Methods of Solving the Problems of the Seller in Order to Find the Optimal Solution for Economic and Legal Problems on the Internet for Export-Oriented Products

Nataliya Boyko[a], Lyubov Zalizna[a], Anna Starukh[b], Yurii Kryvenchuk[a], Yuriy Malynovskyy[a]

[a] *Lviv Polytechnic National University, Lviv79013, Ukraine*
[b] *Ivan Franko National University of Lviv, Lviv 79007, Ukraine*

Abstract

In this paper will be considered several already existing algorithms for solving the problem of a travelling salesman. The purpose of this research is to assess the quality of each of the selected algorithms, compare and analyze the results. Mathematical implementations of "branch and bound" algorithms (Little's Algorithm), the closest neighbor of the Approx-TSP algorithm, i.e. the ant colony optimization algorithm, are included in the overview. The result of this study is a graph with the most optimal route and time of the algorithm. For algorithms that require special operating conditions, the matrix will be adapted and possible error will be taken into account. For the rest - the matrix will contain the same values for more accurate analysis. The article considers the approach of setting the parameters of the algorithm for solving the problems of a travelling salesman. For this purpose, the problem of the salesman and the problem of optimization of parameters on a final grid are formed. The algorithms and their parameters for which the settings will take place are determined. Also, an approach to adjust the parameters of the algorithm is formed.

Keywords 1

Transportation task, Hamiltonian cycle, spanning tree, traveling saleslman problem, cycle, transcomputational problem, algorithm, reduction, algorithm parameters, parameter setting.

1. Introduction

It is known that the result of any algorithm depends on the settings of its parameters. They can improve the accuracy of the solution, speed up the operation of the algorithm. Selection of parameters for algorithms is time consuming and may require an expert group to determine the best parameters of the algorithm [2, 9, 16]

The travelling salesman problem is one of the most common combinatorial optimization problems, among a number of other optimization problems. Such problems, for example, include the search for optimal tourist routes, the multiple travelling officer's problem and the optimization of the route for welding circuit boards. Applied combinatorial optimization algorithms are used to solve this problem. Almost all of them have some number of parameters [1, 5-8].

That is why the study of setting the parameters of algorithms to solve the problem of the travelling salesman is relevant. Hence, has to be developed an approach for setting the parameters of the algorithm for a wide range of tasks.

The objective of the study is to develop a sufficient approach to increase the efficiency of applied algorithms of combinatorial optimization and test it in solving known problems of the salesman.

GMiGIN'2020: 2nd International Workshop on Conflict Management in Global Information Networks, November 30, 2020, Lviv, Ukraine
EMAIL: nataliya.i.boyko@lpnu.ua (N. Boyko); liubov.v.zalizna@lpnu.ua (L. Zalizna); anna.starukh@lnu.edu.ua (A. Starukh); yurkokryvenchuk@gmail.com (Yu. Kryvenchuk); inem.news@gmail.com (Yu. Malynovskyy)
ORCID: 0000-0002-6962-9363 (N. Boyko); 0000-0002-7330-000X (L. Zalizna); 0000-0003-3282-1746 (A. Starukh); 0000-0002-2504-5833 (Yu. Kryvenchuk); 0000-0002-7139-5623 (Yu. Malynovskyy)

To attain this objective, it is necessary to complete the following tasks:

- review the known results for combinatorial optimization algorithms in the field of parameter settings for the algorithm;
- formalize the problem of finding the optimal parameters of the algorithm in a bounded grid;
- to develop software implementation of combinatorial optimization algorithms, for which parameters will be set;
- conduct experiments of setting the parameters of the algorithm for a set of tasks;
- perform an analysis of the results of the experiment.

The object of research - the processes of solving combinatorial optimization problems.
The subject of research - models and methods of parameter setting of combinatorial optimization algorithms.
The scientific novelty of the obtained results is the development of a formalized approach to setting the parameters of applied algorithms of combinatorial optimization, which allows to increase the accuracy of algorithm solution.

