COMP 360 Algorithm Design Techniques *Due November 18, noon*

Assignment 4: Max Flow and Linear Programming

Part of the goal of this homework is to see how the idea of *problem transformation* can be used to help solve a problem by changing it into a problem we (or somebody else) already know(s) how to solve. Later we will use problem transformation for a different purpose, to show that a problem apparently has no polynomial time solution.

(1) *Maximum Flows with Vertex Capacities.*

Suppose that, in addition to the arc capacity and flow conservation constraints, we wish to impose the additional constraints that each vertex v has a maximum capacity u_v, i.e. at most u_v units of flow can pass through a vertex v. Give a general method for transforming this new problem into a standard maximum flow problem, i.e. one that uses only arc capacity and flow conservation constraints.

(2) *The Student Societies Problem*

A group of student representatives from various student societies head over to Duluth St. for a dinner together. To maximize social interaction and networking opportunities, it is decided that no two students from the same organization (e.g., CSUS) should sit at the same table. There are n student societies, and for $1 \leq i \leq n$, society i has s_i members. The restaurant gives them the entire second floor, but the tables have different sizes. In fact, there are t tables, and table j seats up to c_j people, $1 \leq j \leq t$. Show how to determine whether a *good seating arrangement* exists by formulating the problem as a maximum flow problem. (In a good seating arrangment, everyone gets a seat but no two members of the same society are seated at the same table; it is not necessary to fill all the seats.)

(3) *Updating the Maximum Flow.*

(a) Suppose we use Ford-Fulkerson to find a maximum $s-t$ flow X in a graph G with non-negative integer arc capacities and then discover that we used the wrong arc capacity on an arc (i, j). In particular, let the true arc capacity of (i, j) be actually one unit *larger* than we thought. Using X, how can we find the true maximum flow X' in G in $O(|E|)$ time, where $|E|$ is the number of edges?

(b) Suppose we use Ford-Fulkerson to find a maximum $s-t$ flow in a graph G with non-negative integer arc capacities and then discover that we used the wrong arc capacity on an arc (i, j). In particular, let the true arc capacity of (i, j) be actually one unit *smaller* than we thought. Using X, how can we find the true maximum flow X' in G in $O(|E|)$ time?

(4) *Linear Programming.*

(a) Write the following linear programming problem in standard form.

$$\begin{array}{lrcl} \min & 5x_1 - 7x_2 + 2x_3 - x_4 & & \\ \text{s.t.} & 8x_1 + x_2 - x_3 - 3x_4 & \leq & 6 \\ & 2x_2 + x_4 & \geq & 10 \\ & -x_1 + 3x_3 + x_4 & \leq & -3 \\ & x_2 - 5x_3 - 2x_4 & = & 0 \\ & x_1, x_2, x_3, x_4 & \geq & 0 \end{array}$$

(b) Determine the dual problem for the linear programming problem from part a). Express this dual problem in standard form, just the way you did for part a).

(5) *Shortest Paths Problem.*

Recall the problem of finding shortest paths from a source vertex s to every other vertex. Formulate the Shortest Paths Problem for a graph $G = (V, A)$ containing no negative-weight directed cycles as an LP problem in standard form.

(6) *The Kayak Problem.*

A group of n people has a kayak (a kind of one-seat canoe) with which to cross a wide river, ten kilometers wide. For $1 \leq j \leq n$, each person j has a swimming speed s_j and a paddling speed p_j. Only one person can use the kayak at a time. The kayak may be paddled in either direction (toward the opposite shore, or toward the initial shore), and similarly, swimming can be done in either direction. It is not required to bring the kayak to the opposite side, and it is also ok to jump out of the kayak in the middle of the river and let another swimmer climb in, or to abandon the kayak all together.

The problem is to minimize the time t it takes to get everybody to the opposite bank (with or without the kayak). Each person will spend an amount x_j of time swimming toward the opposite bank, an amount x_j' of time swimming toward the initial side, an amount y_j of time paddling toward the opposite side, and an amount y_j' of time paddling toward the initial side.

Formulate the problem of determining the best values for these times as a linear programming problem. (If these values were known, the next problem would be to determine how, if possible, to arrange the times into a schedule, but that is not a concern here.)