Paper:

A general variable neighborhood search for the multiple depots multiple traveling salesmen problem

Miao Wang, Bin Xin, Qing Wang

School of Automation, Beijing Institute of Technology, Beijing 100081, China.
Beijing Advanced Innovation Center for Intelligent Robots and Systems, Beijing 100081, China.
E-mail: brucebin@bit.edu.cn

[Received 00/00/00; accepted 00/00/00]

Abstract. The multiple depots multiple traveling salesmen problem (MDMTSP) is studied in this paper. In this problem, multiple salesmen start from different depots and visit multiple cities. The objective is to find a tour for each salesman to minimize the total travel distance. A general variable neighborhood search (GVNS) approach is proposed in this paper. Three local search methods are designed in this approach, including *k*-reverse, *k*-same-as-previous, and *k*-same-as-next. The MDMTSP consists of two subproblems: one is to assign cities to salesmen, and the other is to solve the travel path of each salesman. The proposed GVNS approach is used to solve the first subproblem, and the second subproblem is solved by the famous Lin-Kernighan Heuristic (LKH). A state-of-the-art market-based approach is implemented as a competitor. Computational results show that the GVNS performs better than the market-based approach in 8 out of 10 instances.

Keywords: Combinatorial optimization, Variable neighborhood search, Multiple depots multiple traveling salesmen problem

1. Introduction

The multiple depots multiple traveling salesmen problem (MDMTSP) is a variant of the multiple traveling salesmen problem (MTSP). In the MDMTSP, the salesmen start from the depots and visit nodes through a closed path. Every node is visited exactly once. The goal of the MDMTSP is to find a tour for each salesman to minimize the total travel distance. This problem has a wide range of applications [3, 4, 16]. A simple scenario of MDMTSP is that multiple drones take off from multiple different control stations and collect information at multiple target points, and finally all return to the control stations.

The classic traveling salesman problem (TSP) was shown to be NP-hard in [11]. As with the TSP, the MDMTSP is strongly NP-hard [19]. The MDMTSP has been solved in a variety of ways.

Two 2-approximation algorithms are available in the literature [3, 14]. Both have polynomial-time complexities, with one being nearly $O(n^4)$, and the other being $O(n^2 \log n)$. An improved algorithm by providing a new non-trivial extension of the well-known Christofides heuristic [5] of the TSP is proposed in [19], which has a tight approximation ratio of $2-1/(2k)$ and time complexity nearly $O(n^4)$.

In [13], Kivelevitch et al. studied an MTSP problem using a market-based approach without the constraint that all nodes must be visited. A hierarchical market-based approach was used to solve the MTSP problem in [12]. A market-based approach for solving the Length Constrained Min-Max MDMTSP was presented in [18], in which each salesman is assigned prices in each iteration and the node assignments are updated accordingly.

In this paper, a general variable neighborhood search (GVNS) approach is presented for solving the MDMTSP. Three local search methods are designed in this approach, including *k*-reverse, *k*-same-as-previous, and *k*-same-as-next methods. Inspired by [18], a state-of-the-art market-based approach is implemented as a competitor to the GVNS. The market-based approach proposed in [18] produces near-optimal results for Length Constrained Min-Max MDMTSP. The computational results show that the GVNS performs better than those of the market-based approach in 8 out of 10 instances.

2. Problem Formulation

The MDMTSP is fomulated as a Mixed-Integer Linear Programming (MILP) intending to minimize the total travel distance of the salesmen. The notations and descriptions are shown in Table 1.

Let N denote the number of depots and M denote the number of cities. Each depot d has one salesman. Each city should be visited by only one salesman. Let D and C denote the sets of depots and cities, respectively. Let L denote the union set of depots and cities, that is, $L = D \cup C$. The sets D and C are defined as

$$D = \{1,2,\ldots,N\}, C = \{N+1, N+2, \ldots, N+M\}. \quad (1)$$

Every salesman will eventually returns to the city where he started. The cities where the salesmen start is in set D, and the cities where the salesmen return is in set D'. The start and end depots of one salesman's tour must

