كلية الهندسة بشبرا
FACULTY OF ENGINEERING AT SHOUBRA
ERJ
PRINT ISSN 3009-6049
ONLINE ISSN 3009-6022

ENGINEERING RESEARCH JOURNAL (ERJ)
Volume (53),Issue (4)
October 2024, pp:119-124
https://erjsh.journals.ekb.eg

Applying Minimum Travel Array on Twenty Traveling Salesman Problems

Mohamed Eleiche

Construction Engineering department, Faculty of Engineering, Egyptian Russain University

E-mail address : *mohamed.eleiche@gmail.com*

Abstract: The exact solution for the Traveling Salesman Problem (TSP) is a main research topic with high priority and importance. This article gives insight analysis to understand this important problem by studying its Minimum Travel Array characteristic. There are symmetric and asymmetric TSP. This article selected ten problems from each type and computed the Minimum Travel Array for each. Then, the sum of the minimum cost of arriving at the node and the sum of the minimum cost of departing from it were computed and compared to the Best-Known solution of each problem. There is a distinct difference between symmetric and asymmetric TSP. This is clear in the results of this research. The mean of both sums is a reliable lower bound for symmetric graphs, and the greater from these sums is a practical lower bound for asymmetric graphs. In both cases, the Minimum Travel Array provides a deep understanding of each TSP and is a first step towards its solution. The TSPLIB is the source for the twenty problems analyzed in this article. The program used and the associated data are freely available online.

Keywords: traveling salesman problem (TSP); graph theory; network analysis; Minimum travel array (Ꙗ).

AMS Subject Classification, 2010
52B05: Combinatorial properties
68R10: Graph theory
90B10: Network models, deterministic
90C35: Programming involving graphs or networks
90C27: Combinatorial optimization

1. Introduction

The traveling salesman problem (TSP) is an optimization famous problem in the theory of graphs and has diversity of applications in engineering and other disciplines. It is mathematically defined by a number of (n) cities with the length of travel between each pair of them. The tour with the least cost to visit each city only once from the starting point and returning back to it again (Applegate, et al., 2006). The travel cost is symmetric if traveling from city a to city b is the same as the cost from b to a. If they are different then it is asymmetric. The TSP is a mathematical full graph with (n) nodes. TSP is a prototype of hard combinatorial optimization problem where the possible solutions are *(n-1)!* and is considered NP-hard and NP-complete (Jungnickel , 2008). There is not an exact algorithm to compute the optimal tour deterministically. There are modern applications for TSP such as the Drone-Assisted Variable Speed Asymmetric TSP which considers variable flight times for the drone (Campuzano, et al., 2023) and the spherical asymmetric multiple TSP where all cities and paths exist on the three-dimension sphere (Huang, et al., 2023).

This research aims to analyze the Minimum Travel Array (Ꙗ) property of the TSP in both symmetric and asymmetric cases, apply it on selected samples from the TSPLIB, and discuss the results.

The traveling salesman problem library (TSPLIB) is a cornerstone in the history of this important and iconic problem. Gerhard Reinelt created it in 1990 (Reinelt, 1991). It is a rich resource for the TSP such as data for symmetric and asymmetric problems with size from 14 nodes up to 85000 nodes, with their best-known solutions. Also, an outstanding reference for many researches exist in the TSPLIB. It is available on open access basis worldwide.

2. Mathematical Background

The TSP is modeled as a weighted complete directed graph $G = (V, E)$, where (V) are the nodes with size (n), and (E) are the edges with size (n^2). $V = \{1, 2, \ldots, n\}$, and $E = \{(u,v) \mid u, v \in V\}\ and\ \{cost(u,v)\ \in\ \mathbb{R} \geq 0\}$ (Bondy & Murty, 1976). This research considers only edges with cost of positive values or zero.

2.1 Input Data

The accepted format for the program TSP_02 which computes Ꙗ is given in Table 1. The array of the cost has the format [u, v, $cost(u,v)$], where (u) is the from-node of the edge, (v) is the to-node of the edge, and ($cost(u,v)$) is the cost of the edge. The data type of (u) and (v) is integer, while ($cost$) is a positive real number and may equals to zero. The cost array has a size of ($3\ n^2$) and its structure is shown in **Table** . It has a single row for each edge (u,v).

