CS 222 Winter 2011. HW 5, due Tuesday Feb. 15

1. In our treatment of the Min Cost Arborescence problem, a fixed root node r was known, and the arborescence was a directed tree rooted at r. Let $A(r)$ denote the cost of the Min Cost Arboresence rooted at node r. Of course, for some choices of r, $A(r) = \infty$ because there is no arborescence rooted at r. Now consider the problem where no root node is fixed, so the problem is to choose a node r to Minimize $A(r)$ over all choices for r. Of course you could just run the original algorithm n times, but that would be inefficient. Explain how to solve the new problem in as efficient a way as possible. The solution might be as efficient as the algorithm for the case of a single, fixed r.

2. Suppose we have the height and weight of every person in the class, and we organize this information into a two dimensional table T where there are n height intervals, and m weight intervals. In the table, the (i, j) entry $T(i, j)$ indicates how many people fall into height category i and weight category j. Suppose that some of the particular cell entries are kept secret (so they are empty), but the number of people falling into each height category is public, as is the number that fall into each weight category. That is, the n row sums and the m column sums of the table are published, along with some, but not all, of the individual cell values of the table.

Devise an algorithm, based on network flow, that finds a way to assign non-negative values to the empty cells so that every row now adds up to its known row total, and every column now adds up to its known column total. That is, model this problem as a network flow problem. Explain what the graph is and how to use the max flow in it to assign the values to the empty cells.

3. Be sure to read the text or scan it on bipartite matching using network flow. This is in Section 7.5. You don't need to understand all of the material in that section, but you do need to understand what a matching is, what a perfect matching is (a matching that touches every node), and you need to understand how to find a perfect matching (if one exists) by setting up a bipartite matching problem.

This next problem is based on a real system that was recently introduced to increase the number of organ transplants from live donors. Previous to this system, a person who needed a specific organ transplant had to find a donor who was willing to donate and who was a good biological match with the recipient. These were usually family members of the patient. However,

it often happens that even family members are not good biological matches for the patient. But suppose a willing family member, say B, of patient A is a good match for patient C, who they don't know, and that a willing family member, D, of patient C, is a good match for person A. B is not willing to donate to C unless A also gets a transplant, and similarly D is not willing to donate unless C receives a transplant. The obvious solution is for B to donate to C and for D to donate to A, and then both A and C receive the organ they need (they have to do the transplants at the same time, so that no one backs out once their person has received a transplant). This scheme has the additional benefit that the donors do not have to donate the same organ types, increasing the chances of finding a good solution (for example, B might donate a kidney to C, but D donates lung tissue to A). This idea can be generalized to longer cycles, and in fact because of the low probability of mutual matches, long cycles are needed to make things work out.

The input to this problem is a directed graph G on n nodes representing n patients in need of a transplant. There is a directed edge from node i to node j, if patient i knows a person who is willing to make a donation (on i's behalf) and who is a good biological match for patient j. A directed cycle in the graph specifies a set of donations that allows all of the patients in the cycle to receive the needed transplant. The problem is to determine if there is a set of cycles in G so that each node is in exactly one of the selected cycles. If so, then there is a way to assign the donations so that every patient gets a transplant.

Show how to solve the problem by computing a maximum matching in a bipartite graph. That is, specify what the bipartite graph consists of and how to use the matching to solve the donation problem.