Modified Traveling Salesman Problem for a Group of Intelligent Mobile Objects and Method for Its Solving

Vitalii Tkachov, Volodymyr Tokariev, Iryna Ilina, and Stanislav Partyka
Department of Electronic Computers, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine
Email: {tkachov; tokarev}@ieee.org; {iryna.ilina; stanislav.partyka}@nure.ua

***Abstract*—A solution of a modified traveling salesman problem for the case, in which a group of intelligent mobile objects is simultaneously involved, is discussed in the article. For this purpose, a review and a comparative analysis of Johnson's algorithms and the branch and bound method have been carried out. It was found that these algorithms are ineffective for solving the modified traveling salesman problem for a group of intelligent mobile objects, therefore, a quest method has been further developed as a solution to this problem. The results of experimental calculations are presented. The directions of further work on the unification of the developed method for constructing structures of fast step-by-step data processing are proposed.**

***Index Terms*—Group interaction, intelligent mobile object, traveling salesman problem**

I. INTRODUCTION

The tasks of organizing step-by-step processing, data collection, or servicing mobile objects are characterized by a constantly replenishing list of practical solutions, taking a traditionally important place in the study of combinatorial optimization problems [1]. In the broad sense, the given functional task is the task of current planning, during which moving objects are selected and trajectories as well as schedules of their movement between points of the system are determined. Under the conditions of each specific task, a description of the point system is provided, which defines a set of possible paths of movement to the target of all moving mobile objects. As a rule, the structural parameters of the point system remain unchanged until the end of the process of solving the problem [1].

Hence, in the field of information technology, there is a direction uniting the tasks of organizing step-by-step data processing [2]. Let us highlight the most prioritized of them. For example, in the process of data mining, data preprocessing is an important step. It includes a set of successive procedures: cleansing, normalization, feature separation, etc. [3]. When designing a computer network to perform the task of data preprocessing, there are two directions. The first, classical, is the creation of polyfunctional data processing subsystems (knowledge extraction). Each of the subsystems performs a complete cycle of processing each of data chunks. This means that a data chunk, when it enters such a computer network, exclusively occupies a subsystem for the entire time of its processing. The second direction is the creation of mono-functional subsystems, which is typical for cloud architectures when implementing clusters from “software as a service” nodes. In other words, a certain data processing step can be performed by a certain number of subsystems. When a data chunk arrives at the boundary subsystem, it is processed and further transferred to the next specialized subsystem, etc. In addition, when using onion architecture, each layer can be responsible for its step of data processing.

Another task is based on the fact that there is no requirement for any sequence of performed operations on servicing mobile objects, however, intelligent mobile objects must pass all points [4]-[6]. Each of the points is characterized by its own set of tools for performing various actions of servicing objects. Such a scenario has been taken from a typical quest game where teams must visit all game points, complete the necessary tasks on them and get to the end point. The simultaneous presence of many teams leads to the need of developing a quasi-optimal algorithm of step-by-step movement planning for each team, taking into account the travel time of points passing and bonuses.

There are also modifications of transport problems, which consider the following: a set of autonomous data storages that are filled with data chunks from information recording systems; and mobile data-collection agents that move between nodes to collect a certain type of data [7]. In this case, the route of the collection agents must be formed so that the time for collecting data from all nodes is minimal. Such a scenario is based on the inverse problem of minimizing the delivery of various products by automobiles to retail outlets within the city location, taking into account the fact that each outlet can simultaneously receive one type of goods from only one automobile for a certain fixed time. The problem of the land transport and methods for solving it are studied in the framework of the scientific direction–transport logistics, the mathematical apparatus of which is represented by graph theory and research on operations [8].

Manuscript received April 22, 2020; revised May 30, 2020; accepted June 14, 2020.

Corresponding author: Vitalii Tkachov (email: tkachov@ieee.org).

doi: 10.18178/ijeetc.10.1.1-7

Given the above varieties of the problem of organizing either step-by-step processing of data or queue-based servicing of objects, this problem can be generally characterized as a modification of the traveling salesman problem, where there may be several such “traveling salesmen” (mobile objects). Moreover, these objects are characterized by downtime at the point (which is determined by the point and/or its queue). The aim is to minimize the travel time of all points for each of the intelligent mobile objects, provided that preliminary agreements between the objects are excluded. The modified versions of traveling salesman problem are common in practice of robotics (indoor transportation [9]) and delivery drones (outdoor transportations [10])

The goal of the article is to develop a modified quest method for solving the traveling salesman problem for the case when more than one intelligent mobile object is involved.

