NAME: _____

Algorithms and Data — Spring, 2009

CS U690 — Professor G. Cooperman

Final Exam

This exam is open book, open notes, and open Wed. However, you absolutely may not discuss any aspect of Algorithms or of this exam with anybody else during the exam period. We will still hold the regular class on Monday, April 13, when I will answer general questions about the Final Exam. Outside of class-time, I can only answer typical exam-time questions, such as "Should the answer be given as an integer or floating point? (Integer.); Do I also have to show the intermediate work? (No.)" The points on this exam purposely add to 110 points, in order to include the possibility of extra credit.

Write your answers on separate paper, with the numbered questions in the same numbered order. This exam is due in my office (336-WVH) by Wednesday, April 15, at 6:00 p.m.

Please staple this cover sheet to your exam, and sign the following honor statement:
On my honor, I have not discussed this exam, nor any topic in algorithms related to this exam, with anybody else during this exam period.

SIGNATURE: _____

Problem 1 (10 points) Recall that if $\lim_{n\to\infty} f(n)/g(n) < c$ for some constant c, then $f(n) = O(g(n))$. For each of the following equations, state if it is true or false.

(a) $3n^2 + 4 = O(n^2)$

(b) $\log n = O(\sqrt{n})$

(c) $\sqrt{n} = O(\log n)$

(d) $e^{2\log n} = O(n^2)$

(e) $n^2 = O(e^{2\log n})$

Problem 2 (10 points) Solve the following recurrence relations. Assume that T(1) = 1 in all cases.

(a) T(n) = T(n-1) + 1

(b) T(n) = T(n/2) + 1

(c) T(n) = 2T(n/2) + 1

(d) T(n) = 2T(n/2) + n

(e) T(n) = T(n-1) + n

(f) T(n) = 2T(n-1)

(g) T(n) = nT(n-1)

Problem 3 (20 points) *Greedy algorithm:* You are given a collection of jobs $1 \ldots n$ with deadlines d_i and (nonnegative) durations t_i. You have to schedule the jobs, i.e. specify their start times s_i. (By definition, the finish time $f_i = s_i + t_i$.) The lateness of a job is $\max(0, f_i - d_i)$. Prove that scheduling the jobs in order of their deadline minimizes the maximum lateness of any job.

Problem 4 (10 points) *Non-greedy algorithm:* Now suppose that the jobs of the previous problem have weight w_i. Give an efficient algorithm to schedule the jobs so as to minimize $\sum_{i=1}^{n} w_i f_i$. Also, state the $O(\cdot)$ complexity of the algorithm.

Given an undirected graph G and an edge e, give a linear time algorithm to determine whether G has a cycle containing e.

Problem 5 (15 points) Suppose that you are the curator of a large zoo. You have just received a grant of m dollars to purchase new animals at an auction you will be attending in Africa. There will be an essentially unlimited supply of n different types of animals, where each animal of type i has an associated cost c_i. In order to obtain the best possible selection of animals for your zoo, you have assigned a value v_i to each animal type, where v_i may be quite different than c_i. (For instance, even though panda bears are quite rare and thus expensive, if your zoo already has quite a few panda bears, you might associate a relatively low value to them.) Using a business model, you have determined that the best selection of animals will correspond to that selection which maximizes your perceived profit (total value minus total cost); in other words, you wish to maximize the sum of the profits associated with the animals purchased.

Devise an efficient algorithm to select your purchases in this manner. State the complexity of your algorithm as a $O(\cdot)$ formula. Provide pseudo-code for your algorithm.

(You may assume that m is a positive integer and that c_i and v_i are positive integers for all i. Note that c_i does not necessarily divide m dollars. So, it is not enough to buy only the most profitable animal.)
HINT: This type of problem is sometimes called the 0-1 knapsack problem.

Problem 6 (15 points) Prof. Curly is planning a cross-country road-trip from Boston to Seattle on Interstate 90, and he needs to rent a car. His first inclination was to call up the various car rental agencies to find the best price for renting a vehicle from Boston to Seattle, but he has learned, much to his dismay, that this may not be an optimal strategy. Due to the plethora of car rental agencies and the various price wars among them, it might actually be cheaper to rent one car from Boston to Cleveland with Hertz, followed by a second car from Cleveland to Chicago with Avis, and so on, than to rent any single car from Boston to Seattle.

Prof. Curly is not opposed to stopping in a major city along Interstate 90 to change rental cars; however, he does not wish to backtrack, due to time constraints. (In other words, a trip from Boston to Chicago, Chicago to Cleveland, and Cleveland to Seattle is out

of the question.) Prof. Curly has selected n major cities along Interstate 90 and ordered them from East to West, where City 1 is Boston and City n is Seattle. He has constructed a table $T[i, j]$ which for all $i < j$ contains the cost of the cheapest single rental car from City i to City j. Prof. Curly wants to travel as cheaply as possible.

Devise an efficient algorithm which solves this problem. State the complexity of your algorithm as a O(·) formula. Provide pseudo-code.

Problem 7 (10 points) Recall from Computer Organization or Computer Architecture that a 2-to-1 multiplexer is a circuit with a control lines, c and two input data lines, d_1 and d_2. If $c = 0$, then the output is the same as the input d_1. If $c = 1$, then the output is the same as the input d_2. This defines a Boolean function $f(c, d_1, d_2)$. For values of the three Boolean variables, the function $f()$ produces 0 or 1 as its answer. Assume a variable ordering in which c comes before d_1, which comes before d_2. You must do the following:

(a) Draw a truth table for $f(c, d_1, d_2)$. (You can use 0 and 1 or T and F, as you prefer.)

(b) Draw a BDD for $f()$. The BDD must be a fully reduced lattice (what is sometimes called an ROBDD).

(c) Write how many solutions (c, d_1, d_2) satisfy $f()$ (how many solutions make the output equal to true or 1).

(d) Write down all solutions, all values of (c, d_1, d_2), that satisfy $f()$.

Problem 8 (10 points) Next, we wish to pass the output of the multiplexer of the previous problem through an "and" gate with the logical "or" of d_1 and d_2. Suppose we have a second Boolean function, $g(d_1, d_2) = d_1 \vee d_2$. (Note that $g(d_1, d_2)$ is "d_1 or d_2".)

(a) Draw a BDD for $g(d_1, d_2)$.

(b) Draw a BDD for the logical "and": $f(c, d_1, d_2) \wedge g(d_1, d_2)$.

(c) How many values of (c, d_1, d_2) satisfy $f(c, d_1, d_2) \wedge g(d_1, d_2)$?

(d) Show all values of (c, d_1, d_2) that satisfy $f(c, d_1, d_2) \wedge g(d_1, d_2)$.

Problem 9 (10 points) Suppose that we have a Boolean function $g(x_1, \ldots, x_n)$. That is, the function g takes n Boolean variables as inputs, and returns either true or false, according to the value of those n Boolean variables. Suppose we define a new function, $g(x_1, \ldots, x_n) = f(x_1, \ldots, x_{k-1}, x_k, x_k, x_{k+2}, \ldots, x_n)$. (Essentially, $g()$ is independent of x_{k+1}, and the f-variables x_k and x_{k+1} are both set equal to the g-variable g_k.)

Prove that the number of nodes of the BDD for $g()$ is less than or equal to the number of nodes for $f()$.