2. Materials and methods

The traveling salesman problem was formed in the 30s of the twentieth century. This is one of the most popular combinatorial optimization problems. Its main idea is to find the most efficient route that would pass through the chosen cities with a return to the starting point. It is usually allowed to visit one city only once. Under this condition, the solution is among the Hamiltonian cycles. Many problems can be formulated on the principle of the traveling salesman problem. For example: transportation task, task of optimal creation of computer networks and so on. This task has a wide range of applications. Nowadays it can be applied in such areas as routing of transport flows, construction of algorithms for the study of grammatical structures, and most of the tasks used in the field of logistics [1, 5, 12].
There are many different approaches and methods that have been derived from theoretical calculations and research that can be used to solve the travelling salesman problem. The most effective methods, i.e. those that significantly reduce the full search, belong to the class of heuristic methods [3, 13].
The result of most of the existing methods is not the most profitable route, but only an approximation to it. You can view the following groups of methods for solving problems of the travelling salesman:
1) Ordered sampling:
This method is also called - the method of "brute force". The essence of this method, to solve the problem is to search for all possible solutions. The complexity of such a sampling depends on the size of the problem to be solved. If the set of solutions is extremely wide, then the solution of the problem by ordered sampling can take not only several years, but also possibly centuries or millennia [4, 15].
2) Random sampling :
There are cases when the solution of the problem can be represented in the form of a sequential sample of solutions. If you form such a sample using a random mechanism, it can significantly reduce the time to find a solution. You only need to remember the "best option", i.e. the most optimal of the received. This approach is quite simple, but there is also a way to improve it. It is implemented by combining the local search method with a heuristic approach to random search. These methods are widely used in the formation of the schedule of airports, trains or in the scheduling of curriculum in schools and universities [11, 17-19].
3) Greedy algorithms:
Greedy algorithm is an algorithm in which we can assume that the local solution at each stage is optimal. These methods include the method of the nearest neighbor or the method of inclusion of the nearest city, the method of the cheapest inclusion.
Also, the existing methods of solving the problems of the travelling salesman can be divided into exact and heuristic (approximate) [11-19].
Examples of exact methods:

- Ordered sampling method

- The branch and bound method

Examples of heuristic methods:

- The nearest neighbor method
- Local improvements method
- Monte-Carlo method (random sampling method)

Exact algorithms use a complete and ordered sampling of all options. Sometimes they allow you to find a solution quickly, but usually the search goes through all n! routes, where n is the number of specified cities. The salesman's task is one of the transcomputational tasks - this means that if you set more than 66 points in the bypass route, the exact solution of this problem, even with the use of modern computers, can take a very long time [6, 10].

Approximate methods of solving the problem of the traveling salesman are heuristic and are quite effective, as they reduce the complete sampling of all the routes. Many of them find not the most effective but the basic route. In the future, this route is improving.

3. Statement of the problem

In this paper, will be considered several existing algorithms for solving the problem of a travelingsalesman. The purpose of this research will be to assess the quality of each of the selected algorithms, compare and analyze the results.

Therefore, the analysis will be performed on the following algorithms:

1) The branch and bound method (Little's Algorithm);
2) The method of the nearest neighbor;
3) Approx-TSP algorithm;
4) Ant colony optimization algorithm;

The branch and bound method

Little's algorithm is one of the special cases of applying the "branch and bound" method to solve a problem of a traveling salesman. The general idea is quite trivial: it is necessary to break down a huge number of options into individual groups and get their appropriate estimates (bottom - for the task of minimization, top - for the task of maximization) to be able to reject options not a single one but whole groups. The difficulty is to find the best division into groups (branches) and such estimates (bounds) that the division process is the most effective [11, 13].

Algorithm:

1. Row reduction method (in each row of the matrix find the minimum element dmin and subtract it from all elements of the corresponding row.
   Lower limit: $H = \sum dmin$
2. Reduction operation on columns (in each row of the matrix find the minimum element dmin and subtract it from all elements of the corresponding column. Lower limit: $H = H + \sum dmin$
3. The constant H is the lower limit for the set of all admissible Hamiltonian contours
4. Search for powers and zeros for the given matrix. To do this, replace the zeros in the matrix with a temporary sign and find the sum of the minimum elements of the row and column, which correspond to this zero
5. An arc *(i; j)* is selected for which the exponent of the zero element is maximal
6. The set of all Hamiltonian contours is divided into two subsets: which contain the selected arc (i; j) from point 5, and do not contain it - *(i *; j *)*
7. The matrix of Hamiltonian contours is reduced with the search of the reduction constants *H (i; j)* and *H (i *; j *)*
8. Compare the lower limits of the subsets of Hamiltonian contours *H (i; j)* and *H (i *; j *)*. If *H (i; j) <H (i *; j *)*, then we reduce by *(i; j)*, otherwise - by *(i *; j *)*
9. If as a result we obtain a matrix of 2 * 2, we determine the Hamiltonian contour and its length.
10. The length of the Hamiltonian contour is compared with the lower limits. If the length of the contour is greater, then the problem is solved.

Consider the mathematical implementation of this algorithm. The input will be a square matrix 4 by 4 with the weights of the transition from city to city.

