CS 473: Algorithms, Fall 2009
HBS 11

Problem 1 (Proving Reductions).

Remember that a matching in a graph $G = (V, E)$ is a set of edges $M \subset E$; such that no two edges share a common vertex. In lecture, we showed how to use network flow to find the maximum cardinality matching in a bipartite graph. Now, we will attempt to find the maximum cardinality matching in a general graph by also using network flow.

Suppose we are given a graph $G = (V, E)$. Define a new graph $G' = (V', E')$ as follows. For each $v \in V$, create two vertices $v_l, v_r \in V'$. Arbitrarily label the vertices of G as $1, \dots, n$. For every (undirected) edge $(u, v) \in E$, where $u < v$, create an edge $(u_l, v_r) \in E'$ with capacity 1. Create a source, s, and for each $v \in V$, create an edge (s, v_l) of capacity 1. Create a sink, t, and for each $v \in V$, create an edge (v_r, t) of capacity 1.

Now we will run the network flow algorithm on G'. We want to check to see that our reduction is correct. A maximum cardinality matching of size k exists in G if and only if the maximum flow of $G' \geq k$.

1. Prove or disprove the forward direction (if statement).
2. Prove or disprove the reverse direction (only if statement).

Problem 2 (Perfect Matchings in Regular Bipartite Graphs).

A *k-regular* graph is a graph $G = (V, E)$ where each vertex has degree k. One can show that every k-regular bipartite graph $G = (L, R, E)$ contains a perfect matching. This question asks for two different proofs of this theorem.

(a) Use Hall's Theorem to show that every k-regular bipartite graph $G = (L, R, E)$ contains a perfect matching.

(b) Given a k-regular bipartite graph $G = (L, R, E)$, we construct a flow network G' as follows. The vertex set of G' is $L \cup R \cup \{s, t\}$. For each edge (ℓ, r) in G, where $\ell \in L$ and $r \in R$, we add the directed edge (ℓ, r) to G'. For each vertex $\ell \in L$, we add the edge (s, ℓ) to G'. For each vertex $r \in R$, we add the edge (r, t) to G'. Finally, we assign capacity 1 to each edge of G'.

    As we have already seen, G has a perfect matching if and only if G' has a flow of value $|L| = |R|$. Prove that G' has a flow of value $|L|$.

Problem 3 (Reducing Shortest Paths to Min-Cost Flow).

The minimum cost flow problem is a generalization of the maximum flow problem. The flow network $G = (V, E)$ now has capacities and costs on the edges: for $e \in E$, $c(e) \geq 0$ is its capacity and $w(e)$ is its cost/weight. The costs can be positive or negative. The objective is to find a maximum flow in G of total minimum cost where the cost of a flow $f : E \to R^+$ is defined as $\sum_{e \in E} w(e) f(e)$.

1. Suppose we are given a flow network G and we would like to compute the min-cost flow of value k in G. Show how to reduce this problem to computing maximum flows of minimum cost by modifying G to obtain a graph G' where the minimum cost flow of value k in G corresponds to a maximum flow of minimum cost in G' and vice versa.

2. Assume that you are given a black box algorithm to compute the minimum cost flow in a flow network $G = (V, E)$, show how you can compute the s-t shortest path in a graph $G' = (V', E')$ with no negative cycle. *Hint:* Consider the flow network G with the same vertex and edge set as G' and capacity 1 on each edge. Let s, t be the source, sink of G, respectively. Show first that a min-cost flow with value 1 in G corresponds to a shortest s-t path in G' and then that a shortest s-t path in G' corresponds to a min-cost flow of value 1 in G.

3. (**Advanced. For thinking outside HBS**) Consider the case where G' may have negative cycles. Show how to reduce the problem of finding a negative cycle to computing min-cost flow. *Hint:* Compare the cost of the min-cost flow of value 1 when all edges in G have capacity 1, with the cost of the min-cost flow of value 1 when all edges in G have capacity 2.