Course: Advanced Algorithm and Problem Solving

WEEK 11 Complexity Classes - P, NP, NP-Complete, and NP-Hard

Lemlem Kassa (Ph.D.)

Addis Ababa Science and Technology University, Ethiopia

June, 2025

WEEK 11 Complexity Classes - P, NP, NP-Complete, and NP-Hard

Content

- Overview of Complexity Theory
- Types of Complexity Classes
  - P and NP class problem
  - NP-Complete Problem
  - NP-Hard Problem

Lecture Learning Outcome

- Understand complexity theory and complexity classes
- Understand types of complexity classes
- Differentiate types of problems, such as Hard problem and easy problems
- Understand P, NP , NP-hard and NP-complete problems

Overview of Complexity Theory

Introduction

- In complexity theory, a complexity class is a set of problems with related complexity
- It is a branch of theory of computation that studies the resources required during computation to solve a given problem.
- In order to understand the problems for which solutions are not there, the problems are divided into Complexity classes
- Time (how much time the algorithm takes to solve a problem) and space (how much memory it takes) are the most common resources.

[1]. Complexity Classes, Duc Duong, The Road Ahead (WordPress), 2017. https://duongquangduc.wordpress.com/2017/02/13/complexity-classes/

Overview of Complexity Theory …cont'd

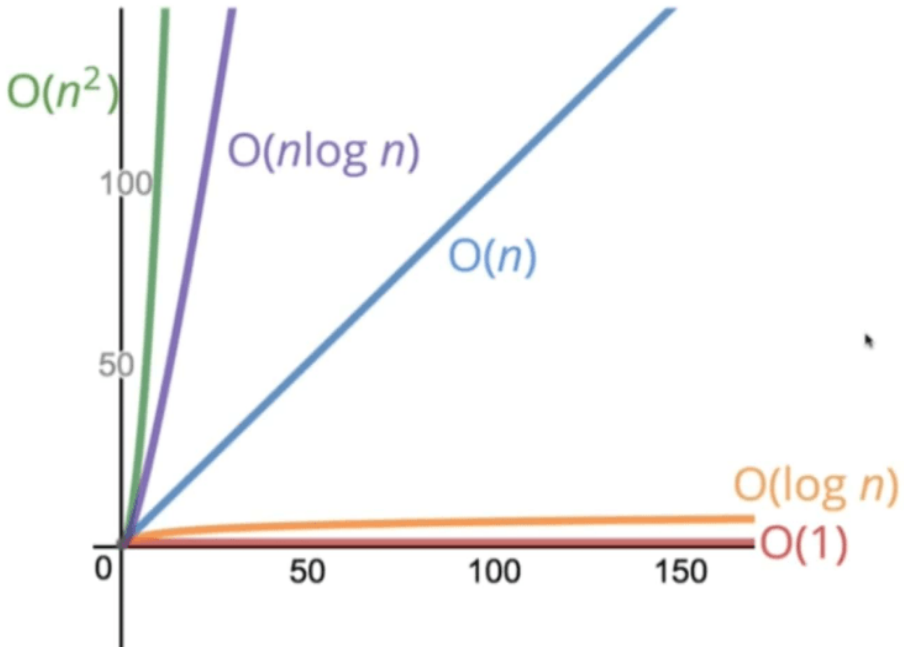
- **Time Complexity:** measurement of the amount of time an algorithm takes to complete with respect to its input size.
- **Big O Notation**: Used to describe how 'fast' an algorithm grows, Big O Notation gives an upper bound of the complexity in the worst-case, thereby providing a worst-case scenario estimate.
- **Time Complexity Classes:** There are various classes of Time Complexity, including: Constant Time (O[1]), Logarithmic Time (O[log(n)]), Linear Time (O[n]), Quadratic Time (O[n^2]), and Exponential Time (O[2^n]).
- **Significance of Time Complexity**: It helps us predict the time an algorithm would take to complete with respect to its input size.

[2]. Computational Complexity Theory: Key Concepts, BotPenguin Team, BotPenguin, 2023. https://botpenguin.com/glossary/computational-complexity-theory

Overview of Complexity Theory …cont'd

- Space Complexity refers to the number of spaces an algorithm needs to run i.e., the amount of memory an algorithm uses with respect to its input size.

