CS141: Intermediate Data Structures and Algorithms

NP-Completeness: A Brief Summary

Amr Magdy
Why Studying NP-Completeness?

Two reasons:

1. In almost all cases, if we can show a problem to be NP-complete or NP-hard, the best we can achieve (NOW) is mostly exponential algorithms.
   - This means we cannot solve large problem sizes efficiently

2. If we can solve only one NP-complete problem efficiently, we can solve ALL NP problems efficiently (major breakthrough)

More details come on what does these mean
Topic Outline

1. Decision Problems
2. Complexity Classes
   - P
   - NP
   - Polynomial Verification
   - Examples
3. NP-hardness
   - Polynomial Reductions
4. NP-Complete Problems
   - Definition and Examples
Decision Problems

- Decision problem: a problem expressed as a yes/no question
Decision Problems

- Decision problem: a problem expressed as a yes/no question
  - Examples:
    - Is graph G connected?
    - Is there a path $P: u \rightarrow v$ of cost 100?
    - Is there a common subsequence of strings A and B of length 5?
    - …etc
## Computation

### Decision Problems

<table>
<thead>
<tr>
<th>Decidable Problems: problems that can be decided using a Turing machine (our computer)</th>
<th>Undecidable Problems: problems that cannot be decided using a Turing machine.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example: the halting problem</td>
<td></td>
</tr>
</tbody>
</table>
Computation

Decision Problems are classified into different complexity classes based on time complexity and space complexity.

Decidable Problems: problems that can be decided using a Turing machine (our computer).

Undecidable Problems: problems that cannot be decided using a Turing machine.

Example: the halting problem
Complexity Class

» Complexity class: A set of problems that share some complexity characteristics
  » Either in time complexity
  » Or in space complexity
Complexity Class

- Complexity class:
  A set of problems that share some complexity characteristics
  - Either in time complexity
  - Or in space complexity

- In this course, our discussion is limited to only three time complexity classes: P, NP, and NP-hard
  - Other courses cover more content (e.g., Theory of Computation course)
P

- P is a complexity class of problems that are *decidable* (for simplicity *solvable*) in *polynomial-time*, i.e., $O(n^k)$
- where $n$ the input length and $k$ is constant
P

P is a complexity class of problems that are **decidable** (for simplicity **solvable**) in **polynomial-time**, i.e., $O(n^k)$

- where $n$ the input length and $k$ is constant

**Examples:**
- Shortest paths in graph
- Matrix chain multiplication
- Activity scheduling problem
- ....
NP

NP is a complexity class of problems that are verifiable in polynomial-time of input length.

For simplicity, given a solution of an NP problem, we can verify in polynomial time $O(n^k)$ if this solution is correct.
NP

NP is a complexity class of problems that are *verifiable* in *polynomial-time* of input length.

For simplicity, given a solution of an NP problem, we can verify in polynomial time $O(n^k)$ if this solution is correct.

Examples:
- Is bipartite graph? Given two subsets of nodes, verify it is bipartite.
- Max clique: Given a clique and $k$, verify it is actually a clique of size $k$.
- Shortest path: Given a path of cost $C$, verify it is a path and of cost $C$.
- .....
Is $P \subset NP$?

- Yes
- What does this mean?
  - Every problem that is solvable in polynomial time is verifiable in polynomial time as well
Is $P \subseteq NP$? or Is $P = NP$?

What does this mean?
Is $P \subseteq NP$? or Is $P = NP$?

- What does this mean?
  - There are polynomial time algorithms to solve NP problems
Is $P \subseteq NP$? or Is $P = NP$?

- What does this mean?
  - There are polynomial time algorithms to solve NP problems

- Nobody yet knows
  - The question posed in 1971
Is $P \subseteq NP$? or Is $P = NP$?

- What does this mean?
  - There are polynomial time algorithms to solve NP problems

- Nobody yet knows
  - The question posed in 1971
  - You think it is old?
    - Check Alhazen’s problem then
Is $P \subseteq NP$? or Is $P = NP$?

- What does this mean?
  - There are polynomial time algorithms to solve NP problems

- Nobody yet knows
  - The question posed in 1971
  - You think it is old?
    - Check Alhazen’s problem then

- Computer Science theoreticians “thinks” $P \neq NP$, but no proof
Is $P \subseteq NP$? or Is $P = NP$?

