ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23
American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

OPTIMIZATION APPROACHES AND MATHEMATICAL MODELING IN THE TRAVELING SALESMAN PROBLEM IN TOURISM COMPANIES

Mamatova Zilolakhon Khabibulloqonovna
Associate Professor at Fergana State University,
Doctor of Philosophy in Pedagogical Sciences (PhD)
E-mail: mamatova.zilolakhon@gmail.com
ORCID ID 0009-0009-9247-3510
Bahriddinova Nozanin Janobidin-zoda
Student at Fergana State University
E-mail:nozanin122003@gmail.com

Annotation: The Traveling Salesman Problem (TSP) is one of the most well-known and studied problems in mathematical optimization and computer science. The objective of the problem is to find the shortest possible route that allows a traveling salesman to visit a set of specified cities, visiting each city only once, and ultimately returning to the starting point. TSP is famous for its fully combinatorial nature and high complexity. It is classified as an NP-complete problem, meaning that finding an exact solution becomes increasingly difficult as the number of cities grows. Various algorithms have been developed to find solutions using computers, including network search algorithms, genetic algorithms, and simulated annealing methods. The Traveling Salesman Problem is applied in many practical fields, such as logistics, transportation, robotics, and various resource management systems. It is also of great theoretical importance as a tool used for optimization and improving efficiency.
Keywords: Traveling Salesman Problem (TSP), optimization, combinatorics, NP-complete problem, shortest path, algorithms, logistics, transportation, robotics, resource management, simulated annealing, genetic algorithms, network search algorithms, solution finding.

Introduction

The Traveling Salesman Problem is one of the important problems in combinatorial optimization, and its goal is to find the shortest route that visits several cities and returns to the starting point with minimal distance. This problem is applied in fields such as transport logistics, optimization of manufacturing processes, and electronic circuit design.
Solution Methods: Bruteforce (Complete Enumeration), Dynamic Programming (Held-Karp Algorithm), Greedy Algorithm, Genetic Algorithm, Simulated Annealing, Linear Programming, and Branch and Bound.

Objective: The objective of the Traveling Salesman Problem (TSP) is to find the shortest and most efficient route for the salesman to visit the given cities. In this problem, each city must be visited only once, and at the end, the salesman must return to the starting city. The main goal of the problem is to minimize the distance traveled between the cities, thereby reducing time and costs. By optimizing the objective of the TSP, the efficiency in real-world applications such as transportation, logistics, and resource management can be improved. Additionally, the algorithms and methods developed to solve the TSP can be applied to many other optimization problems.

ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23

American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

The problem is to find the shortest cycle (in terms of time, distance, or cost) that visits each of the given n cities exactly once. The number of cycles must not exceed $(n-1)!$. This problem is related to finding the minimum length Hamiltonian cycle. The "Branch and Bound" method can be applied to solve the Traveling Salesman Problem. This method is carried out using a graph without cycles and with outer constraints, as well as by constructing tables.

Sample Variant. Let's introduce the concept of generating a table. To do this, we first generate the rows of the table, meaning that the smallest element from each row is subtracted from the elements of that row. After that, the same operation is performed with respect to the columns, and the columns of the table are generated. A table that has had all its rows and columns generated is called the generated table. The sum of the smallest elements of all the rows and columns is denoted by h, which is called the generation coefficient of the table. As an example, let's consider the following train travel schedule across Uzbekistan:

We can input the markings.