Table 1. Notations and descriptions

Symbol	Description
N	Number of depots
M	Number of cities
D	Set of depots
C	Set of cities
L	Union of sets D and C
D'	Copy of set D
L'	Union of sets D, C and D'
c_{ij}	Cost to travel from city i to j
x_{ij}	Decision variable that indicates whether or not city j is visited directly after city i by a salesman
y_{ijk}	Decision variable that indecates whether node j is visited by a salesman originating from depot k after node i
J	Total travel distance of the salesmen
x	A solution of the MDMTSP
H_h	A finite set of pre-selected neighborhood structures
$H_h(x)$	Set of solutions in the h^{th} neighborhood of x
$f(x)$	The travel cost calculated from the solution x
G_i	Set of cities assigned to the salesman originating from depot i
T_i	The cost of the tour of the salesman originating from depot i
$\bar{T}$	Average value of $T_i, \forall i \in D$
p_i	The service price of the salesman originating from depot i
α	Amount of price update in each iteration of the market-based approach

be the same depot. The set D' is defined as

$$D' = \{N+M+1, N+M+2, \ldots, 2N+M\}. \quad (2)$$

The resulting set of all nodes is

$$L' = D \cup C \cup D'. \quad (3)$$

The node i and node $i+N+M$ in L' represent the start and end depots of depot $i \in D$, respectively. Let J denote the total travel distance of the salesmen. The cost to travel from city i to j is denoted by the constant c_{ij}. The decision variable x_{ij} indicates whether or not city j is visited directly after city i by a salesman, which satisfies

$$x_{ij} = \begin{cases} 1 & \text{if city } j \text{ is visited directly after city } i \text{ by a salesman,} \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

The decision variable y_{ijk} indecates whether node j is visited by a salesman originating from depot k after node i, which satisfies

$$y_{ijk} = \begin{cases} 1 & \text{if city } j \text{ is visited directly after city } i \text{ by a salesman originating from depot } k, \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

The MILP formulation of MDMTSP is as below:

$$minimize \quad J = \sum_{i \in L'} \sum_{j \in L'} \sum_{k \in D} c_{ij} y_{ijk} \quad (6)$$

subject to:

$$\sum_{j \in L'} x_{dj} = 1 \quad \forall d \in D, \quad (7)$$

$$\sum_{j \in L'} x_{ij} = 1 \quad \forall i \in C, \quad (8)$$

$$\sum_{i \in L'} x_{ij} = 1 \quad \forall j \in C, \quad (9)$$

$$\sum_{i \in L'} x_{id'} = 1 \quad \forall d' \in D', \quad (10)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \le |S| - 1 \quad \forall S \subset L', 2 \le |S| \le |L'| - 1, \quad (11)$$

$$\sum_{j \in C} x_{djd} = 1 \quad \forall d \in D, \quad (12)$$

$$\sum_{d \in D} \{x_{djd} + \sum_{i \in C} x_{ijd}\} = 1 \quad \forall j \in C, \quad (13)$$

$$\sum_{j \in C} x_{djd} + \sum_{i \in C} x_{ijd} = x_{j(d+M+N)d} + \sum_{i \in C} x_{jid} \quad \forall d \in D, j \in C, \quad (14)$$

$$\sum_{j \in C} x_{djd} = \sum_{j \in C} x_{j(d+M+N)d} \quad \forall d \in D, \quad (15)$$

$$x_{ij} \in \{0,1\} \quad \forall i,j \in L', \quad (16)$$

$$y_{ijk} \in \{0,1\} \quad \forall i,j \in L', k \in D. \quad (17)$$

Eqs. (7)-(10) are assignment constraints which ensure that each node has exactly one incoming arc and one outgoing arc. Eqs. (7) and (10) ensure that exactly one salesman in each depot leaves or returns. Eqs. (8) and (9) ensure that each city j is visited by exactly one salesman.

Eq. (11) is a representation of cycle elimination constraints [6]. The cycle elimination constraints avoid the existence of cycles (subtours) in C, resulting in routes along cities that do not have a salesman associated with them. $|S|$ and $|L'|$ in Eq. (11) represent the cardinality of set S and L', respectively [9].

Eqs. (12)-(15) are representations of cycle imposement constraints [2]. The cycle imposement constraints ensure that each salesman returns to the original depot. Eqs. (12) and (15) ensure that exactly one salesman departs and returns to depot d. Eq. (13) guarantee that each city is visited exactly once. Eq. (14) ensures that a salesman starts at depot d and visits city j first will either continues to another city i or returns to the same depot.

Finally, Eqs. (16) and (17) ensure that the decision variables x_{ij} and y_{ijk} are treated as binary variables.

