Problem Set 7: Flows

1. (Easy) You are given a graph G and a flow f on each edge. Assume that the flow is valid, i.e., it satisfies the capacity constraint and the conservation of flow. Let $value(f)$ denote the flow reaching the destination t. Design an $O(m+n)$ time algorithm to check if $value(f)$ is the maximum flow in G.

2. (Easy) Show that the Ford-Fulkerson also gives you the minimum cut between the source s and the sink t as a byproduct.

3. (Easy) You are given a directed unweighted graph, a source s, a sink t and a number k. You may remove $\leq k$ edges in the graph with the aim to decrease the size of minimum cut between s and t. Design an $O(mn)$ time algorithm for this problem.

4. (Easy) An edge of a flow network is called critical if decreasing the capacity of this edge results in a decrease in the maximum flow. Give an efficient algorithm that finds a critical edge in a network.

5. (Medium) We define a most vital arc of a network as an arc whose deletion causes the largest decrease in the maximum s-t-flow value. Let f be an arbitrary maximum s-t-flow. Either prove the following claims or show through counterexamples that they are false:
   (a) A most vital arc is an arc e with the maximum value of c(e).
   (b) A most vital arc is an arc e with the maximum value of f(e).
   (c) A most vital arc is an arc e with the maximum value of f(e) among arcs belonging to some minimum cut.
   (d) An arc that does not belong to some minimum cut cannot be a most vital arc.
   (e) A network might contain several most vital arcs.

6. (Medium) Given a graph G with source s and sink t, we say that a node $v \in V$ is *upstream* if, for all minimum s-t cuts (S, T) of N, $v \in S$. In other words, v lies on the s-side of every minimum s-t cut. Analogously, we say that v is *downstream* if $v \in T$ for every minimum s-t cut (S, T) of G. We call v *central* if it is neither upstream nor downstream.

   Design an algorithm that takes G and a flow f of maximum value in G, and classifies each of the nodes of G as being upstream, downstream, or central. Your algorithm should run in linear time.

7. (Medium) Suppose you are given a directed graph $G = (V, E)$, with a positive integer capacity c_e on each edge e, a designated source $s \in V$, and a designated sink $t \in V$. You are also given a maximum s-t flow in G, defined by a flow value $f(e)$ on each edge e. The flow $f(e)$ is acyclic: there is no cycle in G on which all edges carry positive flow.
   (a) Suppose we pick a specific edge $e^* \in E$ and increase its capacity by 1 unit. Show how to find a maximum flow in the resulting capacitated graph in time $O(m)$.
   (b) Suppose we pick a specific edge $e^* \in E$ and reduce its capacity by 1 unit. Show how to find a maximum flow in the resulting capacitated graph in time $O(m)$, where m is the number of edges in G.

8. (Medium) There are many common variations of the maximum flow problem. Here are four of them.
   (a) There are many sources and many sinks, and we wish to maximize the total flow from all sources to all sinks.
   (b) Each vertex also has a capacity on the maximum flow that can enter it.

   Each of these can be solved efficiently. Show this by reducing (a) and (b) to the original max-flow problem.

9. (Hard) A graph has a unique minimum cut if there is only one cut that whose weight is the minimum. Design an algorithm that finds if a graph has a unique minimum cut.