Precept Outline

- Review of Lectures 23 and 24:
  - Intractability
  - Algorithm Design

A. Review: Intractability

Your preceptor will briefly review key points of this week’s lectures.

B. P and NP

Suppose that problems X and Y are both in **NP** and X poly-time reduces to Y. Which of the following can you infer?

1. ( ) If X can be solved in poly-time, then **P** = **NP**.
2. ( ) If Y can be solved in polynomial time, then so can X.
3. ( ) If Y can be solved in polynomial time, then **P** $=$ **NP**.
4. ( ) Y is in **P**.
5. ( ) If **P** $=$ **NP**, then X can be solved in polynomial time.
6. ( ) Y is not in **P**.
7. ( ) st-connectivity poly-time reduces to Y.
8. ( ) If **P** $=$ **NP**, then Y can be solved in polynomial time.
9. ( ) If **P** $\neq$ **NP**, then X cannot be solved in polynomial time.
10. ( ) If X cannot be solved in polynomial time, then **P** $\neq$ **NP**.

C. Algorithm Design via Reductions: Princeton MSTs (Fall’23 Final)

Consider the classical minimum spanning tree problem and a variant:

- **Classic-MST**: given a connected edge-weighted graph G, find a spanning tree of G with minimal total weight.
- **Princeton-MST**: given an edge-weighted graph G with each edge colored orange or black, find a spanning tree of G that has minimum total weight among all spanning trees that contain all of the orange edges (or report that no such spanning tree exists).

Design an efficient algorithm to solve the Princeton-MST problem on an edge-weighted and edge-colored graph G. *To do so, you* ***must*** *model it as a Classic-MST problem on a closely related edge-weighted graph* G'.

For full credit, your algorithm must run in $O(E \log E)$ time, where V and E are the number of vertices and edges in G, respectively.

D. Optional bonus problems

E. Detecting Biased Dice

Suppose you have k 6-sided uneven dice. Because they are uneven, each outcome has a different probability p_i, which is given to you. These are always valid probabilities, meaning $p_1 + p_2 + p_3 + p_4 + p_5 + p_6 = 1$ and they are all non-negative numbers (but some could be 0!).

Describe an algorithm that finds the maximum possible sum of the outcomes obtainable by rolling all k dice, as well as the probability of obtaining that sum. The running time of your algorithm should be $\Theta(k)$.

In the space provided, give a concise English description of your algorithm for solving the problem. You may use any of the algorithms that we have considered in this course (e.g., lectures, precepts, textbook, assignments) as subroutines. If you modify such an algorithm, be sure to describe the modification. Feel free to use code or pseudocode to improve clarity.

Given an integer X, your task is to determine what is the probability that the sum of the outcomes of rolling the k dice is exactly X. Describe an algorithm to do so with running time of $\Theta(X \cdot k)$.

In the space provided, give a concise English description of your algorithm for solving the problem. You may use any of the algorithms that we have considered in this course (e.g., lectures, precepts, textbook, assignments) as subroutines. If you modify such an algorithm, be sure to describe the modification. Feel free to use code or pseudocode to improve clarity.