Statement of Integrity: I, Derek Zhu, state that I completed this assignment with integrity and by myself.

Homework 6 is here, Due: Aug 19, 2024, to be submitted at here

Problem 1:

(a) **Variables**

We can chose x_{ij} as our variables to represent the assignment of detections to tracks, where $x_{ij} = 1$ if detection i is assigned to track j. Here i ranges from 1 ro M (number of detections), and j ranges from 1 to N (number of tracks).

Since each x_{ij} is a variable, and there are M detections and N tracks, therefore, the total number of variables is $M * N$.

(b) **Objective**

The objective here is to maximize the sum of the similarity scores for the assigned detections and the tracks. The similarity scores are given by matrix S, where s_{ij} represent the similarity score between detection i and track j. Here is the objection function:

$$f = Maximize\,(\sum_{i=1}^{M}\sum_{j1}^{N} s_{ij} * x_{ij})$$

(c) **Constraints**

- Detection Constraint (each detection is assigned to at most one track): $\sum_{j=1}^{N} x_{ij} \leq 1,\ \forall i \in \{1, 2, ..., M\}$, hence there are M detection constraints.
- Track Constraint (each track is assigned to at most one detection): $\sum_{i=1}^{M} x_{ij} \leq 1,\ \forall j \in \{1, 2, ..., N\}$, hence there are N track constraints.
- Value Constraint (each variables are either 1 to indicate a detection is assigned to a track or 0 to indicate detection is not assigned to a track): $x_{ij} \in \{0, 1\},\ \forall i \in \{1, 2, ..., M\}, \forall j \in \{1, 2, ..., N\}$, hence there are M*N value constraint.
- Total constraints: $M + N + M * N$

Problem 2:

Concept **review** from the guide here provided in this assignment:

$\leq_P$ Polynomially reducible. If we say $X \leq_P Y$, that means there exists an algorithm that runs in polynomial time that converts an instance of problem X into problem Y. More formally, there exists some $f(X) \to Y$ (also written as $f : X \to Y$) where $f(X)$ runs in polynomial—often linear—time.

To prove that $\leq p$ is a transitive relation on languages, we need to show that if $L_1 \leq p\, L_2$ and $L_2 \leq p\, L_3$, then $L_1 \leq p\, L_3$

Given: $L_1 \leq p\, L_2$ and $L_2 \leq p\, L_3$ means:

$x \in L_1 \Leftrightarrow f\ (x) \in L_2$ and $y \in L_2 \Leftrightarrow g(y) \in L_3$

Where $f\ (x)\ and\ g(y)$ are polynomial-time computable functions.

We construct a new composition function $h(z) = g(f\ (z))$. Since $f(x)\ and\ g(y)$ are polynomial-time computable functions, hence the composition function $h(z)$ is also a polynomial-time computable function.

Since:

$y \in L_2 \Leftrightarrow g(y) \in L_3$ (given)

henece: $f(x) \in L_2 \Leftrightarrow g(f(x)) \in L_3$

Therefore: $f(x) \in L_2 \Leftrightarrow h(x) \in L_3$ (by definition of h(x))

Since $x \in L_1 \Leftrightarrow f\ (x) \in L_2$ (given)

Therefore $x \in L_1 \Leftrightarrow f\ (x) \in L_2 \Leftrightarrow h(x) \in L_3$

That means: $x \in L_1 \Leftrightarrow h(x) \in L_3$

That means if $L_1 \leq p\, L_2$ and $L_2 \leq p\, L_3$, then $L_1 \leq p\, L_3$

Proved!

Problem 3:

To obviousely use the method in NP Completeness Short Guide here, we define X,Y as:
Let X to be the Efficient Recruiting problem of this homework;
Let Y to be the known vertex cover problem.

Step 1 (from the short guide: prove $X \in NP$ by describing a polynomial-time algorithm that verifies a solution
of X):

Problem Statement: Given k counselors and n sports, is it possible to hire at most k counselors such that each sport has at least one counselor qualified in it?
Certificate: A subset of at most k counselors.
Verification: Check that each of the n sports has at least one counselor from the subset who is qualified for that sport. This involves checking each sport against the subset of counselors, which can be done in $O(n * k)$ time.
Thus, $X \in NP$ because we can verify a solution in polynomial time.

Step 2 (from the short guide: identify a known problem $Y \in NPC$)

We choose Y for our reduction. Given a graph $G = (V, E)$ and an integer k, is there a subset $V' \subseteq V$ with $|V'| \leq k$ such that every edge in E has at least one endpoint in V'?

Step 3 (from the short guide: describe an algorithm that maps Y into X in polynomial time. Formally stated,prove that X is NP-hard by showing $Y \leq P\ X$):

We construct an instance of X from an instance of Y such that solving the X instance will solve the Y instance.

Given a graph $G = (V, E)$ from Y, construct an instance of X as follows: each vertex v_i in V represents a counselor; each edge $e_j = (v_i, vl_l)$ in E represents a sport that requires counselors v_i and v_j to cover it; the set of m applicants corresponds to the set of vertices V; the set of n sports corresponds to the set of edges E.

Step 4 (from the short guide: prove that a solution of X maps to a solution of Y and vice-versa, in polynomial time):

(a) (from the short guide: Find a polynomial-time function f that maps a solution of problem X to a solution of problem Y. In math-ese, $\exists f \mid x \in X \Leftrightarrow f(x) \in Y$):
Given an instance (G,k) of Y: Construct the corresponding X instance with $m = |V|$ counselors and $n = |E|$ sports. Each counselor is qualified to cover sports corresponding to the edges they are incident to in G.

(b) (from the short guide:find a polynomial-time function g that does the reverse of f, mapping a solution of Y to a solution of X):

Given an instance of X, map it back to Y: The counselors selected in the X solution correspond to the vertices in the Y solution. Each sport covered by a counselor in X corresponds to an edge of G covered by the vertex in the Y.

(c) (from the short guide: make sure you discuss the running time of each, verifying that these functions run in polynomial time):

Both transformations run in polynomial time since they involve simple mappings of vertices to counselors and edges to sports.

By showing the polynomial-time mappings between Y (the known vertex cover problem) and X (the Efficient Recruiting problem), we have proven that X is NP-complete. If there were a polynomial-time algorithm for X, there would be one for Y, implying P=NP. Since Y is NP-complete, X must also be NP-complete