İstanbul Beykent Üniversitesi Fen ve Mühendislik Bilimleri Dergisi 2023, 16(1), 33-42
Istanbul Beykent University Journal of Science and Engineering 2023, 16(1), 33-42

Makale Türü: Konferans Bildirisi
Paper Type: Conference Paper

Geliş tarihi/Received: 12/12/2022
Kabul tarihi/Accepted:12/01/2023

AN EXACT SOLUTION FOR REAL-LIFE TRANSSHIPMENT PATH PROBLEM

GERÇEK HAYAT AKTARMA PROBLEMİNE TAM ÇÖZÜM

https://doi.org/10.20854/bujse.1218139

Zehra Hafızoğlu Gökdağ[1,*], Salih Cebeci[2]

Abstract

In industrial engineering, transportation planning, vehicle routing problem, warehousing, inventory management, and customer service are logistics problems. Graph theory algorithms provide solutions to logistics problems such as the shortest path, minimum spanning tree, and vehicle routing problems. In a logistics company system with branches and transfer centers to which the branches are affiliated, if the sorting process is carried out in the transfer centers, the deliveries collected from the branches must be transported to a transfer center. Thus, there are situations where delivery is transferred in the sending branch, the sending transfer center, the receiving transfer center, and the receiving branch, respectively. In this flow, transferring with a single transfer center without visiting two transfer centers reduces the total cost. While moving from the sender transfer center to the receiver transfer center, stopping by some branches on the way allows us to complete the transfer process with a single transfer center and eliminates the necessity of leaving the vehicle from the receiver transfer center to these branches again. Thus, the number of vehicles that need to go from the receiver transfer center to the branches is reduced. The mentioned logistics structure is defined as a graph that is considered a network design problem. Given the sender transfer center S, the receiver transfer center T, the set of branches A connected to S, and the set of branches C that are not connected to S or T, a counting algorithm that gives the minimum value route among all combinations are designed in order to find the optimal route from the source node $s \in A \cup \{S\}$, to the target node $t = T$. The algorithm has been implemented in Python and Gams and tested by the different number of elements of the set A and the set C.

Keywords: Combinatory Problem, Graph Theory, Logistics, Network Design Problem

Özet

Şubelerin ve şubelerin bağlı olduğu aktarma merkezlerinin bulunduğu bir lojistik firma sisteminde, eğer ayrıştırma işlemi aktarma merkezlerinde yapılıyorsa, şubelerden toplanan gönderilerinbir aktarma merkezine taşınması gerekir. Böylece gönderiler sırasıyla gönderen şubede, gönderen transfer merkezinde, alıcı transfer merkezinde ve alıcı şubede transfer edildiği durumlar ortaya çıkmaktadır. Bu akışta iki aktarma merkezine uğramadan tek bir aktarma merkezi ile aktarma yapılması toplam maliyeti düşürmektedir. Gönderici aktarma merkezinden alıcı aktarma merkezine hareket ederken yol üzerindeki bazı şubelere uğramak aktarma işlemini tek bir aktarma merkezi ile tamamlamamızı sağlar ve alıcı aktarma merkezinden bu şubelere tekrar araç yönlendirme zorunluluğunu ortadan kaldırır. Böylece alıcı aktarma merkezinden şubelere gitmesi gereken araç sayısı azaltılmış oluyor. Söz konusu lojistik yapı, bir ağ tasarım problemi olarak ele alınan bir çizge olarak tanımlanmaktadır. Gönderici transfer merkezi S, alıcı transfer merkezi T, S'ye bağlı A şubeleri kümesi ve S veya T'ye bağlı olmayan C şubeleri kümesi verildiğinde, $s \in A \cup \{S\}$ kaynak düğümünden $t = T$ hedef düğümüne giden en uygun rotayı bulmak için tüm kombinasyonlar arasından minimum değerli rotayı veren bir sayma algoritması tasarlanmıştır. Algoritma Python ve Gams'da uygulanmış ve A ve C kümelerinin farklı eleman sayıları ile test edilmiştir.

