Logistic Network Design
An ILP Approach

Author1 (ID, email), Author2 (ID, email), Author3 (ID, email)

Department of Computer Science,
Shanghai Jiao Tong University, Shanghai, China

Abstract. The abstract should briefly summarize the contents of the paper in 15–250 words.
Keywords: keywords 1, keywords2.

1 Preliminaries

In this section, we describe the variables and denotations we use and the assumptions we propose.

1.1 Denotations

$\mathbf{L} = \{1, 2, 3...\}$: The cities set.
$N = |\mathbf{L}|$: The total number of cities.
$P = 2 * N * N$: The total number of all O-D pairs.
$\mathbf{H} = \{1, 2, 3...\}$: The hubs set.
D_{ij}: Direct distance between city i and j.
T_{ij}: Direct travel time between city i and j.
F_{ij}: Flow demands from city i to city j.
U_i: Update cost for city i.
C_{ijkm}: The transportation cost per unit flow from city i to city j routed from hub k and m.
K_i: The capacity of city i.
X_{ijkm} : The fraction of flow from origin i to destination j routed from hubs k and m, in other words, the fraction of F_{ij} through the path $i \to k \to m \to j$.

1.2 Assumptions

We assume that the transportation cost per unit flow directly from city p and city q is is proportional to the direct distance between p and q and we denote the coefficient as s. Then we have:

$$C_{ijkm} = s(D_{ik} + \alpha D_{km} + D_{mj}) \tag{1}$$

1.3 Decision variables

$$X_{ijkm} \in [0, 1] \tag{2}$$

X_{ijkm} is the fraction of flow from origin i to destination j routed from hubs k and m, in other words, the fraction of F_{ij} through the path $i \to k \to m \to j$.

$$Y_i = \begin{cases} 1 \text{ where city } i \text{ is selected as a hub} \\ 0 \text{ where city } i \text{ is not selected as a hub} \end{cases} \tag{3}$$

$$A_{ik} = \begin{cases} 1 \text{ where city } i \text{ is allocated to hub } k \\ 0 \text{ where city } i \text{ is not allocated to hub } k \end{cases} \tag{4}$$

1.4 Object functions

The main aim of our model is to minimize the update cost, meanwhile the total transportation cost and delivery time are also in consideration but not important.

Obviously, the **update cost** of all the hubs is:

$$\sum_k U_k Y_k \tag{5}$$

Then the **transportation cost** from i to j through network is:

$$\sum_k \sum_m F_{ij} X_{ijkm} C_{ijkm} \tag{6}$$

The **delivery time** from origin i to destination j through network is:

$$\sum_k \sum_m (T_{ik} + T_{km} + T_{mj}) A_{ik} A_{jm} \tag{7}$$

Then the total cost to build the entire network will be:

$$\sum_i \sum_j \sum_k \sum_m F_{ij} X_{ijkm} C_{ijkm} + \sum_k U_k Y_k \tag{8}$$

As the update cost is the main factor we consider, using the average transportation cost insead of total cost is more suitable. Then we get our first object function:

$$\frac{1}{P} \sum_i \sum_j \sum_k \sum_m F_{ij} X_{ijkm} C_{ijkm} + \sum_k U_k Y_k \tag{\textbf{Object 1}}$$

P is the total number of all O-D pairs here.

And the second object function is the average delivery time:

$$\frac{1}{P} \sum_k \sum_m (T_{ik} + T_{km} + T_{mj}) A_{ik} A_{jm} \tag{\textbf{Object 2}}$$

2 Uncapacitated single allocation formulation

In this section, we formulate the uncapacitade single allocation logistic network design as a ILP problem to minimize 2 objects:

1. the cost including transportation cost and update cost .
2. the average travel time between all O-D pairs.

$$
\begin{aligned}
\min \quad & \frac{1}{P} \sum_i \sum_j \sum_k \sum_m F_{ij} X_{ijkm} C_{ijkm} + \sum_k U_k Y_k \\
& \frac{1}{P} \sum_i \sum_j \sum_k \sum_m (T_{ik} + T_{km} + T_{mj}) A_{ik} A_{jm} && \text{(LP1)} \\
\text{s.t.} \quad & Y_k \in \{0,1\} & \forall k \quad & (9) \\
& A_{ik} \in \{0,1\} & \forall i,k \quad & (10) \\
& A_{ik} \le Y_k & \forall i,k \quad & (11) \\
& \sum_k A_{ik} = 1 & \forall i \quad & (12) \\
& \sum_k \sum_m X_{ijkm} = 1, & \forall i,j \quad & (13) \\
& X_{ijkm} \in \{0,1\} & \forall i,j,k,m \quad & (14) \\
& X_{ijkm} = A_{ik} * A_{jm} & \forall i,j,k,m \quad & (15)
\end{aligned}
$$

