doi:10.22306/al.v6i3.123

Received: 09 May 2019
Accepted: 15 Aug. 2019

OPTIMIZING LOGISTICS ROUTES IN THE FIELD OF MAINTENANCE

Robert Frischer
VŠB-Technical University Ostrava, 17. listopadu 15, Ostrava, Czech Republic, EU, robert.frischer@vsb.cz (corresponding author)
Miloš Pollak
VŠB-Technical University Ostrava, 17. listopadu 15, Ostrava, Czech Republic, EU, milos.pollak@vsb.cz
Zora Koštialová Jančíková
VŠB-Technical University Ostrava, 17. listopadu 15, Ostrava, Czech Republic, EU, zora.jancikova@vsb.cz

Keywords: logistics, maintenance, optimization, Traveling Salesman Problem

Abstract: In the event that there is a requirement to provide maintenance with a service intervention somewhere in the area, it is necessary to select a service team which is optimal for the given task for all the assessed aspects. In this article, tests and methods based on TSP principles have been developed that are able to optimize route selection between key points. Due to the complexity of algorithm design, two approaches have been validated. The "Brute Force" method, which can provide information on the choice of the optimal route according to the specified parameters, and the "Nearest Neighbour" method, which is able to quickly calculate a large group of intermediate points, but only provides sufficient results. The aim was to create a methodology, guidance, and direction in evaluating route selection. That is why two basically opposed methods of TSP solution have been chosen, modified by the authors into a form suitable for implementation on virtually any platform. Although the design and program implementation was based on the MATLAB platform, no proprietary functions and libraries are used and the entire software design is implemented without the need for their use.

1 Introduction

Distribution logistics is an important part of logistics chain, as it enables the final distribution of products to a large number of end customers and the provision of related services.

The term distribution logistics refers to the circulatory processes provided by the enterprise. Distribution logistics is responsible for the location and determination of the type, equipment and number of warehouses, type of assortment, quantity of supplies, packaging of products, organization of transportation, etc. Last but not least, the determination of the logistics unit. The logistics unit is the largest amount of transportable goods that can be further subdivided. It plays an important role in international and domestic transport. Its unification achieves savings and accelerates the movement of goods.

The goal of distribution logistics is to ensure that the product is properly placed on the market in the best possible way, with optimal costs. Therefore, distribution logistics should only address a certain segment of a particular circulatory process, as one of the components of marketing logistics. The so-called distribution policy serves her. We can define it as a set, the process of all decisions that must be made in relation to a product or performance path from the manufacturer to the final consumer or processor. Thus, the role of distribution policy is to decide on how to create optimal relationships between production and consumption, as well as optimal distribution links.

2 Logistics in providing maintenance and service

The issue of distribution logistics must also be addressed in the area of maintenance and service processes. In particular, the following factors affect the organizational forms of maintenance and repair activities [1]:

- the number of independent manufacturing companies and firms and their size,
- territorial distributed units,
- the number of installed production facilities and their structure,
- the technical complexity of the equipment and the resulting complexity of technological, technical and economic preparation of maintenance and repair operations,
- the complexity and sophistication of operational control of maintenance and repairs.

Although practice very often seeks to define and subsequently apply some type of organizational solution - such a general and universal solution does not exist nor would such an organizational scheme be useful. There is no general system that could be used by small, medium or large enterprises, as well as by enterprises with advanced management or with completely underdeveloped management, or enterprises with already built technical repair facilities, or enterprises with incomparably lower equipment, because everyone it has its specifics both from a technical and technological point of view, and of course it has an influence on the system for maintenance and repair activities [2].

Copyright © Acta Logistica, www.actalogistica.eu

In order to effectively control multiple service groups on a large area, it is always necessary to choose which group will intervene. This decision depends on several conditions that need to be considered.

The most important decision factors are:

- Distance of service team from service area.
- Time of arrival to the place of service.
- Distance of service team from parent service centre after arrival to service area.
- Priority of service intervention.

If there are more service groups in the terrain or if there are several service centres in the area (Figure 1), the actual distance of the service team from the service intervention decides on the exit.

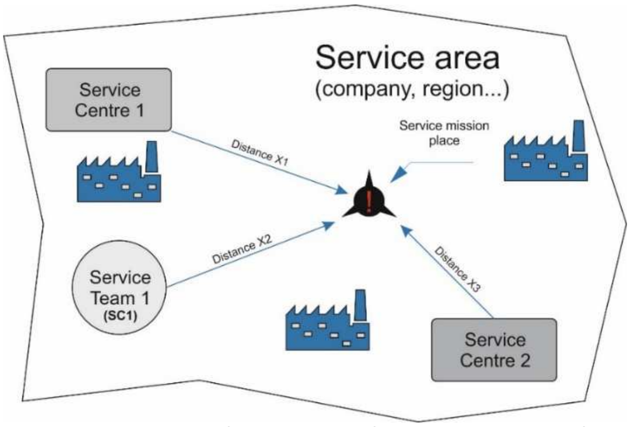

Figure 1 Evaluation of the distance of the intervention site from the current service team position

Of course, this value is important, but it says nothing about the real distance that a service team will offend. The parameters, that defines arrive time, must also take into account the route, that service team travel distance. This step has proved to be a very complex problem with a highly developed scientific background. In short, the issue of the so-called "sales representative" and the methods of solution is a large quantity [3].