II. Problem Statement

Suppose that according to the condition of the problem, a set of intelligent mobile objects U needs to visit all points of a system. Each of the intelligent mobile objects stops at each point for a time set by the point. It is required to determine in which order each of the intelligent mobile objects will visit all points, so that the travel time of points for each of the intelligent mobile objects is minimal. In the classical case, for one such intelligent mobile object this task is similar to the traveling salesman problem provided that the object returns to the starting point [11]. In contrast to the classical traveling salesman problem, there is an open traveling salesman problem [11].

Additional conditions of the problem are: the distances between points (the time taken by an intelligent mobile object to move between points at the maximum arc bandwidth); starting points are assigned according to the arbitrary taken distribution function; the downtime of an intelligent mobile object at a point and the availability of information about the utilization of points by other objects at the current time moment (queue); all queues at the points are organized according to the “first in first out,” principle [12].

Let us perform a mathematical description of the point system. Assume that there is a set of intelligent mobile objects $q_n \in Q$, a set of system points $u_n \in U$ and the weight of the arc between them $t_{u_n \to u_{n+m}}$ where m=1, 2, ···, M. In order to service a certain set of objects, points need to perform a set of actions $f_1, f_2, \cdots f_r$ in time t_{u_n}. Thus, the operation of the point can be represented as:

$$a_0 \frac{d^n y}{dt^n} + a_1 \frac{d^{n-1} y}{dt^{n-1}} + \ldots + a_n y = b_0 \frac{d^m x}{dt^m} + b_1 \frac{d^{m-1} x}{dt^{m-1}} + \ldots + b_m x \quad (1)$$

where $x(t)$ is the setting effect of the object; $y(t)$ is the point status or an output variable; a_i and $b_i$are the types of work carried out at the point.

Points can be combined into clusters according to the actions they perform, and the intelligent mobile object must be serviced using all points in any sequence of passing them. Thus, the task can be clarified according to the following requirements: precedence conditions should not be fulfilled unless specified by other conditions; the sum of the downtimes of an intelligent mobile object at a point should not exceed the relevance of performing the traveling salesman problem as a whole; the sum of the travel times T that take an intelligent mobile object to visit all the points should be minimized.

This problem can also be formulated as a linear programming problem. Let $z_{f,u}$ be an unknown variable taking the values 0 and 1. f is the action performed at the u th point if and only if $z_{f,u} = 1$, and the opposite holds if $z_{f,u} = 0$. Each action with an object must necessarily be performed on one of the points in the cluster of points of the same type for each of the intelligent mobile objects, therefore:

$$\sum_{u=1}^{M} k_{f,u} = 1, \; i = 1, 2, \ldots, n. \quad (2)$$

Sums of the form $\sum_{j=1}^{n} k_{f,u} t_f$ are numerically equal to the sum of the time periods of performing all work at the uth point. These sums are subject to restrictions:

$$\sum_{u=1}^{n} k_{f,u} t_f \le T, u = 1, 2, \ldots, M. \quad (3)$$

If, under the conditions of the problem, a restriction is introduced on the preceding servicing of an intelligent mobile object $f \prec l$, then

$$\sum_{u=1}^{m} k_{f,u} \ge \sum_{u=1}^{m} k_{l,u}, \; m = 1, 2, \ldots, M. \quad (4)$$

$$F(t) = \sum_{u=1}^{m} k_{f,u} t_u \to \min \quad (5)$$

The restrictions introduced above (2) to (4) allow to obtain the formulation of the modified traveling salesman problem for a group of intelligent mobile objects in terms of linear integer programming:

$$\begin{cases} T \ge \sum_{u=1}^{n} k_{f,u} t_u, \; u = 1, 2, \ldots, M; \\ \sum_{u=1}^{M} k_{f,u} = 1, \; f = 1, 2, \ldots, n; \\ \sum_{u=1}^{m} k_{f,u} \ge \sum_{j=1}^{m} k_{l,u}, m = 1, 2, \ldots, M, \; f \prec l; \\ k_{f,u} \ge 0; \\ T \ge 0; \\ E(k_{f,u}) = k_{f,u}. \end{cases} \quad (6)$$

where the restrictions on the bandwidth of arc between points can be additionally introduced.

