CSE 202 Final Exam
Spring, 2008
Due Thursday, June 12, at 6 AM.

Describe and analyze (prove correct and give a time analysis) algorithms for any three of the following four problems. Algorithms may be described at "high level" without actual code. You may use any algorithm or data structure from the text or in class, and their correctness and analysis, but be careful. For example, if you construct a graph with n^2 nodes and n^3 edges and then run Dijkstra's algorithm on the resulting graph, the total run-time is $O(n^3 logn)$,because Dijkstra's algorithm is $O(ElogV)$. If you want to use the correctness of Dijkstra's algorithm on the graph, you must check that edge weights are positive, since otherwise the proof that Dijkstra's algorithm gives shortest paths does not go through.

If a problem has multiple size parameters, you should express the run-time as a function of all relevant parameters; e.g., saying maximum bi-partite matching takes time $O(|V||E|)$ is more accurate than to say it takes time $O(V^3)$, although both are correct.

For some problems, correctness may be trivial; for others the analysis may be trivial. If it is trivial, you do not need to go into detail, but you should at least give a one or two line explanation. (Conversely, if you only give a one or two line explanation, I will view this as implicitly claiming that it is trivial.)

Each problem is worth 10 points. Grading may be based on any of the following that are relevant for the problem: the efficiency of your algorithm; the correctness and proof thereof; and the time analysis. You only get credit for what you state and prove: if you give a time analysis of $O(n^2)$ for your algorithm, and the algorithm is in fact $O(n \log n)$, your score for efficiency will be based on your claimed $O(n^2)$. The number of points depending on each part is given after the problem, as well as a ballpark estimate of the time analysis for my solutions for some of the problem.

Dance partners You are pairing couples for a very conservative formal ball. There are n men and m women, and you know the height and gender of each person there. Each dancing couple must be a man and a woman, and the man must be at least as tall as, but no more than 3 inches taller than, his partner. You wish to maximize the number of dancing couples given this constraint. (4 pts correct poly-time algorithm, 4 pts correctness proof, 2 pts efficiency. My best time is $O(n \log n + m \log m)$).

Approximation for bin filling In the *bin filling* problem, you have n items of positive integral sizes $a_1, ..a_n$ and $m < n$ *bins*, where bin j is of size B_j. You need to assign each item i to a bin $A[i]$, in a way to fill the maximum number of bins, where a bin j is *full* if $\sum_{i|A[i]=j} a_i \geq B_j$. Give an efficient approximation algorithm for this problem. Most of the points will be based on your approximation ratio and the proof that it achieves this ratio. Best possible ratio is to fill at least 1/2 the bins as the optimal solution.

Job scheduling T is a positive integer, specifying the number of time units you have to perform jobs. You have n jobs $Job[1..n]$ each with integer arrival times, deadlines, and time requirements, $a_i, d_i, r_i \in \{1..T\}$ with $a_i \leq d_i$ and $r_i \leq d_i - a_i + 1$. Each job must be scheduled to a set of r_i integer times S_i between a_i and d_i ($a_i \leq t \leq d_i$ for each $t \in S_i$.) In addition, due to maintenance schedules, the number of machines available varies for each time. You have a list $m_1, ..m_t..m_T$ of the number of machines available at each time $1 \leq t \leq T$, and at most m_t jobs can be assigned time t. The problem is to decide whether there is a schedule that meets the above constraints. (6 pts correct poly time algorithm, 4 pts efficiency. My time is $O(n^2T^2)$.)

Grade maximization In the last homework assignment, you gave an $O(nH^2)$ algorithm to assign numbers of hours totalling to H to maximize grades when each of n assignments had a possibly different time-points trade-off function f_i. Give a more efficient algorithm when all of the assignments are identical, i.e., $f_1 = f_2 = ...f_n$. (Almost all points for efficiency. Assume $H \geq n$, since otherwise $n - H$ projects have no time assigned to them. $O(H^2 \log n)$ is best I know so far.)