An empirical study on Multiple Traveling Salesmen

EE 5283 Optimization in Engineering and Data Science
Fall 2023

Tien Van Nguyen
wdp076

Abstract—The Multiple Traveling Salesman Problem (MTSP) extends the classic Traveling Salesman Problem (TSP) by involving multiple salesmen tasked with visiting a set of cities exactly once before returning to their initial position, aiming to minimize travel costs. MTSP finds applications in diverse fields such as transportation, data collection, and network connectivity. This problem involves determining optimal routes for multiple salesmen, each starting and ending at a depot, visiting all cities exactly once. The abstract formulation seeks to minimize the total cost across these routes while ensuring that each city is visited only once. This study focuses on formalizing MTSP as a mixed-integer linear program, presenting deterministic solutions, and assessing the impact of varying salesman numbers on total cost.

Index Terms—Combinatorial problem, Mixed-Integer Linear Program, Multiple agent schedule

I. Introduction

The Multiple Traveling Salesman Problem (MTSP) is a general form of the well-known Traveling Salesman Problem (TSP), where multiple salesmen start from the initial position to visit a given number of cities exactly once and return to the initial position with the minimum traveling cost [1]. The solution is applied for various fields such as Transportation and delivery, WSN data collection and network connectivity, Search and rescue, etc [2]. For instance, in smart city logistics, determining the optimal routes for a fleet of vehicles such as drones, and also minimize total travel distance is a critical task.

MTSP is widely studied and was originally defined as which, given a set of cities, one depot, m salesmen, and a cost function (e.g. time or distance), MTSP aims to determine a set of routes for m salesmen minimizing the total cost of the m routes, such that each route starts and ends at the depot and each city is visited exactly once by one salesman. Figure 1 shows an illustration of MTSP, in which two workers start at depot v_1 to travel to all cities $v_2, \dots, v_9$. The MTSP minimizing the sum of all salesmen's tour costs can be stated in the general form [1] as below.

$$\begin{aligned} &\text{minimize}_{\text{Tour}_{R_i} \in \text{TOURS}} \quad (\sum_{i=1}^{M} C(Tour_{R_i})) \\ &\text{subject to} \quad \text{Tour}_{R_i} \cap \text{Tour}_{R_j} = \emptyset \\ &\qquad \forall i \neq j, 1 \leq i, j \leq m \\ &\qquad \cup_{i=1}^{i=m} \text{Tour}_{R_i} = \{T_j, 1 \leq j \leq n\} \end{aligned} \tag{1}$$

, where $C(\text{Tour}_{R_i})$ is the tour cost of agent R_i and TOURS is the set of all possible Tours, and $\{T_1, \dots, T_n\}$ is a set of n places need to visit. Moreover, the two conditions in (1) guarantee that all targets are visited and that only one agent visits each target. If $m = 1$, the problem becomes the standard (single) Traveling Salesman Problem (TSP).

The variants of MTSP result from considering the different characteristics of the salesmen, the depot, the city, and the problem constraints and objectives [1]. For example, the problem may consider a single depot, or multiple depots, or the salesman's path is closed, or open in some applications, the salesman does not need to return to the depot and it can stay in the last visited city; single or multiple objectives; etc.

In this work, we try to solve MTSP which is stated in (11). The main works in this project include:

- Formalize the Multiple Salesmen Traveling Problem as the mixed-integer linear program
- Show the deterministic solution of the formalized problem.
- Examining the effectiveness of varying numbers of salesmen on total cost.

