Homework 4

Due: **3 December**, 2025 6pm PT

Unless stated otherwise, you should always prove the correctness of your algorithms and analyze their runtimes.

Each problem is graded on the coarse scale of ✓$^+$, ✓, ✓$^-$ and no ✓. It is also assigned a multiplier, denoting the relative importance of the problem. Both correctness and presentation are grading criteria.

Please read and make sure you understand the collaboration policy on the course missive. Extra credit problems are clearly marked below (see course missive for the details of grade calculations).

Remember to prove all your (non-elementary and not shown in class) mathematical claims, unless stated otherwise.

Each group of students should submit only 1 pdf to the corresponding Gradescope assignment.

Problem 1

(6 ✓s total)

A CEO is hosting a networking event to help people come up with startup ideas. She has certain ideas about how her guests should interact. From a list of n potential guests, she knows which guests already know each other, and she wants to use this information to figure out who to invite to this event. In each of the following *distinct, unrelated* cases, determine whether there is a polynomial time algorithm for her, or whether the problem is NP-hard. State **without proof** the polynomial-time algorithm, or **prove NP-hardness**.

1. (2 ✓s) The CEO wants to ensure that everyone has a group to hang out with at the mixer: each guest should know at least 1 other guest. She wants to maximize the number of guests at the party.

2. (2 ✓s) She wants to ensure both that everyone has a group to hang out with and that everyone can meet new people: each person should (1) know at least 1 other guests; and (2) not know at least 1 other guests. She wants to maximize the number of guests at the event.

3. (2 ✓s) But her main goal is to help her guests "network": in this part, each guest must **not** know any other guest at the event. She wants to maximize the number of guests at the event.

Problem 2

(4 ✓s)

A comparison-based algorithm is an algorithm where, instead of being able to do all kinds of numerical operations (e.g. addition, subtraction, multiplication, modulo division), the only numerical operation it can do is to compare two elements a, b and check if $a < b$, $a = b$ or $a > b$.

For the purposes of the following problem, we will restrict attention only to deterministic (non-randomized) algorithms.

It is a well-known fact that if a sorting algorithm is comparison-based, then it has running time at least $\Omega(n \log n)$ when given an input array of size n.

(See `https://www.cs.cmu.edu/~avrim/451f11/lectures/lect0913.pdf`)

The focus on this problem is instead on heaps and priority queues. Recall that a heap/priority queue is a data structure supporting (at least) the following operations:

- Insert(*value*): inserts a value into the heap.
- ExtractMax(): removes the maximum value in the heap and returns it.

The simple binary heap implementations you have seen in intro classes take $\Theta(\log n)$ time for both operations. On the other hand, more sophisticated heap data structures, such as a strict Fibonacci heap, achieves $O(1)$ insertion time while maintaining $O(\log n)$ time for ExtractMax. A natural question, then, is whether it is possible to achieve $O(1)$ time for both operations in the same data structure.

Use the fact about comparison-based sorting to show that a comparison-based heap/priority queue data structure cannot be too efficient. Show that there can be no comparison-based heap/priority queue that implements both operations in $O(1)$ time (in fact, not even $o(\log n)$ time for both operations, if you recall what the little-o notation means).

Problem 3

(**Do not submit**, only an exercise for anyone interested)

Consider the CEO problem again, trying to invite guests to a networking event. Her main goal is to help her guests "network", but it is now ok if her guests know *some* people: in this extra problem, each guest must know at most 1 other guest at the event (the previous problem has the number 0 instead of 1). She wants to maximize the number of guests at the event.

Show that this problem is NP-hard (in fact, the problem remains NP-hard if we replace the number 1 by any larger integer). The reduction is not straightforward like those presented in class, and does require some work, so this problem is "out of scope" for the course.