It is also necessary to take into account that if the closest team is chosen as a suitable team, it can reach such a distance from its service centre that the total distance travelled is no longer profitable [4,5].

This is also related to the priority of the intervention (Figure 2), which will be crucial in evaluating the selection of a service team. Priority may be the urgency of the service action, the quantified loss on equipment, the quantified loss counting the entire logistics chain, etc. [6].

During maintenance management it is necessary to minimize the logistic delay of individual downtime and therefore it is also necessary to pay attention to logistics issues in the field of maintenance. For preventive maintenance, maintenance personnel must pay attention on various sites located in different parts of the enterprise. This requires planning the optimal path to ensure efficient use of service time. In the case of corrective maintenance, when multiple failures are reported in a single time period, the service team must decide on the sequence of restoring the failures to service. One of the partial factors of this optimization task is the magnitude of the logistical delay caused by transport to the place of intervention. To optimize routes, it is possible to take advantage of the traveling salesman problem. This method is a key parameter in minimizing the logistic delay, which significantly affects the total length of time the objects will be restored after a failure. This is positively reflected in the economy of the company [7].

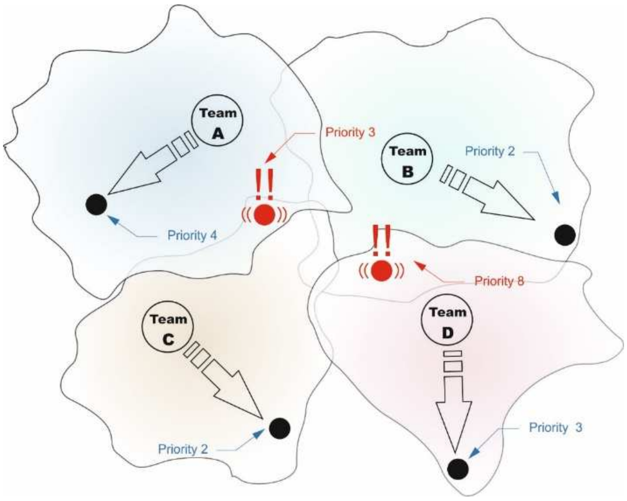

Figure 2 Prioritize service events

3 Traveling Salesman Problem

The issue of finding the optimal route is not just a domain of logistics. Traveling Salesman Problem (TSP) is very similar to the issue addressed by routing protocols in data networks. For low number of intermediate points, it is possible to apply simple algorithms [8-10], which are not very effective, but are fast, computational and low-memory and will fulfil their task. If you need to find out the optimal route, for example, using the "brute force" method, you will have to take into account the very high memory requirements, as well as the high CPU load. The "brute force" method passes through every possible option and selects from it the best parameters [11].

The TSP problem is, to say the least, the assignment that defines that business travellers must promote products in n cities (including the city where they live). After visiting each city (each city can be visited only once) it will return to the default city of departure (option of return with return to starting position). Suppose there is one way to connect each two cities. So what's the best way, with a minimum journey time? It has been shown that TSP is generally a complex problem, i.e. finding a time polynomial algorithm to achieve an optimal solution. TSP is easy to interpret, but it is very difficult to solve. This problem has been arousing the interest of many scientists for a long time, although it was presented in 1932.

Copyright © Acta Logistica, www.actalogistica.eu

However, so far no effective solution has been found. Although TSP is only the shortest circuit problem, there are many real TSP related problems in real life [12-14]. For a better understanding of the issue, possible examples will be explained in the following.

Example 1. - Mail route. Mail delivery is very closely related to the TSP issue. Suppose a mail car needs to collect mail in n locations. Under these circumstances, you can view the route using a drawing containing n + 1 nodes. One node means the post office, from where this postal car leaves and returns. The remaining n nodes are nodes that need to collect shipments. The aim is to find the shortest route while neglecting to optimize the route, taking into account the time lag, fuel consumption, or the need to pay tolls [15,16].

Example 2. - Mechanical Arm. When the mechanical arm is used to secure the nuts and bolts on the assembly line, this mechanical arm will move from its initial position (position where it is necessary to fasten the first nut) to each nut in the correct order and then return to its starting position. The path that moves the mechanical arm is the path that contains the nodes - individual assembly keys. The most economical route will allow the mechanical arm to complete its work in the shortest possible time.

Example 3. - Analog circuit of an integrated circuit. Often, thousands of separate electronic components need to be inserted during integrated circuit manufacturing. When switching from one electronic element to another, some energy is consumed. So how is it possible to arrange a production order for minimum energy consumption? This is also a solution for TSP.

In addition to the examples above, the issues associated with finding the shortest route are linked to issues such as the layout of the transport network, the choice of a walking path, the laying of the pipeline needed for city planning, and engineering structures, etc. Because finding a solution for TSP plays an important role in real life, much attention is being paid to this issue [17-19].