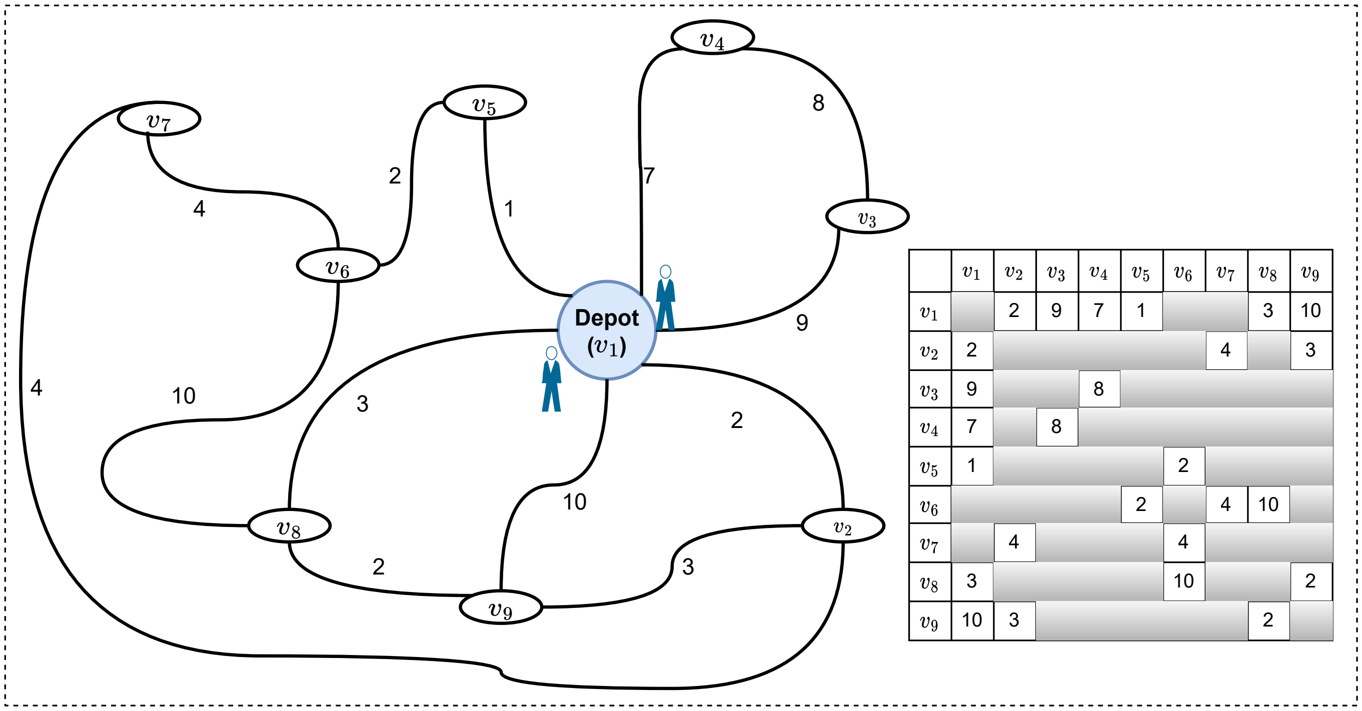
II. Problem formulation

A. Notation

We use graph representation to formalize MTSP. Let denote $G = (V, E)$ is the undirected graph established by V, E, in which $V = \{v_1, \dots, v_n\}, E = \{(i, j)\}$ are the set of vertices, and set of edges respectively. In that, $n = |V|$ is the number of vertices, and (i, j) denotes the path between two vertices v_i, v_j. If G is complete (existing path between any two vertices), then $|E| = n(n-1)/2$. Each city is assigned with the ith vertex. The distance between two vertices v_i and v_j is $d_{ij} > 0$, $\forall i \neq j$, which is stored in a matrix $D \in {\mathbb{R}_+}^{n \times n}$. Due to G is undirected, D is symmetric, or $D^T = D$; and (k, l), (l, k) are the same identical for any pair $(k, l) \in E$. Simply, we only consider the set E containing pairs (k, l) with $k \leq l$.

We use the first vertex v_1 as the beginning location (called depot) of m salesmen. The goal is to find the formulation of tours for all m salesmen, so that:

- all vertices are visited exactly once, and
- selecting edges such that minimize the total traveling distance of all m salesmen.