Input data:

0	11	5	20	5
10	0	13	5	9
11	8	0	7	17
19	15	12	0	19
10	7	18	10	0

As an arbitrary route we choose: (1,2); (2,3); (3,4); (4,5); (5,1). To determine the lower limit, we reduce the matrix, i.e. in each row we look for the minimum element and subtract from the others

					di
-1	11	5	20	5	5
10	-1	13	5	9	5
11	8	-1	7	17	7
19	15	12	-1	19	12
10	7	18	10	-1	7

We do the same for the columns:

-1	6	0	15	0
5	-1	8	0	4
4	1	-1	0	10
7	3	0	-1	7
3	0	11	3	-1
dj3	0	0	0	0

After the first step we get such matrix:

-1	6	0	15	0
2 -1	8	0	4	
1 1	-1	0	10	
4 3	0	-1	7	
0 0	11	3	-1	

The quantities di and dj are called reduction constants. The sum of these constants determines the lower limit of H:

$$H = \sum di + \sum dj = 5+5+7+12+7+3 = 39$$

For each zero we find the sum of the reduction constants (in parentheses)

i,j	1	2	3	4	5	di
1 -1	6	0(0)	15	0(4)	0	
2 2	-1	8	0(2)	4	2	
3 1	1	-1	0(1)	10	1	
4 4	3	0(3)	-1	7	3	
5 0(1)	0(1)	11	3	-1	0	
dj1	1	0	0	4		

The largest sum of reduction constants = 4 for the edge (1,5). Divide the set into two (1.5) and *(1 *, 5 *)*

i,j	1	2	3	4	5	di
1 -1	6	0	15	-1	0	
2 2	-1	8	0	4	0	
3 1	1	-1	0	10	0	
4 4	3	0	-1	7	0	
5 0	0	11	3	-1	0	
dj0	0	0	0	4		

Lower limit: H = 39 + 4 = 43

Exclude all elements belonging to 1 row and 5 columns and in order to avoid looping exclude element (5; 1):

i,j	1	2	3	4	di

2	2	-1	8	0	0
3	1	1	-1	0	0
4	4	3	0	-1	0
5	-1	0	11	3	0
dj	1	0	0	0	

Lower limit: H = 39 + 1 = 40≤43. Therefore, we include an edge (1.5) with a new H = 40.
We reduce and calculate the reduction constants:

i,j	1	2	3	4	di
2	1	-1	8	0(1)	1
3	0(1)	1	-1	0(0)	0
4	3	3	0(11)	-1	0
5	-1	0(4)	11	3	0
dj	1	1	0	0	

Edge exclusion (4,3)

i,j	1	2	3	4	di
2	1	-1	8	0	1
3	0	1	-1	0	0
4	3	3	-1	-1	3
5	-1	0	11	3	0
dj	0	0	8	0	

H = 40+11 = 51
Edge exclusion (4,3)

i,j	1	2	4	di
2	1	-1	0	0
3	0	1	0	0
5	-1	0	3	0
dj	0	0	0	

Lower limit: H = 40+0 = 40≤51. Hence, exclude edge (4,3) with the new limit H = 40.
We reduce and calculate the reduction constants:

i,j	1	2	4	di
2	1	-1	0(4)	1
3	0(2)	1	-1	1
5	-1	0(4)	3	3
dj	1	1	3	

Edge exclusion (2,4)

i,j	1	2	4	di
2	1	-1	-1	1
3	0	1	-1	0
5	-1	0	3	0
dj	0	0	3	

H = 40+4 = 44
Edge exclusion (2,4)

i,j	1	2	di
3	0	1	0
5	-1	0	0
dj	0	0	

Lower limit: H = 40+0 = 40≤44. Hence, exclude edge (4,3)) with the new limit H = 40.

Also, taking into account this matrix, we include in the route edges (3,2) and (5,2). Thus, we obtain the route:

$$(1,5) \rightarrow (5,2) \rightarrow (2,4) \rightarrow (4,3) \rightarrow (3,2)$$

Hungarian algorithm

The specific features of the tasks led to the emergence of an effective Hungarian method of solving them. The main idea of the Hungarian method is to move from the original square matrix of value C to its equivalent matrix Ce with non-negative elements and a system of n independent zeros, of which neither two belong to the same row or the same column. For a given n there is n! acceptable solutions. If in the matrix of destination X arrange n units so that in each row and column there is only one unit, arranged in accordance with the arranged n independent zeros of the equivalent matrix of value Ce, then we obtain valid solutions to the problem [3, 6, 16].

Consider the mathematical implementation of this algorithm. The input data will be a matrix of distances between cities 4 by 4.

0 11	5	20	5	
10	0	13	5	9
11	8	0	7	17
19	15	12	0	19
10	7	18	10	0

We reduce the matrix by rows. In this case, the newly formed matrix in each row will be at least one zero.

					di
-1	6	0	15	0	5
5	-1	8	0	4	5
4	1	-1	0	10	7
7	3	0	-1	7	12
3	0	11	3	-1	7

After that, the same operation is performed on the columns:

-1	6	0	15	0
2	-1	8	0	4
1	1	-1	0	10
4	3	0	-1	7
0	0	11	3	-1
dj3	0	0	0	0

In this step we obtain a completely reduced matrix

Next, we will search for a solution for which all areas have zero value.