- Analogous to Time Complexity, an increase in problem size generally leads to an increase in space complexity. It plays an important role in the design and selection of algorithms.
- One of the major considerations when designing algorithms is considering the trade-off between Time and Space Complexity. Faster algorithms often require more space and vice versa



[2]. Computational Complexity Theory: Key Concepts, BotPenguin Team, BotPenguin, 2023. https://botpenguin.com/glossary/computational-complexity-theory

Overview of Complexity Theory …cont'd

Complexity Classes

- Complexity classes provide a theoretical framework for characterizing the difficulty of computational problems based on the amount of resources required to solve them.
- This helps in understanding the inherent complexity of a problem and in determining the best algorithmic approach for solving it.
- Complexity classes provide a standard benchmark for comparing the efficiency of different algorithms for solving the same problem.
    - This helps in selecting the most efficient algorithm for a given problem and in assessing theoretical feasibility of solving the problem.

[3]. Complexity Classes. Vedant Parvekar, Medium, 2023.
https://medium.com/@parvekarvedant24/complexity-classes-630c8b7563f1

Overview of Complexity Theory …cont'd

Cont'd …….. Complexity Classes

- Complexity classes help in identifying problems that are computationally infeasible to solve in practice, such as NP-Complete and NP-Hard problems.
  - Help in focusing research efforts on developing approximation algorithms or heuristic methods for solving such problems.
- Complexity classes provide insights into the theoretical limits of computation, such as the fact that some problems are inherently unsolvable or require an exponential amount of resources to solve.
  - Help in developing new models of computation or in identifying new research directions.

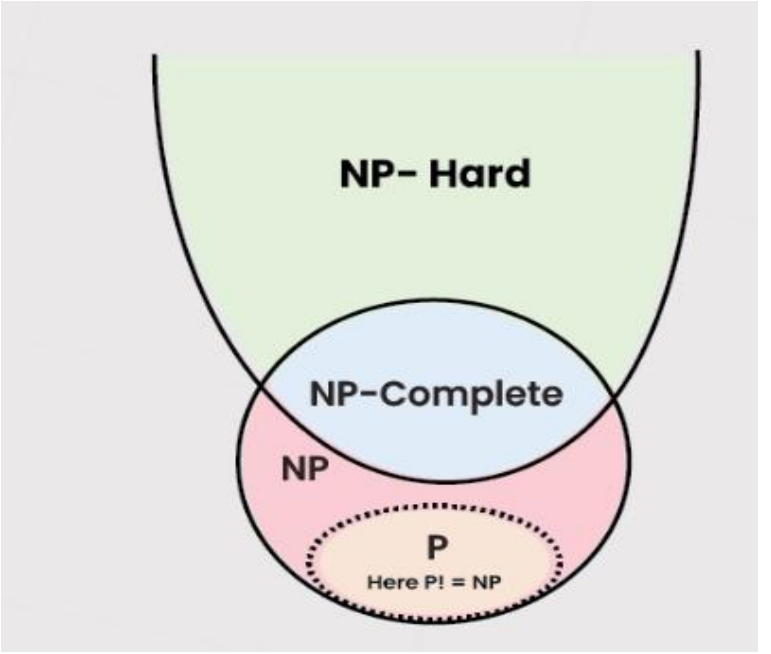
[3]. Complexity Classes. Vedant Parvekar, Medium, 2023.
https://medium.com/@parvekarvedant24/complexity-classes-630c8b7563f1

Overview of Complexity Theory …cont'd

Time complexity and space complexity of an algorithm

- The time complexity of an algorithm is used to describe the number of steps required to solve a problem, but it can also be used to describe how long it takes to verify the answer.

- The space complexity of an algorithm describes how much memory is required for the algorithm to operate.



[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Overview of Complexity Theory …cont'd

Types of Problems

- **Trackable /Easy Problems :** - Problems that can be solvable in a reasonable (polynomial) time.

Examples.