Anahtar Kelimeler: Kombinasyon Problemi, Grafik Teorisi, Lojistik, Ağ Tasarımı Problemi

[1,*] Corresponding Author: Hepsijet & Kadir Has University, Faculty of Engineering and Natural Sciences, Department of Industrial Engineering, zehra.gokdag@hepsijet.com, orcid.org/0000-0002-5804-3105

[2] Hepsijet & Kadir Has University, Faculty of Engineering and Natural Sciences, Department of Industrial Engineering, salih.cebeci@hepsijet.com, orcid.org/0000-0003-2200-6318

1. INTRODUCTION

Logistics network topology analysis studies have been carried out in the literature with the definitions of graph theory. A method to analyze the attributes of nodes and edges in the graph structure of the Urban Logistics system, with some graph theory and complex network definitions, was proposed (Li & Zhang, 2009). In transportation planning, factors such as reducing transportation cost, route length, and the number of machines (or drivers) are of great importance in order to maximize profit (Malandraki et al., 2001).

In the logistics system, some problems such as Shortest Path Problem, Vehicle Routing Problem, Traveling Salesman Problem, and Minimum Spanning Tree are well-known Network Design Problems (Feremans et al., 2003). The Network Design problem is the problem of finding the S_G subgraph of a G graph that satisfies the balance of the flow and side constraints.

Exact algorithms such as Set Partitioning Algorithm (Balinski & Quandt, 1964), Branch-and-Bound Algorithm (Christofides & Eilon, 1969), Dynamic Programming (Eilon et al., 1971), and heuristics such as Clarke and Wright Algorithm (Clarke & Wright, 1964), Nearest Neighbor Algorithm (Bellmore & Nemhauser, 1968), A* Search Algorithm (Hart et al., 1968), Set Partitioning Heuristics (Gillett & Miller, 1974), 2-Opt & 3- Opt (Croes, 1958), Lin Kernighan Heuristic (Lin & Kernighan, 1973), Tabu Search (Glover, 1986), Simulated Annealing (Kirkpatrick et al., 1983), Genetic Algorithm (Holland, 1962) have been developed since the early 20th century for the solution of VRP and TSP.

Kruskal's Algorithm (Kruskal, 1956), Prim's Algorithm (Prim, 1957), Boruvka's Algorithm (Boruvka, 1926) are known as solution algorithms for the minimum spanning tree problem.

The Shortest Path Algorithm is a simple network design algorithm. Various methods such as Dijkstra's Shortest Path algorithm (Dijkstra, 1959), Bellman-Ford Algorithm (Bellman, 1958; Ford, 1956), Floyd- Warshall's Algorithm (Floyd, 1962), and Path Labeling Algorithm (Pandian & Rajendran, 2010) have been developed since the past to solve this problem.

2. NETWORK DESIGN PROBLEM

The Network Design Problem's (NDP) goal is to identify the optimal subgraph $S_G = (V', E')$ in a graph $G = (V, E)$ under the constraints of forcing, flow balance, and side constraints (Magnanti, 1984). In order to explain NDP, sets, parameters, and decision variables notations are defined as

Sets:

- $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes and $v_i \in V$ is the node in the set of nodes
- E contains each edge e_{ij} from v_i to v_j
- k is the commodity in the commodities set K
- M is the constraints set

Parameters:

- c_{ij}^{k} is the cost of one unit $k \in K$ along $e_{ij} \in E$
- f_{ij} is the fixed cost of containing $e_{ij} \in E$
- q_k is the total quantity of commodity $k \in K$
- l_{ij} is the capacity of $e_{ij} \in E$

Decision Variables:

- $x_{ij} = \begin{cases} 1, & if\ e_{ij} \in E\ is\ containing\ in\ the\ S_G \\ 0, & otherwise \end{cases}$
- y_{ij}^{k} is the fraction of commodity $k \in K$ on $e_{ij} \in V$