Save the value (1,5), and the other zeros in row 1 and column 5 will not be taken into account

-1	6	[0]	15	0
2 -1	8	0	4	
1 1	-1	0	10	
4 3	0	-1	7	
0 0	11	3	-1	

We perform the same operation for the remaining zero values. As a result, we obtain a matrix.

-1	6	[0]	15	0
2 -1	8	0	4	
1 1	-1	[0]	10	
4 3	0	-1	7	
0 0	11	3	-1	

Since the position of the zero elements in the matrix does not allow to build a system of 5 independent 0, the solution without modification of the matrix is impossible.

To modify our matrix, cross out the rows and columns with the largest number of zero elements. We obtain the following matrix:

2 -1	4
1 1	10
4 3	7

Find the minimum element of the abbreviated matrix and subtract it from all elements. The result will be the following matrix:

1 -1	3
0 0	0
3 2	6

The minimum element that was subtracted from the elements of the abbreviated matrix, add to the elements that are at the intersection of the crossed rows and columns, we obtain the following matrix:

-1	6	1	16	0
1 -1	8	0	3	
0 0	-1	0	9	
3 2	0	-1	6	
0 0	12	4	-1	

Repeat the step with the reduction as a result of which we obtain a matrix with at least one zero in each row. We also perform reduction on columns.

-1	6	1	16	0
1 -1	8	0	3	
0 0	-1	0	9	
3 2	0	-1	6	
0 0	12	4	-1	

Find directions that have zero value:

-1	6	1	16	0
1 -1	8	0	3	
[0]	0	-1	[0]	9
3 2	0	-1	6	
[0]	0	12	4	-1

The number of found zeros k = 5. As a result, we obtain an equivalent matrix..

-1	6	1	16	0
1 -1	8	0	3	
0 0	-1	0	9	
3 2	0	-1	6	
0 0	12	4	-1	

Cmin = 8 + 5 + 5 + 12 + 10 = 40

So we get the following route:

(3;2), (1,5), (2;4), (4;3), (5,1)

The ant colony optimization algorithm

Ant algorithms are probabilistic greedy heuristics, where probabilities are established based on information about the quality of the solution obtained from previous decisions. The idea of the ant algorithm is to model the behavior of ants related to their ability to quickly find the shortest route from the anthill to the food source and adapt to changing conditions, finding a new shortest route. This is an elementary rule of conduct and determines the ability of ants to find a new route if the old one is unavailable [8, 10, 15].

Algorithm:

1. The starting point is selected, where the ants are placed. At this stage, the initial level of the pheromone is set.
2. The probability of transition from vertex to vertex is by the formula:

$$P_{ij}(t) = \frac{\tau_{ij}^{\alpha}(\frac{1}{d_{ij}})^{\beta}}{\sum \tau_{ij}^{\alpha}(\frac{1}{d_{ij}})^{\beta}}, \quad (1)$$

where, τ_{ij} – level of pheromone – heuristic distance, a α, β – constants. By

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum \frac{Q}{L_k}, \quad (2)$$

where, $\alpha = 0$ the choice of the nearest city, the most probable, i.e. the algorithm becomes greedy. By $\beta = 0$ the choice is based on the pheromone, which leads to suboptimal solutions.

3. The pheromone level is updated according to the following formula:

Algorithm Approx-TSP

At the input of this algorithm, an undirected graph G is given, in which it is necessary to find the Hamiltonian cycle.

To begin, you must select the starting vertex. It is necessary to create the list L in which we will enter necessary vertices. The next step is to build a minimal spanning tree, using one of the algorithms for its construction (For example, the algorithm of Prim, Kruskal, Boruvka, etc.).

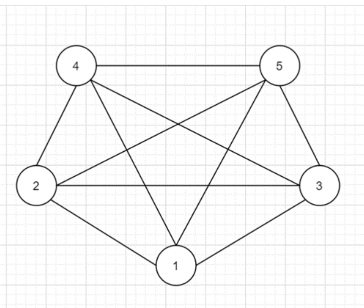
During the construction process, the vertices of the tree are entered in the list L in the same order in which the bypass was made during the construction of the minimum spanning tree. We connect end vertices and we return the approximate solution of the problem of the travelling salesman passing on vertices from the list.

Algorithm:

1. Choose *V1* as the root vertex
2. Construct the minimum spanning tree *T* for *G*, starting from the top *V1*
3. Enter the vertices in the list *L*
4. We return the Hamiltonian cycle, which passes through the vertices of the list *L*.