III. Overview of Known Solutions

In [13], Johnson's algorithm was presented for sparse graphs in the asymptotic limit, where it was shown that it performs better than the multiple matrix squaring algorithm and the Floyd-Warshall algorithm. Johnson's algorithm uses a method of weight changing. Its essence is as follows. If the weights of all the edges in a given graph are non-negative, then it is possible to find the shortest paths between all pairs of vertices, running the Dijkstra algorithm for each vertex one time. If a non-decreasing priority queue is implemented in the form of a Fibonacci pyramid, then the operation time of such an algorithm will be equal to $O(V^2 \lg V + VE)$ where V, E are the vertices of the graph. The drawback of this approach is its practical implementation. In software execution, when scaling the system of service points, the complexity of the algorithm grows, and the amount of redundant data in a real system leads to an overload of the points, recognized as optimal, by redundant data of requests and status checks.

In [14], for a particular case of the problem posed above, there is a solution implementing the branch and bound method strategy, which is used to solve the closed general traveling salesman problem. A feature of the proposed method is the selection of such a solution that contains the smallest number of edges. In [14], an example of a transport network is considered in the form of a connected incomplete graph with arbitrarily given non-negative weights of edges. An assumption is made about the presence of a Hamiltonian cycle. That is, the cost of passing the points by an object is not less than the cost of solving the general traveling salesman problem. The disadvantage of solving this problem is that in the case of adding a new object, it becomes impossible to optimally determine a new pair of points, since the system may not have data on its load or queue at the current time. From the point of view of practical implementation, the disadvantage is also the fact that the algorithm often stops without reaching optimality: either given its design or by necessity. In this case, we can obtain a complete solution for bypassing all points where the smallest boundary for all functioning points is the lower boundary of the optimal given cost. However, the obtained relative error does not exceed the ratio of the cost-boundary difference to the lower boundary itself.

In [5], an algorithm for determining the optimal point for the next processing step in a multipath problem was considered. Its essence lies in the fact that the objects enter a system consisting of several clusters of points, in which, in turn, multifunctional points operate. Each of the points can service the object once, which eliminates the principle of exclusive use of one node for all steps of processing a data chunk. The algorithm for determining the optimal point for the next step of processing takes into account: the time of moving objects between points, in fact, servicing and waiting in the queue of the point. The drawback of the proposed solution is the issue of scaling the point system in order to identify the limit threshold values for the optimum of given solution implementation.

In various special cases, such approximate algorithms as the Ford-Fulkerson method, Edmonds-Karp method, etc. can be used for this problem [15], [16]. In all these methods, the drawback is the inefficiency for calculating all the shortest paths in sparse graphs, when the number of pairs of edges between pairs of vertices is in most cases equal to one, or because of the specifics of the "greedy algorithm" that underlies the nearest neighbor algorithm in the traveling salesman problem.

IV. Application of the Quest Method for Solving the Modified Traviling Salesman Problem

Each intelligent mobile object has a matrix of values of the transition time between all pairs of points in the point system and the downtime in each of the points that determine the priority of the point selection for the next movement step at each moment of time. The point with the highest priority will be selected by the intelligent mobile object for the next movement step (Table I):

TABLE I: Status of Points (General View)

Status	Points			
	u_n	u_{n+1}	…	u_{n+m}
q	s_n	S_{n+1}	…	s_{n+m}

Based on the quest method [17], [18], an algorithm has been developed that is presented below as a sequence of steps.

Step 1. Equality is checked by the formula:

$$H_{u_n} = t_{u_n \to u_{n+m}} - (f_r \times g) - \zeta \quad (7)$$

where $t_{u_n \to u_{n+m}}$ denotes the weight of the arc (transition time of an intelligent mobile object between points); f_r is the weight of the point (downtime of an intelligent mobile object at a point); $g = \sum_{n=1}^{m} q_n$ denotes the number of intelligent mobile objects in the point queue; ζ is the residual value of the point weight (the part of the time that is necessary to finish servicing an intelligent mobile object at the time when the point is assigned to be optimal for the object of another point).

Step 2. Checking the conditions for selecting the next point for intelligent mobile object movement.

Step 2.1. If $H \geq 0$, then $s_n = t_{u_n \to u_{n+m}} + f_r$.

Step 2.2. If $H < 0$, then $s_n = (f_r \times g) - t_{u_n \to u_{n+m}} + f_r$.

Step 3. Next, a search for the minimum set of values is performed: the transition time between the starting point and all other points in the point system and the downtime at each point.