Mathematical Formulation:

$$\min \sum_{k \in K} \sum_{e_{ij} \in E} c_{ij}^{k}\, q_k y_{ij}^{k} + \sum_{e_{ij} \in E} f_{ij} x_{ij} \qquad (1)$$

$$\sum_{\in K} q_k y_{ij}^{k} \le l_{ij} x_{ij} \qquad \forall e_{ij} \in E \qquad (2)$$

$$\sum_{j:e_{ij} \in E} y_{ij}^{k} - \sum_{j:e_{ji} \in E} y_{ji}^{k} = \begin{cases} 1, & if\ i = O(k) \\ -1, & if\ i = D(k), \\ 0, & otherwise. \end{cases} \qquad \forall v_i \in V\,, \qquad k \in K \qquad (3)$$

$$\left(y_{ji}^{k}, x_{ij}\right) \in M \qquad (4)$$

$$x_{ij} \in \{0,1\} \qquad (5)$$

$$0 \le y_{ji}^{k} \le 1 \qquad \forall v_i \in V, \qquad k \in K \qquad (6)$$

The aim of Equation (1) is to minimize the network's total cost, which is made up of the cost of each commodity and the fixed cost of each included arc. The "forcing" limits shown in Equation (2) ensure that the flow on any arc does not go over the capacity designated for that

arc. The flow balancing constraint, also known as the flow conservation constraint, is found in Equation (3) and assures that commodities only enter or exit the network at their respective origin $O(k)$ or destination $D(k)$ nodes. To add additional restrictions to the NDP to adapt it to particular applications, see the side constraints, Equation (4). The range restrictions for the flow and decision variables are Equation (5) and Equation (6). The problem examined in this study is a simple network design problem.

3. PROBLEM DEFINITION

The logistics company has branches and transfer centers to which these branches are affiliated. Packages are collected from branches in the region of Transfer Center 1 (S) and brought to S. The vehicle is taken out from S and transferred to Transfer Center 2 (T), then the vehicles are directed to the branches in the region of T.

In this study, in the scenario given in Figure 1, the optimal route solution that goes to the node T is obtained by visiting all the points in $A = \{A_1, A_2, \dots, A_n\}$, $C = \{C_1, C_2, \dots, C_m\}$, and S. A is the set of branches that are connected to the transfer center S, while C is the set of branches that are connected to different transfer centers that are not connected to the transfer center S or T. This combinatory problem is solved by considering all possibilities with some external factors (weather conditions, traffic, road works, closed roads, etc.).

Figure 1: Problem scenario.

Let $G = (V, E)$ be a graph, the node set be $V = \{S, T\} \cup A \cup C$ (Figure 2) where the source node $\theta \in A \cup \{S\}$, a target node $t = T$. The set of edges E is a distance associated with each edge $(i, j) \in E$, in this study, distances between nodes are taken from the open routing service. At the same time, a random value is added to this value.

In the problem examined in this study, the starting point must be S or a point from the set A. There is no priority between point S and a point of set A, so the arc between A and S is bidirectional in Figure 2. It is not possible to go to any point in set C without visiting point S. After the point S is visited, there is no priority between the elements of set C and set A, and the route ends at the point T.

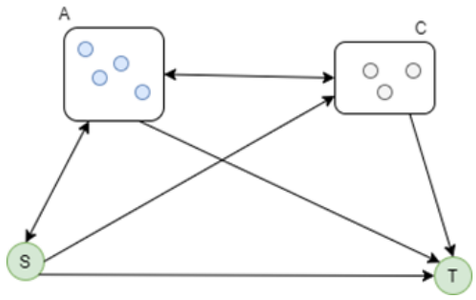

Figure 2: The graph structure of the problem.

Each node in $V \setminus \{T\}$ must be visited once by a vehicle and all vehicle routes will start at the point in $\{S\} \cup A$ and end at point T. The integer programming formulation to determine the route with minimum cost vehicle routes according to the constraints is as follows:

Sets:

- A: the set of branches that are connected to the point S, and $A = \{A_1, A_2, \dots, A_n\}$
- C: the set of branches that are not connected to the point S or the point T, and $C = \{C_1, C_2, \dots, C_m\}$.
- V: the set of all nodes, $V = \{S, T\} \cup A \cup C$.

Parameters:

- d_{ij}: the edge cost, $d_{ij} \in R$
- M: a big number.

Decision variables:

- x_{ij}: the binary variable that take value 1 if the arc of (i, j) belongs to the path.
- δ_i: the order at which location i is visited, $\delta_i > 0$.
- θ: the starting point.