The mathematical description of problems TSP

$$min \sum d_{ij} \cdot x_{ij} \tag{1}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad i = 1,2,\cdots n \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad j = 1,2,\cdots n \tag{3}$$

$$\sum_{i,j\in S} x_{i,j} \leq |S| - 1 \quad 2 \leq |S| \leq n-2, S \subset \{1,2,\cdots n\} \tag{4}$$

$$x_{ij} \in \{0,1\} \quad i,j = 1,2,\cdots n \quad i \neq j \tag{5}$$

where d_{ij} is the distance between the city i and the city j, the decision variable x_{ij} = 1 is the route made by the business traveller (including the route from the city i and the city j), x_{ij} = 0 is the route not made by the passenger. | S | means the number of elements contained in the S set (1).

The objective function (1) defines the total minimum distance, z (2) shows that a business traveller can only leave the city once, (3) defines that a business traveller can enter the city only once. It can be seen from (2) and (3) that business travellers visit each city once, but do not exclude the possibility of any loop. Definition (4) specifies that a vendor should not create any loop in any subset of cities [20].

4 Evolutionary algorithm for TSP solution

As mentioned above, traditional algorithms used to solve TSPs have some significant limitations. With the advent of evolutionary algorithms, many new numerical optimization algorithms appear [21]. These algorithms are to some extent random search algorithms. ACA and PSO are typical parallel algorithms, although they cannot guarantee that they will obtain an optimal solution in limited time, they can provide a satisfactory solution within an acceptable time frame. In order to determine the effect of the solution for the TSP obtained by the optimization algorithm, it is necessary to consider the search capabilities of the search algorithm. For example, a strong optimization algorithm will have a better effect, while an algorithm that easily gets stuck in a local extreme often helps to obtain an optimal solution for a TSP of a local nature. An example of an evolutionary algorithm for solving TSPs is the Ant Antony Algorithm.

Ant Colony Algorithm (ACA) is a relatively new evolutionary algorithm presented by the Italian scientist Dorigo. He called it the "ant system" and achieved a relatively good experimental result with it. As for ACA, n represents the number of cities for TSP, M represents the number of ants in the colony, d_{ij} (i, j = 1,2,…, n) represents the distance between the city and the city *j*, τ_{ij} (t) represents the pheromone concentration on the route between cities *i* and *j* in time *t*. The initial condition at time *t* = 0 is the concentration of pheromone on each route equal, respectively τ_{ij} (0) = C, where C is a constant. During movement in the colony, the ant (*k* = 1.2,…, *m*) determines which direction to follow, according to the pheromone concentration on each route. P_{ij}^{k} *(t)* represents the probability of movement (choice) of ant from city *i* to city *j* at time *t* (6).

$$P_{ij}^{k}(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{s\epsilon allow_k} \tau_{is}^{\alpha}(t)\eta_{is}^{\beta}(t)} & j \notin prohib_k \\ 0 & \end{cases} \tag{6}$$

where *prohib*$_k$ (*k* = 1.2,… *m*) means that the ant has passed through the given city, with only one first at the beginning (default city). During the algorithm, the length of the *prohib*$_k$ variable increases gradually. The term allow *k* = {1,2,…, *n*} - *prohib*$_k$ defines another city, the direction

Copyright © Acta Logistica, www.actalogistica.eu

that an ants are allowed to. The quantity η_{ij} represents visibility and is derived from the length of a given route between two cities *i* and *j*. The variables α and β affect the pheromone concentration τ and visibility η.

Over time, pheromones on each route lose their intensity. The ACA algorithm uses the principle of positive feedback, which accelerates the evolutionary process and also allows the native use of parallelization calculations.

The ongoing process of information exchange and communication between individuals helps to find a better, more effective solution. It is easy to focus on the local extreme when there is only one individual. Colony collaboration, however, allows several subsets of the solution space to be obtained that provide a better environment to implement another solution. The movement of individual individuals in the colony is random. In fact, the measures taken to exclude the possibility of local extreme occurrence slow down the speed of convergence. When the size of an anthill colony grows, it will take longer to find the best route. The principle of an ant colony is very popular today and many scientific papers are devoted to it. Many improvements and optimizations are available. However, it cannot be said that this algorithm was in the final stage of optimization. On the contrary, it seems that it is very suitable for the solution of TSP and is only at the very beginning of its journey [22].

5 Design of optimization method based on TSP principles

5.1 Brute Force Method

In principle, it is a method that can check all variations of routes and calculate their suitability index, resp. price, the English term "Cost". Each selected route has its own specific price. This is an analogy of data networks, where a route with the lowest cost is selected - cost, cost. This criterion was chosen as appropriate and therefore the cost of each route will be calculated. Both presented methods allow for a variable number of specific properties, parameters for each route, so there is no limitation in their number. However, the essential condition is independence from the direction of the selected route. The route price is therefore the same in both directions. This limiting condition is introduced for purely objective reasons, in a significant simplification of the algorithm. Directional constraint is very specific and not very utilized. Multiple algorithms can be created to calculate the path price. The current time, the dynamic path utilization, etc. can play a role. The cost of the trip in this case can be calculated, for example, by the relation (7).

