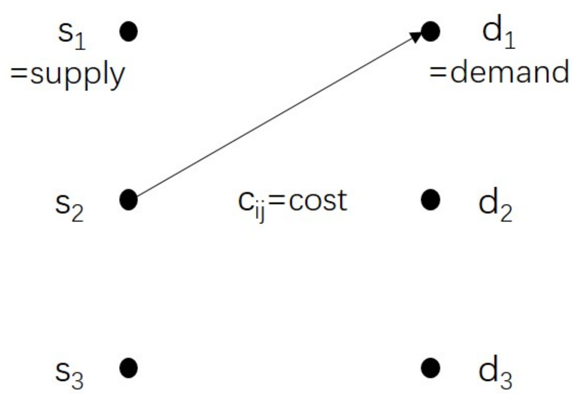
ORIE 6300 Mathematical Programming I September 21, 2016

Recitation 5

Lecturer: Calvin Wylie *Scribe: Xueyu Tian*

The Transportation Problem [1]

Suppose there are a set, S, of suppliers, each with supply s_i and a set, D, of customers each with demand d_j that must be met. Further, suppose that each unit shipped from supply node i to demand node j incurs a cost c_{ij}. Let x_{ij} equal the amount to ship from $i \in S$ to $j \in D$, of c_{ij} per unit. Find a minimum-cost shipping scheme that satisfies all the demand and supply restrictions.



Clearly, this problem is only feasible if $\sum_i s_i \geq \sum_j d_j$. Without loss of generality, suppose $\sum_i s_i = \sum_j d_j$ since otherwise we can set a dummy demand node k with $d_k = \sum_i s_i - \sum_j d_j$. Then we can formulate this as an linear program by:

$$\begin{array}{lrll} \min & \sum_{i \in S, j \in D} c_{ij} x_{ij} & & \\ \text{s.t.} & \sum_{j \in D} x_{ij} & = s_i & \forall i \in S \\ & \sum_{i \in S} x_{ij} & = d_j & \forall j \in D \\ & x_{ij} & \geq 0. & \end{array}$$

In some cases, we might wish to restrict the value of x_{ij} to be integral.

First, note that the rows of the constraint matrix are linearly dependent because:

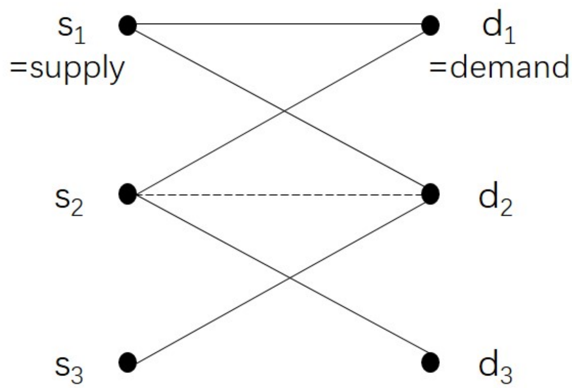
$$\sum_{i \in S} \sum_{j \in D} x_{ij} = \sum_{i \in S} s_i = \sum_{j \in D} d_j = \sum_{j \in D} \sum_{i \in S} x_{ij}$$

i.e., adding all the rows in the first constraint set gives the same result as adding all the rows of the second constraint set. Hence, without loss of generality, we can remove one constraint arbitrarily.

[1] Based on previous notes of Chaoxu Tong

Let $|S| = m, |D| = n$. Rank of constraint matrix is $n+m-1$. Consider a graphical representation of this problem where G is a bipartite graph with vertex sets corresponding to S and D and an edge set $E = S \times D$, i.e., $\{i, j\} \in E$ for every $i \in S,\ j \in D$, i.e., $G = (S \cup D, E_B), E_B = \{(i, j) \in E | x_{ij} > 0\}$. Note that each decision variable corresponds to an edge in the graph. Hence, a basic solution to the LP corresponds to some subgraph.

e.g.