- Divide-and-conquer
- Merge sort.
- May long, but we can solve them in known time

Polynomial time: $O(n^2)$, $O(n^3)$, $O(1)$, $O(n \lg n)$
Not in polynomial time: $O(2^n)$, $O(n^n)$, $O(n!)$

- **Intractable/ Hard problem**:- As they grow large, we are unable to solve them in reasonable time.

  O(n ^ k) algorithm is available, O(n ^n) or O(n!) or O(k ^n) time required

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Overview of Complexity Theory …cont'd

Optimization Problems

- A problem which asks, "What is the optimal solution to problem X?"

Examples:

- 0-1 Knapsack
- Fractional Knapsack
- Minimum Spanning Tree

Decision Problems

- A problems with a "yes" or "no" answer that can be solved by a deterministic machine (our computers) in polynomial time.

Examples: Does a graph G have a MST of weight <= W?

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Types of Complexity Classes …cont'd

P Class

- The P in the P class stands for **Polynomial Time.**
- It is the collection of decision problems(problems with a "yes" or "no" answer) that can be solved by a deterministic machine (our computers) in polynomial time.
    - That is, they are solvable in O(p(n)), where p(n) is a polynomial on n
    - A deterministic algorithm is (essentially) one that always computes the correct answer
- P is often a class of computational problems that are solvable and tractable.

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Types of Complexity Classes …cont'd

- The first set of problems are polynomial algorithms that we can solve in polynomial time, like logarithmic, linear or quadratic time. If an algorithm is polynomial, we can formally define its time complexity as:

$$T(n) = O(C * n^k) \text{ where } C > 0 \text{ and } k > 0$$

- Where C and k are constants and n is input size.
- In general, for polynomial-time algorithms k is expected to be less than n.
- Many algorithms complete in polynomial time

[5]. P, NP, NP-Complete, and NP-Hard, Baeldung Team, Baeldung, 2023. https://www.baeldung.com/cs/p-np-np-complete-np-hard.

Types of Complexity Classes .... Cont'd

Many algorithms complete in polynomial time:

- All basic mathematical operations; addition, subtraction, division, multiplication
- Testing for primacy
- Hash table lookup, string operations, sorting problems
- Shortest Path Algorithms; Dijkstra, Bellman-Ford, Floyd-Warshall
- Linear and Binary Search Algorithms for a given set of numbers

- All of these have a complexity of O(n^k) for some k, and that fact places them all in P. Of course, we don't always have just one input, n. But, so long as each input is a polynomial, multiplying them will still be a polynomial.

[5]. P, NP, NP-Complete, and NP-Hard, Baeldung Team, Baeldung, 2023. https://www.baeldung.com/cs/p-np-np-complete-np-hard

Types of Complexity Classes .... Cont'd

NP Algorithms

- The second set of problems cannot be solved in polynomial time. However, they can be verified (or certified) in polynomial time. We expect these algorithms to have an exponential complexity, which is define as:

$$T(n) = O(C_1 * k^{C_2 * n}) \text{ where } C_1 > 0, C_2 > 0 \text{ and } k > 0$$

- Where C_1, C_2 and k are constants and n is the input size.
- T(n) is a function of exponential-time when at least C_1=1 and C_2=1. As a result, we get O(k^n).

[5]. P, NP, NP-Complete, and NP-Hard, Baeldung Team, Baeldung, 2023. https://www.baeldung.com/cs/p-np-np-complete-np-hard

Types of Complexity Classes .... Cont'd

- For example, we'll see complexities like O(n^n), O(2^n), O(2^{0.000001*n}) in this set of problems. There are several algorithms that fit this description.
- Among them are:
  - ***Integer Factorization and Graph Isomorphism***
- Both of these have two important characteristics: Their complexity is O(k^n) for some k and their results can be verified in polynomial time.
- Those two facts place them all in NP, that is, the set of "Non-deterministic Polynomial" algorithms.

[5]. P, NP, NP-Complete, and NP-Hard, Baeldung Team, Baeldung, 2023.
https://www.baeldung.com/cs/p-np-np-complete-np-hard

Types of Complexity Classes …. Cont'd

- Formally, we also state that Integer Factorization and Graph Isomorphism problems must be decision problems – have a yes or no answer – though note that practically speaking, all function problems can be transformed into decision problems.
- This distinction helps us to nail down what we mean by "verified".
- To speak precisely, then, an algorithm is in NP if it can't be solved in polynomial time and the set of solutions to any decision problem can be verified in polynomial time by a "Deterministic Turing Machine".