Table 1 The input cost array for the program

u	*v*	*Cost*
1	*1*	∞
…	…	…
1	*n*	…
2	*1*	…
n	*n*	∞

2.2 Minimum Travel Cost for Each Node

Each node has a unique characteristic, its own minimal travel cost. This means that each node *i* has very close node *u* to it, as shown in Figure 1, such that the cost of the edge (*u*,*i*) is the minimal cost to arrive to node *i*. Similarly, the node *i* has also a very close node *v* such that the cost of the edge (*i,v)* is the minimal cost to depart from node *i*. The path (*u,i,v)* is the minimal cost to travel through node *i*, there is no less cost than that. In some cases, there is one or more path with the same cost. In case of symmetric graph, by definition, the cost of the edge (*i,j*) is the equal to the edge (*j,i*) but in opposite direction, the path of (*u,i,v)* has the same cost of (*v,i,u)*. The cost of the edge (*u,i*) is the minimum cost for edges ending by *i*, and the cost (*i,v)* is its second minimum. These two values can be achieved by one time sorting edges ending (or starting) by node *i*.

This is not the case of asymmetric graph, where by definition the cost of the edge (*i,j*) is not equal to its opposite edge (*j,i*). The cost (*u,i*) is the minimum cost for edges ending by *i*, the cost (*i,v)* is the minimum cost for edges starting by *i*. *Here, the solution needs* two sorting operations. The first for edges ending by node *i*, and the second for edges starting with it. The complexity to find the minimum travel cost for symmetric TSP is O(n^2), while for asymmetric TSP is O($2n^2$).

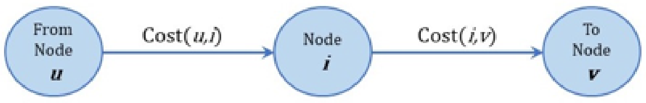

Figure 1 Minimum travel cost for node *i*

2.3 Minimum Travel Array (ɰ) for the Graph

The ɰ is the main output of this algorithm. Its structure is [*InCost, u, i, v, OutCost*], where (*InCost*) is the cost of the edge (*u*,*i*), (*u*) is the incident node to node (*i*), (*i*) is the node of interest ($1 \leqslant i \leqslant n$), (*v*) is the outgoing node from node i, and (*OutCost*) is the cost of the edge (*i*,*v*). The size of ɰ is (*5 n*) (Eleiche, Markus 2010), and Table 2**Error! Reference source not found.** shows its structure.

2.4 TSP_02 Program

It is C++ program implemented to produce the minimum travel array (ɰ) for any graph (Eleiche, 2020). The input data to the program must be in the format in Table 1.

Table 2 Minimum Travel Array (ɰ)

InCost	*u*	*i*	*v*	*OutCost*
…	…	*1*	…	…
InCost	*u*	*i*	*v*	*OutCost*
…	…	*n*	…	…
Sum of InCost				*Sum of OutCost*

3. Data and Methodology

A list of twenty graphs is selected from the TSPLIB to compute the Min Travel Array for each of them. Ten graphs are symmetric with node size from 42 up to 1032 nodes. Another ten graphs are asymmetric ranging from 17 to 443 nodes. In **Error! Reference source not found.**, the name and size of each graph is presented, aside with its best-known solution.

Table 3 Input data for graphs (Source: TSPLIB)

Type	ID	Name	Size	Best Solution
Symmetric	1	US42	42	699
	2	hk48	48	11461
	3	brazil58	58	25395
	4	gr120	120	6942
	5	si175	175	21407
	6	brg180	180	1950
	7	si535	535	48450
	8	pa561	561	2763
	9	college647	647	47149705
	10	si1032	1032	92650
Asymmetric	1	br17	17	39
	2	ftv35	36	1473
	3	p43	43	5620
	4	ry48	48	14422
	5	ft53	53	6905
	6	ftv70	71	1950
	7	kro124	100	36230
	8	ftv170	171	2755
	9	rbg323	323	1326
	10	rbg443	443	2720

The TSPLIB stores the cost data for each graph in different format. These data are downloaded then presented in the format described in **Table** , which is the input format for the program TSP_02. The output file includes detailed data about the computation, however Ꜿ is only extracted in a separate file. This process starts from downloading the data, convert its format, apply the program and extract Ꜿ for each graph is represented in Figure 2 .

The data repository includes the program TSP_02 source code and its executive file, the original data from TSPLIB, the input file, the output file and Ꜿ for each graph.

4. Results

The main result is the minimum travel array (Ꜿ) for each problem as presented in Table 4. This table includes, in addition to the columns of **Error! Reference source not found.**, the Min Incident column which is the sum of the minimal cost edges to arrive to each node in the graph, while the Min Outgoing column is the sum of the minimal cost edge to depart from the node. Finally, the last column named Mean Travel Cost is the arithmetic mean for the Min Incident and Min Outgoing as defined in Equation 1.