1.Tashkent-Bukhara – 994 000

Tashkent-Karshi – 854 000

Tashkent-Samarkand – 500 000

Tashkent-Urgench – 70 000

Tashkent-Termiz – 90 000

Tashkent-Khiva – 102 000

Tashkent-Andijan – 112 000

2.Bukhara-Tashkent – 4 000

Bukhara-Karshi – 9 000

Bukhara-Samarkand – 12 000

Bukhara-Urgench – 900 000

Bukhara-Termiz – 950 000

Bukhara-Khiva – 60 000

Bukhara-Andijan – 70 000

3.Karshi-Tashkent – 860 000

Karshi-Bukhara – 10 000

Karshi -Samarkand – 67 000

ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23

American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

Karshi -Urgench – 78 000

Karshi -Termiz – 100 000

Karshi -Khiva – 222 000

Karshi -Andijan – 55 000

4.Samarkand-Tashkent – 50 000

Samarkand -Bukhara – 14 000

Samarkand -Karshi – 70 000

Samarkand-Urgench – 80 000

Samarkand -Termiz – 11 000

Samarkand -Khiva –8 000

Samarkand -Andijan – 12 000

5.Urgench -Tashkent – 75 000

Urgench -Bukhara – 890 000

Urgench -Karshi – 75 000

Urgench - Samarkand – 95 000

Urgench -Termiz – 12 000

Urgench -Khiva – 24 000

Urgench -Andijan – 26 000

6.Termiz -Tashkent – 95 000

Termiz -Bukhara – 854 000

Termiz -Karshi – 90 000

Termiz -Samarkand – 50 000

Termiz -Urgench – 10 000

Termiz -Khiva – 10 000

Termiz -Andijan – 95 000

7.Khiva -Tashkent – 100 000

Khiva -Bukhara – 50 000

ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23

American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

Khiva -Karshi – 111 000

Khiva -Samarkand – 40 000

Khiva -Urgench – 120 000

Khiva -Termiz – 50 000

Khiva -Andijan – 100 000

8.Andijan -Tashkent – 100 000

Andijan -Bukhara – 80 000

Andijan -Karshi – 50 000

Andijan -Samarkand – 22 000

Andijan -Urgench – 24 000

Andijan -Termiz – 18 000

Andijan -Khiva – 90 000

Table 1.

B/S	1	2	3	4	5	6	7	8	"The smallest in the row."
1		994	854	500	70	90	102	112	70
2	4		9	12	900	950	60	70	4
3	860	10		67	78	100	222	55	10
4	50	14	70		80	11	8	12	8
5	75	890	75	95		12	24	26	12
6	95	854	90	50	10		10	95	10
7	100	50	111	40	120	50		100	40
8	100	80	50	22	24	18	90		18

To generate the rows of Table 1, we write the smallest element of the corresponding row on its right side and subtract it from the row elements. This results in the following Table 2.

Table-2

ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23

American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

B/S	1	2	3	4	5	6	7	8
1		924	784	430	0	20	32	42
2	0		5	8	896	946	56	66
3	850	0		57	68	90	212	45
4	42	6	62		78	3	0	4
5	63	878	63	83		0	12	14
6	85	844	80	40	0		0	85
7	60	10	71	0	80	10		60
8	82	62	32	4	6	0	72	
"The smallest element in the column."	0	0	5	0	0	0	0	4

To generate the columns of the resulting Table 2, the smallest element of the corresponding column is written below the table and subtracted from the column elements. As a result, the following Table 3 is obtained.

Table-3

B/S	1	2	3	4	5	6	7	8
1		924	779	430	$0^{(20)}$	20	32	38
2	$0^{(42)}$		$0^{(27)}$	8	896	946	56	62
3	850	$0^{(47)}$		57	68	90	212	41
4	42	6	57		78	3	$0^{(0)}$	$0^{(10)}$
5	63	878	58	83		$0^{(10)}$	12	10
6	85	844	75	40	$0^{(0)}$		$0^{(0)}$	81
7	60	10	66	$0^{(14)}$	80	10		56
8	82	62	27	4	6	$0^{(4)}$	72	

ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23
American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

