MTech CSE – 1st Semester

Student Name: SIDDHARTH KUMAR
Student ID: A125021

Question 14

Is the 2-SAT problem NP-hard? Can it be solved in polynomial time? Explain your answer.

Answer:

Summary

The 2-SAT problem is *not* NP-hard (unless **P** = **NP**). Moreover, 2-SAT can be solved efficiently in polynomial time, and in fact admits a linear-time algorithm based on graph techniques.

Definition of the 2-SAT Problem

The 2-SAT problem is a restricted form of the Boolean satisfiability problem in which the formula is written in conjunctive normal form (CNF) and each clause contains at most two literals.

A typical 2-CNF formula has the form:

$$(x_1 \vee \neg x_2) \ \wedge \ (\neg x_1 \vee x_3) \ \wedge \ (x_2 \vee x_3).$$

The objective is to determine whether there exists an assignment of truth values to the variables that satisfies all clauses.

Is 2-SAT NP-Hard?

Key Observation

2-SAT is not NP-hard unless $\mathbf{P} = \mathbf{NP}$.

Justification

While the general SAT problem and the restricted 3-SAT problem are NP-complete, 2-SAT belongs to a simpler class of problems.

If 2-SAT were NP-hard, then a polynomial-time algorithm for 2-SAT would imply polynomial-time solutions for all problems in **NP**. This would lead to the conclusion:

$$\mathbf{P} = \mathbf{NP},$$

which is widely believed to be false. Hence, 2-SAT is not NP-hard.

Polynomial-Time Solvability

2-SAT can be solved in linear time:

$$O(n + m),$$

where:

- n is the number of variables,
- m is the number of clauses.

This efficiency is achieved by transforming the problem into a graph reachability problem.

Implication Graph Construction

Each clause of the form:

$$(a \vee b)$$

is logically equivalent to the pair of implications:

$$(\neg a \Rightarrow b) \quad \text{and} \quad (\neg b \Rightarrow a).$$

Using this equivalence, we construct a directed graph, called the *implication graph*, where:

- each literal x and $\neg x$ is represented as a vertex,
- each implication corresponds to a directed edge.

Strongly Connected Components Criterion

Satisfiability Condition

A 2-SAT instance is satisfiable if and only if, for no variable x, both x and $\neg x$ appear in the same strongly connected component of the implication graph.

Reasoning

If both x and $\neg x$ belong to the same strongly connected component, then each implies the other. Assigning either truth value to x would therefore lead to a logical contradiction.

Algorithm Outline

The standard algorithm for solving 2-SAT proceeds as follows:

- Construct the implication graph.
- Compute its strongly connected components using Tarjan's or Kosaraju's algorithm.
- For each variable x, check whether x and $\neg x$ lie in the same SCC.

All steps can be performed in linear time.

Comparison with 3-SAT

The crucial difference between 2-SAT and 3-SAT lies in their structural properties:

- 2-SAT constraints naturally form implications that can be analyzed using graph algorithms.
- 3-SAT allows more complex interactions among variables, leading to combinatorial explosion and NP-completeness.

Thus, restricting clauses to at most two literals significantly reduces the computational complexity.

Final Conclusion

2-SAT is not NP-hard (unless $\mathbf{P} = \mathbf{NP}$) and can be solved in polynomial time. In fact, efficient linear-time algorithms exist using implication graphs and strongly connected components.

This makes 2-SAT a classic example of how imposing structural restrictions on a problem can dramatically simplify its computational complexity.