CSC373 – Problem Set 3

Remember to write your **full name(s)** and **student number(s)** prominently on your submission. To avoid suspicions of plagiarism: at the beginning of your submission, **clearly state any resources (people, print, electronic) outside of your group, the course notes, and the course staff, that you consulted**.

Remember that you are required to submit your problem sets as both LaTeX `.tex` source files and `.pdf` files. There is a 10% penalty on the assignment for failing to submit both the `.tex` and `.pdf`.

Due Dec 9, 2022, 22:00; required files: ps3.pdf, ps3.tex

Answer each question completely, always justifying your claims and reasoning. Your solution will be graded not only on correctness, but also on clarity. Answers that are technically correct that are hard to understand will not receive full marks. Mark values for each question are contained in the [square brackets].

You may work in groups of up to THREE to complete these questions.

Please see the course information sheet for the late submission policy.

1. [**15 points**] In this problem, we will show that the maximum-flow problem is equivalent to the max-single-edge-flow problem.

   Recall that in a max-flow instance, we require that for every vertex, the in-flow is the same as out-flow, except the source (s) and target (t) vertices.

   In the **max-single-edge-flow** problem, we are given a directed graph $G(V, E)$ with capacities $c_e > 0$ on edges $e \in E$, and a special (directed) edge $e^\star \in E$. Our goal is to find a flow f_e for $e \in E$, that satisfies the following constraints:

   - f satisfies the direction and capacity constraints for all edges, i.e., $\forall e \in E, 0 \leqslant f_e \leqslant c_e$,
   - f satisfies in-flow equals out-flow for *all* vertices, i.e.,

   $$\forall v \in V, \qquad \sum_{u:(u\to v)\in E} f_{(u\to v)} = \sum_{u:(v\to u)\in E} f_{(v\to u)},$$

   and subject to the above constraints, our goal is to maximize the flow on the special edge $e^\star$.

   (a) (6 points) Give a reduction from the max-single-edge-flow problem to the max-flow problem. That is, given an instance of max-single-edge-flow, construct a max-flow instance such that solving the max-flow instance allows you to solve the max-single-edge-flow instance. Justify both sides of the reduction as done in class.

   Your algorithm for constructing the max-flow instance must run in $O(|V| + |E|)$ time, where $G(V, E)$ is the graph for the max-single-edge-flow instance.

   (b) (9 points) Give a reduction from the max-flow problem to the max-single-edge-flow problem. That is, given an instance of max-flow, construct an instance of the max-single-edge-flow problem such that solving the max-single-edge-flow instance allows you to solve the max-flow instance. Justify both sides of the reduction.

   Your algorithm for constructing the max-single-edge-flow instance must run in $O(|V| + |E|)$ time, where $G(V, E)$ is the graph for the max-flow instance.

2. **[15 points]** The city of Toronto wants to monitor all traffic that enters Toronto from Mississauga, and leaves Toronto towards Scarborough. It is planning to do so via cameras mounted at select intersections throughout the city.

   We are going to model this as an instance of the **Undirected Bottleneck Problem**. The road network of Toronto is given as an undirected graph $G(V, E)$, with vertices V representing intersections, and edges E representing roads connecting these intersections. We assume for simplicity that a car can only travel along a sequence of roads, and every road can be traversed in either direction, i.e., the edges are undirected. Say $|V| = n$, and $|E| = m$.

   We are given 2 subets of these intersections. $M \subseteq V$ is the set of intersections where traffic from Mississauga can enter Toronto. Similarly, $S \subseteq V$ is the set of intersections where traffic can leave Toronto for Scarborough.

   Given $G(V, E)$ and M, S, the goal of the **Undirected Bottleneck Problem** is to determine a *bottleneck* set $C \subseteq V$ of intersections with the smallest possible cardinality where cameras can be mounted (one per such intersection), such that every car starting from Mississauga, i.e., from some intersection in M, traveling along some sequence of roads, and leaving Toronto at some intersection in S, must go through one of the intersections in C.

   (a) (10 points) Give a reduction from this problem to maximum-flow, i.e. construct a max-flow instance such that running a max-flow algorithm (such as the Ford-Fulkerson algorithm), allows us to find the minimum number of checkpoints needed. Your construction must run in $O(m + n)$ time.

   Fully justify your reduction. Remember that a reduction consists of 2 parts. The first part involves showing that if every bottleneck in our **Undirected Bottleneck Problem** has large cardinality, then the resulting max-flow instance has a flow with large flow-value.

   For the second part, show that any flow in the max-flow instance with large flow value implies that every bottleneck set in the original Undirected Bottleneck Problem instance has a large cardinality.

   (b) (5 points) Say we use the Ford-Fulkerson algorithm to solve the max-flow instance constructed in part a. Using the max-flow returned by the Ford-Fulkerson algorithm, and an additional $O(m + n)$ time at most, find a set of checkpoints $C \subseteq V$ that is a solution to the **Undirected Bottleneck Problem** with minimum cardinality.

3. **[15 points]** You are organizing a hackathon at the Department of Mathematical & Computational Sciences (MCS). Students have organized themselves into teams of sizes between 4 and 6. Note that each student can participate in multiple teams. Assume that there are a total of n students, and t teams.

   You wish to identify a set C of student representatives. We want the set of representatives C to contain at least one student from each of the t teams across the department. Note that one student representative could represent multiple teams. Our goal is to find a set of representatives C with the smallest possible size.

   (a) (3 points) Design an approximation algorithm for this problem that runs in time polynomial in n, t and achieves an approximation ratio of $O(\log n)$ for the problem of finding the smallest set of representatives. Justify the correctness of your algorithm, its running time and approximation ratio guarantees.

   (b) (12 points) Design another approximation algorithm for this problem that runs in time polynomial in n, t and achieves an **approximation ratio of 6** for the problem of finding the smallest set of representatives. Justify the correctness of your algorithm, its running time and approximation ratio guarantees. (Hint: consider an algorithm like GreedyVertexCover).