CS3230 Semester 1 2024/2025

Design and Analysis of Algorithms

Tutorial 10
NP-completeness
For Week 12

Document is last modified on: November 4, 2024

1 Lecture Review: NP-completeness

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
- NP-**hard**: ... can be **polynomial-time reduced from all** problems in NP
- NP-**complete**: ... is **both** in NP and is NP-hard

To prove `SOMETHING` is NP-complete, we need to show that:

1. Prove `SOMETHING` is in NP
   You can verify the 'Yes' instance in polynomial time via a certificate.
   State the certificate, then show it verifies the 'Yes' instance in polynomial time.
2. Prove `SOMETHING` is NP-hard
   Show that it is harder than (or equal to) any pre-existing NP-hard problem
   Show that `A-PROVEN-NP-HARD-PROBLEM` $\leq_p$ `SOMETHING`

2 Tutorial 10 Questions

Q1). is hidden.
It is a simple question involving lec10.

Q2) and Q3). involves `SUBSET-SUM`

Definitions (also digitized at `https://visualgo.net/en/reductions?slide=10`):

- The decision problem `SUBSET-SUM` is defined as follows:
  Given a multiset S of n (usually non-negative) Integers $\{S_1, S_2, \ldots, S_n\}$ and an integer W, the `SUBSET-SUM` problem asks:
  Is there exists a subset $I \subseteq \{1, 2, \ldots, n\}$ 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.

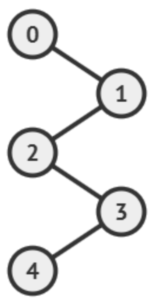
Q3). Prove that `SUBSET-SUM` is NP-hard.
Hint: Use `3-SAT` (digitized at `https://visualgo.net/en/reductions?slide=5`).

Q4) and Q5). involves `FIND-FAMILY`

Definitions (I am not sure if I want to digitize this at VisuAlgo (not a classic NP-complete problem)):

- 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 $\geq k$?

For example, given the following bipartite graph G with $L = \{0, 2, 4\}$ and $R = \{1, 3\}$, then 0 and 2 are siblings, 2 and 4 are siblings, but $\{0, 2, 4\}$ is not a family because 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.