GLPK Case Study 5 - 60016 Operations Research

A company wishes to build 3 distribution centers in London for distributing its products to all 33 London boroughs (see Figure 1).

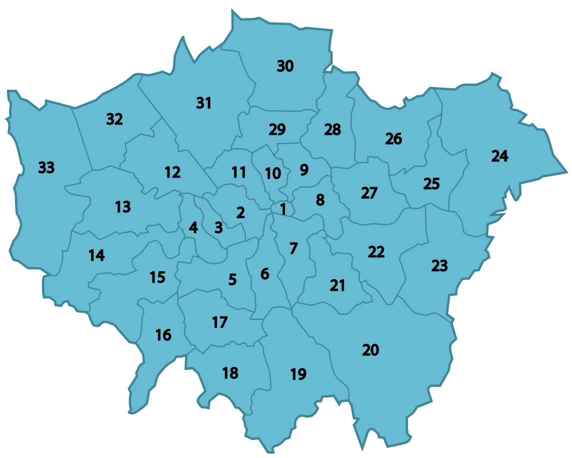

Figure 1. London boroughs

The selected locations should minimize the total transportation cost from the distribution centers to the remaining boroughs. You are given a GLPK data file ('supply_chain.dat') containing the transportation costs for all pairwise boroughs and you need to define the GMPL model that will be used to generate the location of the distribution centers and the assignment from the distribution centers to the boroughs. The integer programming formulation must take into account the following requirements:

- The decision variables are $x \in \{0,1\}^{33}$ and $Y \in \{0,1\}^{33\times 33}$. The i-th element of x is used to indicate whether a distribution center is built in the i-th borough. the (i,j)-th element of Y is used to indicate whether an assignment is made from the i-th borough to the j-th borough.
- At most 3 distribution centers can be built.
- There must be one assignment to each borough from any of the three distribution centers.
- The objective is to minimize the total cost of the chosen assignment (*i.e.*, $\min \sum_{ij} \mathrm{COST}_{ij} Y_{ij}$, where COST is the given cost matrix).

(a) Write the GMPL model for the integer programming problem and find its optimal solution with `glpsol`.

(b) The company has decided to build the distribution centers in boroughs 1, 18, and 30. Using the given GLPK data file 'supply_chain_assignment.dat', write the GMPL model for the assignment problem and find its optimal solution with `glpsol`.

(c) Rerun `glpsol` for the assignment problem (b) with the `--nomip` parameter which relaxes all integer variables as continuous variables. What is the optimal objective value now? Try also with different combinations of the location of distribution centers. What do you observe? Why?

Solution

(a) See Listing "supply_chain.mod". Since the data section is not included in the model file, `glpsol` needs to be called using the syntax `glpsol -m supply_chain.mod -d supply_chain.dat -o supply_chain.out` . For the given dataset, the optimal objective value is 213 and the optimal solution is to build the distribution centers in boroughs 4, 10, and 22, *i.e.*, Hammersmith, Islington, and Greenwich, respectively (see Figures 2 and 3).

Listing 1. supply_chain.mod

```
set I := {1..33};

param NW;
param COST {i in I, j in I};

# DECISION VARIABLES
var x {i in I}, binary;
var Y {i in I, j in I}, binary;

# INTEGER PROGRAM
minimize cost : sum {i in I, j in I} COST[i,j] * Y[i,j];

s.t.
MaxCenters: sum {i in I} x[i] <= NW;
Assignment {j in I} : sum {i in I} Y[i,j] == 1;
Build {i in I, j in I} : Y[i,j] <= x[i];

# SOLVE INTEGER PROGRAM
solve;
end;
```

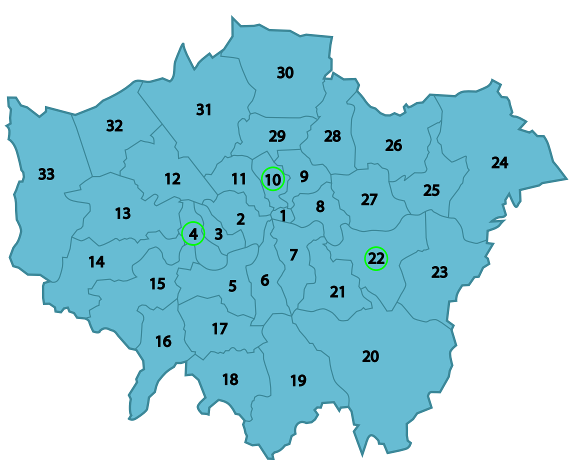

Figure 2. Locations of optimal distribution centers

(b) See Listing "supply_chain_assignment.mod". The optimal objective value is now 267 and the optimal assignment is shown in Figure 4.

