Homework 10 – Due Thursday, April 25 at 11:59 PM

Reminder Collaboration is permitted, but you must write the solutions *by yourself without assistance*, and be ready to explain them orally to the course staff if asked. You must also identify your collaborators and write "Collaborators: none" if you worked by yourself. Getting solutions from outside sources such as the Web or students not enrolled in the class is strictly forbidden. Collaboration is not allowed on problems marked "INDIVIDUAL."

Note You may use various generalizations of the Turing machine model we have seen in class, such as TMs with two-way infinite tapes, stay-put, or multiple tapes. If you choose to use such a generalization, state clearly and precisely what model you are using. **You may describe Turing machines at a high-level on this assignment.**

Problems There are 4 required problems.

1. **(Closer vertices)** If $G = (V, E)$ is an undirected graph, and u, v are vertices in V, let $d_G(u, v)$ denote the length of the shortest path from u to v in G. (Define $d_G(u, v) = \infty$ if no path exists.) Let $CLOSER = \{\langle G, u, v, w\rangle \mid G \text{ is an undirected graph}, u, v, w \text{ are vertices, and } d_G(u, v) < d_G(u, w)\}$. This corresponds to the following computational problem: Given a graph G and vertices u, v, w, is u closer to v than it is to w?

   Show that $CLOSER \in \mathsf{P}$ by i) giving a high-level description of a polynomial-time algorithm deciding $CLOSER$, ii) analyzing the correctness of your algorithm, and iii) explaining why your algorithm runs in polynomial time.

   You don't need to specify the exact polynomial runtime that your algorithm runs in, since this may depend on implementation details that are suppressed in a high-level description. Just give a convincing argument that the runtime is polynomial as in the examples in Chapter 7.2 of Sipser.

2. **(Project scheduling)** You are trying to plan a schedule for completing your course projects next semester. You will be given k course projects. For each project i, you know the the release time r_i, the length of time ℓ_i you have to spend on it, and the due date d_i. You would like to determine whether you can find an *uninterrupted schedule* where you are working on *exactly one* project at a time from the beginning to the end and satisfying all the deadlines. For simplicity, you can think of all numbers involved as positive integers. For example, if you have Project 1 released at time 1, due at time 11, taking 2 time units, Project 2 released at time 3, due at time 9, taking 4 time units, and Project 3 released at time 2, due at time 7, taking 3 time units, you can have an uninterrupted schedule as follows: you do Project 3 from 2 to 5, Project 2 from 5 to 9, and Project 1 from 9 to 11.

   We formalize this problem as a language as follows: $PS = \{\langle k, (r_1, \ell_1, d_1), \ldots, (r_k, \ell_k, d_k)\rangle \mid$ all numbers in the input are positive integers and there exists an uninterrupted schedule$\}$.

   Show that $PS \in \mathsf{NP}$ by giving a **polynomial-time nondeterministic algorithm**. Make sure to explain why your algorithm is correct and why it runs in (nondeterministic) polynomial time.

3. **(NTM satisfiability)**

   (a) Consider the language

   $$TMSAT = \{\langle N, w, 1^t\rangle \mid \text{ NTM } N \text{ accepts input } w \text{ within } t \text{ steps}\}.$$

   Prove that $TMSAT \in \mathsf{NP}$ by giving a **polynomial-time verifier** for it.

   (b) If t was written in binary, would your solution to part (a) still work? Why or why not?

4. (**Decision vs. search**) You are consulting for a record label that wants to improve the morale of all the formulaic pop bands it's cranking out. A pop band consists of a vocalist, a guitarist, and a drummer. The record label sends you sets X, Y, and Z, with r individuals each, corresponding to the three roles. You also receive a set M of triples, where each triple corresponds to a potential pop band where all 3 individuals can perform together without fighting. An individual can appear in multiple triples. The record label is asking you to find a way to arrange the largest possible number of happy pop bands. That is, your goal is to select a largest subset of triples from M so that each individual appears in at most one triple. (There may be more than one largest subset, in which case, you are free to pick any largest subset.)

   In this problem, you will prove that if $\mathsf{P} = \mathsf{NP}$, you can *find* a largest set of happy pop bands in polynomial time.

   (a) Consider the following language:
   $BAND = \{\langle X, Y, Z, M, k\rangle \mid |X| = |Y| = |Z|$, each element of M is a triple (x, y, z) where $x \in X, y \in Y$ and $z \in Z$, and M contains a subset of size k where each element appears in at most one triple$\}$.
   Prove that $BAND \in \mathsf{NP}$. You can either give a nondeterministic poly-time algorithm or a poly-time verifier.

   (b) If $\mathsf{P} = \mathsf{NP}$, the solution you gave to part (a) implies that $BAND \in \mathsf{P}$. That is, in polynomial time you can decide whether it is possible to have k happy pop bands. Assume you have a subroutine that does it, and use it repeatedly to *find* a largest set of happy pop bands in polynomial time.
   Hint: Similar to Solved Problem 7.40 in Sipser.