

CSci 280, Spring 2009, Final Take-Home Exam

This take-home exam is due **Thursday, April 30**, at 10am. You should submit your solution on paper, either handwritten or typed. E-mailed solutions will not be accepted.

Answer only five of the following six problems. If you answer all problems, I will grade only the first five. Each problem is worth 20 points, so your solutions will be graded out of 100 points. The quality of writing and mathematical argument will be a central component in how your solutions are graded. *There are no rewrites on this final take-home exam.*

You should not obtain help from anybody or anything, except your instructor, your written class notes, the textbook (Dasgupta, Papadimitriou, & Vazirani, available on paper or on-line), and the course Web site.

1. Google makes money by selling keywords to advertisers. Each advertiser bids on possible search keywords, saying how much the advertiser is willing to pay each time a customer enters that word as a query and sees the advertiser's ad. Google has decided they want to build a program that helps advertisers decide which keywords to buy. (In fact, my brother works for Google on such a program for advertisers. I don't know all the details of his project, though, and in any case I'm changing several of the details of how Google's bidding scheme works.)

The advertiser will enter into the program a list of keywords w along with the number n_w of viewers who will see an ad associated with each keyword, the price p_w of purchasing the keyword, and the value v_w of each ad view for the keyword. (If I'm selling plane tickets, I'll be much more interested in showing my ad to people who are querying for "Oahu" than to those querying "Obama," so I'll place a higher value for Oahu ads than Obama ads.) The advertiser also enters the total amount C that can be expended on advertising and some minimum number N of people that should be reached by the selection of ads. (We'll not worry about people who view the ad twice under different keywords.) Your program should then display how much should be spent on each keyword without exceeding the amount of money available so as to reach at least N people and maximize the value of the keywords chosen.

Show how to formulate this problem as a linear programming problem, with some explanation of the meaning of your formulation.

2. The postmaster has hired you to compose a program for locating post offices in rural communities. The input to your program is a series of addresses given as two-dimensional coordinates (x_i, y_i) . We want to place our post office at the point $(x_{\text{post}}, y_{\text{post}})$ minimizing the maximum distance from that point to all addresses. (There are no restrictions on which point in the plane we choose for our post office.) We compute the distance between the post office to an address using the Manhattan distance:

$$\text{dist}(x_i, y_i) = |x_i - x_{\text{post}}| + |y_i - y_{\text{post}}|$$

For example, given the points $(0, 0)$, $(1, 4)$, $(9, 4)$, and $(10, 0)$, the best post office location is $(5, 1.5)$: With this choice, the distance to all points is 6.5.

Show how to formulate this problem as a linear program, and describe why your linear program will compute the correct answer.

3. An airline with N airplanes must inevitably confront the issue of how to assign its airplanes to flights each day. The flight schedule is already made: Each flight has the name of the source airport, the name of the destination airport, the departure time, and the arrival time. Because flying planes is so expensive, we won't allow flights to occur during the day unless

the flight is already on our schedule. Thus, if an airplane flies into an airport, the only way it will get to another airport is by taking another of the scheduled flights.

For simplicity, we won't worry about where each airplane starts and ends its day. The starting and ending airports need not be the same. You may assign each plane to start at any airport and stop at any airport you wish.

Show how to transform this problem so it can be solved using a maximum flow algorithm, and argue that your solution is correct.

4. [My original "solution" to this question was flawed, and so I've deleted it from the exam. You may feel free to submit your own solution to the original, though.]
5. On a parallel system, we have an array of "messages" represented as integers and an array saying how far each message should be copied into the array. For example, given the two arrays `msgs` and `dist` below, we want to arrive at the solution `recv`.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-------------------|----|----|----|----|----|----|----|----|---|
| <code>msgs</code> | 3, | 0, | 0, | 8, | 0, | 0, | 5, | 6, | 0 |
| <code>dist</code> | 3, | 0, | 0, | 2, | 0, | 0, | 1, | 2, | 0 |
| <code>recv</code> | 3, | 3, | 3, | 8, | 8, | 0, | 5, | 6, | 6 |

In this example, `dist[0]` is 3, and so we see that `msgs[0]` occurs in the three entries of `recv` starting at index 0. Similarly, `dist[7]` is 2, so `msgs[7]` is copied into the two entries of `recv` starting at index 7. You may assume that `dist` is structured so that no interval overlaps (for example, if `dist[0]` is 4, then `dist[1]` through `dist[3]` will necessarily be 0) and the final interval does not go off the array's end.

Show how this job can be accomplished using a parallel prefix scan. This is a matter of identifying an operation \otimes , explaining why you know this operation does its job, and showing that the operation is associative.

6. Draw a cube. You will receive full credit for any reasonable interpretation.
7. Given a dag with two identified vertices s and t , we want to identify as many paths as possible that connect s and t *without sharing any other vertices*. That is, no vertex (except s and t) should have more than one selected path going through it. Show how this problem could be solved using maximum flow. (You'll likely want to add some new vertices and edges.)