3. Algorithm Design

In this paper, a GVNS approach is proposed for solving the MDMTSP. A market-based approach is implemented as a competitor to the GVNS approach.

Algorithm 1: Steps of the GVNS

```
x ← initial_assignment()
h ← 1  // neighborhood counter
repeat
    i ← 1  // neighbor counter
    repeat
        x′ ← argmin(f(x), f(x_i)), x_i ∈ H_h(x)
        if f(x′) < f(x) then
            x ← x′
            i ← 1
        else
            i ← i + 1
    until i > |H_h(x)|
until h > h_max
```

Fig. 1. Steps of the GVNS

3.1. General Variable Search Approach

Variable Neighborhood Search (VNS) [1, 8, 15] is a meta-heuristic algorithm for constructing heuristic algorithms. The basic idea of VNS is the systematic change of the neighborhood structure in the local search algorithm. To construct different neighborhood structures and perform systematic searches, it is necessary to provide some (quasi) metrics for the solution space and then induce neighborhoods from them. Different neighborhood structures can be exploited in both deterministic and stochastic methods. Basic VNS combines these two methods. If a better solution is found, the new incumbent is obtained and the search is re-centered around it. The VNS used in this article only adopts deterministic methods.

GVNS is an extended version of basic VNS. It uses multiple neighborhoods in a local search. This local search is called Variable Neighborhood Descent (VND). The steps of GVNS implements in this paper are given in Fig. 1.

The MDMTSP consists of two subproblems: one is to assign cities to salesmen, and the other is to solve the travel path of each salesman. The proposed GVNS approach is used to solve the first problem, and the second problem is solved by the famous Lin-Kernighan Heuristic (LKH).

Let H_h for $h = 1, 2, \ldots, h_{max}$ denote a finite set of preselected neighborhood structures, and $H_h(x)$ denote the set of solutions in the h^{th} neighborhood of x. The total travel cost calculated from the solution x is denoted by $f(x)$. A detailed description of the steps of GVNS is as follows.

Initial Assignment. Let x denote a complete solution to the problem. x contains the assignment relationship between the cities and the salesmen and the tour of each salesman. The set of cities assigned to the salesman originating from depot i is represented by G_i. At the initial assignment stage, G_i is determined by Eq. (18):

$$G_i = \{ j \quad | \quad \underset{k \in D}{argmin} \quad c_{kj} = i \} \quad \forall i \in D. \quad . \quad . \quad (18)$$

The cities were assigned to the salesman nearest to themselves. The tour for each salesman is calculated by Lin-Kernighan Heuristic (LKH) [10].

Variable Neighborhood Descent. A sequential VND is used in this paper. In sequential VND, the neighborhood structures are explored one by one in the given sequence. The solution update strategy can be first improvement (a move made when an improvement in the neighborhood is found for the first time) or best improvement strategy (a move to the best solution in the neighborhood) [7]. The latter is also known as the steepest descent. The first improvement approach is used in this paper. As shown in the pseudocode, if a solution improvement occurs, the current local search method will be restarted. Since no random search is used in this VND, the result of each run is certain.

Three local search methods are designed in this approach, including k-reverse, k-same-as-previous, and k-same-as-next. In the k-reverse method, the salesmen assigned to the k consecutive cities in C will be reversed. In the k-same-as-previous method, the salesmen assigned to k consecutive cities in C will be the same as the salesmen assigned to their previous city (if they have any). In the k-same-as-next method, the salesmen assigned to k consecutive cities in C will be the same as the salesmen assigned to their next city (if they have any). In this paper, 2-reverse, 3-reverse, 1-same-as-previous, 2-same-as-previous, 1-same-as-next, and 2-same-as-next are used.

Neighborhood change. The neighborhood change procedure compares the value $f(x')$ with the incumbent value $f(x)$. The solution x' is obtained by local search methods. If an improvement is obtained, the incumbent is updated and the current local search method is restarted by setting i to 1. If an improvement cannot be found in the current neighborhood, the neighborhood will be replaced.

3.2. Market-Based Approach

Inspired by [18], a market-based approach is implemented as a as a competitor for the variable neighborhood search approach. The salesmen are treated as resources or services. The cities individually choose the best salesman for them. The pseudo code of the algorithm is shown in Fig. 2.