$$c_i^{a\to b} = \sum_{k=1}^{n} w_k \cdot \sigma_k \qquad (7)$$

where c_i is the price of the route between two points *a* and *b*, w_k the weight of the specific route parameter *i* and σ_k determines the value of the specific route parameter *i*. The specific property, parameter such as route difficulty, static load, toll etc. is multiplied by the appropriate weight and the sum of all parameters determines route price. It should be borne in mind that, for example, a faster route will show a lower price because the lowest cost of the route is the criterion of optimality. Only the length of the route is taken into account within the algorithm, other features and weights have been omitted for reasons of debugging and clarity. The route length was calculated in two-dimensional space according to formula (8).

$$\begin{gathered} l_i^{a\to b} = \sqrt[2]{dX + dY} \\ dX = (a_x - b_x)^2;\; dY = \left(a_y - b_y\right)^2 \\ c_i^{(a,b,\dots,n)} = \sum_1^n l_i \end{gathered} \qquad (8)$$

where $l_i^{a\to b}$ corresponds to the length of the route between points *a* and *b*, dX and *dY* are the distances between points in the *X* and *Y* axes, a_x, b_x, and *y*, would be the coordinates of the given axis for that point. Then the total length of the route ci between points *a* to *n* is the sum of the partial lengths l_i.

The generated algorithm generates key points in the desktop automatically and randomly for testing purposes. Fixed points can generate overly optimistic results, and the algorithm could gravitate to optimize for a given point array in the face. Thus, the random distribution of key points helps to test the developed algorithm. The number of parameters of each route is defined and the area size in which the optimal route is sought is determined. These values correspond to the size of the business, the area where the service teams are located.

Furthermore, the number of route points is defined, which significantly influences the solution time and the optimum route.

Subsequently, the coordinates of the individual through points of the route, which are in the range ⟨0; dimX⟩ and ⟨0; dimY⟩. The result is a three-dimensional matrix with the depth of parameter + 1 and the rank of dimX, respectively dimY.

The following is the *"%% Start and End Point Definition*" section, which aims to determine the starting and ending points of the route. No extra point is generated, both are selected from existing points.

In real-time deployment, the starting position would be the current location of the service team and the end point instead of service intervention. In this case, a random generator of both points is created.

A very important step is to create a matrix of all points (cities) on the map.

In fact, the map_1 variable represents the entire area at a given resolution, and only in places where there are through points is the cell value set to 1. Individual points, including coordinates, are stored in trace_1. The procedure that passes all points in the matrix map_1 and stores them in the new variable trace_1 is named *"%% Saving all* *waypoints to trace_1 without optimization*". Individual points are stored in the matrix in the order in which the procedure algorithm encounters them. The order of points is not optimized.

The algorithm for finding the optimal route itself is in the section called *"%% Finding the Shortest Way - BruteForce Method*". The aim of the method is to find the optimal route by analysing each individual route combination. The condition is that the individual waypoints are not duplicated and the route passes from the starting point to the end point. The computing complexity is $\frac{2^{n-2}}{2}$ operations, where n is the total number of through points. By simple calculation it is possible to find out that in case of 4 points of the route 2 combinations of points in the route will result. Two points are always fixed, that is, the starting and ending points.

There is a lot of material describing the issue of combinatory such as here [23,24]. However, the combination creation algorithm is not directly listed. The most complicated part of the "Brute Force" method was to create a complete combination matrix where each line would represent one unique route from the starting point to the end point. To this end, a binary math based combination matrix method has been developed. The method is not optimized, it can create duplicate routes, but it does not change the result. The principle is based on the binary notation of the sequence of numbers presented in on the left.

It is apparent from the binary notation that the ones and zeros create a pure combination structure. Thus, the method of the combination matrix method was that if the individual positions in binary expression were replaced by individual through points, it would be possible to effectively create their combination matrix. If we introduce binary expressions of numbers in the range $\langle 0, n \rangle$ as a matrix (e.g. matrix A), individual cells at the level of weights represent through points [25]. If a value of 1 is entered in matrix A at weight level 1, the through point is shifted one position to the left to be exchanged with its neighbour. Basically, the "swap" function is performed on the adjacent two cells of matrix A. In this way, the individual through points are "bubbled" to the right places.

If the cell value of matrix A at the highest weight level is equal to 1, the outer cells are exchanged (right at Figure 3).

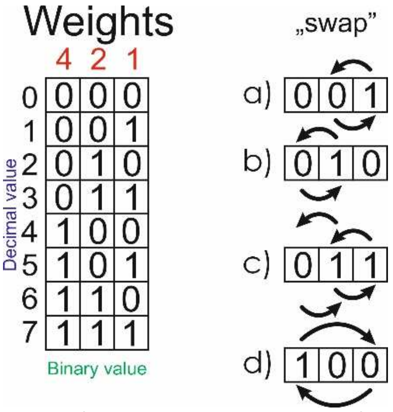

Figure 3 Display of decimal numbers in binary form and route point migration method

An example of the algorithm is shown in Figure 4, the coordinates of the through points are shown in Table 1 and an example of the combination matrix of the *x* values of the through points in Table 2.

