CS 6150: HW4 – Graphs, Flows and Cuts

Submission deadline: Friday, Nov 4, 2016, 11:59PM

This assignment has 4 questions, for a total of 50 points. Unless otherwise specified, complete and reasoned arguments will be expected for all answers.

Question	Points	Score
Unique Minimum Spanning Trees	5	
Max Flow Basics	10	
More Reductions to Flow	20	
Unexpected Reductions to Flow/Matchings	15	
Total:	50	

Question 1: Unique Minimum Spanning Trees .... **[5]**

(a) [**2**] Let G be a weighted, undirected graph with all edge weights being distinct integers. Prove that there is a *unique* minimum weight spanning tree.

(b) [**3**] Even if the weights are not distinct, there could be graphs in which the MST is unique. Given a weighted undirected graph G, give a polynomial time algorithm to determine if G has a unique minimum weight spanning tree.

Question 2: Max Flow Basics .... **[10]**

(a) [**2**] Let G be a directed graph with all edges having an integer capacity. Prove that for any s, t, there exists a maximum flow between s, t in which every edge has an integral flow on it. [This is what we used in class to reduce the maximum matching problem in bipartite graphs to flows.]

(b) [**4**] Suppose you have already computed the max $s - t$ flow in a network with *integer* capacities. Give an efficient algorithm to update the maximum flow after one edge's capacity is increased/decreased by 1 (i.e., give algorithms for both the cases). The running times should be significantly faster than recomputing the maximum flow from scratch.

(c) [**4**] Let G be a directed graph with all edges having capacity 1. Further, suppose the *in-degree* of every vertex is equal to its *out-degree*. Prove that for any two vertices s, t and integer $k \geq 1$, there exist k edge disjoint paths from s to t iff there exist k edge disjoint paths from t to s.

Question 3: More Reductions to Flow .... **[20]**

We will see some more examples of problems reducing to flow/matching.

(a) [**5**] Consider the problem of forming a univeristy wide faculty committee that has one representative from each department. Suppose the university has n departments, and m faculty. Each faculty member may have appointments at multiple departments, and thus he/she may be a represetative for any of those departments. However, two departments are not allowed to pick the same faculty member as their representative.

Note that forming such a committee easily reduces to a max flow problem. Now, suppose we have the additional constraint that the committee has equal representation from all ranks of professors (assistant, associate, full). Give a polynomial time algorithm that incorporates this constraint and comes up with a committee, or concludes that it is impossible.

(b) [**5**] Suppose you are given an $n \times n$ checkerboard with some of the squares deleted. You have a collection of 2×1 dominoes, which can be used to cover any two adjacent squares. Describe an algorithm to determine if the board can be "tiled" with dominoes, i.e., if we can place dominoes on the board (either horizontally or vertically) such that each one covers two squares (neither of which is deleted), and no two dominoes overlap. As an example, consider Figure 1.

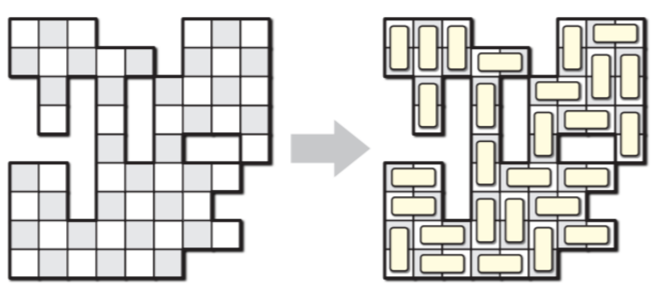

Figure 1: Checkerboard with some squared deleted on the left, and a feasible domino tiling on the right.

(c) [**5**] A *cycle cover* of a directed graph $G = (V, E)$ is a set of vertex-disjoint cycles in the graph that cover all the vertices. Design a polynomial time algorithm for finding such a cycle cover, or conclude that it is impossible.

(d) [**5**] Alice and Bob play the following game: Alice starts by naming an *actress* A_1. Bob must then name an *actor* B_1 who has co-appeared in a movie with A_1. Then Alice must name an actress A_2 who co-apparead with B_1, and so on. (Alice must always pick from the set of actresses, and Bob must pick from the set of actors.) The catch is that the players are not allowed to name anyone they have named already. The game ends and a player loses if he/she cannot name an actor/actress who hasn't been named already.

Suppose we are given as input a set of all "eligible" movies and their cast, and suppose that the total number of actresses is equal to the total number of actors. We can view it as a bipartite graph between actresses and actors, in which there is an edge iff the two have co-appeared in a movie. Prove that if this graph has a perfect matching, then there exists a winning strategy for Bob. (I.e., no matter how Alice plays, Bob can win.)

Bonus [**5**]: Prove the other direction, i.e., if there is no perfect matching, then Alice has a winning strategy.

Question 4: Unexpected Reductions to Flow/Matchings .... [**15**]

(a) [**5**] Finding "communities" in social networks is a problem of significant interest in network science. A common problem here is the following: we are given an undirected graph $G = (V, E)$, and a target *connectivity* λ. The goal is to find a subset S of users, such that the number of edges in G both of whose end points lie in S is at least $\lambda|S|$. I.e., on average, a user in S is connected to 2λ other users in S. By reasoning about the graph in Figure 2, give a polynomial time algorithm to find such a set S, or conclude that none exists.

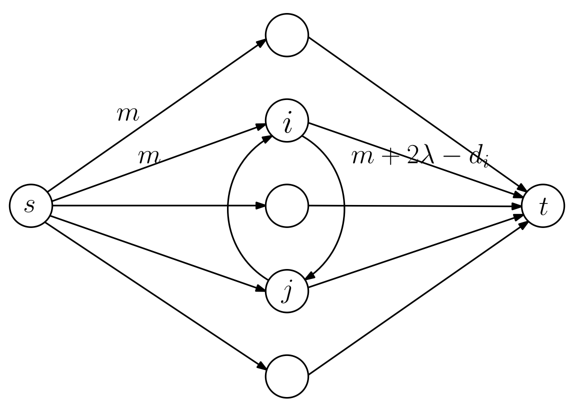

Figure 2: This graph has one node corresponding to every vertex in G, along with additional vertices s and t. The capacities of the edges to/from s, t are as shown. m is the total number of edges in G, and d_i refers to the degree of vertex i. For every edge $i, j \in E$, there are two directed edges of capacity 1.

(b) [**10**] Suppose we have a collection of n rectangular tiles lying on the floor, with the ith one having dimensions $a_i \times b_i$. We wish to reduce the amount of "floor space" used by these tiles by stacking them up as much as possible. However, the tiles are quite delicate, and tile j can be placed on top of tile i only if it is "fully contained" in tile i (by potentially rotating by 90 degrees), i.e., if either ($a_j \leq a_i$ and $b_j \leq b_i$) or ($a_j \leq b_i$ and $b_j \leq a_i$).

Design an algorithm that, given $\{a_i, b_i\}_{i=1}^n$, produces the stacking that leads to the minimum amount of total floor space used.

E.g.: Suppose we have tiles A, B, C, D, with dimensions $(10, 20), (15, 10), (5, 15), (10, 10)$ respectively. Then, one way to stack might be to have two stacks: first with C on top of B on top of A, and second with just D. This stacking yields a total floor space of $200 + 100 = 300$. On the other hand, having the first stack have D on top of B on top of A, and the second just C uses a total floor space of $200 + 75 = 275$.

[*Hint:* You may use the fact that the weighted version of maximum bipartite matching is solvable in polynomial time. (I.e., the version in which edges have weights and the goal is to maximize the sum of the weights of the edges in the matching.)]