Listing 2. supply_chain_assignment.mod

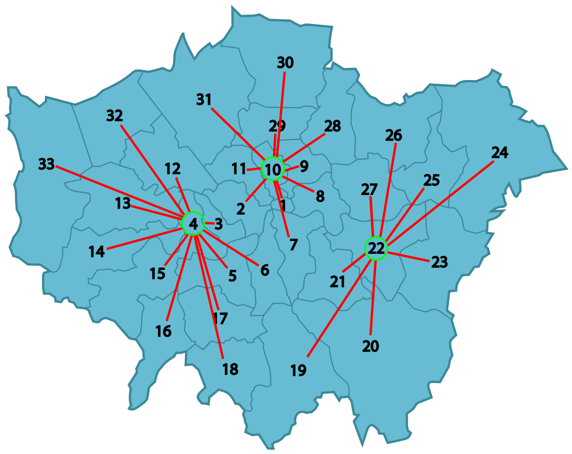

Figure 3. Optimal assignment from the distribution centers to the boroughs

```
set I := {1..33};

param NW;
param COST {i in I, j in I};
param x {i in I};

# DECISION VARIABLES
var Y {i in I, j in I}, binary;

# INTEGER PROGRAM
minimize cost : sum {i in I, j in I} COST[i,j] * Y[i,j];

s.t.
MaxCenters: sum {i in I} x[i] <= NW;
Assignment {j in I} : sum {i in I} Y[i,j] == 1;
Build {i in I, j in I} : Y[i,j] <= x[i];
# SOLVE INTEGER PROGRAM
solve;
end;
```

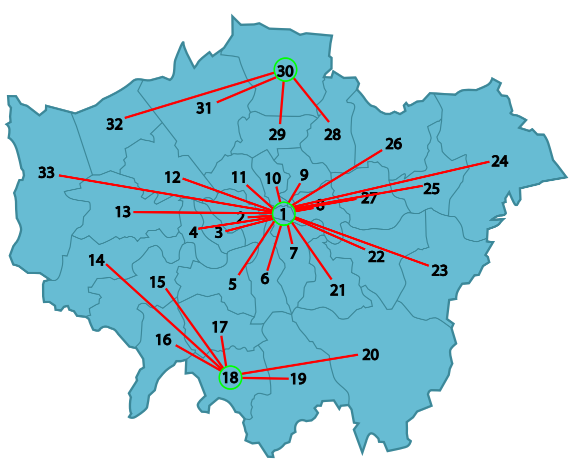

Figure 4. Optimal assignment from the distribution centers to the boroughs

(c) The LP relaxation of the assignment problem also produces an objective value of 267. Indeed, if the location of the distribution centers is fixed, the arising integer assignment problem belongs to the class of totally unimodular integer program, which is equivalent to its LP relaxation.

While most integer programming problems are perceived to be intractable (or NP-hard) due to their combinatorial nature, a subset of them admit a polynomial time solution. If the matrix A is totally unimodular and the vector b is integral for the LP: $\min\{c^\top x : Ax \leq b,\ x \geq 0\}$, then it can be shown that the optimal solution of this LP is integral. Thus, the integer program: $\min\{c^\top x : Ax \leq b,\ x \geq 0,\ x \text{ integer}\}$ can be solved as an LP, which is polynomial time solvable. Integer programming problems that satisfy the total unimodularity properties are, among others, shortest path, linear assignment, bipartite matchings, max flow, etc.

A sufficient condition for A to be totally unimodular are:

- Every column of A contains at most two non-zero entries
- Every entry in A is 0, +1, or −1
- The rows of A can be partitioned into two disjoint subsets $\mathcal{F}$ and $\mathcal{G}$ such that:
  - If two non-zero entries in a column of A have the same sign, then the row of one is in $\mathcal{F}$, and the other in $\mathcal{G}$
  - If two non-zero entries in a column of A have opposite signs, then the rows of both are in $\mathcal{F}$, or both in $\mathcal{G}$

We may represent the assignment problem as the integer program

$$\begin{aligned} \min \quad & \sum_{ij} \mathrm{COST}_{ij} Y_{ij} \\ \text{s.t.} \quad & A \begin{bmatrix} y_1 \\ \vdots \\ y_{33} \end{bmatrix} \leq b \\ & y_1, \ldots, y_{33} \in \{0, 1\}, \end{aligned} \tag{1}$$

where y_i is the i-th column of the matrix Y, for all $i = 1, \ldots, 33$. Matrix A in (1) is defined as

$$A = \begin{bmatrix} -\mathbf{1} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & -\mathbf{1} & \cdots & \mathbf{0} \\ \vdots & \cdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & -\mathbf{1} \\ \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \cdots & \mathbf{0} \\ \vdots & \cdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{2}$$

where $\mathbf{0}$ and $\mathbf{1}$ are row vectors of zeros and ones of length 33, respectively, $\mathbf{I}$ is an identity matrix of order 33, and $\mathbf{0}$ is a matrix of zeros of order 33. The vector b in (1) is defined as

$$b = \begin{bmatrix} -\mathbf{1}^\top \\ x \\ \vdots \\ x \end{bmatrix}. \tag{3}$$

Note that $x = [x_1 \ \cdots \ x]^\top$ is a vector whose elements are fixed integer constants. It can be easily verified that A satisfies the sufficient condition of total unimodularity given above. Since b is also integral, we conclude that the integer program (1) is equivalent to its LP relaxation.