Objective Function:

$$min \sum_{i \in V} \sum_{j \in V} d_{ij}.x_{ij} \qquad (7)$$

Constraints:

$$\sum_{i \in V} x_{ij} = 1, \quad \forall j \in V \setminus \{\theta\} \tag{8}$$

$$\sum_{j \in V} x_{ij} = 1, \quad \forall i \in V \setminus \{t\} \tag{9}$$

$$\sum_{i \in V} x_{i\theta} = 0 \tag{10}$$

$$\sum_{j \in V} x_{tj} = 0 \tag{11}$$

$$x_{ii} = 0, \qquad \forall i \in V \tag{12}$$

$$\delta_j > \delta_i - M(1 - x_{ij}), \qquad \forall i, j \in V, \ i \neq j \tag{13}$$

$$\delta_c > \delta_S, \qquad \forall c \in C \tag{14}$$

Constraints (8) and (9) specify that only one arc enters each vertex, and only one arc leaves from each vertex, respectively. Constraints (10) and (11) specify that there is no arc entering θ, and there is no arc leaving from t, respectively. Constraint (12) means that no node comes back to itself; there is no loop. Constraint (13) is for subtour elimination. (If $x_{ij} = 1$, $\delta_j > \delta_i$.) Constraint (14) means that any point of the set C cannot be visited without visiting point S.

4. METHODOLOGY AND NUMERICAL EXPERIMENTS

The formula $f(n, m)$ for the number of all feasible solutions where $n = |A|$ and $m = |C|$ is as follows:

$$f(n, m) = \sum_{i=0}^{n} \binom{n}{i} . i! \, (m + n - i)!$$

The algorithm is implemented in Python 3.9. All experiments are implemented on a Laptop with a Core 5 CPU, 64-bit operating system, and 16 GB ram. The algorithm generates all feasible solutions and calculates the total path distance for every solution. Then we choose the solution with the minimum total path distance. The number of all feasible solutions for different m and n values, and algorithm run-time are given in Table 1 and Table 2, respectively.

Table 1: The number of feasible solutions.

f(x)	m=0	m=1	m=2	m=3	m=4	m=5	m=6
n=0	1	1	2	6	24	120	720
n=1	2	3	8	30	144	840	5760
n=2	6	12	40	180	1008	6720	51840
n=3	24	60	260	1260	8064	60480	518400
n=4	120	360	1680	10080	72576	604800	5702400
n=5	720	2520	13440	90720	725760	6652800	68428800
n=6	5040	20160	120960	907200	7983360	79833600	889574400

Table 2: Algorithm run-time (miliseconds).

t	m=0	m=1	m=2	m=3	m=4	m=5
n=0	0	0	0	0	0	0
n=1	0	0	0	0	0	0
n=2	0	0	0	0	0	15
n=3	0	0	0	0	16	113
n=4	0	0	0	17	188	1443
n=5	0	0	16	171	1585	14794

The mathematical model is also modeled by Gams 36. The real-life problem given above is also solved by this Gams model, and it was seen that the same optimal solution was obtained.

A real-life problem instance where $S = \{\dot{I}zmir\ TM\}$, $T = \{Antalya\ TM\}$, 6 branches connected to S (n=6) and 3 branches not connected to S and T (m=3) is examined. For these specific m and n values, the number of feasible solutions is 907200, and the run-time of the algorithms is measured as 1569 milliseconds. The optimal solution is shown in Figure 3.