As Eq 1 shows, we have $C_{ijkm} = s(D_{ik} + D_{km} + D_{mj})$.
Eq 9,10 assure that Y_k and A_{ik} are binary variables.
Eq 11 enforces that when city i is allocated to city k, city k must be selected as a hub.
Eq 12 enforces that one city i must be allocated to a single hub k.
Eq 13 enforces that all required flow between city i and j must be routed through proper hub k and m, as the sum of the fractions of flow distributed to different paths is 1.
Eq 14 sets X_{ijkm} to binary variable, which means the flow between city i and j could only be routed through a singe path.
Eq 15 enforces the route from city i and j through hub k and m is valid, in other words, city k and m must all be selected as hubs.

The decision variable X_{ijkm} makes the model easier to extend to capacitated or multiple allocation problems, but in fact it is redundant here, as it would be either 0 or 1 and could be replaced with $X_{ijkm} = A_{ik} * A_{jm}$. Thus we could simplify the model as:

$$\begin{aligned}
\min \quad & \frac{1}{P}\sum_i\sum_j\sum_k\sum_m F_{ij}A_{ik}A_{jm}C_{ijkm} + \sum_k U_kY_k \\
& \frac{1}{P}\sum_i\sum_j\sum_k\sum_m (T_{ik}+T_{km}+T_{mj})A_{ik}A_{jm} && \text{(LP2)} \\
\text{s.t.} \quad & Y_k \in \{0,1\} && \forall k && (16)\\
& A_{ik} \in \{0,1\} && \forall i,k && (17)\\
& A_{ik} \le Y_k && \forall i,k && (18)\\
& \sum_k A_{ik} = 1 && \forall i && (19)
\end{aligned}$$

2.1 Solution

This is a multi-objective programming problem, there are 2 common approaches to solve this kind of problems:

1. Focus on the first object function and solve the problem, get the optimal value *opt*. Then set the value as a up bound and find the optimal value for for the second object function.
2. Simply add the 2 object function with proper weights w_1 and w_2, and to optimize the new object function $w_1 * OBJ1 + w_2 * OBJ2$.

We prefer the first approach.

3 Uncapacitated multiple allocation formulation

Similarly to the single allocation problem, we just modify the constrains of A_{ik} and X_{ijkm} to model the multiple allocation problem. Obviously, the flow demands from city i and j could be distributed to multiple different paths here.

$$\begin{aligned}
\min \quad & \frac{1}{P}\sum_i\sum_j\sum_k\sum_m F_{ij}X_{ijkm}C_{ijkm} + \sum_k U_kY_k \\
& \frac{1}{P}\sum_i\sum_j\sum_k\sum_m (T_{ik}+T_{km}+T_{mj})A_{ik}A_{jm} && \text{(LP3)} \\
\text{s.t.} \quad & Y_k \in \{0,1\} && \forall k && (20)\\
& A_{ik} \in \{0,1\} && \forall i,k && (21)\\
& A_{ik} \le Y_k && \forall i,k && (22)\\
& \sum_k\sum_m X_{ijkm} = 1, && \forall i,j && (23)\\
& 0 \le X_{ijkm} \le 1 && \forall i,j,k,m && (24)\\
& X_{ijkm} \le A_{ik} * A_{jm} && \forall i,j,k,m && (25)
\end{aligned}$$

Comparing the single allocation formulation, we make the following changes:

1. Remove the constraint $\sum_k A_{ik} = 1, \forall i$ to assure one city i could be allocated to multiple hubs.
2. X_{ijkm} represents the flow distributed fractions of city i to city j, we modify it from a binary variable 0 or 1 to a decimal number in range of $[0, 1]$.
3. Eq 25 means that, if there are some flows routed through the path $i \to k \to m \to j$, then city i must be allocated to hub k and city j must be allocated hub m.

4 Capacitated single allocation formulation

We simply add a capacity constraint for the sum of all valid paths routed through hub m.

$$
\begin{aligned}
\min \quad & \frac{1}{P}\sum_i\sum_j\sum_k\sum_m F_{ij}X_{ijkm}C_{ijkm} + \sum_k U_kY_k \\
& \frac{1}{P}\sum_i\sum_j\sum_k\sum_m (T_{ik}+T_{km}+T_{mj})A_{ik}A_{jm} && \text{(LP4)} \\
\text{s.t.} \quad & Y_k \in \{0,1\} & \forall k \quad & (26) \\
& A_{ik} \in \{0,1\} & \forall i,k \quad & (27) \\
& A_{ik} \le Y_k & \forall i,k \quad & (28) \\
& \sum_k A_{ik} = 1 & \forall i \quad & (29) \\
& \sum_k\sum_m X_{ijkm} = 1, & \forall i,j \quad & (30) \\
& X_{ijkm} \in \{0,1\} & \forall i,j,k,m \quad & (31) \\
& X_{ijkm} = A_{ik} * A_{jm} & \forall i,j,k,m \quad & (32) \\
& \sum_i\sum_j\sum_k F_{ij}X_{ijkm} \le K_m & \forall m \quad & (33)
\end{aligned}
$$