	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
v_1	▨	2	9	7	1	▨	▨	3	10
v_2	2	▨	▨	▨	▨	▨	4	▨	3
v_3	9	▨	▨	8	▨	▨	▨	▨	▨
v_4	7	▨	8	▨	▨	▨	▨	▨	▨
v_5	1	▨	▨	▨	▨	2	▨	▨	▨
v_6	▨	▨	▨	▨	2	▨	4	10	▨
v_7	▨	4	▨	▨	▨	4	▨	▨	▨
v_8	3	▨	▨	▨	▨	10	▨	▨	2
v_9	10	3	▨	▨	▨	▨	▨	2	▨

Fig. 1. Example for the multiple salesman traveling problem. The graph (on the left) shows that there are two salesmen, the v_1 is used as a depot where all salesmen start and also end their travel. There are 9 vertices (cities), some pairs of cities have the direct path moving between each other (e.g., $v_2, v9$), but some others do not (e.g., v_2, v_3). The table on the right side is the distance matrix which is constructed from the cost, especially the distance between cities. If there is a direct path between v_i, and v_j, the value corresponding to (v_i, v_j) is filled by the distance shown in the graph, otherwise, they are displayed by gray cells.

Also, denote $X \in \{0,1\}^{n\times n}$, in which x_{ij} indicate whether we choose the pair (v_i, v_j) in the tour of solution or not. If (i,j) is chosen, then $x_{ij} = 1$, otherwise it would be 0.

$$x_{ij} = \begin{cases} 1 & ,(i,j) \text{ is included.} \\ 0 & ,o.w \end{cases} \tag{2}$$

B. Objective and constraints

The goal is to minimize the total distance of all m salesmen which is stated as the objective function below.

$$f(X) = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} \quad ,\forall\ (i,j) \in E, \text{and } i \neq j \tag{3}$$

with variables are x_{ij}, $1 \leq i,j \leq n$. We explicitly set that if $(i,j) \notin E$, $d_{ij} = \infty$; and $\forall i \in [1,n]$, then $d_{ii} = \infty$, to eliminate $(\forall\ (i,j) \in E, \text{and } i \neq j)$ in (3). Therefore, the objective function becomes:

$$f(X) = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} \tag{4}$$

To explain setting these values above to infinity we want to minimize the objective function, and we also are not able to select edges are not exist in the graph, therefore extremely large value is justifiable, and it also helps the optimization algorithm not select this path to the routine solution ($x_{ij} = 0$). The same explanation for setting d_{ii} is the same. Although the distance between v_i and itself is 0, we do not want the agents to jump to the same position (not moving). Instead, setting d_{ii} would push the salesmen to validate other reachable vertices and help the algorithm work.

Besides minimizing the cost, we also design X such that satisfies these constraints:

- Every vertex v_i is included exactly once and every salesman, only enters and exits a vertex exactly once.

$$\sum_{j=1}^{n} x_{ij} = 1 \quad ,\forall\ v_i \in V \setminus \{v_1\} \tag{5}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad ,\forall\ v_j \in V \setminus \{v_1\} \tag{6}$$

- All m salesmen must start at the depot.

$$\sum_{j=2}^{n} x_{1j} = m \tag{7}$$

- All m salesmen must end their journey at the depot.

$$\sum_{i=2}^{n} x_{i1} = m \tag{8}$$

- In addition, to guarantee sub-tour elimination, the found solutions must satisfy the following:

$$\begin{gathered} u_i - u_j + n \times x_{ij} \leq n - 1 \\ \forall (i,j) \in E,\ 2 \leq i \neq j \leq n \end{gathered} \tag{9}$$

, with $u_i, \forall\ 1 \le i \le n$, be an integer decision variable for each vertex i, that represents visiting order in the tour (e.g., $u_3 = 4$ indicate before visiting the vertex v_3, there are already 3 vertices were cross through before). Thanks to setting $d_{ii} = \infty$, and $d_ij = 0, \forall\ (i,j) \notin E$, we can eliminate checking condition from $(\forall\ (i,j) \in E,\ 2 \le i \ne j \le n)$ to $(\forall\ 1 \le i,j \le n)$. Hence, (9) is written equivalently as below.

