1. Given a directed graph $G$ and two vertices $s$ and $t$, use a max-flow solver to compute the maximum number of edge-disjoint $s-t$ paths. Can you extend your approach to compute the maximum number of vertex-disjoint $s-t$ paths?

2. (a) If a flow network $(V,E)$ has two minimum $s-t$ cuts $(S_1,T_1)$ and $(S_2,T_2)$ where $S_1, S_2 \subset V$ are the source sets (the set of vertices in the cut that contains the source) and $T_1, T_2 \subset V$ are the sink sets (the set of vertices in the cut that contains the sink), then show that $S_1 \cap S_2$ and $S_1 \cup S_2$ are also source sets for minimum cuts.

(b) Design an algorithm that finds a minimum $s-t$ cut $(S,T)$ in a flow network such that the size of the source set $|S|$ is minimum among all possible source sets in all minimum $s-t$ cuts in the network.

(c) Give a polynomial time algorithm to decide whether a flow network has a unique minimum $s-t$ cut.

3. Give a proof or a counterexample for each of the following two assertions:

   (a) For a flow network with integer capacities, if all edge capacities are even, then there exists a maximum $s-t$ flow such that the amount of flow on each edge is even.

   (b) For a flow network with integer capacities, if all edge capacities are odd, then there exists a maximum $s-t$ flow such that the amount of flow on each edge is odd.

4. Our algorithms for network flow show that, if all edge capacities are integral, then there always exists an integral maximum flow. Show by example that some network flow instances with integral edge capacities also have non-integral (rational, even irrational) optimal solutions.

5. Show how to phrase the TSP approximation using an MST as a minimum-cost, perfect matching problem, followed by a simple Eulerian path (a path that crosses every edge of its given graph, something that always exists in a connected graph in which every vertex has even degree).

   Note: this is called Christofides’ algorithm, after its author, who first published it in 1976. You can of course trivially find this on Wikipedia, but the idea is to work it out on your own. Hint: shortcuts must go from one vertex of odd degree to another vertex of odd degree.