It is important to note that when performing Step 2.2, the intelligent mobile object enters the service queue at the point. Each of the points, at which the intelligent mobile object has been serviced, is excluded from the list of available points. The algorithm is repeated again for all points until all intelligent mobile objects pass all points.

Let us consider a theoretical example of the proposed solution. Suppose there is a fixed number of intelligent mobile objects and points. Starting distribution of objects by points is proposed to be carried out according to the principle of division with the remainder:

$$u_0 = 1 + n \bmod U. \quad (8)$$

where u_0 is the starting point that is defined for the object; n is the number of the intelligent mobile object; U is the number of points.

Suppose for each intelligent mobile object there is a step-by-step time matrix in which reference times of passing all points at the current moment of time are entered, and the columns of the matrix of passed points get the assigned value (-1), which blocks returning to this point. Filling the matrix with values starts when an intelligent mobile object declares finishing of passing the starting point. The minimum value among the columns of time points in the row of the second step is the optimal point that is assigned to the intelligent mobile object for further completing of the task. In this case, the point receives the (+1) value of the queue, which affects the search for the optimal point for all other objects. After passing the last point, the system should direct the intelligent mobile object to the end point. Let us consider an example of such a matrix (Table II).

The starting point u_0 for the object q_1 has been defined as u_2 (8). For all further steps, this point is excluded so that the object will not pass it again. After passing the starting point, the system calculates the time values for all other points of the second step. In our case, they are: point 1 has value 4; point 2 is excluded; point 3 has value 7; etc. Next, the intelligent mobile object selects the minimum value from the matrix (Table II) and determines the corresponding point as an optimal one (in our case, this is point No. 4 – value 2). If there are several minimum values, the system will select the point, the number (identifier) of which is less (as, for example, in the third step, points 1 and 3 have the same value 12. The optimal point will be point 1, since its number is less than 3). After determining the optimal point and assigning it to the intelligent mobile object, the value g at the point itself increases by one q. The selected (passed) point at all further steps becomes inaccessible for selection by the same intelligent mobile object (record (–1)).

At the last step, when there are no more available points for passing, the system determines the next point – the end point u_{finish}.

Calculation of time for each point at each step (except for determination of u_{finish}), taking into account (7), is carried out according to the scheme below:

$$\begin{cases} t_{u_n \to u_{n+m}} \geq f_r(g+1),\ npu\ g = 0,1,2,... \Rightarrow \\ \qquad \Rightarrow H_{u_n} = t_{u_n \to u_{n+m}} + f_r; \\ t_{u_n \to u_{n+m}} < f_r(g+1),\ npu\ g = 0\ \Rightarrow \\ \qquad \Rightarrow H_{u_n} = t_{u_n \to u_{n+m}} + f_r; \\ t_{u_n \to u_{n+m}} < f_r(g+1),\ npu\ g = 1,2,... \Rightarrow \\ \qquad \Rightarrow H_{u_n} = f_r(g+1). \end{cases} \quad (9)$$

TABLE II: OBJECT ROUTE MATRIX

Step	Points				
	u_1	u_2	u_3	u_4	u_{finish}
1	-1	0	-1	-1	-1
2	4	-1	7	2	-1
3	12	-1	12	-1	-1
4	-1	-1	6	-1	-1
5	-1	-1	-1	-1	0

The proposed solution has also been successfully applied to the following subtasks.

Subtask 1. Is it possible for an intelligent mobile object to go to a point despite the determined next point, if the weight of the arc to it is minimal? Solution: before the transition of an intelligent mobile object to a point, it is checked whether it is determined to be optimal for this intelligent mobile object. If this condition is not met, the point cannot service such an object.

Subtask 2. Intelligent mobile objects temporarily stopped passing points (for example, due to external influence). What effect will this have on the system? There are two possible solutions. The first solution: until an intelligent mobile object reaches a point determined for it, it will occupy the queue of this point. However, those intelligent mobile objects, for which this point will be actually defined as an optimal one, can be serviced by the point provided there is be no other intelligent mobile objects in the queue when they transfer to this point. As previously agreed, the queue is formed according to the "first in first out," principle [13]. The second solution: it may be necessary to provide for the creation of time limits, and the end point will be assigned as the next point for all intelligent mobile objects when they reach those limits.

V. VERFICATION OF SOLUTION EFFICIENCY

Let us consider an example. Three points u_2, u_3, u_4 are set, which are available for movement of the intelligent mobile object from the starting point u_1. It is necessary to choose from these three points an optimal point for the next step of the movement of the intelligent mobile object (Fig. 1).