Figure 1

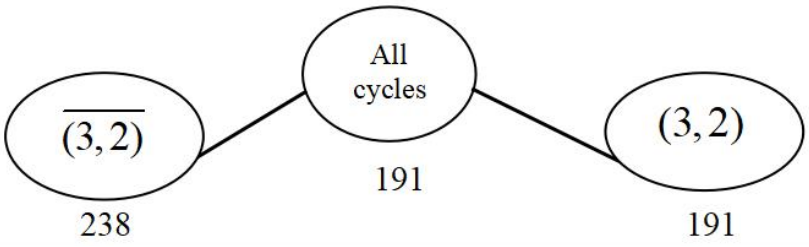


Table 3 is generated, and in each row and column of the table, there is at least one zero element. The generation coefficient h of the given table is equal to the following number:"

$$h = 70+4+10+8+12+10+40+18+0+0+5+0+0+0+0+4 = 191$$

$$h_1 = 70+4+10+8+12+10+40+18+0+0+5+0+0+0+0+4 = 191$$

$$h_1^{'} = h_1 + 47 = 238$$

In general, the Branch and Bound method consists of two important stages:
1. Branching;
2. Determining the lower bounds.
During the problem-solving process, both stages are carried out simultaneously. To perform these stages, the following tasks must be completed sequentially:

A) Generate the initial table;

B) Determine the generation coefficient h ;

C) Identify the level of zero elements in the generated table;

D) Perform branching based on these levels;

E) Determine the lower bounds of the cycles formed by the branching results;

F) Reduce the table size by one;

G) Prevent the formation of incomplete cycles;

H) Continue this process until a (2x2) table is formed;

I) Identify the cycle corresponding to the final branching result;

J) Compare all the bounds (for all cases);

K) If necessary, regenerate the table corresponding to the smallest bound and continue branching.

ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23
American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

During the application of this method, all calculations are carried out using the given table, and the results are displayed in a separately constructed graph. At the end of this process, the optimal (least-cost) cycle is identified.

The graph consists of interconnected circles, each of which defines a set of cycles with specific properties. The boundary values written next to these circles represent the lower bounds of the costs for the cycles that belong to that circle. The initial part of the graph is shown in Figure 1. In this case, the first initial circle defines a set that contains all cycles, and it indicates that for any cycle, the cost is not less than the value h .In the example discussed above, h=191, meaning that there is no cycle with a cost smaller than 191.

The row and column with the largest degree of zeros are selected and branched accordingly. If there are multiple zeros with the highest degree, one of them is chosen randomly. In this case, the circle on the right side represents the set of all cycles that include a route from city i to city j , and it is denoted by (i, j) . The circle on the left side, on the other hand, represents the set of routes that do not include a route from city (i, j) to city (i, j) , and it is denoted by (i, j) .

The zero element with the highest degree, 47, is selected, and thus the branching graph will look like Figure 1. The boundary value on the left circle will have the generation coefficient h plus the highest degree of the zero element, 47, resulting in the value 238. To determine the lower bound of the costs corresponding to the circle on the right, the 3rd row and 2nd column of Table 3 are removed (i.e., the table size is reduced by one). It is important to emphasize that the order of the cities must always be preserved, otherwise confusion may arise. After that, the formation of all incomplete cycles is prohibited, and the cycle $i \to j \to i(i \to j$ (indicating the transition from one city to another) is discarded. For this, the element is replaced and written as $c_{23} =$.

Let's continue our work by constructing the tables again

Table-4

B/S	1	3	4	5	6	7	8	
1		779	430	0	20	32	38	0
2	0		8	896	946	56	62	0
4	42	57		78	3	0	0	0
5	63	58	83		0	12	10	0
6	85	75	40	0		0	81	0
7	60	66	0	80	10		56	0
8	82	27	4	6	0	72		0
	0	27	0	0	0	0	0	

$h_2 = 191+27=218\ h_2^{'} = h_2 +50=268$

ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23
American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