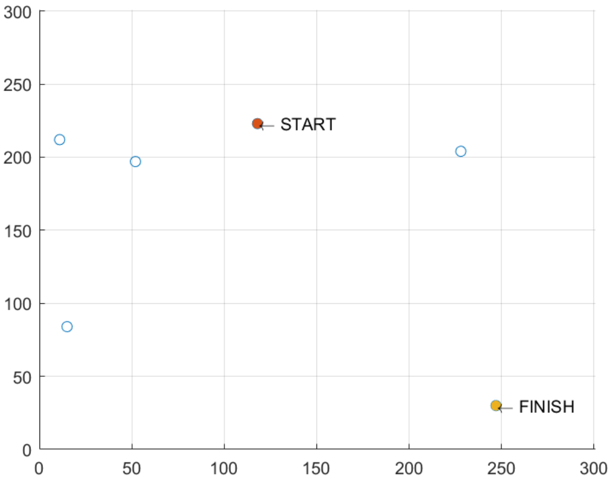

Figure 4 An example of the placement of 4 through points with the starting and ending points

Table 1 Coordinate of through points

Point order	Y-coordinate	X-coordinate
1.	10	5
2.	5	9
3.	8	2
4.	2	10

Table 2 Combination of x-coordinate of through points

A combination of X-coordinate points			
10	5	8	2
10	8	5	2
10	5	8	2
10	5	8	2

Copyright © Acta Logistica, www.actalogistica.eu

The creation of the combinatorial matrix is dealt with in the program section called "%% Finding the Shortest Way - BruteForce Method".

The output of this part of the program is the variable var_mapx and var_mapy, where all combinations of *x* and *y* coordinates of through points are stored.

The following is the calculation of the length of the individual routes in the section marked with the heading "%% Calculate the length of individual routes". The lengths of each route are calculated according to formula (8). Path lengths are stored in the trace_2 variable unchanged.

The following is the shortest route ("%% Finding the shortest path") by a simple algorithm by browsing the individual path length records. The output of the algorithm is *mnm* and *mxm*. Examples of output routes when using the "Brute Force" method are see in the Figure 5.
Computational testing also took place as part of the verification. See Figure 6 for details. This method is suitable for a low number of through points. With the number of points, computational complexity increases, and from a certain threshold, the method on conventional machines is unusable. Its main advantage is finding the optimal route, according to the given criteria.

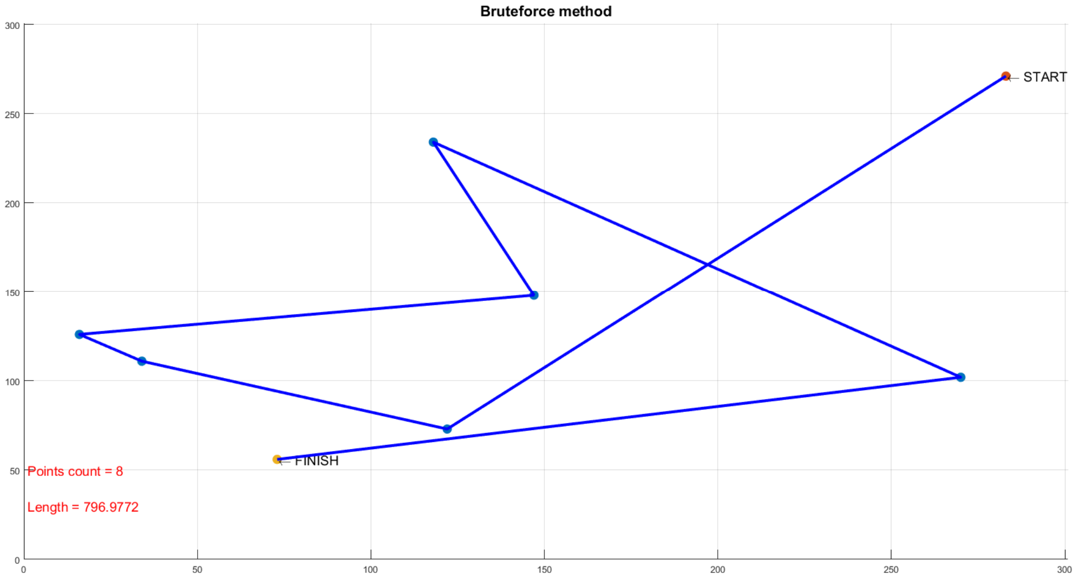

Figure 5 Example of output routes when using the "Brute Force"