0 11	5	20	5	
10	0	13	5	9
11	8	0	7	17
19	15	12	0	19
10	7	18	10	0

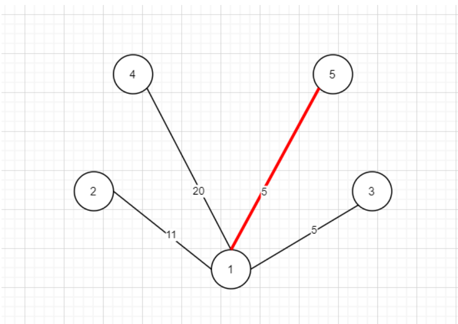
Based on the matrix, we construct a graph:



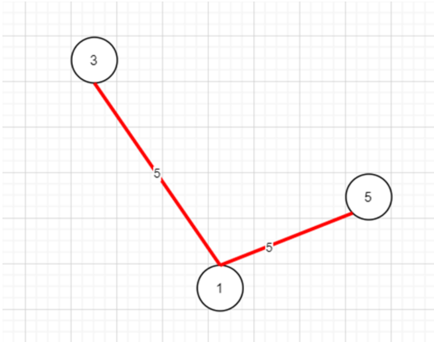
Choose the starting vertex *V0* = 1

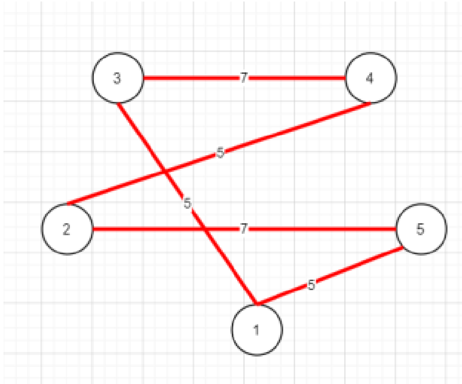
According to Prim's algorithm, we find the minimum spanning tree:

The initial vertex was chosen vertex 1. Add to the tree all the edges with the least weight that have a vertex 1



Add the vertex (1, 5) to the tree. The next vertex is the closest of the selected 1 or 5. This will be the vertex (1,3)

Next, we add a  vertex according to this algorithm (5,2), (2,4), (4,3):



Having formed the list L we receive the following route: (1,5), (5,2), (2,4), (4,3), (3,1)

4. Analysis of the output results

The result of this study will be a graph with the best route and time of the algorithm. For algorithms that require special operating conditions, the matrix will be adapted and possible error will be taken into account. For the rest - the matrix will contain the same values for more accurate analysis.

The branch and bound method

During the operation of this program, the operating time of the algorithm was recorded. The cost of the route is also calculated - the sum of the entire constructed route. Studies were performed up to N = 15. The results are presented in the form of a table.

Table 1

The results of calculations for the branch and bound method le

N	Operating time	Cost of the route
5	0,01637	439
6	0,02671	443,44
7	0,04085	535,79
8	0,06163	477,7
9	0,08351	489,29
10	0,11281	550,27
11	0,54384	554,26
12	0,93642	572,14
13	1,6512	615,54
14	2,95794	632,03
15	5,4982	677,7

Analyzing the results obtained during the experiments and listed in table 1, it is seen that with increasing size of the matrix (increasing the number of cities to be visited by the salesman), respectively, increases not only the operating time, but also the costs of moving between cities are also increasing.

The Figure 1 below will show the dependence of time on the number N in the matrix.

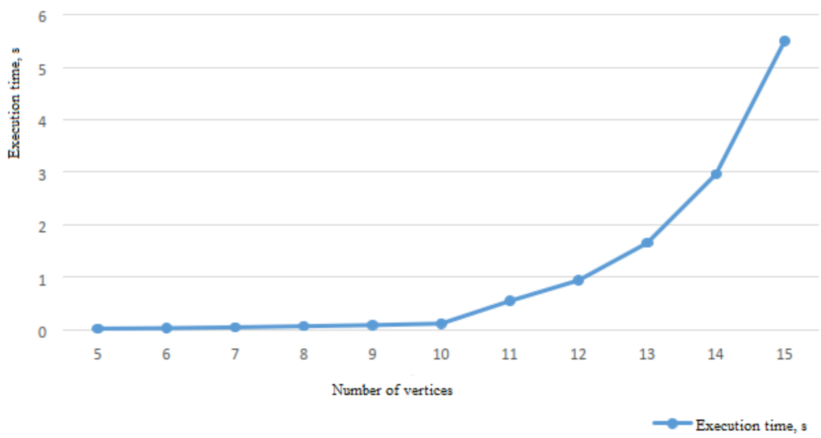


Figure 1: Dependence of time on the number of elements of the matrix

Based on the results of the experiment, we can conclude that the operating time of this algorithm increases sharply with the number of elements. This is confirmed by the complexity of this method - *O (n * log2 (n)).*

Algorithm Approx-TSP

The implementation of this algorithm also allows you to specify n. During the experiments we will continue to record the time spent on the operation of the algorithm and the cost of the route..

The results of the experiment are shown as a table.