[5]. P, NP, NP-Complete, and NP-Hard, Baeldung Team, Baeldung, 2023.
https://www.baeldung.com/cs/p-np-np-complete-np-hard

Types of Complexity Classes …. Cont'd

- If someone can provide us with the solution to the problem, we can find the correct and incorrect pair in polynomial time.
- Thus for the NP class problem, the answer is possible, which can be calculated in polynomial time.
- This class contains many problems that one would like to be able to solve effectively:
    - Boolean Satisfiability Problem (SAT).
    - Hamiltonian Path Problem.
    - Graph coloring.

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Types of Complexity Classes .... Cont'd

NP Problems : - Boolean Satisfiability Problem

- Boolean Satisfiability or simply **SAT** is the problem of determining if a Boolean formula is satisfiable or unsatisfiable.
  - **Satisfiable :** If the Boolean variables can be assigned values such that the formula turns out to be TRUE, then we say that the formula is satisfiable.
  - **Unsatisfiable :** If it is not possible to assign such values, then we say that the formula is unsatisfiable.

Examples:

- $F = A \wedge \bar{B}$, is satisfiable, because A = TRUE and B = FALSE makes F = TRUE.
- $G = A \wedge \bar{A}$, is unsatisfiable, because:

A	$\bar{A}$	G
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE

[6]. 2-Satisfiability (2-SAT) Problem, GeeksforGeeks, GeeksforGeeks 2024. https://www.geeksforgeeks.org/2-satisfiability-2-sat-problem/

Types of Complexity Classes .... Cont'd

Co-NP Class

- Co-NP stands for the complement of NP Class.
- It means if the answer to a problem in Co-NP is No, then there is proof that can be checked in polynomial time.

Features:

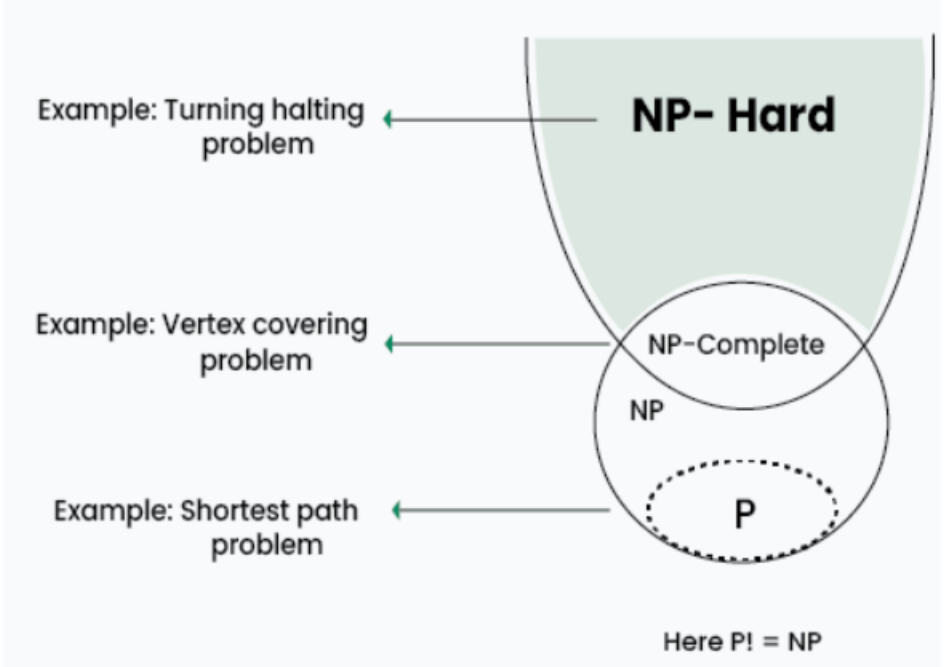
- If a problem X is in NP, then its complement X' is also in Co-NP.
- For an NP and Co-NP problem, there is no need to verify all the answers at once in polynomial time, there is a need to verify only one particular answer "yes" or "no" in polynomial time for a problem to be in NP or Co-NP.

