cs473: Algorithms	Assigned: *Thu., Oct. 31, 2019*

Problem Set #8

Prof. Michael A. Forbes
Prof. Chandra Chekuri

Due: *Thu., Nov. 7, 2019 (10:00am)*

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 (non-optional) problems are of equal value.

1. Let $G = (L \sqcup R, E)$ be a d-regular undirected bipartite graph (each node has degree exactly d), and where each part has size $|L| = |R| = n$.
   (a) Prove that G has a perfect matching via two methods.
      i. Use Hall's theorem.
      ii. Exhibit a feasible fractional flow in an appropriate network, which implies a matching of size n in G.
   (b) Using the previous part, prove that the edges of G can be partitioned into d perfect matchings, and hence the edges of G can be colored with d colors such that no two adjacent edges (edges that share a vertex) have the same color — this is called an *edge coloring*.

(c) Use the previous part to show that a bipartite graph with maximum degree d (not necessarily regular) has an edge coloring with at most d colors.

2. Let $G = (V, E)$ be a graph with edge weights given by $c : E \to \mathbb{R}$. In the min-cost perfect matching problem the goal is to find a minimum-cost perfect matching M in G (if there is one, otherwise the algorithm has to report that there is none) where the cost of a matching M is $\sum_{e \in M} c(e)$; note that the costs can be negative. In the maximum-weight matching problem the input is a graph $G = (V, E)$ and *non-negative* weights $w : E \to \mathbb{R}_{\geq 0}$, and the goal is to find a matching M of maximum weight. Note that a maximum weight matching may not be a maximum cardinality matching, that is, may not be a perfect matching. Describe an efficient reduction from min-cost perfect matching to max-weight matching.

   Note: Your reduction must work even when G is not bipartite.

3. Given points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ in the plane the *linear regression problem* asks for real numbers a and b such that the line $y = ax + b$ fits the points as closely as possible according to some criterion. The most common fit criterion is minimizing the L_2 error, defined as follows:

$$\epsilon_2(a, b) = \sum_{i=1}^{n} (y_i - (ax_i + b_i))^2 .$$

   But there are several other fit criteria, some of which can be optimized by linear programming.

   (a) The L_∞ error of the line $y = ax + b$ is defined by

$$\epsilon_\infty(a, b) = \max_{i=1}^{n} |y_i - (ax_i + b)| .$$

   Describe a linear program whose solution describes a line with the minimum L_∞ error.

   (b) The L_1 error of the line $y = ax + b$ is defined by

$$\epsilon_1(a, b) = \sum_{i=1}^{n} |y_i - (ax_i + b)| .$$

   Describe a linear program whose solution describes a line with the minimum L_1 error.

   Note: In general the points can be in $\mathbb{R}^d$ for some d, and in that case one fits a hyperplane. We are here considering the simple case of $d = 2$.

4. (**optional**, *not* for submission) Let $G = (V, E)$ be a flow network with integer edge capacities. An edge e is called *critical* if e is in every minimum (s, t)-cut. Describe an efficient algorithm that checks whether a given edge e is critical or not.