The allocation of the cities is in line 3 of the pseudocode. This results in a set of nodes assigned to each salesman, and a TSP is posed for each salesman to find its tour. Same as GVNS approach, the LKH method [10] is used to find the tour for each salesman. The calculation of TSP tours is shown in line 4 of the pseudocode. In each iteration, the price of each salesman is updated according to the resultant tours, and new assignments are again calculated. Lines 5-10 in the pseudocode show the update process of the price. T_i in the pseudo code is the cost of the tour of the salesman originating from the depot i. $\bar{T}$ is the average value of $T_i, \forall i \in D$.

The allocation of the cities is shown in the Eq. (19):

$$G_i = \{ j \quad | \quad \underset{k \in D}{argmin} \quad (c_{kj} + p_k) = i \} \quad \forall i \in D, (19)$$

Algorithm 2: Market Procedure

$p \leftarrow \mathbf{0}_{1\times N}$
for Q Iterations **do**
    Build Assignment Sets G_i
    $T_i \leftarrow Tour(G_i \cup \{i\}), i \in D$
    for Each Salesman **do**
        if $T_i > \bar{T}$ **then**
            $p_i = p_i + \alpha$
        else
            $p_i = p_i - \alpha$
    end
end

Fig. 2. Market Procedure

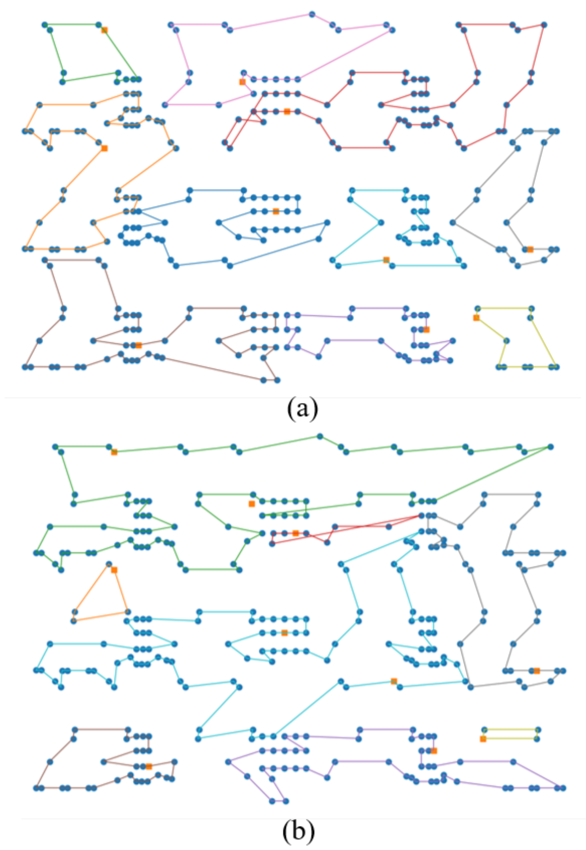

Fig. 3. A solution of instance lin318d10. (a) A solution obtained by the GVNS. (b) A solution obtained by the market-based approach.

where c_{ij} is the cost to travel from node i to j. Price p_i is only associated with a salesman, and it does not depend on the cities assigned to the salesman. The update of p_i in the iterative process helps to balance the load of different salesmen. α is the updated amount of p_i in each iteration.

4. Computational Results and Analysis

In this section, the computational results of the presented GVNS methods and their comparison with the market-based approach are given. TSPLIB [17] is a library of sample instances for the TSP and related problems from various sources and of various types. The instances in this paper is obtained by marking some nodes as depots on some instances in TSPLIB.

All tests were carried out on the Intel (R) Xeon (R) CPU E5-2678 @ 2.50GHz with 8 GB RAM running the Linux operating system. The algorithms were coded in Python programming language. A solution of an instance with 318 nodes obtained by the GVNS approach is shown in Fig. 3. In the figure, the squares represent the depots, and the dots represent the city.

The results of GVNS and market-based approach for 10 instances are presented in Table 2. There are two integers in the name of the instances. The first integer represents the total number of nodes in the instance, and the second integer represents the number of depots in the instance.

From Table 2, it can be noticed that the GVNS performs better than the market-based approach in 8 out of 10 instances. GVNS consumes less time on all 10 instances than the market-based approach.