Claim 1 *The subgraph corresponding to a basic solution to the LP doesn't contain cycles. Cycle =* $\{(i_1, j_1), (j_1, i_2), ..., (j_k, i_1)\}$

Proof: Let C be a cycle in the corresponding subgraph induced by the basic solution. Then $C = i_1 \to j_1 \to \ldots \to j_k \to i_1$ where $i_l \in S,\ j_l \in D$. i.e., $C = \{(i_1, j_1), (j_1, i_2), ..., (j_k, i_1)\}$.

Consider the vector y_{ij} which equals 1 if y_{ij} corresponds a cycle edge from S to D, -1 if y_{ij} corresponds to a cycle edge from D to S, and 0 otherwise.

Note that for every node in the cycle, there is exactly one edge 'entering' and one 'leaving'. Hence, it must be that $Ay = 0$, where A is our constraint matrix.

So, consider any basis B and suppose C is a cycle using only our basis variables. Let A_B be the corresponding basis matrix and y_B be the matching components of y. Since $y_i = 0$ for $i \notin B$.

$$0 = Ay = \begin{bmatrix} A_B & A_N \end{bmatrix} \begin{bmatrix} y_B \\ y_N \end{bmatrix} = A_B y_B + A_N y_N = A_B y_B + A_N 0$$

constraints A_B having full rank.

Noting that our graph has $n + m$ vertices, and each basis has $n + m - 1$ edges (since there are that many constraints), and each basis cannot contain a cycle, it follows that each basic solution corresponds to a spanning tree of our graph. □

Dual:

$$\begin{array}{ll} \max & \sum_{i=1}^{m} u_i + \sum_{j=1}^{n} v_i \\ \text{s.t.} & \quad u_i + v_j \le c_{ij}, \quad \forall i, j \end{array}$$

Since one constraint in the primal was redundant, one variable in the dual can be set arbitrarily.

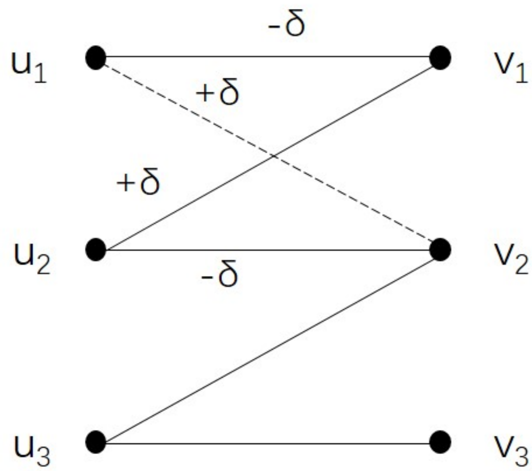
Algorithm Idea

Assume we start with an initial basic feasible solution, (x_B, x_N)

Solve $u_i + v_j = c_{ij}, \forall (i,j) \in B$ (Easy since one can be set arbitrarily, then solve for all others since basis induces a spanning tree)

(complementary slackness condition)

Check if $u_i + v_i \le c_{ij}, \forall (i,j) \in N$ (Dual feasibility). If so, it's optimal. Otherwise, $u_{\hat{i}} + v_{\hat{j}} > c_{\hat{i}\hat{j}}$, for some $(\hat{i},\hat{j}) \in N$. We'll see in lecture that putting $(\hat{i},\hat{j})$ in the basis will improve solution. Add edge $(\hat{i},\hat{j})$ to the subgraph. This will create a cycle.



Alternatingly increase (from $i \to j$) and decrease (from $j \to k$) flow around edges in the cycle to preserve 0 net flow. Let $\delta = \min_{(i,j)}\{x_{ij} : x_{ij}$ is a basic variable , (i,j) is a decreasing edge$\}$. Increase all 'forward' edges by δ, decrease all 'backward' edges by δ. Remove from the basis an edge which achieves the minimum.