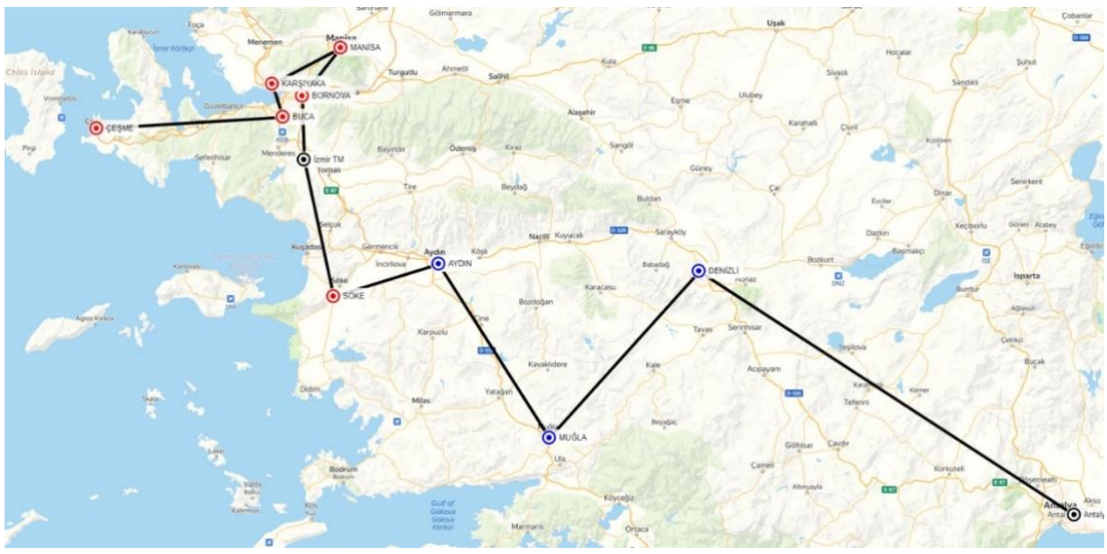

Figure 3: The optimal solution of the real-life example with n=6 branches (red), m=3 branches (blue), and transfer centers (black).

5. CONCLUSION

In this study, the problem of transshipment between transfer centers in logistics structure is discussed. While transferring between two transfer centers, the brute force optimal result was determined by adding the branches to the route and examining all combinations with the counting algorithm for the optimal route result. It is concluded that it is a suitable algorithm for real-life small-sized problems. However, the heuristic algorithm can be proposed to solve large-sized problems in the future.

ACKNOWLEDGMENT

This publication has been created by utilizing TUBITAK-2244 Industrial Doctorate Program (Project No: 119C147). However, all responsibility for the publication belongs to the owners of the publication. Financial support from TUBITAK does not constitute an endorsement by TUBITAK of the scientific content of the publication.
I would also like to thank the logistics company that provided data support for this study.

REFERENCES

Balinski, M. L., & Quandt, R. E. (1964). On an integer program for a delivery problem. *Operations Researc*h, *12*(2), 300–304. https://doi.org/10.1287/opre.12.2.300

Bellmore, M., & Nemhauser, G. L. (1968). The traveling salesman problem: A survey. *Operations Research*, *16*(3), 538–558. https://doi.org/10.4018/978-1-7998-3970-5.ch006

Boruvka, O. (1926). Borůvka, Otakar: Scholarly works. *The Czech Digital Matematics Library*. 37-58.

Christofides, N., & Eilon, S. (1969). An algorithm for the vehicle-dispatching problem. *Journal of the Operational Research Society*, *20*(3), 309–318.

Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, *12*(4), 568–581. https://doi.org/10.1287/opre.12.4.568

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, *6*(6), 791-812.

Eilon, S., Watson-Gandy, C. D. T., & Heilbron, A. (1971). A vehicle fleet costs more. *International Journal of Physical Distribution*, *1*(3), 126–132. https://doi.org/10.1108/eb038836

Feremans, C., Labbé, M., & Laporte, G. (2003). Generalized network design problems. *European Journal of Operational Research*, *148*(1), 1–13. https://doi.org/10.1016/S0377-2217(02)00404-6

Gillett, B. E., & Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations Research*, *22*(2), 340–349.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, *13*(5), 533–549. https://doi.org/10.1016/0305-0548(86)90048-1

Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). Formal basis for the heuristic determination Eijj. *Systems Science and Cybernetics*, *4*(2), 100–107.

Holland, J. H. (1962). Outline for a logical theory of adaptive systems. *Journal of the ACM (JACM)*, *9*(3), 297–314. https://doi.org/10.1145/321127.321128

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680. https://doi.org/10.1007/978-3-642-24974-7_7

Kruskal, J. B., Jr. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *American Mathematical Society*, *7*(1), 48–50.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, *21*(2), 498–516. https://doi.org/10.1007/s10489-006-8514-7

Open Routing Service. (n.d.). https://openrouteservice.org/

Prim, R. C. (1957). Shortest connection networks and some generalizations. *Bell System Technical Journal, 36*(6), 1389–1401. https://doi.org/10.1002/j.1538-7305.1957.tb01515.x