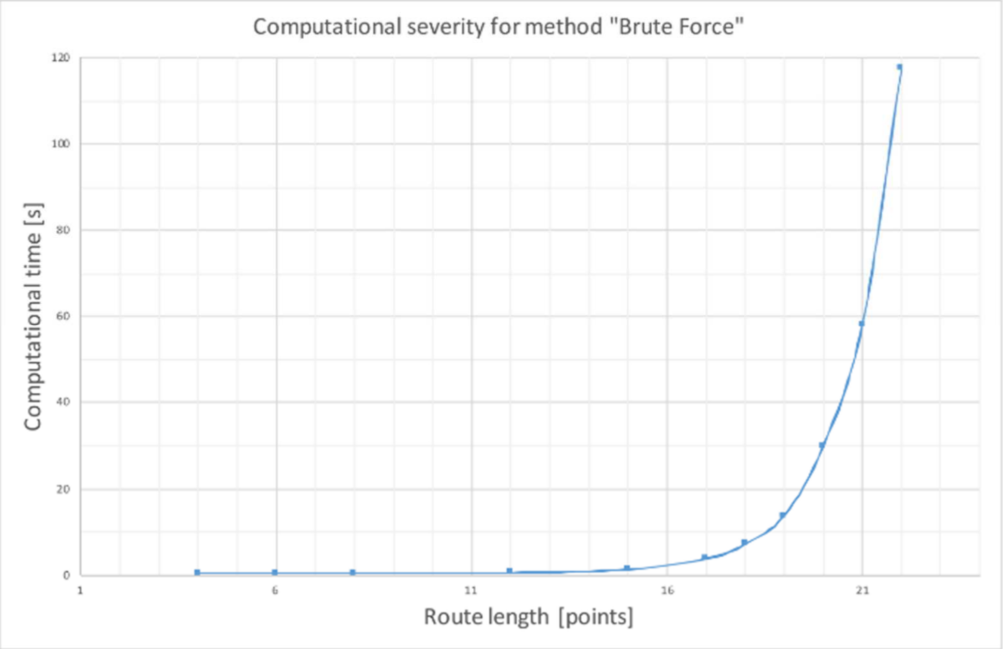

Figure 6 The computational complexity of the method "Brute Force"

Copyright © Acta Logistica, www.actalogistica.eu

5.2 Nearest Neighbour Method

Due to the limitations of the "Brute Force" method, it was also necessary to verify one of the methods, which, although not leading to an optimal result, will be applicable within certain compromises. One of these methods is the method called "Nearest Neighbour", respectively nearest neighbour. This method has been elaborated and modified for use by the author of this thesis to find the optimal path for the service team.

Again, as with the previous method, two endpoints, default and endpoint, are selected. The method is based on the assumption that the ideal route choice will be through the nearest next point.

The initial setting is the same as the previous method. A map of the area with through points is created. The area size is defined by the dimX and dimY variables. If there are not many through points (<6), they will be printed on the MATLAB application window. Following is the definition of the start and end points, the start and fin variables, then a simple matrix of all through points (trace_2) is created as they are stored, without optimization. Up to now, only the input data has been edited into an appropriate format for analysis. The core of the algorithm is the partition named *"%% Nearest Neighbour method*". In the cycle *"% Calculate the length between the waypoints*", the distance from the starting point to all the other points according to formula (8) is calculated. Only a simple distance between points is considered. Additional parameters are neglected. If these parameters are available, the calculation according to formula (7) and (8) is used. The nearest point (the *"% found shortest route*") to the starting point is marked as the starting point after the cycle is completed and the process is repeated. The output of this part of the algorithm is the newly arranged trace_3 matrix, where the route is already optimized. The following is the calculation of the total length of the route in the section called '% compute total route length' [26]. On Figure 7 are examples of output routes when using the "Nearest Neighbour" method.
Computational tests were performed as part of the verification. See Figure 8 for details. This method is suitable for both low and high throughput points. The number of points increases the computational complexity of the method linearly. Its main disadvantage is only approximate optimization of the route, according to the specified criteria.

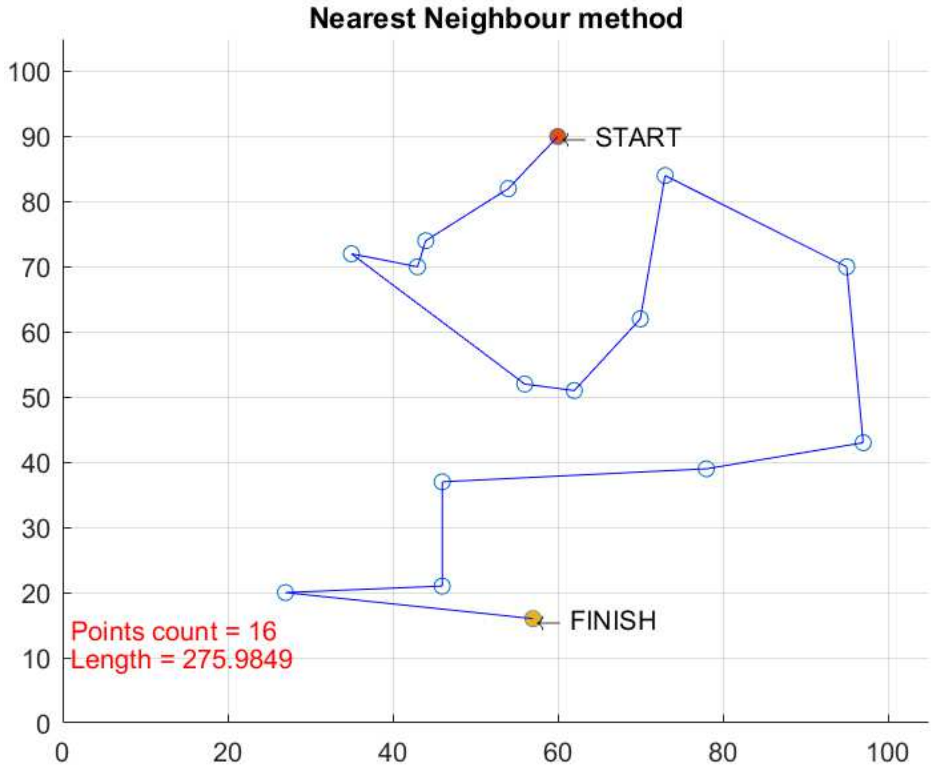

Figure 7 Example of output routes when using the "Nearest Neighbour"

