ORIE 6300 Mathematical Programming I October 15, 2008

Recitation 7

Lecturer: Maurice Cheung *Topic: The Assignment Problem*

The Assignment Problem [1]

In the assignment problem, there is a set of n workers and a set of n tasks to which they must be assigned, satisfying that each worker is assigned to exactly one task. Associated with each assignment, there is an associated cost c_{ij} specifying how long it takes for worker i to complete task j. For now, assume $c_{ij} \geq 0$.

Note that this is a special case of the transportation problem where the number of supply nodes equals the number of demand nodes and each associated supply and demand is 1. Hence, there is a natural integer and linear programming formulation as before in which every basic feasible solution is integer-valued.

In the last recitation, we saw that by using the structure of the linear programming model for network flow problems, we can speed up each iteration of the Simplex Method. We will see that we can obtain a faster algorithm using the additional structure of the assignment problem.

Suppose that the costs associated with assigning worker i to each job were reduced by γ. Then, since any feasible assignment must assign worker i to some job, the cost of every feasible assignment will be reduced by gamma. Similarly, if the cost of assigning any worker to job j were reduced by γ, the cost of every feasible assignment would also be reduced by γ.

Hence, the idea will be to reduce the costs associated with a specific worker **OR** a specific job in a way that maintains non-negative costs. If we can do this and get a feasible assignment with modified total cost of 0, then it must be that we have an optimal solution.

Let's begin with some definitions:

Definition 1 *A **line** is any column or row. A line **covers** all elements contained in it.*

Definition 2 *A **0-cover** is a collection of lines such that every zero-cost entry is contained in some line.*

Definition 3 *A feasible assignment is an **All-0 assignment** if $c_{ij} = 0$ whenever worker i is assigned to job j.*

For example, consider the following input:

			⇓
	1	2	0
	2	2	0
⇒	0	0	2

Then row 3 and column 3 form a zero-cover, but there is no all-zero assignment associated with this input.

[1] Based on previous notes of Dennis Leventhal

Claim 1 *If there is no all-zero assignment, then there is a zero-cover using fewer than n lines.*

Proof: Consider the following max-flow problem: For vertices, there is a source node s, a sink node t, a node for each row and a node for each column. For edges, let the source be connected to each row node with capacity one and let each column node be connected to the sink with capacity one. Finally, let row node i be connected to column node j with infinite capacity.

Clearly, the maximum flow value is always no more than n and it equals n if and only if there is an all-0 assignment. Hence, if there is no all-zero assignment, consider the minimum cut.

Next, take as a cover the rows not in the cut and the columns that are in the cut. Clearly, if there is no all-0 assignment, there are fewer than n. Finally, suppose some $c_{ij} = 0$ is not covered by this 0-cover. Then the vertex corresponding to row i is in the cut and the vertex corresponding to column j is not in the cut. Hence, (i, j) is a cut edge which is impossible since it has infinite capacity. □

Further, note that by solving the max-flow problem above, it provides us with an all-0 assignment if one exists.

Now, we can turn this into an algorithm commonly referred to as the **Hungarian Algorithm**:

1. If the current 0-cover has n lines, **Stop**. We have an all 0-assignment.
   Otherwise, let k be the smallest value not in any line.
2. Subtract k from every c_{ij} value.
3. Add k to every row in the 0-cover.
4. Add k to every column in the 0-cover.

Proof of Algorithm:

First, note that at each iteration, the cost of any assignment is strictly decreasing. To see this, suppose we are not optimal, hence there are $l < n$ lines in a 0-cover. Since every c_{ij} value is decreasing by $k \geq 1$, the total assignment cost decreases by nk. However, note that since every line must contain exactly one entry in an assignment, the total cost increases by lk. Hence, the net change in cost is $k(l - n) \leq -1$ at every iteration.

Since we know that we always have a cover of fewer than n lines unless we're at an all-0 solution, and an all-0 solution is optimal, and the objective value decreases by at least 1 at every iteration, this algorithm must terminate.