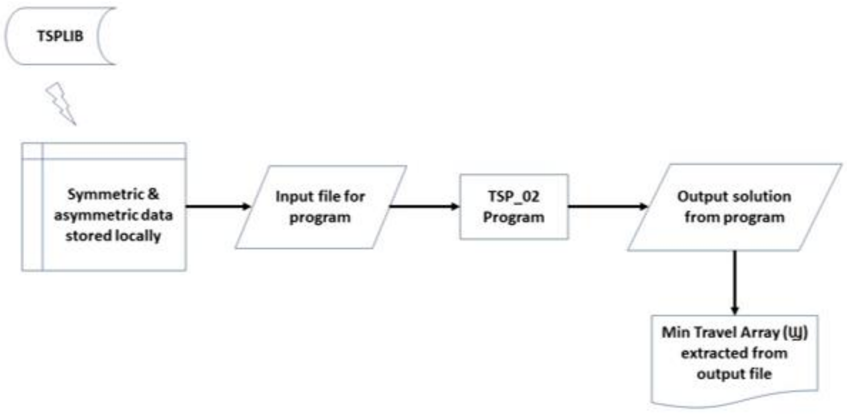
TSPLIB
Symmetric & asymmetric data stored locally
Input file for program
TSP_02 Program
Output solution from program
Min Travel Array (Ꜿ) extracted from output file

Figure 2 The flow of computing Minimum Travel Array (Ꜿ)

Table 4 Results for the graphs

Type	ID	Name	Size	Best Solution	Sum of Min Incident	Sum of Min Outgoing	Mean Travel Cost
Symmetric	1	US42	42	699	454	732	593
	2	hk48	48	11461	8757	11738	10247.5
	3	brazil58	58	25395	14627	21455	18041
	4	gr120	120	6942	1218	2855	2036.5
	5	si175	175	21407	19720	21804	20762
	6	brg180	180	1950	0	3600	1800
	7	si535	535	48450	44066	48197	46131.5
	8	pa561	561	2763	2101	2879	2490
	9	college647	647	47149705	25615458	42777203	34196331
	10	si1032	1032	92650	90158	92796	91477
Asymmetric	1	br17	17	39	0	24	12
	2	ftv35	36	1473	1068	1213	1140.5
	3	p43	43	5620	159	214	186.5
	4	ry48	48	14422	12987	11964	12475.5
	5	ft53	53	6905	3580	3989	3784.5
	6	ftv70	71	1950	1415	1509	1462
	7	kro124	100	36230	31425	30716	31070.5
	8	ftv170	171	2755	2141	2361	2251
	9	rbg323	323	1326	517	219	368
	10	rbg443	443	2720	37	352	194.5

$$\text{Mean Travel Cost} = \frac{\text{Min Incident+ Min Outgoing}}{2} \quad (1)$$

Where Mean Travel Cost is the computed value for each graph, Min Incident is the sum of Min Incident for each graph computed from the minimum travel array (ꟺ) and Min Outgoing is the sum of Min Outgoing for each graph computed from the minimum travel array (ꟺ).

The results included in Table 4 are statistically presented in Figure 3 for symmetric graphs and in Figure 4 for asymmetric graphs. The horizontal axis represents the graph ID. The vertical axis is the normalized percentage of the sum of Min Incident, the sum of Min Outgoing, and the Mean columns are the relative percentage to the Best-Known solution. As example, for the first symmetric graph: percentage of Min Incident = (454/699 = 64%), percentage of Min Outgoing = (732/699 = 104%), and finally percentage of Mean = (593/699 = 84%). The light vertical bar at the left is for the sum of incident cost, the dark vertical bar at the right represents the sum Min Outgoing Cost, and the red dot represents the mean value define in Equation 1. The horizontal red line represents the Best-known solution for the graph as 100%, it is the metric reference to measure the deviation of both costs from the Best-Known solution.

The results of symmetric graphs displayed in Figure 3 showed that the sum of outgoing cost is greater than the sum of incident cost for all symmetric graphs. The graph with ID equals 6 had outlier behavior with zero incident cost and very high outgoing cost. The sum of the incident cost for all the tested graphs is lower than the best-known solution. The sum of the outgoing cost for 50% the tested graphs is higher than the best-known solution, for the graph with ID equals 10 has equal value, and for the remaining 40% has lower value. The mean travel cost for 40% the tested graphs is between 95% and 98% from the best-known solution, while the best-known solution is higher than the mean travel cost for all the tested graphs.