$$u_i - u_j + n \times x_{ij} \le n-1 \quad ,\forall\ 1 \le i,j \le n \tag{10}$$

C. Optimization problem

Based on these formulas above, we come up with the optimization problem which is written explicitly as. Given $V = \{v_1, \ldots, v_n\}$, $E = \{(i,j)\}_{i,j=1,i\ne j}^{n}$ and $D \in {\mathbb{R}_+}^{n\times n}$, containing entries d_{ij}, we want to find the solution $X \in \{0,1\}^{n\times n}$ including x_{ij} such that:

$$\begin{aligned}
\min_{X\in\{0,1\}^{n\times n}} \quad & f_0(X) = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} \\
\text{subject to} \quad & \sum_{j=1}^{n} x_{ij} = 1 \quad ,\forall\ v_i \in V \setminus \{v_1\} \\
& \sum_{i=1}^{n} x_{ij} = 1 \quad ,\forall\ v_j \in V \setminus \{v_1\} \\
& \sum_{j=2}^{n} x_{1j} = m \\
& \sum_{i=2}^{n} x_{i1} = m \\
& u_i - u_j + n \times x_{ij} \le n-1 \\
& \quad \forall\ 1 \le i,j \le n
\end{aligned} \tag{11}$$

, which is a mixed-integer linear program with variables $X \in \{0,1\}^{n\times n}$, and $u_i \in \mathbb{Z}_+$. To be more concise, we can present (11) as the optimization problem with matrix variable (see section A).

$$\begin{aligned}
\min_{X\in\{0,1\}^{n\times n}} \quad & f_0(X) = tr(DX) \\
\text{subject to} \quad & X1 = \tilde{1} \\
& X^T1 = \tilde{1} \\
u - \mathbf{1}\mathbf{k}^T(j)u & + nX\mathbf{k}(j) \preceq \mathbf{1}(n-1) \\
1 \le & j \le n
\end{aligned} \tag{12}$$

with variables $X \in \{0,1\}^{n\times n}$, $u \in \mathbb{Z}_+^n$

III. Solution approach

A. Convexity

The problem (12) have $2n$ equalities ($X1 = \tilde{1}$, and $X^T1 = \tilde{1}$), and n^2 inequalities (n ones for each j corresponding $u - \mathbf{1}\mathbf{k}^T(j)u + nX\mathbf{k}(j) \preceq \mathbf{1}(n-1)$).

Although the objective function is linear, all constraints are linear respective to X, it is non-convex (in general, the mixed integer linear program is non-convex) due to the domain of the problem, $X \in \{0,1\}^{n\times n}$, a discrete set, which is non-convex. Or we can show that it does not qualify the convexity condition, $\forall\ x,y \in \{0,1\}$,

$$\theta x + (1-\theta)y \in \{0,1\} \tag{13}$$

do not holds $\forall \theta \in [0,1]$.

B. Deterministic approach

For finding an exact solution, the deterministic approaches such as [3] formalized it as integer linear programming, then proposed a customized branch-and-cut algorithm that reached the optimal solution within 300s using an instance of 100 targets and 5 vehicles.

However, this computational complexity for looking at closed-form solutions is an expenditure, especially in the context of problem size growth. Meta-heuristic is one of the approaches that can reduce the search cost, but can near-optimal solutions. The typical algorithms in this category are Generic algorithm [4], or PSO-based approaches [5].

In this work, to find the solution for MTSP, we utilize the cvxpy[1] to effectively gain the solution.

IV. Numerical tests

In this section, we show a numerical example of MTSP and explain the solution and why it makes sense.

A. Data input

Let's consider the MTSP with 9 locations, in which we start from v_1 and travel to all other vertices $v_2, \ldots, v_9$. In particular, our vertices are:

$$\begin{gathered}
v_1(7,6),\ v_2(1,6),\ v_3(3,3) \\
v_4(1,10),\ v_5(5,8),\ v_6(9,3) \\
v_7(9,10),\ v_8(12,6),\ v_9(13,1)
\end{gathered} \tag{14}$$

By calculating the distance between any pairs, we can construct the distance matrix that is required to feed into the solver. Supposedly, we use Euclidean distance as measurement, then we have the distance matrix D as below.