There are three local search methods used in the GVNS approach, and the effect of each method is compared through computational experiments. The computational results are shown in Table 3. The data in the column of the name of the local search method is the test result of GVNS without that method. The data in the gap column is the percentage of the difference between the result of the GVNS and that of the GVNS without each local search method in the result of the GNVS. The larger the gap, the greater the effect of the local search method on improving the solution. From Table 3, the k-same-as-next local search method has the greatest effect on the improvement of algorithm results. The k-reverse and k-same-as-previous methods contribute to improving the solutions in some instances. Although the removal of the k-reverse method makes GVNS get better solutions on the two instances pr124d8 and bier127d5, it plays an important role in solving std70d5. It can be seen that after removing any local search method, the performance of GVNS will decline in some instances.

The time consumed by GVNS without each local search method is shown in Table 4. The data in the gap column is the percentage of the difference between the time consumed of the GVNS and that of the GVNS without each local search method in the time consumed of GNVS. From the data in Table 3 and Table 4, it can be seen that the k-same-as-next method has the most significant effect on the solution improvement, and it also consumes the most time. The k-same-as-previous works for the improvement of the solution in most instances. The k-reverse method only improves the solution in one instance, but it consumes a lot of time. In most instances, the best solution is obtained when all three local search methods are used.

Table 2. The computational results

Instances	Parameters		GVNS		market-based approach	
	N	*M*	Travel cost	Time (s)	Travel cost	Time (s)
berlin52d4	4	48	**7732.662**	**17.485**	7899.659	37.901
st70d5	5	65	760.797	**35.372**	**727.636**	57.093
pr76d10	10	66	**127807.868**	**32.123**	130659.402	45.746
pr124d8	8	116	**58341.677**	**103.421**	59491.333	331.929
pr136d8	8	128	**99625.715**	**314.615**	100509.815	555.077
lin105d6	6	99	**15653.697**	**97.366**	17512.991	172.616
bier127d5	5	122	**131806.971**	**378.331**	135151.128	799.630
ch150d6	6	144	**7063.975**	**287.491**	7280.026	682.880
tsp225d9	9	216	**4109.338**	**993.728**	4358.130	2416.077
lin318d10	10	308	46540.380	**2622.519**	**46520.958**	5397.010

Table 3. The computational results of GVNS without each local search method

Instances	GVNS	(-) *k*-reverse		(-) *k*-same-as-previous		(-) *k*-same-as-next	
		Travel cost	Gap (%)	Travel cost	Gap (%)	Travel cost	Gap (%)
berlin52d4	**7732.662**	7732.662	0.000	7732.662	0.000	8228.437	*6.411*
st70d5	**760.797**	773.548	*1.676*	761.778	0.129	763.463	0.350
pr76d10	**127807.868**	127807.868	0.000	128310.246	0.393	129994.772	*1.711*
pr124d8	58341.677	**57430.767**	-1.561	58368.084	0.045	58932.708	*1.013*
pr136d8	**99625.715**	99625.715	0.000	100223.359	0.600	100961.146	*1.340*
lin105d6	**15653.697**	15653.697	0.000	16044.991	2.500	16775.975	*7.169*
bier127d5	131806.971	**130524.632**	-0.973	135821.160	*3.046*	134312.663	1.901
ch150d6	**7063.975**	7063.975	0.000	7063.975	0.000	7263.851	*2.830*
tsp225d9	**4109.338**	4109.338	0.000	4167.103	*1.406*	4164.482	1.342
lin318d10	46540.380	46540.380	0.000	**46497.956**	-0.091	47219.625	*1.459*

Table 4. The time consumed by GVNS without each local search method

Instances	GVNS	(-) *k*-reverse		(-) *k*-same-as-previous		(-) *k*-same-as-next	
		Time (s)	Gap (%)	Time (s)	Gap (%)	Time (s)	Gap (%)
berlin52d4	17.485	12.128	-30.637	13.712	-21.578	*9.690*	*-44.585*
st70d5	35.372	24.252	-31.437	23.620	-33.224	*23.421*	*-33.788*
pr76d10	32.123	22.917	-28.658	23.734	-26.117	*20.053*	*-37.576*
pr124d8	103.421	73.998	-28.450	72.189	-30.199	*71.324*	*-31.036*
pr136d8	314.615	*237.907*	*-24.382*	248.977	-20.863	247.719	-21.263
lin105d6	97.366	74.572	-23.411	78.699	-19.173	*47.565*	*-51.149*
bier127d5	378.331	*228.901*	*-39.497*	283.649	-25.026	305.727	-19.191
ch150d6	287.491	*173.705*	*-39.579*	211.901	-26.293	181.983	-36.700
tsp225d9	993.728	741.175	-25.415	683.236	-31.245	*621.834*	*-37.424*
lin318d10	2622.519	2187.162	-16.601	1986.418	-24.255	*1381.278*	*-47.330*

