Bilkent University

IE 400: Principles of Engineering Management

Term Project: Linear Programming Model

Submitted by: Team 16

Mert İnan, ID: 21402020
mert.inan@ug.bilkent.edu.tr

Nashiha Ahmed, ID: 21402950
nashiha.ahmed@ug.bilkent.edu.tr

İmge Gökalp, ID: 21402076
imge.gokalp@ug.bilkent.edu.tr

December 21, 2017

Linear Programming Formulation

Our team's assigned city is Istanbul. In order to find the shortest paths from Istanbul to every other city, the following Linear Programming formulations can be implemented.

There are two possibilities to implement an LP model. As the cities will be given during the demo, either all shortest paths to each city can be found or an LP model with a fixed starting node but an undecided terminal node—given as a parameter—that finds a single shortest path from that starting node to the given terminal node can be designed. We are presenting both of these LP formulations in this report.

Following is the model to find the shortest paths from our assigned city Istanbul to every other city in Turkey.

$$min \sum_{\{\forall i,j \in V\}} d_{i,j} x_{i,j}$$

$$s.t \sum_{\{\forall j \in V | (i,j) \in E\}} x_{i,j} - \sum_{\{\forall j \in V | (j,i) \in E\}} x_{j,i} \begin{cases} 80 & i = Istanbul \\ -1 & otherwise \end{cases} \quad \forall i \in V$$

$where,$
$x_{i,j} \geq 0,\ integer\ \forall i,j \in V$
$d_{i,j} \in D$
$V = \{v_1, ..., v_{81} \mid v_i = Adana, Ankara, ... ; i = 1, ...81\} : vertex\ set\ of\ the\ graph\ G$
$E = \{(Eskisehir, Ankara), (Ankara, Eskisehir), ...\} : edge\ set\ of\ the\ graph\ G$
$D = \{d_{i,j} \mid i,j = Adana, Ankara, ... ; \forall i,j \in E\} : distances\ between\ neighbor\ vertices\ of\ graph\ G$

As there is a constraint that the starting node should have 80 flow, this model finds all the optimal shortest paths from Istanbul to every other city.

On the other hand, if the terminal node is given, then the following LP model can be used, which yields a single path from Istanbul to that given city with less computations.

$$min \sum_{\{\forall i,j \in V\}} d_{i,j} x_{i,j}$$

$$s.t \sum_{\{\forall j \in V | (i,j) \in E\}} x_{i,j} - \sum_{\{\forall j \in V | (j,i) \in E\}} x_{j,i} \begin{cases} 1 & i = Istanbul \\ 0 & i \neq Istanbul, t \\ -1 & i = t \in V - \{Istanbul\} \end{cases} \quad \forall i \in V$$

$where,$
$x_{i,j} \geq 0,\ integer\ \forall i,j \in V$
$d_{i,j} \in D$
$V = \{v_1, ..., v_{81} \mid v_i = Adana, Ankara, ... ; i = 1, ...81\} : vertex\ set\ of\ the\ graph\ G$
$E = \{(Eskisehir, Ankara), (Ankara, Eskisehir), ...\} : edge\ set\ of\ the\ graph\ G$
$D = \{d_{i,j} \mid i,j = Adana, Ankara, ... ; \forall i,j \in E\} : distances\ between\ neighbor\ vertices\ of\ graph\ G$

This model can only be used, if “t”—the terminal node—is known in advance. Hence, this model finds a single shortest path from Istanbul to “t”. It is different than the previous model in a way that every edge is not considered. Hence, if “t” is known—which will be during the demo—then this model achieves the shortest path quicker than the previous model. However, as of now, “t” is not known, first model is also necessary, which finds all the shortest paths.

In the following pages, graph G of Turkey is shown, which connects every neighboring city with an edge. Each city is considered to be a vertex of the graph and only neighboring cities have edges between them. Distances are also shown on the edges of the graphs.