This homework is due at 10:00PM on Sunday, November 24. Write your solution using \LaTeX. Submit this homework using github as a \texttt{.tex} file. This is a \textbf{partnered homework}. You should primarily be discussing problems with your homework partner.

It’s ok to discuss approaches at a high level with others. However, you should not reveal specific details of a solution, nor should you show your written solution to anyone else. The only exception to this rule is work you’ve done with a lab partner \textit{while in lab}. In this case, note (in your homework submission poll) who you’ve worked with and what parts were solved during lab.

1. **Multiple-Interval-Scheduling** (K&T 8.14) In this problem, there is a machine that is available to run jobs over some period of time, say 9AM to 5PM. People submit jobs to run on the processor; the processor can only work on one job at any single point in time. However, in this problem, each job requires a set of intervals of time during which it needs to use the machine. Thus, for example, one job could require the processor from 10AM to 11AM and again from 2PM to 3PM. If you accept this job, it ties up your machine during these two hours, but you could still accept jobs that need any other time periods (including the hours from 11AM to 2PM).

Now, you’re given an integer $k$ and a set of $n$ jobs, each specified by a set of time intervals, and you want to answer the following question: is it possible to accept at least $k$ of the jobs so that no two of the accepted jobs have any overlap in time?

In this problem, you are to show that $\text{Multiple-Interval-Scheduling} \in \text{NP-complete}$. To assist you, we’ve broken down this problem into smaller parts:

(a) First, show that $\text{Multiple-Interval-Scheduling} \in \text{NP}$. 
(b) In the remaining two parts, you will reduce $\text{Independent-Set} \leq_p \text{Multiple-Interval-Scheduling}$.

Given input $(G = (V, E), k)$ for \text{Independent-Set}, create a valid input for \text{Multiple-Interval-Scheduling}. First, divide the processor time window into $m$ distinct and disjoint intervals $i_1, \ldots, i_m$. Associate each interval $i_j$ with an edge $e_j$. Next, create a different job $J_v$ for each vertex $v \in V$. What set of time intervals should you pick for job $J_v$?

(c) Finally, run the \text{Multiple-Interval-Scheduling} algorithm on the input you create, and output \text{yes} iff the \text{Multiple-Interval-Scheduling} algorithm outputs \text{yes}. Argue that the answer to \text{Multiple-Interval-Scheduling} gives you a correct answer to \text{Independent-Set}.

2. **Intersection-Inference** (K&T 8.16) Consider the problem of reasoning about the identity of a set from the size of its intersections with other sets. You are given a finite set $U$ of size $n$, and a collection $A_1, \ldots, A_m \subset U$ of subsets of $U$. You are also given integers $c_1, \ldots, c_m$. The question is: does there exist $X \subset U$ such that for each $i = 1, 2, \ldots, m$, the cardinality of $X \cap A_i$ equals $c_i$? We will call this an instance of the \text{Intersection-Inference} problem, with input $U, \{A_i\}, \{c_i\}$.

Prove that $\text{Intersection-Inference} \in \text{NP-complete}$, \textbf{Hint}: reduce from the following problem, which you may assume is \text{NP-complete}:

\textbf{Problem One-In-Three-Sat:}

\textbf{Inputs:} $n$ variables $x_1, \ldots, x_n$ and $m$ clauses $c_1, \ldots, c_m$ where each clauses is the or of three literals e.g., $c_i = (x_1 \lor \bar{x_2} \lor x_3)$.

\textbf{Output:} \text{yes} iff there is a truth assignment to the variables such that for each clauses there is \text{exactly} one satisfied variable.

\textbf{Hint:} Let $U$ be the set of literals. You’ll have to work to ensure that a variable and its negation cannot both end up in $X$. 

1
3. In the Four-Coloring problem, the input is a graph $G = (V, E)$, and you should output YES iff the vertices in $G$ can be colored using at most four colors such that each edge $(u, v) \in E$ is bichromatic. Prove that Four-Coloring \( \in \text{NP-complete}. \)

4. (Extra Challenge Problem.) Show that One-In-Three-Sat \( \in \text{NP-complete}. \)

5. (Extra Challenge Problem.) Does $P = \text{NP}$? Answer YES or NO. Justify your response with a formal proof.