5 Capacitated multiple allocation formulation

Similarly to the capacitated single allocation formulation, we add a capacity constraint for the sum of all valid paths routed through hub m.

$$
\begin{aligned}
\min \quad & \frac{1}{P}\sum_i\sum_j\sum_k\sum_m F_{ij}X_{ijkm}C_{ijkm} + \sum_k U_kY_k \\
& \frac{1}{P}\sum_i\sum_j\sum_k\sum_m (T_{ik}+T_{km}+T_{mj})A_{ik}A_{jm} && \text{(LP5)} \\
\text{s.t.} \quad & Y_k \in \{0,1\} & \forall k \quad & (34) \\
& A_{ik} \in \{0,1\} & \forall i,k \quad & (35) \\
& A_{ik} \le Y_k & \forall i,k \quad & (36) \\
& \sum_k\sum_m X_{ijkm} = 1, & \forall i,j \quad & (37) \\
& 0 \le X_{ijkm} \le 1 & \forall i,j,k,m \quad & (38) \\
& X_{ijkm} \le A_{ik} * A_{jm} & \forall i,j,k,m \quad & (39) \\
& \sum_i\sum_j\sum_k F_{ij}X_{ijkm} \le K_m & \forall m \quad & (40)
\end{aligned}
$$

6 n-hub allocation approximation algorithm

We fix the hubs number to n before network design to decrease the problem complexity and propose a approximation algorithm to solve this kind of n-hub allocation problem.

Intuitively, the distance between any 2 hubs k and m should in an acceptable range, which menas the value of d_{km} could not be too large or too small. Thus we indtroduce 2 distance threshholds τ_{ld} and τ_{hd}. For every 2 hubs k and m, the distance between them must satisfy $\tau_{ld} \leq d_{km} \leq \tau_{hd}$. This constraint assure our algorithm could terminate.

For a city i and two hubs k and m, we introduce 2 metrics $hc(i,k,m)$ and $ht(i,k,m)$ to measure the transportation cost and travel time through the path $i \to k \to m$.

$$hc(i,k,m) = s(F_{ik}D_{ik} + \alpha F_{km}D_{km}) \tag{41}$$

$$ht(i,k,m) = T_{ik} + T_{km} \tag{42}$$

Here s is the coefficient between transportation cost and flow-distance product and α is the transportation cost discount factor between hubs and non-hubs.

To optimize both the 2 values, we simply add them with weights w_1 and w_2 to get the final determine metric:

$$hscore(i,k,m) = w_1 \cdot hc(i,k,m) + w_2 \cdot ht(i,k,m) \tag{43}$$

We focus on following 2 conditions to find a reasonable good hub allocation:

1. n hubs $\mathbf{Hubs} = \{h_1, h_2, ..., h_n\}$ are determined and we need to assign the owned hub for a free city i.

   In this condition, we introduce *city-hub-score* to determine the best hub for city i. For a city i and a hub k, *city-hub-score*$(i,k,\mathbf{Hubs})$ is the sum of *hscore* from the city i to every hub m through hub k:

   $$\textit{city-hub-score}(i,k,\mathbf{Hubs}) = \sum_{m\in\mathbf{Hubs}} hscore(i,k,m) \tag{44}$$

   City i will be allocated to the hub k which has the highest *city-hub-score*.
2. $n-1$ hubs $\mathbf{Hubs} = \{h_1, h_2, ..., h_{-1}\}$ are determined and we need to find the last hub in the rest cities $\mathbf{Cities} = \{city_1, city_2, ..., city_t\}$ to form a complete network.

   In this condition, to determine a city k if should be selected as a hub, we introduce the metric *hub-score*$(k, \mathbf{Hubs}, \mathbf{Cities})$ to measure the choice:

   $$\textit{hub-score}(k,\mathbf{Hubs},\mathbf{Cities})) = \sum_{i\in\mathbf{Cities}} \sum_{m\in\mathbf{Hubs}} hscore(i,k,m) \tag{45}$$

   For specific city k, the metric means the total score of the paths any city $\to k \to$ all hubs.