- What does this mean?
  - There are polynomial time algorithms to solve NP problems

- Nobody yet knows
  - The question posed in 1971
  - You think it is old?
    - Check Alhazen’s problem then

- Computer Science theoreticians “thinks” $P \neq NP$, but no proof

NP

\[
P \subseteq NP \quad NP = P
\]
We could solve the biggest problem in maths in the next decade

By Jacob Aron

One of the biggest open problems in mathematics may be solved within the next decade, according to a poll of computer scientists. A solution to the so-called P versus NP problem is worth $1 million and could have a profound effect on computing, and perhaps even the entire world.
NP-hard Problems

Informally: an NP-hard problem $B$ is a problem that is at least as hard as the hardest problems in NP class.

Formally: $B$ is NP-hard if $\forall A \in \text{NP}, A \leq_p B$ (i.e., $A$ is polynomially reducible to $B$).
Polynomial Reductions

- Polynomial reduction $A \leq_P B$ is converting an instance of $A$ into an instance of $B$ in polynomial time.
NP-Complete Problems

- B is NP-complete problem if:
  1. B ∈ NP
  2. B is NP-hard
NP-Complete Problems

NP-Hard

NP-Complete

NP

P

P \neq NP

Complexity

NP-Hard

P = NP = NP-Complete

P = NP

25
NP-Complete Problems: Examples

- CIRCUIT-SAT
  - SAT
  - 3-CNF-SAT
    - CLIQUE
    - SUBSET-SUM
      - VERTEX-COVER
        - HAM-CYCLE
        - TSP
NP-Complete Problems: Examples

- **Hamiltonian Cycle Problem**: Given an undirected or directed graph $G$, is there a cycle in $G$ that visits each vertex exactly once?
NP-Complete Problems: Examples

Example: Travelling Salesman Problem
Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?

![Diagram of a graph with cities A, B, C, and D connected by edges with distances labeled: A to B: 20, A to C: 42, A to D: 30, B to C: 35, B to D: 34, C to D: 12, C to B: 28]
NP-Complete Problems: Examples

Example: Travelling Salesman Problem
Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?

How to solve this problem?

![Graph showing cities A, B, C, D with distances between them.]

- A to B: 20
- A to C: 42
- A to D: -
- B to C: 30
- B to D: 35
- C to D: 12
- D to A: -
Example: Travelling Salesman Problem
Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?

How to solve this problem?
- Brute force: $O(n!)$
NP-Complete Problems: Examples

▷ Example: Travelling Salesman Problem
  Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city and returns to the origin city?

▷ How to solve this problem?
  ▶ Brute force: O(n!)
  ▶ Dynamic programming: O(n2^n)
Travelling Salesman Movie

https://www.youtube.com/watch?v=6ybd5rbQ5rU
NP-Complete Problems: Examples

Example: SAT Problem

Given a Boolean circuit $S$, is there a satisfying assignment for $S$? (i.e., variable assignment that outputs 1)
Example: **SAT Problem**

Given a Boolean circuit S, is there a satisfying assignment for S? (i.e., variable assignment that outputs 1)
NP-Complete Problems: Examples

Example: 3-CNF Problem
Given a Boolean circuit S in 3-CNF form, is there a satisfying assignment for S? (i.e., variable assignment that outputs 1)

3-CNF formula: a set ANDed Boolean clauses, each with 3 ORed literals (Boolean variables)

Example: \( \lor = \text{OR}, \land = \text{AND}, \neg = \text{NOT} \)
\[ (x_1 \lor \neg x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor x_3) \land (x_1 \lor x_2 \lor x_3) \]
Example: 3-CNF Problem
Given a Boolean circuit S in 3-CNF form, is there a satisfying assignment for S? (i.e., variable assignment that outputs 1)

3-CNF formula: a set ANDed Boolean clauses, each with 3 ORed literals (Boolean variables)

Example: \( \lor = \text{OR}, \land = \text{AND}, \lnot = \text{NOT} \)
\((x_1 \lor \lnot x_2 \lor \lnot x_3) \land (\lnot x_1 \lor x_2 \lor x_3) \land (x_1 \lor x_2 \lor x_3)\)

Solution: \(O(k2^n)\) for k clauses and n variables
NP-Complete Problems: Examples

Example: (Max) Clique Problem
Given a graph $G=(V,E)$, find the clique of maximum size.
Clique: fully connected subgraph.
NP-Complete Problems: Examples

Example: (Max) Clique Problem
Given a graph $G=(V,E)$ of $n$ vertices, find the clique of maximum size.
Clique: fully connected subgraph.

Solution:

- Assume max clique size $k$ and $|V| = n$
- Brute force: $O(n2^n)$
- Combinations of $k$: $O(n^k k^2)$
  - Try for $k=3,4,5,…$
- $k$ is not constant, so this is not polynomial
Book Readings

- Ch. 34