Here: the travel time between points: $t_{u_1 \to u_2}$ = 3 s; $t_{u_1 \to u_3}$ = 2 s; $t_{u_1 \to u_4}$ = 4 s; the reference time of passing the point $u_2 : f_r$ = 1 s; $u_3 : f_r$ = 6 s; $u_4 : f_r$ = 2 s; the number of objects in the queue of the point: $u_2 : g$ = 3; $u_3 : g$ = 1; $u_4 : g$ = 1.

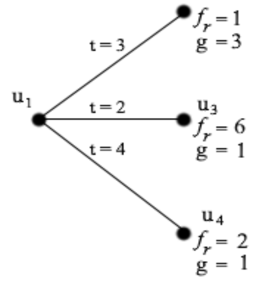

Fig. 1. The considered scenario.

Step 1. Let us make calculations for the point u_2. We verify the equalities from (9). Since the queue for servicing of the point u_2 is not zero, we exclude the second equality from (8) from the verification:

$$\begin{cases} 3 \geq 1(3+1), \text{ condition is not fulfilled} \\ 3 < 1(3+1), \ H_{u_2} = 1(3+1), \text{ condition is fulfilled} \end{cases}$$

$H_{u_2} = 4$ s.

Step 2. Let us perform the calculations for the point u_3. We verify the equalities from (9). Since the queue for servicing of the point u_3 is not zero, we exclude the second equality from (8) from the verification:

$$\begin{cases} 2 \geq 6(1+1), \text{ condition is not fulfilled} \\ 2 < 6(1+1), \ H_{u_3} = 6(1+1), \text{ condition is fulfilled} \end{cases}$$

$H_{u_3} = 12$ s.

Step 3. Let us perform calculations for the point u_4. We verify the equalities from (9). Since the queue for servicing the point u_4 is not zero, we exclude the second equality from (9) from the verification:

$$\begin{cases} 4 \geq 2(1+1), \ H_{u_4} = 4+2, \text{condition is fulfilled} \\ 4 < 2(1+1), \text{ condition is not fulfilled} \end{cases}$$

$H_{u_4} = 6$ s.

For the points u_2, u_3, u_4 the following time values were obtained: 4, 12 and 6 s. The optimal point for the movement of an intelligent mobile object is the point u_2.

VI. Model Experiment

A model experiment has been performed in the MATLAB environment. There have been set: 5 points characterized by the service time of intelligent mobile objects; transition time between points; 4 intelligent mobile objects; starting point numbers corresponding to intelligent mobile objects (Fig. 2).

In general, the system is described as follows:

$$\begin{cases} u_1 \left\langle f = 2;\ t_{u_1 \to u_2} = 1;\ t_{u_1 \to u_3} = 8;\ t_{u_1 \to u_4} = 2;\ t_{u_1 \to u_5} = 1 \right\rangle \\ u_2 \left\langle f = 3;\ t_{u_2 \to u_1} = 1;\ t_{u_2 \to u_3} = 3;\ t_{u_2 \to u_4} = 6;\ t_{u_2 \to u_5} = 7 \right\rangle \\ u_3 \left\langle f = 1;\ t_{u_3 \to u_1} = 8;\ t_{u_3 \to u_2} = 3;\ t_{u_3 \to u_4} = 2;\ t_{u_3 \to u_5} = 1 \right\rangle \\ u_4 \left\langle f = 4;\ t_{u_4 \to u_1} = 2;\ t_{u_4 \to u_2} = 6;\ t_{u_4 \to u_3} = 2;\ t_{u_4 \to u_5} = 3 \right\rangle \\ u_5 \left\langle f = 1;\ t_{u_5 \to u_1} = 1;\ t_{u_5 \to u_2} = 7;\ t_{u_5 \to u_3} = 1;\ t_{u_5 \to u_4} = 3 \right\rangle. \end{cases}$$

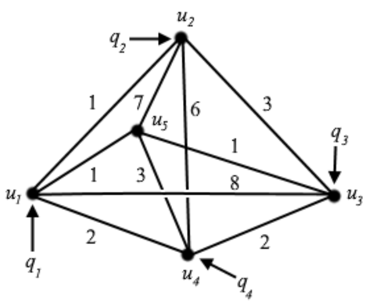

Fig. 2. The initial conditions.

