Homework 11 – Due Tuesday, April 29, 2025 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.

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.

Problems There are 5 required problems.

1. (**Poly-time Reductions**) For each of the following, give a language (if it exists) with the stated property. Explain why your language has the given property. If you think no such language can exist, explain why, and in particular, explain whether your answer depends on whether $\mathsf{P} = \mathsf{NP}$.

   (a) $A \leq_{\mathrm{p}} \overline{A}$.

   (b) $SAT \leq_{\mathrm{p}} B$ and B is not NP-complete.

   (c) $C \leq_{\mathrm{p}} SAT$ and C is NP-hard.

   (d) $D \in \mathsf{TIME}(n^2)$ and is NP-complete.

2. (**NP-Completeness Mad-Libs**) Recall the language HUB from Homework 10 defined as follows. A *valid course plan* is a collection T of courses that, taken together, supply you with all m Hub requirements for some natural number m: $\cup_{i \in T} S_i = [m]$. Define the language $HUB = \{\langle S_1, \ldots, S_k, r\rangle \mid$ there exists a valid course plan $T \subseteq [k]$ of size $|T| \leq r\}$.

   Now we will argue that HUB is NP-hard by giving a reduction showing $VERTEX - COVER \leq_{\mathrm{p}} HUB$. (See page 312 of Sipser for discussion of this problem.) A vertex cover of a graph G is a set of vertices T such that every edge in the graph is incident to at least one vertex in T. The language $VERTEX - COVER = \{\langle G, r\rangle \mid G$ has a vertex cover of size at most $r\}$.

   In the reduction described below, fill in the blank labeled (ii) with a description of what the algorithm computing the reduction should output.

Algorithm: VERTEX-COVER to HUB Reduction

Input : $\langle G, r\rangle$ where $G = (V, E)$ is a graph and $r \in \mathbb{N}$

1. Relabel the vertices and edges of the graph so that $V = [k]$ and $E = [m]$.
2. For each $i = 1, \ldots, k$:
   Let $S_i = \{j \in [m] \mid$ edge j is incident to vertex $i\}$
3. Output __(ii)__.

Your job is now done, but here are explanations of correctness and runtime for this reduction.

Correctness: If $\langle G, r\rangle \in VERTEX - COVER$, then there exists a set T of at most r vertices such that every edge in the graph is incident to a member of T. After relabeling, that means T is

a set of courses such that every requirement in $[m]$ appears in at least one of the sets S_i, so T is a valid course plan of size at most r. Conversely, if there is a valid course plan T of size at most r in the instance of HUB produced, then T corresponds to a set of vertices such that every edge in G is incident to a member of T, and hence $\langle G, r\rangle \in VERTEX - COVER$.

Runtime: Suppose for concreteness that we are working with the adjacency list representation of G on a multi-tape. Inside the main loop of step 2, constructing each set S_i takes time linear in m, the number of edges of the graph. So overall, the algorithm runs in time $O(km + \log r)$ which is polynomial in the description length of the input.

3. (**Satisfiability**) Recall the language $XSAT = \{\langle \varphi_1, \varphi_2\rangle \mid$ there exists an assignment x satisfying exactly one of $\varphi_1, \varphi_2\}$ from Homework 10. There, you showed that $XSAT \in \mathsf{NP}$. Show that $SAT \leq_{\mathrm{p}} XSAT$. Explain why your reduction is correct and why it runs in **deterministic** polynomial time. Explain how you can conclude from this that $XSAT$ is NP-complete.

4. (**Boolean Roots**) Let $p(x_1, \ldots, x_n)$ be an n-variate polynomial with integer coefficients. A *boolean root* of p is point $(b_1, \ldots, b_n) \in \{0,1\}^n$ such that $p(b_1, \ldots, b_n) = 0$. For example, $(0, 1, 1)$ is a boolean root of the polynomial $7x_1^2 + 2x_2x_3 - 2x_3^8$. Define the language $BROOT = \{\langle p\rangle \mid$ p is an integer polynomial that has at least one boolean root$\}$.

   (a) Show that $BROOT \in \mathsf{NP}$ by either describing a poly-time NTM or a deterministic verifier. Make sure to explain runtime and correctness.

   (b) Show that $3SAT \leq_{\mathrm{p}} BROOT$ and conclude that $BROOT$ is NP-complete. Make sure to explain runtime and correctness.

   Hint: First determine how to transform a single clause $u \vee v \vee w$ into a polynomial $p(u, v, w)$ such that $p(u, v, w) = 0$ iff at least one of u, v, w is set to 1. Then create a polynomial q that evaluates to 0 if and only if all of its inputs are 0. Finally, use q to combine the individual polynomials that correspond to the clauses of your $3SAT$ instance.

5. (**Hiring Influencers**) You manage a trendy t-shirt company and wish to recruit social media influencers to help advertise your brand. There are k influencers, labeled $i = 1, \ldots, k$, and each has a number of followers $f_i \in \mathbb{N}$. You wish to hire a set of influencers $T \subseteq [k]$ so as to maximize the total follower count $\sum_{i \in T} f_i$ reached. However, there is an important caveat: If you hire two influencers who are currently engaged in a feud, all of the resulting drama will destroy the reputation of your brand. Let E denote a list of unordered pairs of the form $\{i, j\}$ where influencers i and j are feuding.

   Consider the following language: $INF = \{\langle f_1, \ldots, f_k, E, r\rangle \mid$ there exists $T \subseteq [k]$ such that $\sum_{i \in T} f_i \geq r$ and $\{i, j\} \notin E$ for all $i, j \in T\}$. This corresponds to the problem of determining whether you can hire influencers with total follower count at least r such that none of them are feuding.

   (a) Show that $INF \in \mathsf{NP}$ by either describing a poly-time NTM or a deterministic verifier. Make sure to explain runtime and correctness.

   (b) Show that INF is NP-complete. Make sure to explain runtime and correctness of your reduction.

   Hint: Reduce from $IND - SET$.

6. (**Bonus**) In a directed graph, the ***indegree*** of a node is the number of incoming edges and the ***outdegree*** of a node is the number of outgoing edges. Show that the following problem is NP-complete. Given an undirected graph G and a subset S of the nodes in G, determine whether it

possible to convert G to a directed graph by assigning directions to each of its edges so that every node in S has indegree 0 or outdegree 0, and all remaining nodes in G have indegree at least 1.