The results of the asymmetric graphs displayed in Figure 4 showed different behavior compared to symmetric one. Sometimes the sum of incident cost is greater than the sum of outgoing cost, and other times vice versa. The best-known solution is greater than all the other three quantities, the closest value to it is the higher from the incident and the outgoing sum.

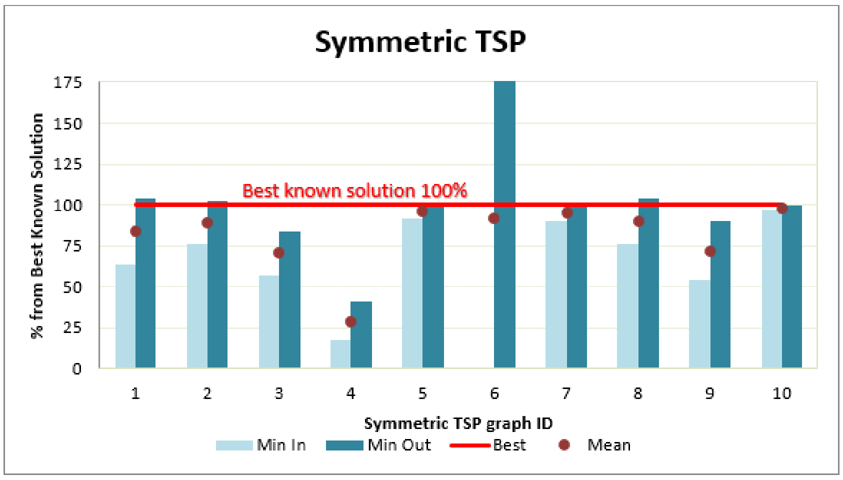

Figure 3 Symmetric graphs results

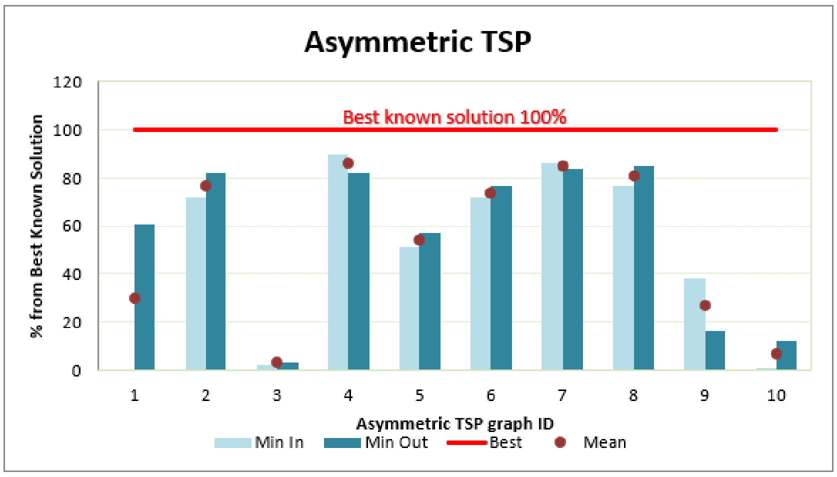

Figure 4 Asymmetric graphs results

5. Discussion

There are several algorithms to compute a lower bound value for the TSP, such as the Held-Karp lower bound which modifies the original graph by assigning weight to vertices and change cost between them (Held & Karp, 1970) and the deleted vertex algorithm which is based on removing a vertex and compute the minimum spanning tree (Rose & Tarjan, 1978). In Table 5, a comparison between these two algorithms with the minimum travel array algorithm. It is clear that the Held-Karp lower bound is most accurate with highest complexity and expensive computation compared to the other two, while the minimum travel array keeps the original graph without modifications compared to the deleted vertex algorithm.

Table 5 comparison with similar algorithms

Algorithm	Held-Karp lower bound	deleted vertex algorithm	Minimum Travel Array
Complexity	$O(2n^3)$	$O(n^2)$	$O(n^2)$ $O(2n^2)$
Computation	expensive	efficient	efficient
Accuracy	high	medium	medium
Technique	modify graph	modify graph	No modifications

6. Conclusion

The results showed clearly the distinct difference between symmetric and asymmetric graphs. The sum of incident cost is always lower than sum of outgoing cost in the symmetric graphs, and this is not the same case in asymmetric graphs, where it may be greater or lower.