TABLE III: ROUTE OF OBJECT q_1

Step	Point				
	u_1	u_2	u_3	u_4	u_5
1	-1	-1	-1	-1	-1
2	-1	4	9	6	2
3	-1	10	3	7	-1
4	-1	6	-1	6	-1
5	-1	-1	-1	11	-1
6	-1	-1	-1	-1	-1

TABLE IV: ROUTE OF OBJECT q_2

Step	Point				
	u_1	u_2	u_3	u_4	u_5
1	-1	-1	-1	-1	-1
2	3	-1	4	11	9
3	-1	-1	9	6	2
4	-1	-1	3	7	-1
5	-1	-1	-1	6	-1
6	-1	-1	-1	-1	-1

TABLE V: ROUTE OF OBJECT q_3

Step	Point				
	u_1	u_2	u_3	u_4	u_5
1	-1	-1	-1	-1	-1
2	10	5	-1	7	2
3	5	10	-1	7	-1
4	-1	4	-1	6	-1
5	-1	-1	-1	10	-1
6	-1	-1	-1	-1	-1

TABLE VI: ROUTE OF OBJECT q_4

Step	Point				
	u_1	u_2	u_3	u_4	u_5
1	-1	-1	-1	-1	-1
2	6	9	3	-1	4
3	10	7	-1	-1	3
4	3	10	-1	-1	-1
5	-1	4	-1	-1	-1
6	-1	-1	-1	-1	-1

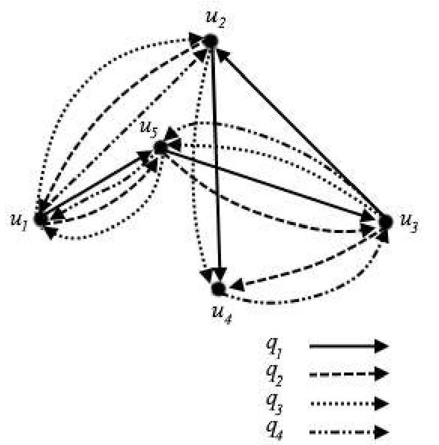

Fig. 3. Visualization of intelligent mobile objects movement route.

As a result of the method performance, time matrices for intelligent mobile objects (Table III to Table VI) were obtained, and movement paths for objects were constructed (Fig. 3).

Table VII shows the state of the points when servicing intelligent mobile objects, forming queues for servicing. The table shows that the total service time of all points took 27 s.

TABLE VII: STATUS OF POINTS

System operation time, (s)	Status of point									
	u_1		u_2		u_3		u_4		u_5	
	t	q	t	q	t	q	t	q	t	q
1	2	1	3	2	1	3	4	4	0	
2	1	1	2	2	0		3	4	0	
3	0		1	2	0		2	4	0	
4	0		0		0		1	4	1	3
5	2	2	0		0		0		1	1
6	1	2	0		0		0		0	
7	2	3	0		1	4	0		0	
8	1	3	0		1	1	0		1	2
9	0		0		1	2	0		0	
10	0		3	3	0		0		1	4
11	0		2	3	-		0		-	
12	0		1	3	-		0		-	
13	2	4	3	1	-		4	2	-	
14	1	4	2	1	-		3	2	-	
15	0		1	1	-		2	2	-	
16	-		0		-		1	2	-	
17	-		3	4	-		0		-	
18	-		2	4	-		0		-	
19	-		1	4	-		4	3	-	
20	-		0		-		3	3	-	
21	-		-		-		2	3	-	
22	-		-		-		1	3	-	
23	-		-		-		4	1	-	
24	-		-		-		3	1	-	
25	-		-		-		2	1	-	
26	-		-		-		1	1	-	
27	-		-		-		0		-	

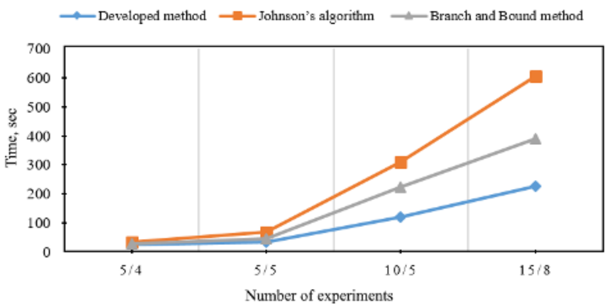

Fig. 4. The results of experiments

Experiments were carried out. As alternative methods for solving the traveling salesman problem, software realizations of Johnson's algorithm [7], [19] and the branch and bound method [20], [21] were used. The results are shown in Fig. 4.