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Types of Complexity Classes .... Cont'd

NP-hard class

- A 'P' problem is said to be NP-Hard when all n^k' belonging in NP can be reduced in polynomial time (where k is some constant) 'P' assuming a solution for 'P' takes 1 unit time.
- NP-Hard is a computational complexity theory that acts as a defining property for the class of problems that are "at least as hard as the hardest problems in NP".
- We can reduce Problem B to Problem A if, given a solution to Problem A, we can easily construct a solution to Problem B.
- "easily" means "in polynomial time.".



Problem-B Problem-A

Example: lcm(m, n) = m * n / gcd(m, n),

Types of Complexity Classes .... Cont'd

Features of NP-hard problems :

- All NP-hard problems are not in NP.
- It takes a long time to check them. This means if a solution for an NP-hard problem is given then it takes a long time to check whether it is right or not.
- A problem A is in NP-hard if, for every problem L in NP, there exists a polynomial-time reduction from L to A.

- Some of the examples of problems in **NP-hard a**re:
  - Halting problem.
  - Qualified Boolean formulas.
  - No Hamiltonian cycle.

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Types of Complexity Classes …. Cont'd

- **NP-hard problems Example- Halting problem.**
  - The Halting Problem asks whether a given program or algorithm will eventually halt (terminate) or continue running indefinitely for a particular input.
  - "Halting" means that the program will either accept or reject the input and then terminate, rather than going into an infinite loop.
- **The Core Question is,** can we create an algorithm that determines whether any given program will halt for a specific input?
- **Answer:** No, it is impossible to design a generalized algorithm that can accurately determine whether any arbitrary program will halt.
- The only way to know if a specific program halts is to run it and observe the outcome. This makes the Halting Problem an **undecidable problem**.

[7]. Halting Problem in Theory of Computation, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/halting-problem-in-theory-of-computation/

Types of Complexity Classes …. Cont'd

- Are all problems solvable in polynomial time?
  - **No**: Turing's "Halting Problem" is not solvable by any computer, no matter how much time is given.

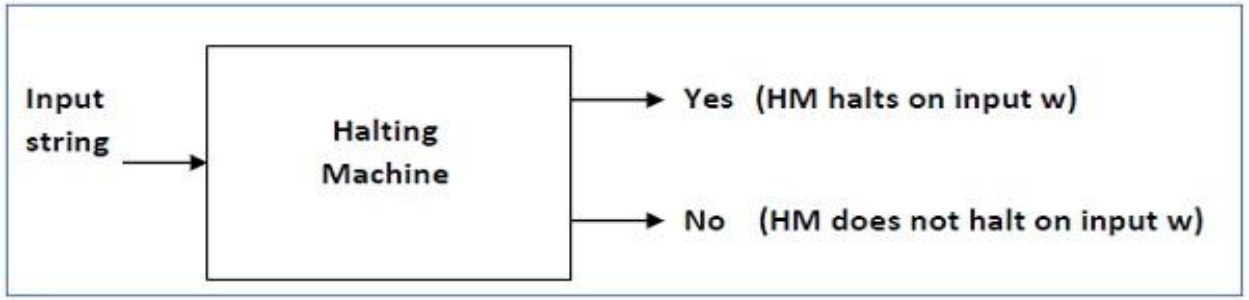
Halting Problem examples

a. If the instructions are *take a number x, which is not negative, and keep multiplying it by 2 until the result is bigger than 1.* Then the program will stop as long as the input x is not 0. If it is 0, it will keep going forever

b. Suppose a program consists of the instruction that *take every number between 1 and 10, add 2 to it and then output the result*. It's obvious that this program halts after 10 steps

Types of Complexity Classes .... Cont'd

- **Input** – A Turing machine and an input string w.
- **Problem** – Does the Turing machine finish computing of the string **w** in a finite number of steps? The answer must be either yes or no.
- **Proof** – At first, we will assume that such a Turing machine exists to solve this problem and then we will show it is contradicting itself.
  - We will call this Turing machine as a Halting machine that produces a yes or no in a finite amount of time. If the halting machine finishes in a finite amount of time, the output comes as yes, otherwise as no. The following is the block diagram of a Halting machine .
  - Hence, the halting problem is **undecidable**.



