NP-completeness part-I (DAA, M.Tech + Ph.D.)

By:

Sunil Kumar Singh, PhD
Assistant Professor,
Department of Computer Science and Information Technology

School of Computational Sciences, Information and Communication Technology,
Mahatma Gandhi Central University, Motihari
Bihar, India-845401

Outline

- Introduction
- P & NP
- NP-complete
- NP-Hard
- Determinism & Non-determinism
- Examples
- Conclusion
- References

Introduction

- Some Algorithm we have seen like.
  - ✓ Sorting - $O(N \log N)$
  - ✓ Binary Searching -$O(\log N)$
  - ✓ Shortest Path Finding - $O(N^2)$
- However, some problems only have
  - ✓ Exponential Time algorithm -$O(2^N)$

Size vs time complexity

N	10	20	30	40	50	60
$O(N)$	.00001 second	.00002 second	.00003 second	.00004 second	.00005 second	.00006 second
$O(N^2)$	.0001 second	.0004 second	.0009 second	.0016 second	.0025 second	.0036 second
$O(N^3)$	.001 second	.008 second	.027 second	.064 second	.125 second	.216 second
$O(N^5)$	1 second	3.2 seconds	24.3 seconds	1.7 minutes	5.2 minutes	13.0 minutes
$O(2^N)$	.001 second	1.0 second	17.9 minutes	**12.7 days**	**35.7 years**	**366 centuries**
$O(3^N)$	.059 second	58 minutes	6.5 years	**3855 centuries**	**$2*10^8$ centuries**	**10^{13} centuries**

P and 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 sometime used as the definition of fast or quickly.

- P is the set of all decision problems which can be solved in polynomial time by a deterministic Turing machine.

- Since it can be solved in polynomial time, and also can be verified in polynomial time. Therefore P is a subset of NP.

NP Complete

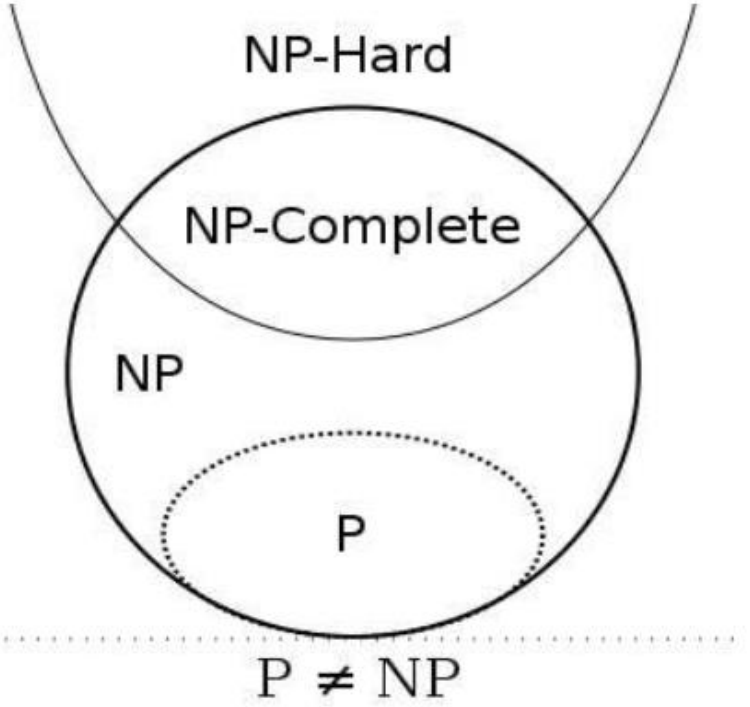
- A problem X that is in NP is also in NP-Complete it and only if every other problem in NP can be transformed into X in polynomial time.

- So what makes NP-Complete so interesting is that if any one of the NP-Complete problems was to be solved in polynomial time then all NP problems can be solved quickly.

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.



Determinism vs. Non-determinism

- Nondeterministic algorithms produce an answer by a series of “correct guesses”

- Deterministic algorithms (like those that a computer executes) make decision based on information.

Examples

- **NP-Complete Problems**
  - ✓ Determining whether a graph has a Hamiltonian cycle
  - ✓ Determining whether a Boolean formula is satisfiable
- **NP-Hard Problems**
  - ✓ The circuit-satisfiability problem
  - ✓ Set Cover
  - ✓ Vertex Cover
  - ✓ Travelling Salesman Problem

TSP is NP-Complete

- The traveling salesman problem consists of a salesman and set of cities.
- The sales man has to visit all the cities starting from one of the city.
- Challenging problem is to minimize the length of the trip.
- Justify that why it is NP-Complete?

References

1. Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, 2009.
2. Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. "Introduction to algorithms second edition." *The Knuth-Morris-Pratt Algorithm, year* (2001).
3. Seaver, Nick. "Knowing algorithms." (2014): 1441587647177.

Thank You