VII. CONCLUSION

As a result of the review and analysis carried out, a modified traveling salesman problem is formulated for the case when a group of intelligent mobile objects is simultaneously involved. The goal of minimizing the travel time each intelligent mobile object takes to pass the route through all points was achieved.

The novelty of the work lies in the fact that the quest method has been further developed, which, in contrast to the classical solutions of the traveling salesman problem, allows to achieve the target functional at a minimum time due to local optimization of the directions of intelligent mobile objects' movement through points.

As further research, it is proposed to consider a modification of the point system with a dynamically variable number of points. Some issues that deserve attention are related to the optimization of control of data transfer processes [22], reducing redundancy and improving the reliability of information delivery between point systems, asymmetric weights of arcs between points. In addition, it is necessary to conduct real experiments, for example, in tasks improving the energy efficiency of large data centers [23]. [10], [24]

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

All authors conducted the research; Vitalii Tkachov and Volodymyr Tokariev wrote the paper; Iryna Ilina and Stanislav Partyka conducted an experiment; all authors had approved the final version.

REFERENCES

[1] L. A. Wolsey and G. L. Nemhauser, *Integer and Combinatorial Optimization*, 1st ed., USA: John Wiley & Sons Inc., 1999, pp. 50-55.

[2] L. Wanga, Y. Ma, J. Yan, V. Chang, and A. Y. Zomaya, "pipsCloud: High performance cloud computing for remote sensing big data management and processing," *Future Generation Computer Systems*, vol. 78, no. 1, pp. 353-368, January 2018.

[3] S. Zhang, C. Zhang, and Q. Yang, "Data preparation for data mining," *Applied Artificial Intelligence*, vol. 17, no. 5-6, pp. 375-381, November 2003.

[4] S. Bielievtsov, I. Ruban, K. Smelyakov, and D. Sumtsov, "Network technology for transmission of visual information," in *Proc. XVIII International Scientific and Practical Conference "Information Technologies and Security*, Kyiv, Ukraine, November 27, 2018, pp. 160-175.

[5] V. Tkachov, V. Tokariev, Y. Dukh, and V. Volotka, "Method of data collection in wireless sensor networks using flying ad hoc network," in *Proc. 5th Int. Scientific-Practical Conf. Problems of Infocommunications. Science and Technology*, Kharkiv, Ukraine, October 9-12, 2018, pp. 197-201.

[6] V. Tkachov and V. Savanevych, "Method for transfer of data with intermediate storage," in *Proc. First Int. Scientific-Practical Conf. Problems of Infocommunications Science and Technology*, Kharkov, 2014, pp. 105-106.

[7] O. Ulyanov, M. Plakhov, A. Shevtsova, O. Ulyanova, A. A. Skoryk, and V. Tkachev, "Testing the speed of the FFT using the NVIDIA graphic cards," in *Proc. International Young Scientists Forum on Applied Physics*, Dnipropetrovsk, Ukraine, 2015, p. 1.

[8] N. Deo, *Graph Theory with Applications to Engineering and Computer Science*, Mineola, New York, USA: Courier Dover Publications, 2017, pp. 112-137.

[9] M. Foumani, A. Moeini, M. Haythorpe, and K. Smith-Miles, "A cross-entropy method for optimising robotic automated storage and retrieval systems," *International Journal of Production Research*, vol. 285, no. 2, pp. 444–457, 2020.

[10] P. Baniasadi, M. Foumani, K. Smith-Miles, and V. Ejov, "A transformation technique for the clustered generalized traveling salesman problem with applications to logistics," *European Journal of Operational Research*, vol. 285, no. 2, pp. 444–457, 2020

[11] J. Li, X. Meng, and X. Dai, "Collision-free scheduling of multi-bridge machining systems: a colored traveling salesman problem-based approach," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 139-147, January 2018.

[12] C. Semeria, *Supporting Differentiated Service Classes: Queue Scheduling Disciplines*, Sunnyvale, CA 94089, USA: Juniper Networks, 2001, pp. 11-14.

[13] L. Engebretsen, “Simplified tight analysis of Johnson's algorithm,” *Information Processing Letters*, vol. 92, no. 4, pp. 207-210, November 2004.

[14] T. A. Tarray and M. R. Bhat, “A nonlinear programming problem using branch and bound method,” *Investigación Operacional*, vol. 38, no. 3, pp. 291-298, January 2018.

[15] U. Zwick, “The smallest networks on which the Ford–Fulkerson maximum flow procedure may fail to terminate,” *Theoretical Computer Science*, vol. 148, no. 1, pp. 165-170, August 1995.