Contructed on *city-hub-score* and *hub-score*, the basic idea of our algorithm is:

1. Randomly choose n hubs at the begining.
2. Calculate the *city-hub-score* for all cities and then allocate these cities to hubs based on this score.
3. Travel the hubs set, fro every hub i and the cities it dominate, remove i from hubs set and mark these cities as free cities, calculate the *hub-score* for all no-hub cities(including the cities owned by other hubs) then select a new hub for the free cities. Put the new hub to hubs set.
4. If the new hubs set equals to the previous one, then algorithm terminates.
5. Repeat from step 2 until the distance between every 2 hubs satisfies the threshold constraint $\tau_{ld} \leq d_{km} \leq \tau_{hd}$ or the algorithm reaches step 4.

Algorithm 1: n-hub(n, τ_{ld}, τ_{hd}, $F[1..N][1..N]$, $D[1..N][1..N]$, $T[1..N][1..N]$)

```
Cities ← {1, 2, ..., N};
Owner[1..N] ← Init an array represents the owner hub of cities;
Hubs ← Randomly select n hubs;
while true do
    for i ∈ Cities and i ∉ Hubs do
        bestowner ← arg min_k city-hub-score(i, k, Hubs) forall k ∈ Hubs;
        owner[i] ← bestowner;
    end
    OldHubs ← Hubs;
    for h ∈ Hubs do
        Hubs ← Hubs \ {h};
        FreeCities ← CitiesOwnedBy(h) ∪ {h};
        betterhub ← arg max_k hub-score(k, Hubs, FreeCities)) forall k ∈ Cities;
        Hubs ← Hubs ∪ {betterhub};
        Owner[betterhub] ← betterhub;
        for i ∈ Cities do
            Owner[i] ← betterhub;
        end
    end
    if Hubs ≠ OldHubs or τ_ld ≤ d_km ≤ τ_hd, ∀k, m ∈ Hubs then
        break;
    end
end
```

For N cities, to find a k-hub allocation, the complexity of our algorithm is $O(ck^2N^2)$, here c is the total rounds our algorithm execute. Cenerally c would be a very small constant. The parameter τ_{ld} and τ_{hd} will decrease the execute time our this algorithm. We could also set them to $-\infty$ and ∞ to for better accuracy. In our experiments we just ignore these 2 parameters because the performance of the program is in an acceptable range.

7 Evalution

We evalute our LP2 model on the given dataset. To assure the transportation cost and the update cost are in the same order of magnitude, we set $s = 0.00001$. Here $N = 81$ then $P = 2 * N * N = 13122$.

2-stage muli-obejctive optimizition costs two much time, thus we simply add the two object functions to get the final object function :

$$
\begin{aligned}
& \frac{1}{P}\sum_i\sum_j\sum_k\sum_m F_{ij}A_{ik}A_{jm}C_{ijkm} + \sum_k U_kY_k + \frac{1}{P}\sum_i\sum_j\sum_k\sum_m (T_{ik}+T_{km}+T_{mj})A_{ik}A_{jm} \\
= \quad & \sum_k U_kY_k \\
& + \frac{s}{P}\sum_i\sum_j\sum_k\sum_m F_{ij}A_{ik}A_{jm}(D_{ik}+D_{km}+D_{mj}) \\
& + \frac{1}{P}\sum_i\sum_j\sum_k\sum_m (T_{ik}+T_{km}+T_{mj})A_{ik}A_{jm} \\
= \quad & \sum_k U_kY_k \\
& + \frac{0.00001}{13122}\sum_i\sum_j\sum_k\sum_m F_{ij}A_{ik}A_{jm}(D_{ik}+D_{km}+D_{mj}) \\
& + \frac{1}{13122}\sum_i\sum_j\sum_k\sum_m (T_{ik}+T_{km}+T_{mj})A_{ik}A_{jm} \qquad (46)
\end{aligned}
$$

The constraints are same with LP2:

$$
\begin{aligned}
\text{s.t.} \quad & Y_k \in \{0,1\} && \forall k && (47)\\
& A_{ik} \in \{0,1\} && \forall i,k && (48)\\
& A_{ik} \le Y_k && \forall i,k && (49)\\
& \sum_k A_{ik} = 1 && \forall i && (50)
\end{aligned}
$$

We build this model with OPL language and solve the problem in IBM CPLEX,

Acknowledgements

Here is your acknowledgements. You may also include your feelings, suggestion, and comments in the acknowledgement section.

References

1. Author, F.: Article title. Journal **2**(5), 99–110 (2016)
2. Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016).
3. Author, F., Author, S., Author, T.: Book title. 2nd edn. Publisher, Location (1999)
4. Author, A.-B.: Contribution title. In: 9th International Proceedings on Proceedings, pp. 1–2. Publisher, Location (2010)
5. LNCS Homepage, `http://www.springer.com/lncs`. Last accessed 4 Oct 2017