CSCI E-124 Data Structures and Algorithms — Spring 2015

Problem Set 8

Due: 11:59pm, Wednesday, April 22

See homework submission instructions at
`http://sites.fas.harvard.edu/~cs124/cs124/problem_sets.html`

Problem 5 is extra credit. Problems 1-4 constitute 100% of this problem set.

Upon completing this problem set, if you have roughly two minutes to spare then please fill out this survey: `http://goo.gl/forms/jV1IbeMedj`

Problem 5, the programming problem, is both **extra credit** and **non-collaborative**. You may not discuss any aspect of solving problem 5 with others (e.g. other students, teaching staff, or people outside the course).

Problem 1

You are managing HBO GO's communications network N, which is comprised of a set of network towers T that can send information back and forth. Some towers can communicate with each other, and the maximum bandwidth (flow of data) allowed from tower u to tower v is described by $c(u, v) \geq 0$. N has a set of (not necessarily disjoint) tower pairs $P = (s_1, t_1)...(s_k, t_k)$, each of which demands at least d_i of bandwidth - s_i must be able to transmit a stream of d_i bits that t_i eventually receives. If the bandwidth needs are not satisfied will be of poor quality, Harvard students will unintentionally read spoilers on Buzzfeed before seeing the latest episode, and your company will tank! Your goal is to have s_i send at least $y \cdot d_i$ bits to t_i for all i, where y is maximized. Ideally you want $y = 1$ to be achievable; if $y > 1$ is possible, then of course you'd be even happier, but if not then you just want y to be as large as possible. (If for example the best you can do is $y = .8$, then that means each pair is only getting 80% of the bandwidth it desires, which means you'd have to switch to some slightly poorer quality video for everyone to be able to stream the show, and you want to avoid this as much as possible.)

Part 1

An enterprising CS 124 student, you initially think you can just maximize the total data flow across your network. Explain briefly (in 3 sentences or less) why this will not necessarily result in a satisfactory solution.

Part 2

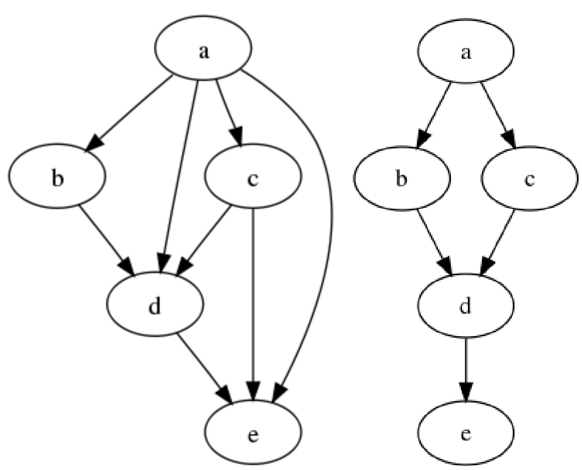
Reformulate this problem as a linear program. Note that data transmitted from tower s_i is not identical to data transmitted from tower s_j. Define $f^i(\cdot,\cdot)$ as the data flow from s_i to t_i, and $f^i(u,v)$ as the amount of "i" data flowing from node u to node v. Find a solution such that maximizes y such that the total flow from s_i to t_i is at least $y \cdot d_i$ for all i.

Problem 2

The class NP was defined as a class of decision problems. However, typically we have an optimization problem we would like to solve and not just a decision problem. For example, in the HAMILTONIANCYCLE problem we must decide whether a directed graph G has a simple cycle of length n. This is in contrast with the non-decision FINDHAMCYCLE problem of actually *finding* a cycle. Similarly in the VERTEXCOVER$_k$ problem we must decide whether a vertex cover exists of size at most k, as opposed to the MINVERTEXCOVER problem of finding a vertex cover of minimum size. Show that a polynomial time algorithm for HAMILTONIANCYCLE implies a polynomial time algorithm for FINDHAMCYCLE, and polynomial time algorithms for VERTEXCOVER$_k$ for all k imply a polynomial time algorithm for MINVERTEXCOVER.

Problem 3

A transitive reduction of a directed graph is a graph with as few edges as possible that has the same reachability relation as the given graph. Said another way, a a graph G' is a transitive reduction of G, there is a path from vertex u' to v' if and only if there is a path from vertex u to v. Additionally, G' has the smallest number of edges possible. Here is an example of a graph and its transitive reduction:



Show that it is NP-complete to determine whether a given graph G has a transitive reduction with at most k edges. (The input G need not be acyclic.) You may use that the directed HAMILTONIANCYCLE problem is NP-hard when when the input graph is promised to be strongly connected.

Problem 4

Consider the two-player game given by the following matrix. (A positive payoff goes to the row player.)

$$\begin{bmatrix} 4 & 1 & 0 & -3 \\ 6 & -3 & -2 & 0 \\ -3 & -2 & 5 & -3 \\ -4 & 4 & -5 & 5 \end{bmatrix}$$

- Write down the linear program to determine the row player strategy that maximizes the value of the game to the row player. Do the same for the column player.
- Find an LP solver and use it to solve these linear programs, and give the proper strategies for both players. There are many online you can use, but we used `http://www.phpsimplex.com/simplex/simplex.htm?l=en`. Note that on many solvers, all decision variables are assumed to be positive - make sure your variables respect this assumption when plugging them into the solver!
- What is the value of the game? Should the column player pay the row player to play, or vice versa, and how much should one player pay the other to make the game fair?

Problem 5 (extra credit, non-collaborative)

Solve "Problem A - Airplanes" on the programming server; see the "Problem Sets" part of the course web page for the link. Since this problem is for extra credit, extra credit is only received if the problem is fully solved (i.e. 100 points on the programming server).