cs473: Algorithms Assigned: *Mon., Oct. 26, 2020*

Problem Set #7

Forbes/Chekuri Due: *Thu., Nov. 5, 2020 (5:00pm)*

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 value, 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 max-flows) or to other standard graph problems (for example: reachability, minimum spanning tree, shortest paths) have similar requirements.

All problems are of equal value.

1. The maxflow-mincut theorem and its specializations (e.g., Hall's Theorem, Menger's Theorem) can translate questions between the language of flows to the language of cuts, and vice versa. This can facilitate proving useful and interesting facts that would otherwise be difficult to show directly. Prove the following.

   - Suppose G is an Eulerian directed graph, that is, a graph where the in-degree of each vertex is the same as the out-degree of each vertex. Prove that if there are k edge-disjoint paths from u to v then there are k edge-disjoint paths from v to u.
   - Suppose G is d-regular bipartite graph. Prove that it has a perfect matching.

2. Let $G = (V, E)$ be an undirected graph. Let V be partitioned into two sets of nodes R and N where R is the set of reliable nodes and N is the set of non-reliable nodes. The reliable nodes never fail but non-reliable nodes and edges can fail. Call $N \cup E$ the *elements*. Given two reliable nodes s, t describe an algorithm that computes the minimum number of elements whose failure results in s and t being disconnected from each other. See Fig 1.

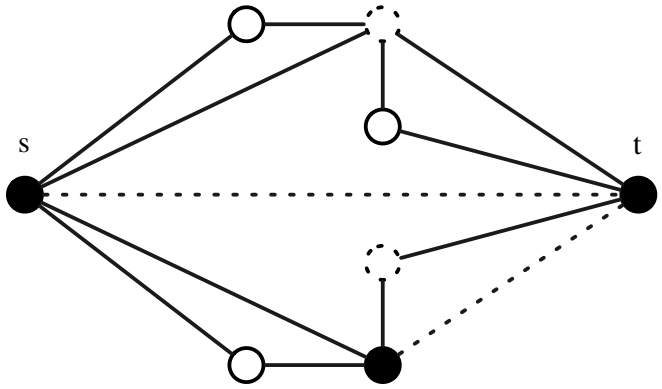

Figure 1: Black nodes are reliable. Removing dotted edges and nodes disconnects s from t.

3. Problem 16 in Chapter 11 of Jeff Erickson's Algorithms book. Only parts (a), (b).