CSC375F **Problem Set 1** **Fall, 2009**

Due: Wed, October 7, beginning of lecture

NOTE: Each problem set only counts 5% of your mark, but it is important to do your own work (but see below). Similar questions will appear on the first term test. You may consult with others concerning the general approach for solving problems on assignments, but you must write up all solutions entirely on your own. Anything else is *plagiarism*, and is subject to the University's Code of Behavior. You will receive 1/5 points for any (non bonus) question/subquestion for which you say "I do not know how to answer this question". You will receive .5/5 points if you just leave the question blank.
Advice: Do NOT spend an excessive amount of time on any question and especially not on a bonus question. If you wish to spend "free time" thinking about (say) bonus questions that is fine but you should not sacrifice time needed for other courses.

1. (30 points)

   Consider the unweighted MIS problem for graphs induced by the intersection of axis aligned squares in the plane. That is given n squares S_i with dimensions $d_i \times d_i$, consider the induced graph $G = (V, E)$ where $V = \{S_i | i = 1, ..., n\}$ and $(S_i, S_j) \in E$ iff S_i and S_j intersect.

   (a) Show that this class of graphs are not chordal graphs.

   (b) Design a greedy algorithm which has a 2-approximation ratio when all the squares have the same dimension $d_i = d$.

   (c) Design a greedy algorithm which has a 4-approximation ratio when the dimensions d_i are arbitrary.

2. (a) (10 points)

   Consider the unweighted interval scheduling problem and the third non-optimal greedy algorithm given in the text; namely in each iteration select an interval $I \in \mathcal{I}$ that intersects the fewest remaining intervals in $\mathcal{I}$. Then remove that I and all the intervals it intersects. Give an argument showing that this algorithm is a 2-approximation algorithm.
   Hint: Consider the iteration when the greedy algorithm chooses an interval I, and argue why three non-intersecting intervals (say in OPT) cannot intersect I.

   (b) (Bonus: 10 points) The text gives an example where this algorithm accepts three intervals whereas an optimal algorithm would accept 4 intervals showing that the approximation ratio is at best 4/3. Can you find an example that improves upon the (4/3) lower bound?

   (c) (Bonus: 25 points) Can you prove a better approximation ratio than 2 for this algorithm?

3. (20 points) The following problem relates to Dijkstra's shortest path algorithm and is motivated by routing on networks. As in the shortest path problem we have an

edge weighted directed graph $G = (V, E, s, b)$ where $s \in V$ is a distinguished start vertex and $b : E \rightarrow \Re$ is a real valued weight function. (In fact, unlike the case for shortest paths, in what follows it is not necessary to insist that b be non-negative.) We think of the edge weight $b(e)$ to be the bandwidth of that edge. For a path π in G, the bandwidth $b(\pi) = \min\{b(e)|e \in \pi\}$, that is, the minimum bandwidth (the bottleneck) of any edge on the path. The maximum bandwidth from s to u is the $\max\{b(\pi)|\pi$ is a path from s to $u\}$. By convention, we define the empty path to have ∞ bandwidth so that the maximum bandwidth from any u to itself is ∞.

Modify Dijkstra's algorithm so as to compute the bandwidth from s to every vertex $u \in V$. Prove that your algorithm optimally computes the maximum bandwidth from s to all $u \in V$.

4. For a fixed $m > 1$, consider the m-machine weighted interval scheduling problem. That is, we want to compute a feasible schedule $\sigma : \{1, \ldots, n\} \rightarrow \{0, 1, m\}$ so as to maximize profit where now each interval I_i has a profit v_i and $\sigma(i) = j > 0$ indicates that the i^{th} interval has been schedule on machine j and $\sigma(i) = 0$ indicates that the i^{th} interval is not scheduled.

   (a) (30 points)
   Design a polynomial time dynamic programming algorithm for computing an optimal solution for this problem. What is the complexity (as a function of n and m) of your algorithm. Provide both semantic and computationally defined arrays and briefly justify the equivalence between these arrays.
   Hint: See problem set 2 from the fall, 2007 course.

   (b) (Bonus 20 points)
   Consider solving the m-machine problem as follows:

```
$\mathcal{I}$ := set of weighted intervals $\{I_1, \ldots, I_n\}$
For $j$ := 1..$m$
    optimally schedule intervals in $\mathcal{I}$ on machine $j$
      and let $S$ be set of intervals that have been scheduled on machine $j$
    remove intervals in $S$ from $\mathcal{I}$
End For
```

   Thats is, the algorithm considers one machine at a time and schedules optimally the remaining intervals on that machine.

   Show that the algorithm is not an optimal algorithm and derive an approximation bound (better than the obvious m) for this algorithm.