Copyright © Acta Logistica, www.actalogistica.eu

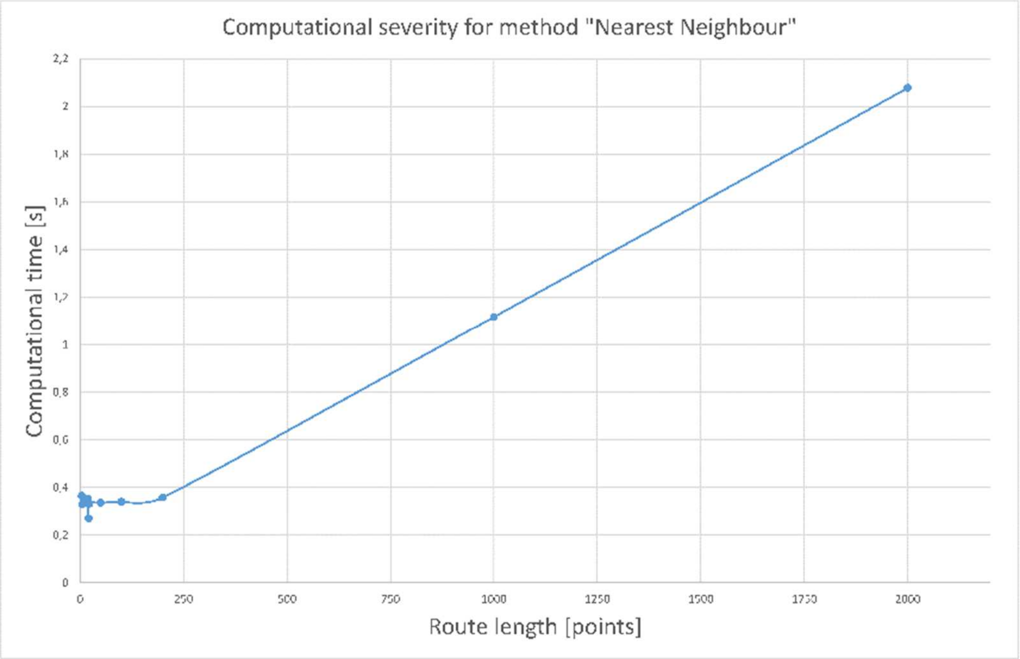

Figure 8 The computational complexity of the method Nearest Neighbour

6 Conclusion

In the paper, two approaches to the optimal route using TSP were presented. Two solutions were chosen, the "Brute Force" method and the "Nearest Neighbour" method. The "Brute Force" method provides better results from the optimal route selection, but is only suitable for low through points. The algorithm created in MATLAB was able to optimize the route with a maximum of 22 through points in reasonable time and with enough memory. The output of the algorithm is the optimal route, determined according to the parameters of individual routes between neighbouring points.

Although the "Nearest Neighbour" method does not provide the optimum route from the starting point to the end point compared to the "Brute Force" method, its computational complexity is of the order of magnitude and is also capable of handling thousands of waypoints.

As you can see from the pictures, the "Nearest Neighbour" method gives a good or a good comparable results with the Brute Force method for a low number of through points. A higher number of points shows insufficient optimization of the algorithm and a temporary jam in the local extreme.

It is important to realize that the data used was of random nature within objectivity and it is likely that the results of the Nearest Neighbour will be better when using real position data.

The aim is to transform the localization data into a simpler form, to create a time link to the location data and, in particular, to optimize the intervention of the service teams according to their current location and priority of the intervention. The output is a system of original algorithms and procedures that can be part of the maintenance process optimization. The developed methodology can be deployed in industrial and non-industrial processes. It can be assumed that good maintenance management with good work management can significantly reduce the likelihood of production equipment failures and prevent additional costs.

The data processing algorithms created are inherently unique, even though they are based on the previously described theories. Both presented optimization methods are adapted for the needs in the field of service team management (or in other areas where intervention is needed at given time with given priority). The methods are optimized (from the program code point of view) with regard to current needs and computational complexity, which is the main scientific contribution.

Acknowledgement

The work was supported by the specific university research of Ministry of Education, Youth and Sports of the Czech Republic No.: SP 2019/17 and SP 2019/62.

References

[1] SAKURAI, Y., TSURUTA, S.: *A Multi-World Intelligent Genetic Algorithm to Optimize Delivery Problem with Interactive-Time*, Croatia, InTech, 2010. doi:10.5772/12872

[2] PEMMARAJU, S.: *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, Cambridge, Cambridge University Press, 2009.

