HW 8 (due Tuesday, at noon, November 4, 2014)

CS 473: Fundamental Algorithms, Fall 2014 Version: **1.01**

Make sure that you write the solutions for the problems on separate sheets of paper. Write your name and netid on each sheet.

Collaboration Policy: The homework can be worked in groups of up to 3 students each.

---

1. (30 PTS.) The good, the bad, and the middle.

uppose you're looking at a flow network G with source s and sink t, and you want to be able to express something like the following intuitive notion: Some nodes are clearly on the "source side" of the main bottlenecks; some nodes are clearly on the "sink side" of the main bottlenecks; and some nodes are in the middle. However, G can have many minimum cuts, so we have to be careful in how we try making this idea precise.

Here's one way to divide the nodes of G into three categories of this sort.

- We say a node v is ***upstream*** if, for all minimum s-t cuts (A, B), we have $v \in A$ – that is, v lies on the source side of every minimum cut.
- We say a node v is ***downstream*** if, for all minimum s-t cuts (A, B), we have $v \in B$ – that is, v lies on the sink side of every minimum cut.
- We say a node v is ***central*** if it is neither upstream nor downstream; there is at least one minimum s-t cut (A, B) for which $v \in A$, and at least one minimum s-t cut (A', B') for which $v \in B'$.

Give an algorithm that takes a flow network G and classifies each of its nodes as being upstream, downstream, or central. (Hint: The running time of your algorithm should be within a constant factor of the time required to compute a *single* maximum flow.)

2. (40 PTS.) Augmenting Paths in Residual Networks.

You are given an integral instance G of network flow. Let C be the value of the maximum flow in G.

(A) (8 PTS.) Given a flow f in G, and its residual network G_f, describe how to compute, as fast as possible, the highest capacity augmenting path flow from s to t. Prove the correctness of your algorithm.

(B) (8 PTS.) Prove, that if the maximum flow in G_f has value T, then the augmenting path you found in (A) has capacity at least T/m.

(C) (8 PTS.) Consider the algorithm that starts with the empty flow f, and repeatedly applies (A) to G_f (recomputing it after each iteration) until s and t are disconnected. Prove that this algorithm computes the maximum flow in G.

(D) (8 PTS.) Consider the algorithm from (C), and the flow g it computes after m iterations. Prove that $|g| >= C/10$ (here 10 is not tight).

(E) (8 PTS.)Give a bound, as tight as possible, on the running time of your algorithm, as a function of n, m, and C.

3. (30 PTS.) Edge-Disjoint Paths.

You are given a directed graph $G = (V, E)$ and a natural number k.

(A) We can define a relation $\rightarrow_{G,k}$ on pairs of vertices of G as follows. If $x, y \in V$ we say that $x \rightarrow_{G,k} y$ if there exist k mutually edge disjoint paths from x to y in G. Is it true that for every G and every $k \geq 0$ the relation $\rightarrow_{G,k}$ is transitive? That is, is it always the case that if $x \rightarrow_{G,k} y$ and $y \rightarrow_{G,k} z$ then we have $x \rightarrow_{G,k} z$? Give a proof or a counterexample.

(B) Suppose that for each node v of G the number of edges into v is equal to the number of edges out of v. Let x, y be two nodes and suppose there exist k mutually edge-disjoint paths from x to y. Does it follow that there exist k mutually disjoint paths from y to x? Give a proof or a counterexample.