$$\begin{bmatrix}
0 & 6 & 5 & 7.21 & 2.83 & 3.61 & 4.47 & 5 & 7.81 \\
6 & 0 & 3.61 & 4 & 4.47 & 8.54 & 8.9 & 11 & 13 \\
5 & 3.61 & 0 & 7.28 & 5.39 & 6 & 9.22 & 9.49 & 10.2 \\
7.21 & 4 & 7.28 & 0 & 4.47 & 10.63 & 8 & 11.70 & 15 \\
2.83 & 4.47 & 5.39 & 4.47 & 0 & 6.40 & 4.47 & 7.28 & 10.63 \\
3.61 & 8.54 & 6 & 10.63 & 6.4 & 0 & 7 & 4.24 & 4.47 \\
4.47 & 8.94 & 9.22 & 8 & 4.47 & 7 & 0 & 5 & 9.85 \\
5 & 11 & 9.49 & 11.70 & 7.28 & 4.24 & 5.0 & 0 & 5.10 \\
7.81 & 13 & 10.12 & 15 & 10.63 & 4.47 & 9.85 & 5.10 & 0
\end{bmatrix} \tag{15}$$

From (12), we seek to final matrix $X \in \{0,1\}^{9\times 9}$, and vector $u \in \mathbb{Z}_+^9$, therefore the number of variables are $9 \times 9 + 9 = 90$.

[1]https://www.cvxpy.org/

B. Deterministic solution

With the input given, supposedly we have $m = 2$ salesmen. We denote Tour_{R_i} as an order set, which represents the traveling process of salesmen R_i. The optimally exact solution found by cvxpy for each salesman is:

$$\begin{aligned} \text{Tour}_{R_1} &= \{v_5, v_4, v_2, v_3\} \\ \text{Tour}_{R_2} &= \{v_7, v_8, v_9, v_6\} \end{aligned} \tag{16}$$

, corresponding the optimal value $f(X^*) = 42.55$, and optimal point:

$$X^* = \begin{bmatrix} 0&0&0&0&1&0&1&0&0 \\ 0&0&1&0&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0 \\ 1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&1&0&0&0 \end{bmatrix} \tag{17}$$

It means that the first agent will start from the depot v_1, then move to the v_5, v_4, v_2, v_3 orderly, then back to v_1. Therefore, the total traveling cost for 1st salesman is $2.83 + 4.47 + 4 + 3.61 + 5 = 19.91$. Similarly, the total cost of 2nd salesman is $4.47 + 5 + 5.10 + 4.47 + 3.61 = 22.65$. Compared to the optimal value found and the sum of all traveling costs, we can see that they are the same.

The value of the minimum objective, obtained by different setting numbers of salesmen m, will be impacted. Especially, the curve in figure 2 change, in that the number of salesmen varies from 1 to 8 and accordingly the optimal objective would be changed increasingly.

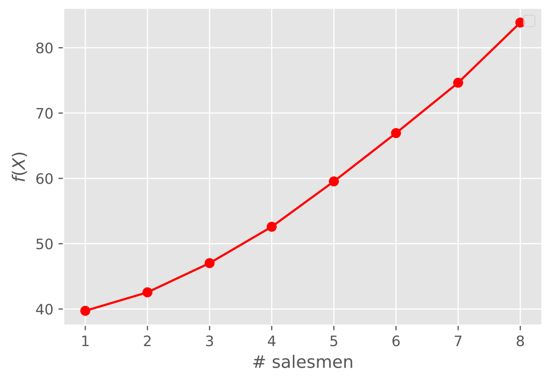


Fig. 2. The growth of objective function in case of the increasing number of salesmen.

Take a simple comparison between $m = 1$, and in case of $m = 8$. To satisfy allocating 8 agents to other vertices, we must send an agent to 1 vertex, and let them back to the depot due to the number of vertices needed to visit is 8. Therefore, in the case of $m = 8$, each agent must pay the cost for traveling to the assigned vertex and the same cost for backing to the depot. It is more expensive than the total cost in case we only have 1 agent, the smart tour should visit all vertices exactly once and not choose the previous paths to go. The same terminology is used for other cases $m = 2, \ldots, 7$.

