Cable Laying Problem

July 17, 2011

1 Problem statement

Given a network of houses in a fully connected trench network, find the location of distribution centers (DCs) and assignment of houses to minimize cable length.

2 Preliminaries

The combinatorial nature of this problem likely puts it in the class of NP-hard problems. This means that there is no known algorithm to compute an optimal solution in a period of time that does not grow exponentially with the size of the problem. The present method avoids this difficulty by decomposing the problem into two parts. The first part is the assignment of DCs to junction nodes. The second is the assignment of houses to DCs. The proposed method is not guaranteed to find an optimal solution. However, experiments with real data have shown a degree of effectiveness.

The algorithm is an alternating assignment process. It starts with an initial assignment of houses to DCs, and then loops through two optimization processes:

1. We fix the assignment of houses to DCs and find optimal locations (with respect to the current house assignment) for each of the DCs.

2. We fix the DC locations, and find optimal housing assignments (with respect to the current fixed DC locations) of houses to DCs.

Each individual optimization process is solved optimally and reduces the cable length. The algorithm is terminated when the assignment stops changing. Experimentally, the algorithm requires 5 to 10 iterations until no more improvement is possible.

3 Random

It is possible to start the algorithm from random starting points than select the best solution. This could help avoid the chance of being stuck in a bad local minima. We have not conducted any experiments on this.

4 Notation and variables

h	number of houses
i	index over houses
c	number of DCs
j	index over DCs
n	number of junctions
k	index over junctions
$A \in \mathbb{R}^{h+n \times h+n}$	adjacency matrix
$D \in \mathbb{R}^{h \times n}$	distance matrix between houses and junctions
$X \in \{0,1\}^{h \times c}$	assignment matrix of houses to DCs
$Y \in \{0,1\}^{c \times n}$	assignment matrix of DCs to junctions
δ	DC capacity

5 Solution process

5.1 Construct weighted adjacency matrix

The network data is loaded into Matlab. The edge data is used to construct an adjacency matrix A which encodes the distance between connected nodes of the network. Element A_{ij} is the distance (in meters) between node i and node j. The network is undirected, so A is symmetric, that is $A_{ij} = A_{ji}$. Note that a node can be either a "house" or a "junction". Node types are specified in separate index arrays.

5.2 Computation of distance matrix

Both optimization routines require access to the shortest-path distance between any "house" and any "junction". For this purpose a distance matrix D is precomputed, where D_{ij} is the distance from house i to junction j. This is done using an implementation of Dijkstra's algorithm from Matlab BGL.

5.3 Distribution center placement

Given housing assignment matrix X and distance matrix D the optimal DC assignment matrix Y is easily computed. First, compute the matrix of cable lengths for any DC placement:

$$Z = X^T D \in \mathbb{R}^{c \times n}.$$

DC assignment matrix Y is constructed to assign DC j to junction $\text{argmin}_k \; Z_{jk}$ for all j.

This is a simple computation and does not require optimization software. It may be possible to design a greedy algorithm for this part that does not require the precomputation of the distance matrix. However, the storage and computation abilities of modern computers are able to do the above computation very quickly for the presented problem sizes.

5.4 Assignment of houses to DCs

Given DC assignment matrix Y and distance matrix D the optimal housing assignment matrix X is computed using a LP/MIP solver. First, compute the matrix of distances from each house to the fixed DC locations:

$$C = DY^T \in \mathbb{R}^{h \times c}.$$

Second, use LP/MIP software to solve the optimization problem:

$$\begin{aligned} \text{minimize} \quad & \text{vec}(C)^T \, \text{vec}(X) && (1) \\ \text{subject to} \quad & X \in \{0,1\}^{h \times c} && (2) \\ & X e_c = e_h && (3) \\ & X^T e_h \leq \delta e_c. && (4) \end{aligned}$$

Vectors e_c and e_h are vectors of ones of length c and h respectively. The $\text{vec}(A)$ operator takes matrix A and stacks the columns into a vector. Constraint (2) says that the assignment variables must be binary, (3) says that a house can only be connected to one DC, (4) says that a DC can be connected to no more than δ houses.

The Matlab implementation uses GUROBI to solve the MIP. It turns out that the system of equations formed by (3) and (4) is totally unimodular. This means that there exists a binary solution to the LP where constraint (2) replaced by $0 \leq X \leq 1$ (element-wise). Note, there is an chance that a standard LP solver return a non-binary solution that has an equivalent objective. Using a LP/MIP solver will guarantee a binary solution.

5.5 Iterate

The main algorithm repeatedly applies operations described in 5.3 and 5.4 until no more improvement is possible. Each step uses the solution from the previous. On the examples provided, the algorithm took 5 to 10 iterations and completed in less than 15 seconds.

6 Recommended LP/MIP solvers

The following opensource LP/MIP solvers should work. They have licenses that allow linking from software distributed under a difference license:

- lp_solve
- CoinMP

We don't have experience with MS solvers. The example with 1600 houses lead to an optimization problem with about 66,000 variables, but only about 130,000 nonzeros in the constraint matrix.