The best-known solution is greater than the sum of incident and/or outgoing cost for all asymmetric graphs, and it may be lower (or greater) than the outgoing cost for the symmetric graphs.

The Mean value is a reliable lower bound for symmetric graphs, and the greater from the sum of the incident and outgoing cost is a practical lower bound for asymmetric graphs.

The exact solution for the Traveling Salesman Problem (TSP) is still a main scientific and engineering challenge. It is vital to understand the problem well in order to achieve its solution.

The Ш is an intrinsic characteristic for the TSP and furnish a good start to understand its type. Although in this study 20 problems were tackled, still more investigations and research are required.

Data and Program

The data in this article is free to access on GitHub https://github.com/meleiche/TSP_Data. The data was collected from TSPLIB and consists of ten symmetric graphs and another ten asymmetric graphs with varying node size. The data is available without license, which allows its open use freely without any constraint. The data is well-documented and includes a README file that provides instructions on how to download and use the data. Each graph has its own folder, and it includes the original data from TSPLIB, the input format for the TSP_02 program and its output solution, and finally the Ш which is the main target.

The TSP_02 is a C++ program developed to compute Ш for any graph. The input file is the cost file in the format included with the program. The output file is a detailed file in its end exist the required array. The URL of the program is https://github.com/meleiche/TSP_Min_Travel_Array, and it is freely available resource on GitHub.

References

[1] Applegate, D., Bixby, R., Chvátal , V. & Cook, W., 2006. The Traveling Salesman Problem: A Computational Study. s.l.:Princeton University Press.

[2] Bondy, J. & Murty, U., 1976. Graph Theory with Applications. London: Macmillan.

[3] Campuzano, G., Lalla-Ruiz, E. & Mes, M., 2023. The drone-assisted variable speed asymmetric traveling salesman problem. Computers & Industrial Engineering, Volume 176.

[4] Eleiche, M., 2020. Exact Minimum Lower Bound Algorithm for Traveling Salesman Problem. International Journal of Industry and Sustainable Development, 1 July, 1(2), pp. 76-84.

[5] Eleiche, M. & Markus, B., 2010. Applying Minimum Travel Cost Approach On 17–Nodes Travelling Salesman Problem. Geomatikai Közlemények, XIII(2), pp. 15-22.

[6] Held, M. & Karp, R. M., 1970. The Traveling-Salesman Problem and Minimum Spanning Trees. Operations Research, 7(4), pp. 1138-1162.

[7] Huang, H., Wei, Y., Zhou, Y. & Luo, Q., 2023. Spherical vector-based artificial gorilla troops optimization for spherical asymmetric multiple traveling salesman problem. Evolving Systems.

[8] Jungnickel , D., 2008. Graphs, Networks, and Algorithms. Springer, pp. 433-472.

[9] Reinelt, G., 1991. TSPLIB—A Traveling Salesman Problem Library. ORSA Journal on Computing, 3(4).

[10] Rose, D. J. & Tarjan, R. E., 1978. Algorithmic Aspects of Vertex Elimination on Directed Graphs. SIAM Journal on Applied Mathematics, 34(1), pp. 176-197.

Appendix (A)

Min Travel Array for the problem US_42

Incident cost	Incident node	Node	Outgoing node	Outgoing cost
3	41	1	42	5
8	1	2	41	11
9	4	3	8	15
9	3	4	5	15
15	4	5	6	17
6	7	6	5	17
6	6	7	8	10
5	9	8	7	10
5	8	9	7	15
14	25	10	9	20
11	12	11	10	23
11	11	12	10	26
29	17	13	16	34
10	15	14	16	31
10	14	15	16	27
21	17	16	18	26
21	16	17	23	27
26	16	18	19	26
22	20	19	18	26
22	19	20	21	30
21	22	21	23	27
5	23	22	21	21
5	22	23	17	27
8	25	24	27	9
8	24	25	26	11
3	27	26	25	11
3	26	27	24	9
12	29	28	27	20
12	28	29	30	20
8	31	30	32	15
8	30	31	32	12
11	33	32	31	12
11	32	33	34	21
9	35	34	31	14
9	34	35	37	13
17	37	36	35	18
12	38	37	35	13
9	39	38	37	12
6	40	39	38	9
6	39	40	38	15
3	1	41	42	6
5	1	42	41	6

$\sum_1^{42} \text{Outgoing cost} = 732 \sum_1^{42} \text{Incident cost} = 454$