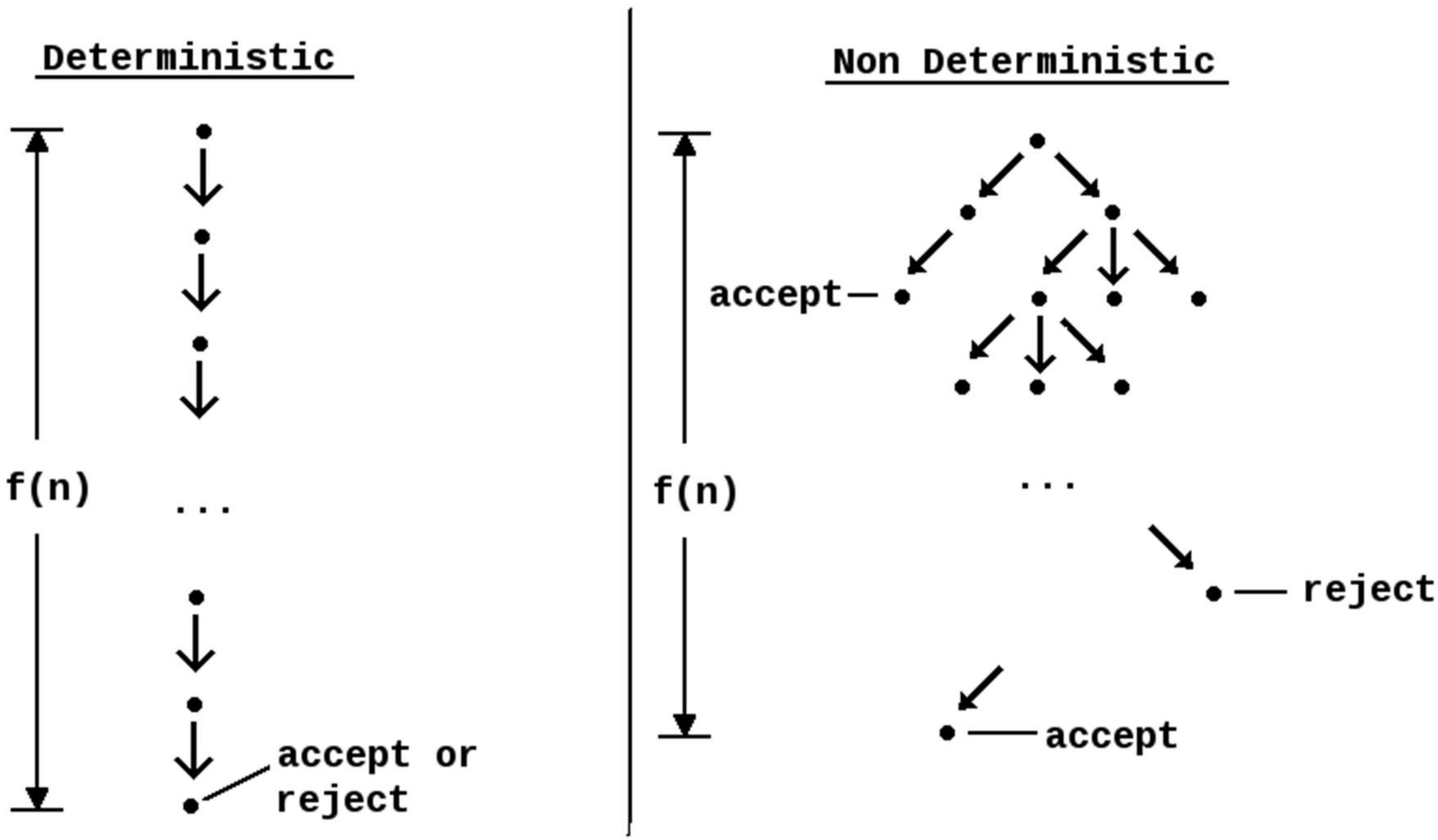
What are P, NP, NP-Complete, and NP-Hard?

- Two models of computer :

  Deterministic and non-deterministic.

  - A deterministic computer is the regular computers which we are using.
  - A non-deterministic computer is one that has unlimited parallelism.

- In deterministic machine each computation can be viewed as a linear ordered sequence (finite or infinite) of computations.
- Nondeterministic machines are not restricted to such ordered computations. At each moment, it will be allowed to choose among a number of possibilities. Each computation corresponding to a particular sequence of choices is called a computation.