[8]. Turing Machine Halting Problem, TutorialsPoint Team, TutorialsPoint, 2025. https://www.tutorialspoint.com/automata_theory/turing_machine_halting_problem.htm

Types of Complexity Classes .... Cont'd

NP-complete class

- A problem is NP-complete if it is both NP and NP-hard.
- NP-complete problems are the hard problems in NP.

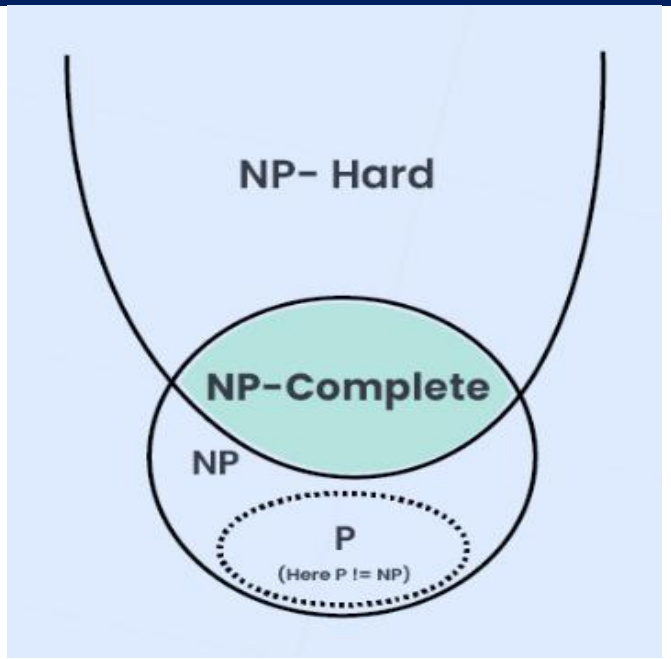
• Features:

- NP-complete problems are special as any problem in NP class can be transformed or reduced into NP-complete problems in polynomial time.
- If one could solve an NP-complete problem in polynomial time, then one could also solve any NP problem in polynomial time.

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Types of Complexity Classes .... Cont'd

- Some example NP-complete problems include:
  - Hamiltonian Cycle.
  - Satisfiability.
  - Vertex cover.



- **Note:** A Turing machine is a mathematical model of computation. It is a general example of a CPU that controls all data manipulation done by a computer.
- Turing machine can be halting as well as non halting and it depends on algorithm and input associated with the algorithm.

Types of Complexity Classes …cont'd

Decision vs Optimization Problems

- NP-completeness applies to the realm of decision problems.
- It was set up this way because it's easier to compare the difficulty of decision problems than that of optimization problems.
- In reality, though, being able to solve a decision problem in polynomial time will often permit us to solve the corresponding optimization problem in polynomial time (using a polynomial number of calls to the decision problem).
- So, discussing the difficulty of decision problems is often really equivalent to discussing the difficulty of optimization problems.

Example

- Consider the vertex cover problem optimization problem-(Given a graph, find out the minimum sized vertex set that covers all edges).
- The corresponding decision problem- given undirected graph G and k, is there a vertex cover of size k?

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

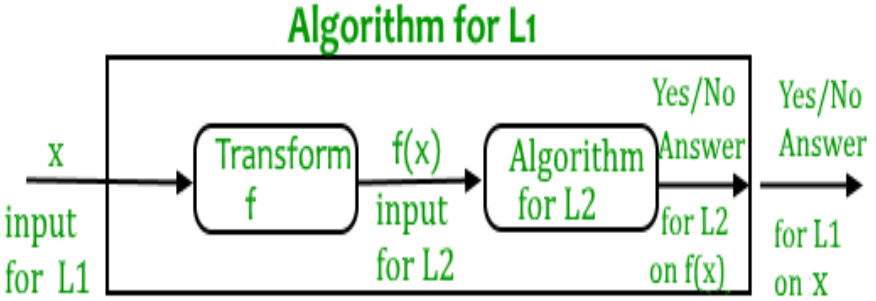
Types of Complexity Classes ...cont'd

Reduction

- A problem L1 can be *reduced* to another problem L2 if any instance of L1 can be rephrased to an instance of L2, the solution to which provides a solution to the instance of L1. This rephrasing is called a *transformation.*
- Intuitively: If L1 reduces in polynomial time to L2, L1 is "no harder to solve" than L2

