HW 11

Due on **Wednesday, November 30, 2022 at 9am CST**

CS/ECE 374A: Intro. Algorithms & Models of Computation, Fall 2022 Version: **1.12**

Submission instructions as in previous homeworks.

21 (100 PTS.) My friend, departing time is pending.

The following question is long, but not very hard, and is intended to make sure you understand the following problems, and the basic concepts needed for proving NP-Completeness.

All graphs in the following have n vertices and m edges.

For each of the following problems, you are given an instance of the problem of size n. Imagine that the answer to this given instance is "yes", and that you need to convince somebody that indeed the answer to the given instance is `yes`. To this end, describe:

(I) An algorithm for solving the given instance (not necessarily efficient). What is the running time of your algorithm?
(II) The format of the proof that the instance is correct.
(III) A bound on the length of the proof (its have to be of polynomial length in the input size).
(IV) An efficient algorithm (as fast as possible [it has to be polynomial time]) for verifying, given the instance and the proof, that indeed the given instance is indeed `yes`. What is the running time of your algorithm?

(EXAMPLE)

Shortest Path

Instance: A weighted undirected graph G, vertices s and t and a threshold w.
Question: Is there a path between s and t in G of length at most w?

Solution:

(I) **Algorithm**: We seen in class the Dijkstra algorithm for solving the shortest path problem in $O(n \log n + m) = O(n^2)$ time. Given the shortest path, we can just compare its price to w, and return yes/no accordingly.
(II) **Certificate**: A "proof" in this case would be a path π in G (i.e., a sequence of at most n vertices) connecting s to t, such that its total weight is at most w.
(III) **Certificate length**: The proof here is a list of $O(n)$ vertices, and can be encoded as a list of $O(n)$ integers. As such, its length is $O(n)$.
(IV) **Verification algorithm**: The verification algorithm for the given solution/proof, would verify that all the edges in the path are indeed in the graph, the path starts at s and ends at t, and that the total weight of the edges of the path is at most w. The proof has length $O(n)$ in this case, and the verification algorithm runs in $O(n^2)$ time, if we assume the graph is given to us using adjacency lists representation.

21.A. (20 PTS.)

Friendly Set

> **Instance**: An undirected graph G, integer k
> **Question:** Is there a friendly set in G of size k?

A set $X \subseteq \mathsf{V}(\mathsf{G})$ is ***friendly*** if every vertex has at least two vertices in X that are its neighbors. Formally, for a vertex $u \in \mathsf{V}(\mathsf{G})$, let $\Gamma(u) = \{v \mid uv \in \mathsf{E}(\mathsf{G})\}$ be its set of ***neighbors***. The set X is friendly if for all $v \in \mathsf{V}(\mathsf{G})$, we have that $|\Gamma(v) \cap X| \geq 2$.

21.B. (20 PTS.)

Max Triangle Free

> **Instance**: An undirected graph G with n vertices and m edges, a parameter k.
> **Question:** Is there a subset S of k edges in the graph, such that no three edges of S form a triangle?

A ***triangle*** is a cycle of length 3.

21.C. (20 PTS.)

Wiggle to target

> **Instance**: S: Set of positive integers. t: An integer number (target).
> **Question:** Are there disjoint subsets $X, Y, Z \subseteq S$ such that $\sum_{x \in X} x - \sum_{y \in Y} y + 2\sum_{z \in Z} z = t$?

21.D. (20 PTS.)

NotMuchOverlap

> **Instance**: X a set of n elements, $\mathcal{F} = \{f_i \subseteq X \mid i = 1, \ldots, m\}$, and a parameter α.
> **Question:** Is there a subset $S \subseteq \mathcal{F}$ of α sets, such that no pair of sets $f, g \in S$ share more than one element.

21.E. (20 PTS.)

SET DISJOINT PACKING

> **Instance**: $(\mathsf{U}, \mathcal{F}, k)$:
> U: A set of n elements
> $\mathcal{F}$: A family of m subsets of U, s.t. $\bigcup_{X \in \mathcal{F}} X = \mathsf{U}$.
> k: A positive integer.
> **Question:** Are there k pairwise-disjoint sets $S_1, \ldots, S_k \in \mathcal{F}$?

Formally, the sets $S_1, \ldots, S_k$ are ***pairwise-disjoint*** if for all $i \neq j$, we have that $S_i \cap S_j = \emptyset$.

22 (100 PTS.) Set picking.

SET PICKING

> **Instance**: A set $U = \{1, \ldots, m\}$, and sets $\mathcal{F} = \{f_1, \ldots, f_n, g_1, \ldots, g_n \subseteq U\}$.
> **Question:** Is there a good selection set $\mathcal{X} \subseteq \mathcal{F}$?

A ***good selection*** is a set $\mathcal{X} \subseteq \mathcal{F}$, such that $|\mathcal{X} \cap \{f_i, g_i\}| = 1$, for all i, and furthermore $\cup\mathcal{X} = \cup_{h \in \mathcal{X}} h = U$.

22.A. (20 PTS.) Prove that SET PICKING is in NP.

22.B. (40 PTS.) Show a polynomial time reduction from SET PICKING to SAT. (You need to prove your reduction is correct.)

22.C. (40 PTS.) Show a polynomial time reduction from SAT to SET PICKING. (You need to prove your reduction is correct.)