CS 473: Algorithms, Spring 2018
HW 7 (due Wednesday, March 31st at 8pm)

This homework contains three problems. **Read the instructions for submitting homework on the course webpage**.

Collaboration Policy: For this home work, each student can work in a group with up to three members. Only one solution for each group needs to be submitted. Follow the submission instructions carefully.

For problems that use maximum flows as a black box, a full-credit solution requires the following.

- A complete description of the relevant flow network, specifying the set of vertices, the set of edges (being careful about direction), the source and target vertices s and t, and the capacity of every edge. (If the flow network is part of the original input, just say that.)
- A description of the algorithm to construct this flow network from the stated input. This could be as simple as "We can construct the flow network in $O(n^3)$ time by brute force."
- A description of the algorithm to extract the answer to the stated problem from the maximum flow. This could be as simple as "Return TRUE if the maximum flow value is at least 42 and False otherwise."
- A proof that your reduction is correct. This proof will almost always have two components. For example, if your algorithm returns a boolean, you should prove that its TRUE answers are correct and that its FALSE answers are correct. If your algorithm returns a number, you should prove that number is neither too large nor too small.
- The running time of the overall algorithm, expressed as a function of the original input parameters, not just the number of vertices and edges in your flow network.
- You may assume that maximum flows can be computed in $O(VE)$ time. Do *not* regurgitate the maximum flow algorithm itself.

Reductions to other flow-based algorithms described in class or in the notes (for example: edge-disjoint paths, maximum bipartite matching, minimum-cost circulation) or to other standard graph problems (for example: reachability, minimum spanning tree, shortest paths) have similar requirements.

---

0. **Not to submit:** Please do Problems 1 and 2 from `https://courses.engr.illinois.edu/cs473/sp2017/homework/hw6.pdf`.

1. Consider a directed network $G = (V, E)$ with capacity $c(e)$ on edge $e \in E$, and a feasible s-t flow $f : E \to \mathbb{R}^+$.

   We say that flow f is *acyclic* if the subgraph of directed edges with positive flow contains no directed cycle.

   (a) Show that given any feasible flow f in G, there is a feasible acyclic flow of the same value (This implies that some maximum flow is acyclic).

   (b) Show that if f is an integral flow in G, then there is a feasible acyclic integral flow of the same value.

2. Let $G = (V, E)$ a directed unit-capacity graph, i.e., $c(e) = 1$ for each $e \in E$.

   - (Menger's Theorem) Given an integer $k > 0$, show that G has k edge disjoint paths from s to t if and only if there is an s-t flow of value k in G. (This shows that the maximum number of edge disjoint paths from s to t is exactly the value of the maximum s-t flow.)
   - Let $G = (V, E)$ be a directed graph and let u, v, w be distinct vertices. Suppose there are k edge disjoint paths from u to v in G, and k edge disjoint paths from v to w in G. Note that the paths from u to v can share edges with the paths from v to w. Prove that there are k edge disjoint paths from u to w in G.
     [Hint: Use Maxflow-Mincut or the Menger's theorem.]

3. The Computer Science Department at UIUC has n professors. They handle department duties by taking part in various committees. There are m committees and the j'th committee requires k_j professors. The head of the department asked each professor to volunteer for a set of committees. Let $S_i \subseteq \{1, 2, \ldots, m\}$ be the set of committees that professor i has volunteered for. A committee assignment consists of sets $S'_1, S'_2, \ldots, S'_n$ where $S'_i \subseteq \{1, 2, \ldots, m\}$ is the set of committees that professor i will participate in. A *valid* committee assignment has to satisfy two constraints: (i) for each professor i, $S'_i \subseteq S_i$, that is each professor is only given committees that he/she has volunteered for, and (ii) each committee j has k_j professors assigned to it, or in other words j occurs in at least k_j of the sets $S'_1, S'_2, \ldots, S'_n$.

   (a) **Not to submit:** Describe a polynomial time algorithm that the head of the department can employ to check if there is a valid committee assignment given m, $k_1, k_2, \ldots, k_m$ the requirements for the committees, and the lists $S_1, S_2, \ldots, S_n$. The algorithm should output a valid assignment if there is one.

   (b) The head of the department notices that often there is no valid committee assignment because professors naturally are inclined to volunteer for as few committees as possible. To overcome this, the definition of a valid assignment is relaxed as follows. Let ℓ be some integer. An assignment $S'_1, S'_2, \ldots, S'_n$ is now said to be valid if (i) $|S'_i - S_i| \leq \ell$ and (ii) each committee j has k_j professors assigned to it. The new condition (i) means that a professor i may be assigned up to ℓ committees not on the list S_i that he/she volunteered for. Describe an algorithm to check if there is a valid committee assignment with the relaxed definition.

   No proof of correctness necessary but we recommend a brief justifaction. And make sure you have a clear and understandable algorithm.