Table 2
Calculation results for the algorithm Approx - TSP

N	Operating time	Cost of the route
5	0,01542	439
6	0,02217	443,44
7	0,03052	537,91
8	0,04736	481,27
9	0,09513	493,25
10	0,1944	552,27
11	0,3168	557,63
12	0,4958	576,41
13	0,7324	617,46
14	0,9655	635,32
15	1,3632	681,47

From the above table, we can conclude that despite the fact that the operating time of the algorithm increases with increasing number of cities, this increase is not critical compared to other algorithms.

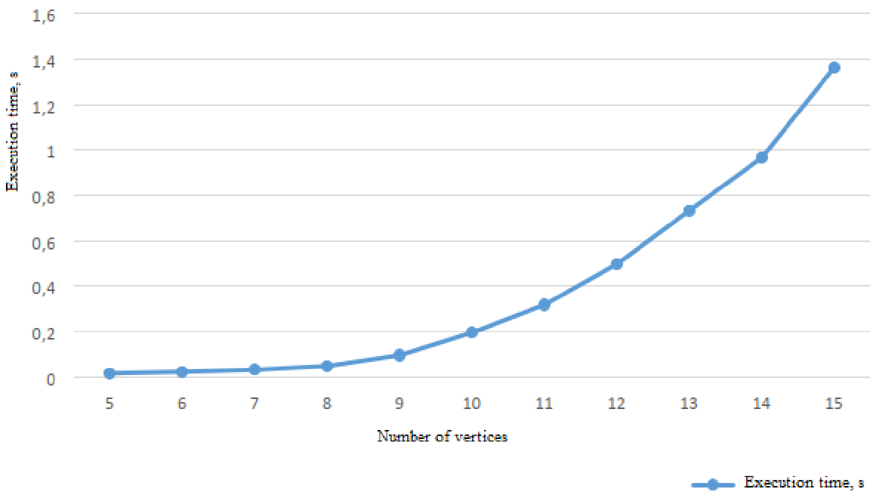


Figure 2: Dependence of time on the number N in the matrix

In the Figure 2, the dependence of time on the number N in the matrix is shown.

Analyzing the obtained results, it is seen that this algorithm also increases with increasing elements, but has a relatively high speed.

5. Comparison of implemented algorithms

The complexity of the algorithm Approx-TSP - *O (n)*. This means that as the size of the matrix increases, the time spent will increase proportionally.

The complexity of the method of bounds and branch is *O (n * log2 (n))*, i.e. as the matrix increases, the time will also increase proportionally, but the complexity of the algorithm calculations will increase with increasing *n*. Even at small values, the algorithm will spend a lot of time. The considered algorithms work at almost the same speed

Table 3

Consideration of algorithms that work at the same speed

N	Branch and bound method	Approx-TSP
5	0,01637	0,01542
6	0,02671	0,02217
7	0,04085	0,03052
8	0,06163	0,04736
9	0,08351	0,09513
10	0,11281	0,1944
11	0,54384	0,3168
12	0,93642	0,4958
13	1,6512	0,7324
14	2,95794	0,9655
15	5,4982	1,3632

But this will only happen until a certain point, then the growth stops and time will be linear. We combine the results of the methods into one table.This table shows that in the initial stages, the considered algorithms work at almost the same speed, but with a significant increase in the number of cities, the Approx-TSP algorithm shows much better results in terms of speed-code of the algorithm.

Representation of the results of these methods on a graph.

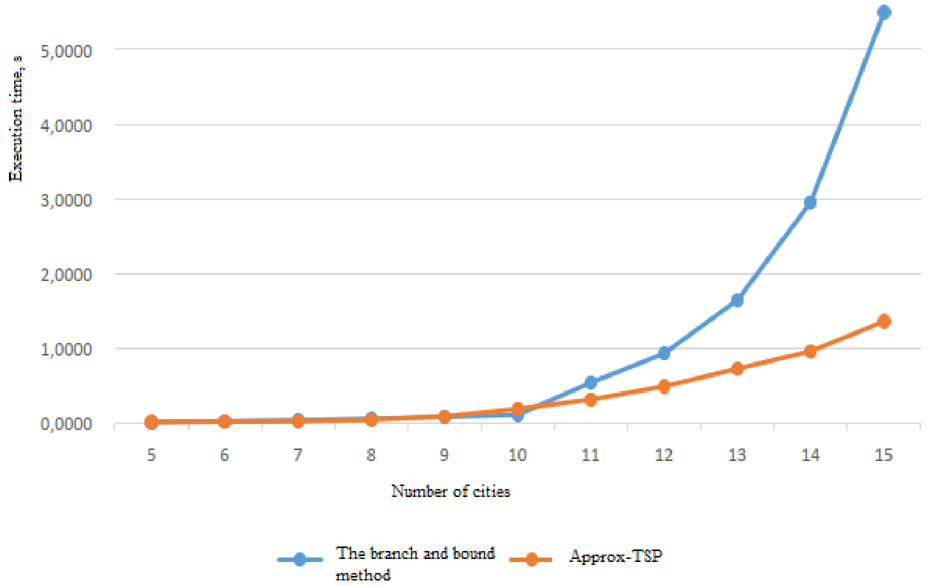


Figure 3: Representation of the results f the the branch and bound method and Approx-TSP

The graph shows that the Approx - TSP method works much faster than the method of branch and bounds. But to determine the efficiency of an algorithm must take into account not only the time of its operation.

