Sample Midterm 222B

Instructions: This is an open book, open notes exam. Communicate your ideas **clearly** and **succinctly**. Show all work. **Note: this is longer than the real midterm will be**

(30) Problem 1. Suppose you have an undirected unweighted graph where each node i in the graph has a weight w_i. We want to find a minimum weight set of vertices whose removal disconnects the graph (the weight of a set of vertices is the sum of the weights of the vertices in the set).

a) Describe how to use one of the algorithms we discussed to efficiently solve this problem.

b) Justify the correctness of your solution technique.

(30) Problem 2. In class (and Ap. 6.4 in the book) we discussed how to solve a scheduling problem with n jobs where each job has a release time r_i, a deadline d_i and a processing time p_i, and we schedule them with preemptions allowed on m processors using network flow.

a) Suppose now we have two types of jobs. Type A can be run on any of the m processors, but type B jobs can only be run on machine 1 (e.g. from time zero to T we can do at most T units of processing on the type B jobs). You can assume jobs number $1, 2, \ldots k$ are of type B, and the rest of type A (note, these are just the job names, they are not ordered in any other way). Describe how to modify the network flow formulation to efficiently solve this new setting.

b) We are back to the original setting (so all jobs are type A),but we have only two processors,and we want to minimize the number of preemptions (so among feasible schedules, we want the one with the fewest preemptions). Prove that this problem is NP-hard (try and use a reduction which makes the best case possible about the hardness of this problem).

(25) Problem 3. For an undirected graph $G = (V, E)$ an *edge cover* C of the graph is a subset of E such that every vertex in V is an endpoint of at least one edge in C.

Give a good algorithm to find an edge cover of minimum size (Hint: use matching to get part of the solution). Justify the correctness of your algorithm and give its run time on an n vertex m edge graph.

(30) Problem 4. You are given a complete bipartite graph G with n nodes in each of the left and right sides of the graph. Each arc in the graph has cost 0 or 1, and m_0 is the number of arcs of cost 0 and m_1 is the number of arcs of cost 1.

a) Describe a fast algorithm for finding a minimum-cost perfect matching in G.

b) Briefly justify the correctness of your algorithm.

c) What is the running time of your algorithm?

(30) Problem 5. Disjoint Paths

Suppose we have a directed graph G and a start vertex s and a destination vertex t (in G). We want to find two paths between s and t that are edge disjoint (no edge is on both paths).

In each case below, describe an efficient solution and give its run time.

a) We want the total number of edges on our two paths to be as small as possible.

b) Now suppose that each edge (i, j) in our graph has an associated *distance* $d_{i,j}$, and we want the total length (sum of the distances) on the two paths to be as small as possible (the number of edges doesn't matter now, just the total distance).

(30) Sample HW problems :

Problems 3.5 and 3.7 in the text book.