[16] J. Edmonds and R. M. Karp, “Theoretical improvements in algorithmic efficiency for network flow problems,” *Journal of the ACM*, vol. 19, no. 2, pp. 248-264, April 1972.

[17] S. D. Pogorilyy and M. S. Slynko, “Research and development of Johnson's algorithm parallel schemes in GPGPU technology,” *Problems in Programming*, vol. 2-3, pp. 105-112, January 2016.

[18] V. Tkachov and M. Hunko, “Quest method for organizing cloud processing of airborne laser scanning data,” in *Proc. IEEE 8th Int. Conf. on Advanced Optoelectronics and Lasers*, Sozopol, Bulgaria, 2019, pp. 565-569.

[19] B. Borchers and J. E. Mitchell, “An improved branch and bound algorithm for mixed integer nonlinear programs,” *Computers & Operations Research*, vol. 21, no. 4, pp. 359-367, April 1994.

[20] T. Filimonchuk, M. Volk, I. Ruban, and V. Tkachov, “Development of information technology of tasks distribution for grid-systems using the grass simulation environment,” *Eastern European Journal of Enterprise Technologies*, vol. 3, no. 9, pp. 43-53, 2016.

[21] O. Lemeshko and O. Yeremenko, “Enhanced method of fast re-routing with load balancing in software-defined networks,” *Journal of Electrical Engineering*, vol. 68, no. 6, pp. 444-454, 2017

[22] O. Lemeshko, O. Yeremenko, and N. Tariki, “Solution for the default gateway protection within fault-tolerant routing in an IP network,” *International Journal of Electrical and Computer Engineering Systems*, vol. 8, no. 1, pp. 19-26, June 2017.

[23] G. Kuchuk, A. Kovalenko, I. E. Komari, A. Svyrydov, and V. Kharchenko, “Improving big data centers energy efficiency: Traffic based model and method,” in *Green IT Engineering: Social, Business and Industrial Applications. Studies in Systems, Decision and Control*, Springer, Cham, 2019, vol. 171, pp. 161-183.

[24] A. Shamraev, E. Shamraeva, A. Dovbnya, A. Kovalenko, and O. Ilyunin, “Green Microcontrollers in Control Systems for Magnetic Elements of Linear Electron Accelerators,” in *Green IT Engineering: Concepts, Models, Complex Systems Architectures*, V. Kharchenko, Y. Kondratenko, and J. Kacprzyk, Ed. Springer International Publishing Switzerland, 2017, pp. 283-305.

Vitalii Tkachov received B.Sc. degree in telecommunications from the Kharkiv National University of Radio Electronics, Ukraine in 2010; M.Sc. degree in telecommunications, Kharkiv National University of Radio Electronics, Ukraine, 2011; and Ph.D. degree from the Kharkiv National University of Radio Electronics, Ukraine in 2015. He is a member of IEEE Ukraine Section. Currently, he is employed as a Senior Lecturer of the Department of Electronic Computers, Kharkiv National University of Radio Electronics, Ukraine. Research interests: telecommunications technology, information technology.

Volodymyr Tokariev received Specialist in engineering and design of radio equipment from the Kharkiv National University of Radio Electronics, Ukraine in 1988; Ph.D. degree from the Kharkiv National University of Radio Electronics, Ukraine in 2001. He is a member of IEEE Ukraine Section. Currently, he is employed as an Associate Professor of the Department of Electronic Computers, Kharkiv National University of Radio Electronics, Ukraine. Head of the educational and scientific laboratory “Reconfigurable and mobile systems”. Research interests: information technology.

Iryna Ilina received Specialist in Machines and technology of foundry production from the National Technical University “Kharkiv Polytechnic Institute”, Ukraine in 1994; Ph.D. degree from the the National Technical University “Kharkiv Polytechnic Institute”, Ukraine in 2010. She is employed as an Associate Professor of the Department of Electronic Computers, Kharkiv National University of Radio Electronics, Ukraine. Research interests: information technology, computer engineering.

Stanislav Partyka received M.Sc. degree in computer engineering from the Kharkiv National University of Radio Electronics, Ukraine in 2010; M.Sc. degree in telecommunications, Kharkiv National University of Radio Electronics, Ukraine, 2003. He is employed as a Senior Lecturer of the Department of Electronic Computers, Kharkiv National University of Radio Electronics, Ukraine. Research interests: information technology, computer engineering.