Figure-2

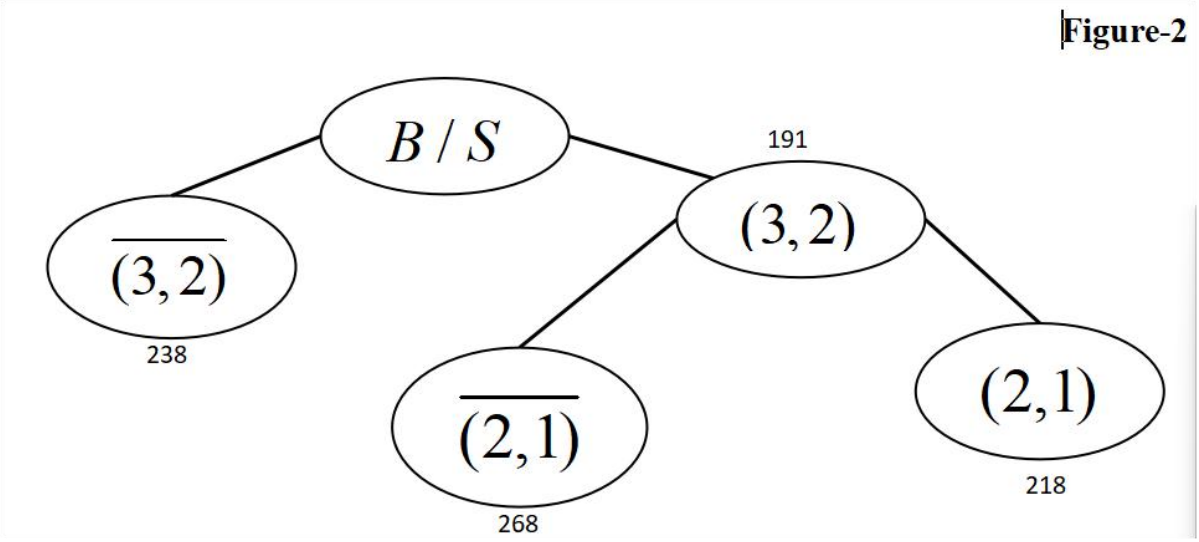


Table-5

B/S	1	3	4	5	6	7	8
1		752	430	$0^{(20)}$	20	32	38
2	$0^{(50)}$		8	896	946	56	62
4	42	30		78	3	$0^{(0)}$	$0^{(10)}$
5	63	31	83		$0^{(12)}$	12	10
6	85	48	40	$0^{(0)}$		$0^{(0)}$	81
7	60	39	$0^{(18)}$	80	10		56
8	82	$0^{(30)}$	4	6	$0^{(0)}$	72	

$c_{21} = 0^{(50)}$

$c_{12} =$

We can also compile and create the final version of Figure 2 and similar graphs at the end.

Table-6

B/S	3	4	5	6	7	8	
1		430	$0^{(20)}$	20	32	38	0
4	30		78	3	$0^{(0)}$	$0^{(10)}$	0
5	31	83		$0^{(10)}$	12	10	0
6	48	40	$0^{(0)}$		$0^{(0)}$	81	0
7	39	$0^{(14)}$	80	10		56	0
8	$0^{(30)}$	4	6	$0^{(0)}$	72		0

ORIGINAL ARTICLE

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23

American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

	0	0	0	0	0	0	

$h_3 = 218 \qquad h_3^{'} = 248$

$c_{83} = 0^{(20)}$

Table-7

B/S	4	5	6	7	8	
1	430	$0^{(20)}$	20	32		0
4		78	3	$0^{(0)}$	$0^{(10)}$	0
5	83		$0^{(13)}$	12	10	0
6	40	$0^{(0)}$		$0^{(0)}$	81	0
7	$0^{(50)}$	80	10		56	0
	0	0	0	0	0	