References

[1] Cheikhrouhou, Omar, and Ines Khoufi. "A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy." Computer Science Review 40 (2021): 100369.

[2] Sundar, Kaarthik, and Sivakumar Rathinam. "Algorithms for heterogeneous, multiple depot, multiple unmanned vehicle path planning problems." Journal of Intelligent & Robotic Systems 88 (2017): 513-526.

[3] Miller, Clair E., Albert W. Tucker, and Richard A. Zemlin. "Integer programming formulation of traveling salesman problems." Journal of the ACM (JACM) 7.4 (1960): 326-329.

[4] Yuan, Shuai, et al. "A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms." European journal of operational research 228.1 (2013): 72-82.

[5] Wei, Changyun, Ze Ji, and Boliang Cai. "Particle swarm optimization for cooperative multi-robot task allocation: a multi-objective approach." IEEE Robotics and Automation Letters 5.2 (2020): 2530-2537.

Appendix

A. Compact form of original problem

Consider the product between 2 matrix $X \in \{0,1\}^{n\times n}$ containing column vectors $\{x_1, \ldots, x_n\}$ and $D \in \mathbb{R}_+{}^{n\times n}$, which is symmetric matrix, containing column vectors $\{d_1, \ldots, d_n\}$, we have:

$$\begin{aligned} (DX)_{ii} = d_i^T x_i &= \sum_{j=1}^{n} d_{ij} x_{ji} \\ &= \sum_{j=1}^{n} d_{ji} x_{ji} \end{aligned} \tag{18}$$

Consider the trace of DX, we have:

$$\begin{aligned} tr(DX) = \sum_{i=1}^{n} (DX)_{ii} &= \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ji} x_{ji} \\ = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij} x_{ij} &= f_0(X) \end{aligned} \tag{19}$$

Noticeably, $XD \neq DX$ in general, however $Tr(XD) = Tr(DX)$ holds, for any X, D satisfy the setting above. Therefore, the objective function 4 can be written equivalently as below.

$$f_0(X) = tr(DX) \tag{20}$$

The constraints (5) and (7) can be written equivalently as below.

$$X\mathbf{1} = \tilde{1} \tag{21}$$

, with $\mathbf{1}$ is an n-dimensional column vector with all entries equal 1, and $\tilde{1} = [m, 1, \ldots, 1]^T$ is the n-dimensional column vector modified from $\mathbf{1}$ by replacing the 1st element by m.

Similarly, constraints (6) and (8) together are equivalent with:

$$X^T\mathbf{1} = \tilde{1} \tag{22}$$

Denote $u = [u_1, \dots, u_n] \in \mathbb{Z}_+^n$, The constraints (10) can be written as below.

$$u - \mathbf{1}u_j + nx_j \leq \mathbf{1}(n-1) \quad , 1 \leq j \leq n \tag{23}$$

By defining $\mathbf{k}(j) : \mathbb{Z}_+ \mapsto \mathbb{Z}_+^n$ containing only one value 1 at j-position and other entries are 0, we can rewrite the equation above as below.

$$u - \mathbf{1k}^T(j)u + nX\mathbf{k}(j) \preceq \mathbf{1}(n-1) \quad , 1 \leq j \leq n \tag{24}$$

Therefore, the problem in (11) can be written as a compact form below. Given $D \in \mathbb{R}_+^{n\times n}$, and $m \in \mathbb{Z}_+$, we seek to find binary matrix X, vector u such that:

$$\begin{aligned} \min_{X\in\{0,1\}^{n\times n}} \quad & f_0(X) = tr(DX) \\ \text{subject to} \quad & X1 = \tilde{1} \\ & X^T1 = \tilde{1} \\ u - \mathbf{1k}^T(j)u + & nX\mathbf{k}(j) \preceq \mathbf{1}(n-1) \\ & 1 \leq j \leq n \end{aligned} \tag{25}$$

with variables $X \in \{0,1\}^{n\times n}$, $u \in \mathbb{Z}_+^n$