5. Conclusions

The MDMTSP is studied in this paper. A GVNS approach is proposed in this paper and three local search methods are designed in the GVNS, including *k*-reverse, *k*-same-as-previous, and *k*-same-as-next methods. A state-of-the-art market-based approach is implemented as a as a competitor for the GVNS approach. The use of three local search methods changes the neighborhood structure of the solution and expands the search range. The computational results show that the GVNS performs better than the market-based approach in 8 out of 10 instances. In most instances, the best solution is obtained when all three local search methods are used.

Acknowledgements

This work was supported by the National Outstanding Youth Talents Support Program under Grant 61822304.

References:

[1] J. Brimberg, P. Hansen, and N. Mladenovic, “Attraction probabilities in variable neighborhood search”, 4OR, 8(2):181–194, 2010.

[2] M. Burger, Z. Su, and B. De Schutter, “A node current-based 2-index formulation for the fixed-destination multi-depot travelling salesman problem”, European Journal of Operational Research, 265(2):463–477, 2018.

[3] T. Carnes and D. B. Shmoys, “Primal-dual schema and Lagrangian relaxation for the k-location-routing problem”, In Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, pp. 99–110, Springer, 2011.

[4] P. R. Chandler, M. Pachter, D. Swaroop, J. M. Fowler, J. K. Howlett, S. Rasmussen, C. Schumacher, and K. Nygard, “Complexity in UAV cooperative control”, In Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301), volume 3, pp. 1831–1836. IEEE, 2002.

[5] N. Christofides, “Worst-case analysis of a new heuristic for the travelling salesman problem”, Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[6] G. Dantzig, R. Fulkerson, and S. Johnson, “Solution of a large-scale traveling-salesman problem”, Journal of the operations research society of America, 2(4):393–410, 1954.

[7] P. Hansen and N. Mladenović, “First vs. best improvement: An empirical study”, Discrete Applied Mathematics, 154(5):802–817, 2006.

[8] P. Hansen, N. Mladenović, and J. A. M. Pérez, “Variable neighbourhood search: methods and applications”, Annals of Operations Research, 175(1):367–407, 2010.

[9] J. M. Harris, ”Combinatorics and graph theory”, Springer, 2008.

[10] K. Helsgaun, “An extension of the Lin-Kernighan-Helsgaun TSP solver for constrained traveling salesman and vehicle routing problems”, Roskilde: Roskilde University, 2017.

[11] R. M. Karp, “Reducibility among combinatorial problems”, In Complexity of computer computations, pp. 85–103, Springer, 1972.

[12] E. Kivelevitch, B. Sharma, N. Ernest, M. Kumar, and K. Cohen, “A hierarchical market solution to the min–max multiple depots vehicle routing problem”, Unmanned Systems, 2(01):87–100, 2014.

[13] E. H. Kivelevitch, K. Cohen, and M. Kumar, “A market-based solution to the multiple depots, multiple traveling salesmen problem with profits”, In AIAA Infotech@ Aerospace (I@ A) Conference, pp. 4666, 2013.

[14] W. Malik, S. Rathinam, and S. Darbha, “An approximation algorithm for a symmetric generalized multiple depot, multiple travelling salesman problem”, Operations Research Letters, 35(6):747–753, 2007.

[15] N. Mladenović and P. Hansen, “Variable neighborhood search”, Computers & operations research, 24(11):1097–1100, 1997.

[16] C. Prodhon and C. Prins, “A survey of recent research on location-routing problems”, European Journal of Operational Research, 238(1):1–17, 2014.

[17] G. Reinelt, “TSPLIB—A traveling salesman problem library”, ORSA journal on computing, 3(4):376–384, 1991.

[18] D. Scott, S. G. Manyam, D. W. Casbeer, and M. Kumar, “Market approach to length constrained min-max multiple depot multiple traveling salesman problem”, In 2020 American Control Conference (ACC), pp. 138–143. IEEE, 2020.

[19] Z. Xu and B. Rodrigues, “An extension of the Christofides heuristic for the generalized multiple depot multiple traveling salesmen problem”, European Journal of Operational Research, 257(3):735–745, 2017.