Since the branch and bound method is a representative of the class of exact algorithms, and Approx–TSP – approximations, it is possible to calculate the ratio (error) of one algorithm to another.

Calculate the error of the exact method (the branch and bound method) in relation to the approximate (Approx - TSP).

Table 4

Calculate the error of the exact method (the branch and bound method) in relation to the approximate (Approx – TSP)

N	Cost of route (the branch and bound method)	Cost of route (Approx – TSP)
5	439	439
6	443,44	443,44
7	535,79	537,91
8	477,7	481,27
9	489,29	493,25
10	550,27	552,27
11	554,26	557,63
12	572,14	576,41
13	615,54	617,46
14	632,03	635,32
15	677,7	681,47

Analyzing the obtained results, it can be seen that the error of the approximate algorithm is not higher than 10% and averages about 5%. The error was calculated based on this formula:

$$\Delta = \frac{x - x_{icm}}{x_{icm}} \qquad (3)$$

We visualize the dynamics of the change in error relative to the increase in the number of cities. As can be seen from the graph, the Approx-TSP method has a relatively low error compared to the branch and bound method.

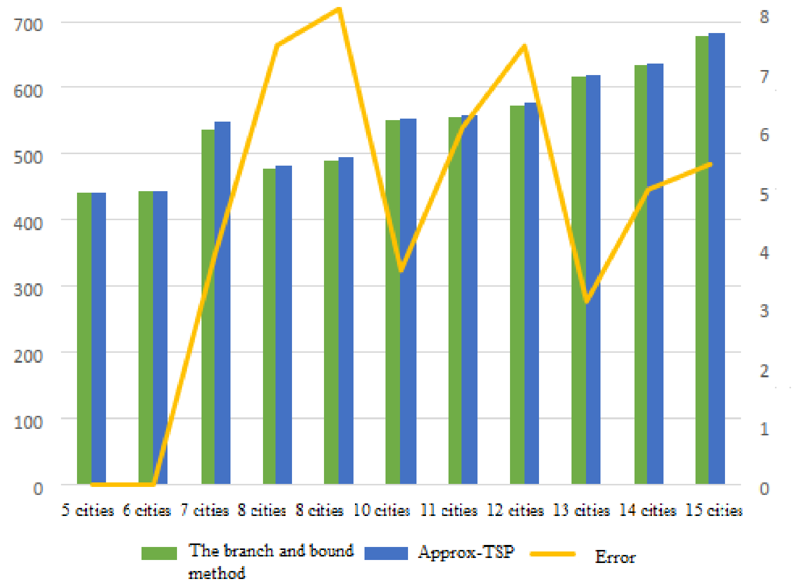


Figure4: The dynamics of the change in error relative to the increase in the number of cities

6. Conclusion

After the conducted experiments, it is difficult to finally choose which of the algorithms to use in a given situation. The branch and bound method is a representative of the class of exact algorithms. The advantage of its operation is the accuracy of the results, but a significant disadvantage is the operating time. On large data arrays, the running time of this algorithm can lead to significant time losses.

While the Approx-TSP algorithm has significantly faster run times than other algorithms, its results are not as accurate. The error when using this algorithm ranges from 5-10%.

Therefore, the feasibility of using a specific of the considered algorithms should be determined based on the general needs and requirements of each case.

7. References

[1] S. Rana, Examining the Role of Local Optima and Schema Processing in Genetic Search, 1999.

[2] N. Boyko, Lesia Mochurad L., Uliana Parpan U., Oleh Basystiuk O. Usage of Machine-based Translation Methods for Analyzing Open Data in Legal Cases, Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019) co-located with 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019), Kyiv, Ukraine, November 30, 2019, pp. 328-338 pp.

[3] Boyko N., Mochurad L., Chervinska O. Cost-Effective Use of Mathematical Methods for the Organization and Management of Data in the Cloud, Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019) co-located with 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019), Kyiv, Ukraine, November 30, 2019, pp. 444-457.

[4] K.K. Lai, J.W.M. Chan, Developing a simulated annealing algorithm for the cutting stock problem, Computers & Industrial Engineering, vol. 32, 1997, pp. 115-127.