$h_4 = 218 \qquad h_4^{'} = 268$

$c_{74} = 0^{(50)}$

Table-8

B/S	5	6	7	8	
1	$0^{(20)}$	20	32		0
4	78	3		$0^{(13)}$	0
5		$0^{(15)}$	12	10	0
6	$0^{(0)}$		$0^{(12)}$	81	0
	0	0	0	0	

$h_5 = 218 \qquad h_5^{'} = 238$

$c_{15} = 0^{(20)}$

Table-9

B/S	6	7	8	
4	3		0	0
5	0	12		0
6		0	81	0
	0	0	0	

$h_6 = 208 \qquad h_6^{'} = 311$

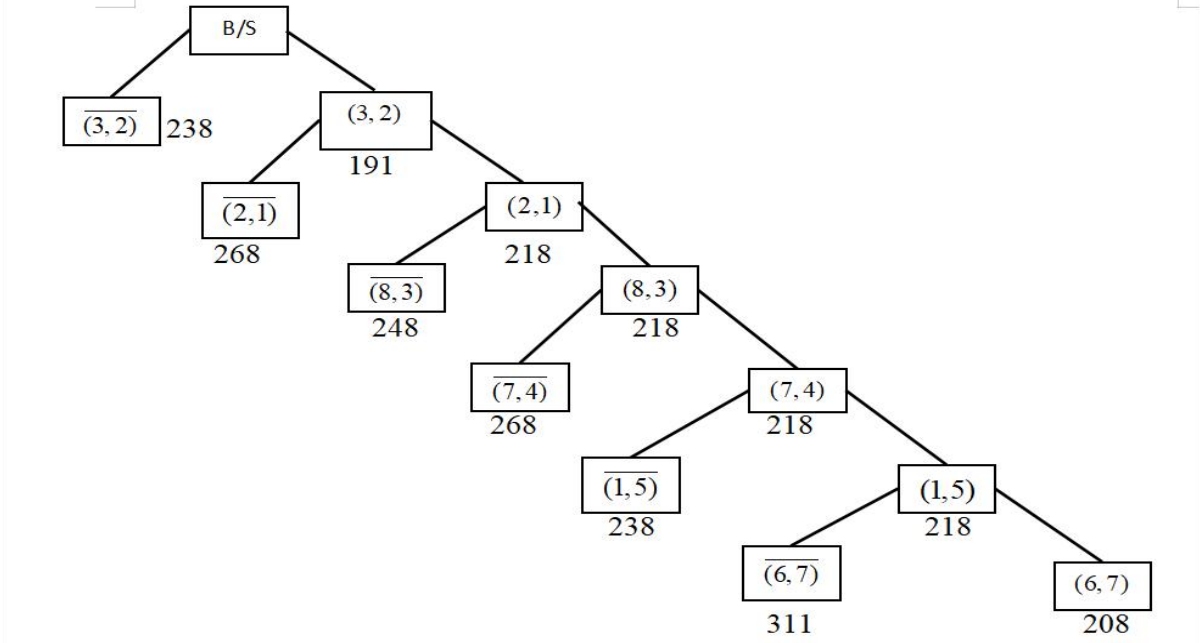
INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE

ISSN: 2692-5206, Impact Factor: 12,23
American Academic publishers, volume 05, issue 05,2025

Journal: https://www.academicpublishers.org/journals/index.php/ijai

$c_{67} = 0^{(103)}$

B/S	6	8
4		0
5	0	



The shortest route (optimal path):
$1 \to 5 \to 6 \to 7 \to 4 \to 8 \to 3 \to 2 \to 1$
Minimal distance: 208 units.

Conclusion

The Traveling Salesman Problem is a classic mathematical problem used to model and optimize many real-world issues. Due to the computational complexity of finding the optimal solution, in many cases, approximate algorithms or heuristic approaches are used.

References:

1. Papadimitriou, C. H., & Steiglitz, K. (1998). Combinatorial Optimization: Algorithms and Complexity. Dover Publications.
2. Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (2006). The Traveling Salesman Problem: A Computational Study. Princeton University Press.
3. Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., & Shmoys, D. B. (1985). The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. Wiley.
4. Gutin, G., & Punnen, A. P. (2002). The Traveling Salesman Problem and Its Variations. Springer.
5. Reinelt, G. (1994). The Traveling Salesman: Computational Solutions for TSP Applications. Springer.