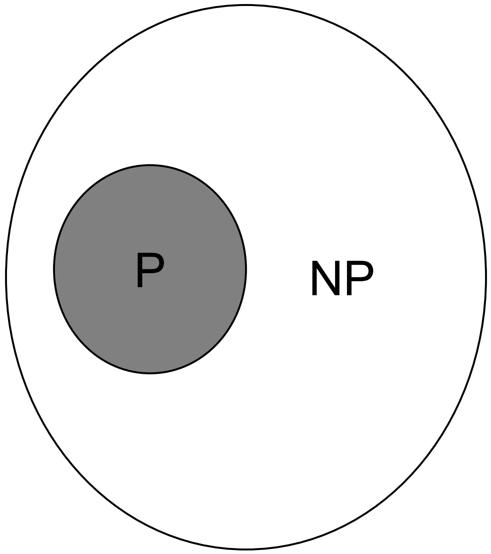
Deterministic
f(n)
...
accept or
reject
Non Deterministic
f(n)
accept
...
reject
accept

- P is the set of all decision problems solvable by deterministic algorithms in polynomial time.
  - Polynomial time: $O(n^2)$, $O(n^3)$, $O(1)$, $O(n \lg n)$
  - Not in polynomial time: $O(2^n)$, $O(n^n)$, $O(n!)$

- NP is the set of all the set of all decision problems solvable by non-deterministic algorithms in polynomial time.

- Since deterministic algorithms are just a special case of nondeterministic ones, we conclude that P is a subset of NP

- Most problems that can not be solved in polynomial-time algorithms are either optimization or decision problems.
- Optimization Problems
  - An optimization problem is one which asks, “What is the optimal solution to problem X?”
  - Examples:
    - 0-1 Knapsack
    - Fractional Knapsack
    - Minimum Spanning Tree
- Decision Problems
  - An decision problem is one with yes/no answer
  - Examples:
    - Does a graph G have a MST of weight $\leq$ W?
    - Is there a hamiltonian cycle of weight $\leq$ k



Commonly believed relationship between P and NP

Sample Problems in P

- Fractional Knapsack
- MST
- Sorting
- Others?

Sample Problems in NP

- Fractional Knapsack
- MST
- Sorting
- Others?
  - Hamiltonian Cycle (Traveling Salesman)
  - Graph Coloring

- The most famous unsolved problem in computer science is whether P=NP or $P \neq NP$

- **Reducibility**
  - Problem L1 reduces to L2 if and only if there is a way to solve L1 by a deterministic polynomial time algorithm using a deterministic algorithm that solves L2 in polynomial time.
  - It is represented as L1 α L2
  - This definition implies that if we have a polynomial time algorithm for L2, then we can solve L1 in polynomial time.

NP-Complete

- Q is an NP-Complete problem if:

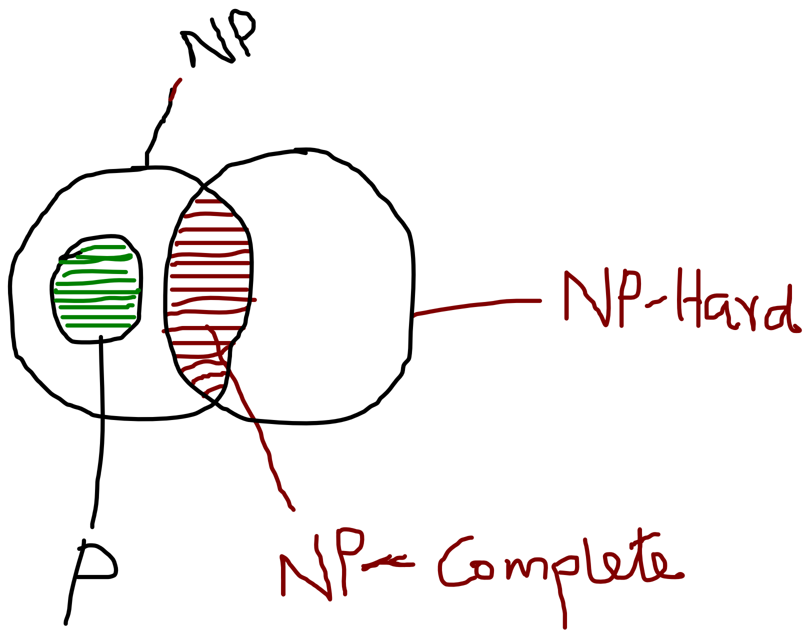
- 1) Q is in NP
- 2) every other NP problem polynomial time reducible to Q

Note: Q is the hardest problem in this group.

NP-Hard

- A problem is NP-hard if and only if it's at least as hard as an NP-complete problem.

Commonly believed relationship among p, NP, NP-Complete, and NP-Hard problems



- Normally decision problems are NP-Complete.
- Optimization problems are NP-Hard.
- There are some NP-Hard problems that are not NP-Complete. For example, halting problem. The halting problem states that: “ Is it possible to determine whether an algorithm will ever halt or enter in loop on certain input?
- Two problems P and Q are said to be polynomially equivalent iff P α Q and Q α P.
- A problem L is ______ if and only if satisfiability reduces to L.
  - Ans: NP-Hard (it is theorem in book, verify ……………)

Cook’s Theorem

- It States that satisfiability is in p iff P=NP
- Boolean Satisfiability(SAT) problem is NP Complete. (see DDA A.A.Puntambekar)