Example

- Let **L1** and **L2** be two decision problems.
- Suppose algorithm **A2** solves **L2**.
- That is, if **y** is an input for **L2** then algorithm **A2** will answer Yes or No depending upon whether **y** belongs to **L2** or not.
- The idea is to find a transformation from **L1** to **L2** so that algorithm **A2** can be part of algorithm **A1** to solve **L1**.



[9]. Introduction to NP-Complete Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2024. https://www.geeksforgeeks.org/introduction-to-np-completeness/

Types of Complexity Classes …cont'd

How to prove that a given problem is NP-complete?

- By definition, it requires us to that show every problem in NP is polynomial time reducible to L.
- Fortunately, there is an alternate way to prove it. The idea is to take a known NP-Complete problem and reduce it to L.
- If a polynomial-time reduction is possible, we can prove that L is NP-Complete by transitivity of reduction (If an NP-Complete problem is reducible to L in polynomial time, then all problems are reducible to L in polynomial time).

[9]. Introduction to NP-Complete Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2024. https://www.geeksforgeeks.org/introduction-to-np-completeness/

Types of Complexity Classes …cont'd

Complexity Class	Characteristic feature
P	Easily solvable in polynomial time.
NP	Yes, answers can be checked in polynomial time.
Co-NP	No, answers can be checked in polynomial time.
NP-hard	All NP-hard problems are not in NP and it takes a long time to check them.
NP-complete	A problem that is NP and NP-hard is NP-complete.

[4]. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/

Summary

- Complexity class P encompasses decision problems solvable by deterministic Turing machines in polynomial time
- NP (Nondeterministic Polynomial time) comprises decision problems with solutions verifiable in polynomial time by deterministic Turing machines
- NP-complete problems represent the most challenging problems in NP, allowing reduction of all other NP problems to them in polynomial time
- P forms a subset of NP due to polynomial-time solvable problems being inherently verifiable in polynomial time
- Solving any NP-complete problem in polynomial time would establish P = NP
- In reality, though, being able to solve a decision problem in polynomial time will often permit us to solve the corresponding optimization problem in polynomial time (using a polynomial number of calls to the decision problem).

References

1. Complexity Classes, Duc Duong, The Road Ahead (WordPress), 2017. https://duongquangduc.wordpress.com/2017/02/13/complexity-classes/
2. Computational Complexity Theory: Key Concepts, BotPenguin Team, BotPenguin, 2023. https://botpenguin.com/glossary/computational-complexity-theory.
3. Complexity Classes. Vedant Parvekar, Medium, 2023. https://medium.com/@parvekarvedant24/complexity-classes-630c8b7563f1
4. P, NP, CoNP, NP hard and NP complete | Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/types-of-complexity-classes-p-np-conp-np-hard-and-np-complete/
5. P, NP, NP-Complete, and NP-Hard, Baeldung Team, Baeldung, 2023. https://www.baeldung.com/cs/p-np-np-complete-np-hard.
6. 2-Satisfiability (2-SAT) Problem, GeeksforGeeks, GeeksforGeeks, 2024. https://www.geeksforgeeks.org/2-satisfiability-2-sat-problem/
7. Halting Problem in Theory of Computation, GeeksforGeeks, GeeksforGeeks, 2025. https://www.geeksforgeeks.org/halting-problem-in-theory-of-computation/
8. Turing Machine Halting Problem, TutorialsPoint Team, TutorialsPoint, 2025. https://www.tutorialspoint.com/automata_theory/turing_machine_halting_problem.htm
9. Introduction to NP-Complete Complexity Classes, GeeksforGeeks, GeeksforGeeks, 2024. https://www.geeksforgeeks.org/introduction-to-np-completeness/

Thank You!

For your attention