

CS3230 Semester 1 2025/2026
Design and Analysis of Algorithms

Tutorial 10
Reductions and Computational Complexity (cont.)
For Week 11

Document is last modified on: October 23, 2025

1 Lecture Review: Reductions and Computational Complexity (cont.)

So far, we have been told the following classes of problems:

- **P**: ... can be **solved** in polynomial time
- **NP**: ... can be **verified** in polynomial time
Prove **SOMETHING** is in NP:
 1. You can verify the ‘Yes’ instance in polynomial time via a certificate.
 2. State the certificate, then show it verifies the ‘Yes’ instance in polynomial time.
- **NP-hard**: ... can be **polynomial-time reduced from all** problems in NP
Prove **SOMETHING** is in NP-hard:
 1. Show that it is harder than (or equal to) any pre-existing NP-hard problem.
 2. Show that **A-PROVEN-NP-HARD-PROBLEM** \leq_p **SOMETHING**.
- **NP-complete**: ... is **both** in NP and is NP-hard
Prove **SOMETHING** is in NP-complete:
 1. Prove **SOMETHING** is in NP.
 2. Prove **SOMETHING** is NP-hard.

2 Tutorial 10 Questions

Q1). Which of the following imply $P = NP$?

- (a) There is a problem in P that is also NP-complete
- (b) There is a problem in P that is also in NP
- (c) There is a problem in NP that is also NP-hard

Q2) and Q3). involves SUBSET-SUM (see also <https://visualgo.net/en/reductions?slide=10>):

Given a multiset S of n (usually non-negative) Integers $\{S_1, S_2, \dots, S_n\}$ and an integer W , the SUBSET-SUM problem asks: Is there exists a subset $I \subseteq \{1, 2, \dots, n\}$ such that $\sum_{i \in I} S_i = W$?

For example, given $n = 5$, $S = \{5, 1, 5, 1, 4\}$, and $W = 7$, then it is a YES-instance with certificate indices $\{0, 1, 3\}$ or values $\{5, 1, 1\}$ that sums to 7.

We want to prove that SUBSET-SUM is NP-complete, using the usual two-steps proofs.

Q2). Prove that SUBSET-SUM is in NP.

Q3). Prove that SUBSET-SUM is NP-hard.

Hint: Use 3-SAT (see <https://visualgo.net/en/reductions?slide=5>).

Q4) and Q5). involves FIND-FAMILY:

- An undirected bipartite graph $G = (L \cup R, E)$ has two disjoint vertex sets L and R with each edge has one endpoint in L and another in R .
- We call a pair $u, v \in L$ as **siblings** if there exists a vertex $r \in R$ such that both edges (u, r) and (r, v) are present.
- A subset $F \subseteq L$ is said to be a **family** if for all distinct $u, v \in F$, u and v are siblings.
- The decision problem FIND-FAMILY is thus defined as follows:
Given an undirected bipartite graph $G = (L \cup R, E)$, is there a **family** of size $\geq k$?

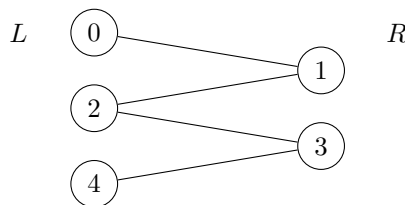


Figure 1: Example bipartite graph with $L = \{0, 2, 4\}$ and $R = \{1, 3\}$. Here, 0 and 2 are siblings, 2 and 4 are siblings, but $\{0, 2, 4\}$ is not a family since 0 and 4 are not siblings.

We want to prove that FIND-FAMILY is NP-complete, using the usual two-steps proofs.

Q4). Prove that FIND-FAMILY is in NP.

Q5). Prove that FIND-FAMILY is NP-hard.

Optional Q6) - time permitting. Discuss the following LeetCode task during tutorial:

TA can choose to discuss in high-level only or show the partial/full code (in C++/Python/Java)

- Wednesday class: combination-sum-ii
- Thursday class: ones-and-zeroes
- Friday class: coin-change-ii