[5] N. Boyko, L. Mochurad, I. Andrusiak, Yu. Drevnytskyi, Organizational and Legal Aspects of Managing the Process of Recognition of Objects in the Image, Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019) co-located with 1st International Conference on Cyber Hygiene and Conflict Management in Global Information Networks (CyberConf 2019), Kyiv, Ukraine, November 30, 2019, pp. 571-592.

[6] N. Boyko, L. Mochurad, O. Chervinska, Cost-Effective Use of Mathematical Methods for the Organization and Management of Data in the Cloud, Proceedings of the International Workshop on Cyber Hygiene (CybHyg-2019) co-located with 1st International Conference on Cyber Hygiene

and Conflict Management in Global Information Networks (CyberConf 2019), Kyiv, Ukraine, November 30, 2019, pp. 444-457.

[7] P. Larra Naga, C.M.H. Kuijpers, R.H.Murga, I. Inza, S. Dizdarevic, Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators,. Artificial Intelligence Review, Kluwer Academic Publishers. Printed in the Netherlands, Vol. 13(2), 1999, pp. 129–170.

[8] J. Majumdar, A.K. Bhunia, Genetic algorithm for asymmetric traveling salesman problem with imprecise travel times. Journal of Computational and Applied Mathematics, Elsevier, Vol. 235, Issue 9, 2011, pp. 3063-3078.

[9] O. E. Semenkina, E. A. Popov, O. E. Semenkina, Self-configuring evolutionary algorithms for travelling salesman problem. Journal of Siberian State Aerospace University named after academician M. F. Reshetnev , Vol. 4(50), 2013, pp. 134-139.

[10] M. Mitchell, An Introduction to Genetic Algorithms, Cambridge USA, London UK: MIT Press, 1999.

[11] A. Shabalov, E. Semenkin, P. Galushin, Automatized Design Application Of Intelligent Information Technologies for Data Mining Problems. Joint IEEE Conference, The 7th International Conference on Natural Computation & The 8th International Conference on Fuzzy Systems and Knowledge Discovery, Shanghai, China, 2011, pp. 2659–2662.

[12] E. Semenkin, M. Semenkina, Self configuring genetic algorithm with modified uniform crossover operator, Advances in Swarm Intelligence, ICSI 2012, Part 1, LNCS 7331, Springer, Heidelberg, 2012, pp. 414.–421.

[13] X. Chen, P. Zhang, G. Du, F. Li, Ant colony optimization based memetic algorithm to solve bi-objective multiple traveling salesmen problem for multi-robot systems, Received March 14, 2018, accepted April 12, 2018, date of publication April 19, 2018, date of current version May 9, 2018. Digital Object Identifier 10.1109/ACCESS.2018.2828499

[14] L. Grady, E. L. Schwartz, Isoperimetric Graph partitioning for Image segmentation. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.28(3), 2006, pp.469-475.

[15] J. Gaber, M. Bakhouya, An Immune Inspired-based Optimization Algorithm: Application to the Traveling Salesman Problem. Advanced Modeling and Optimization, Vol. 9(1), 2007, pp. 105–116.

[16] S. Fedushko, V. Ortynskyy, V. Reshota, V. Tereshchuk, Legal And Economic Aspects of the PR Campaign of Scientific Conference in Social Networks. CEUR Workshop Proceedings. Vol 2616: Proceedings of the 2nd International Workshop on Control, Optimisation and Analytical Processing of Social Networks (COAPSN-2020), Lviv, Ukraine, May 21, 2020. p. 342-352. http://ceur-ws.org/Vol-2616/paper29.pdf

[17] S. Fedushko, Yu. Syerov, O. Tesak, O. Onyshchuk, N. Melnykova, Advisory and Accounting Tool for Safe and Economically Optimal Choice of Online Self-Education Services. Proceedings of the International Workshop on Conflict Management in Global Information Networks (CMiGIN 2019), Lviv, Ukraine, November 29, 2019. CEUR-WS.org, Vol-2588. pp. 290-300. http://ceur-ws.org/Vol-2588/paper24.pdf

[18] Y. Liu and X. Zhang, "Evaluating the Undergraduate Course based on a Fuzzy AHP-FIS Model," IJMECS, vol. 12, no. 6, pp. 55–66, Dec. 2020, doi: 10.5815/ijmecs.2020.06.05.

[19] M.F. Yakovlev, T.O. Gerasymova, A.N. Nesterenko, Characteristic feature of the solving both of non-linear systems and systems of ordinary differential equations on parallel computer. In Proceedings of international symposium Optimization problems of computations (OPC - XXXV), Kyiv: V.M. Glushkov Institute of cybernetics of NAS of Ukraine, 2009. Kyiv: Vol. 2. P. 435-439. (2009).

[20] M.F. Yakovlev, A.N. Nesterenko, V.N. Brusnikin, Problems of the efficient solving of non-linear systems on multi-processor MIMD-architecture computers. Mathematical machines and systems. (4). P. 12-17. (2014).