[3] CONWAY, J. H., GUY, R. K.: *The Book of Numbers*, New York, Springer-Verlag, 1996.

Copyright © Acta Logistica, www.actalogistica.eu

[4] MUIR, T.: *Treatise on the Theory of Determinants*, New York, Nabu Press, 1960.

[5] GELFAND, S. I., GERVER, M. L., KIRILLOV, A. A., KONSTANTINOV, N. N. *Sequences, Combinations, Limits*, Dover, Dover Publications, 2002.

[6] GREGOR, M., HODON, R., BINASOVA, V., DULINA, L., GASO, M.: Design of simulation-emulation logistics system, *MM Science Journal*, Vol. 2018, No. October, pp. 2498-2502, 2018. doi:10.17973/MMSJ.2018_10_201878

[7] FRISCHER, R., POLLAK, M., SVEC, P., TUHY, T.: *Metallurgical systems control with localization instrumentation support,* METAL 2016: 25th International Conference on Metallurgy and Materials, pp. 1776-1781, 2016.

[8] MATAI, R., SINGH, S., MITTAL, M.: *Traveling Salesman Problem: An Overview of Applications, Formulations and Solution Approaches*, Croatia, InTech, 2010. doi:10.5772/12909

[9] YONG-HYUN, Ch.: *An Efficient Solving the Travelling Salesman Problem: Global Optimization of Neural Networks by Using Hybrid Method*, Croatia, InTech, 2010. doi:10.5772/12855

[10] DIABY, M.: *Linear Programming Formulation of the Multi-Depot Multiple Traveling Salesman Problem with Differentiated Travel Costs*, Croatia, InTech, 2010. doi:10.5772/12927

[11] CHAUDHURI, A., KAJAL, D.: *A Study of Traveling Salesman Problem Using Fuzzy Self Organizing Map*, Croatia, InTech, 2010. doi:10.5772/13270

[12] DAVENDRA, D., ZELINKA, I., SENKERIK, R., BIALIC-DAVENDRA, B.: *Chaos Driven Evolutionary Algorithm for the Traveling Salesman Problem*, Croatia, InTech, 2010. doi:10.5772/13107

[13] YUTAO, F., L., MA, J., GONG, M.: *A High Performance Immune Clonal Algorithm for Solving Large Scale TSP*, Croatia, InTech, 2010. doi:10.5772/13034

[14] FRANCISCO C., ADRIÃO D., DORIA N.: *Hybrid Metaheuristics Using Reinforcement Learning Applied to Salesman Traveling Problem*, Croatia, InTech, 2010. doi:10.5772/13343

[15] SALDANA, H.: *A Sociophysical Application of TSP: The Corporate Vote*, Croatia, InTech, 2010. doi:10.5772/12917

[16] Hirotaka, I.: *The Method of Solving for Travelling Salesman Problem Using Genetic Algorithm with Immune Adjustment Mechanism*, Croatia, InTech, 2010. doi:10.5772/13319

[17] Lu, J., Xie, M.: *Immune-Genetic Algorithm for Traveling Salesman Problem,* Croatia, InTech, 2010. doi:10.5772/12936

[18] SAKUMA, J., SHIGENOBU, K.: *Privacy-Preserving Local Search for the Traveling Salesman Problem,* Croatia, InTech, 2010. doi:10.5772/12933

[19] LUPS, L., CHIOREAN, I., LUPS, R., NEAMT, L.: *Some Special Traveling Salesman Problems with Applications in Health Economics*, Croatia, InTech, 2010. doi:10.5772/13365

[20] MO, Y.: *The Advantage of Intelligent Algorithms for TSP*, Croatia, InTech, 2010. doi:10.5772/12897

[21] MANDZIUK, J.: *Solving the Travelling Salesman Problem with a Hopfield - type neural network*, Warszawa, Demonstratio Mathematica, 1996.

[22] FRITZSCHE, P., REXACHS, D., LUQUE, E.: *Predicting Parallel TSP Performance: A Computational Approach*, Croatia, InTech, 2010. doi:10.5772/13206

[23] SIQUEIRA, P. H., STEINER, M., SCHEER, S.: *Recurrent Neural Networks with the Soft Winner Takes All Principle Applied to the Traveling Salesman Problem*, Croatia, InTech, 2010. doi:10.5772/13069

[24] YAN, X., WU, Q., LI, H.: *A Fast Evolutionary Algorithm for Traveling Salesman Problem*, Croatia, InTech, 2010. doi:10.5772/12949

[25] POLLAK, M., TUHY, T., PRAZAKOVA, V., FRISCHER, R.: *Modern communication technologies usage for controlling metallurgical processes,* METAL 2014: 23rd International Conference on Metallurgy and Materials, pp: 1678-1683, 2014.

[26] FRISCHER, R., POLLAK, M., TUHY, T., PRAZAKOVA, V.: *Usage of clustering analysis in diagnostics of metallurgical devices*, METAL 2013: 22nd International Conference on Metallurgy and Materials, pp. 1881-1886, 2013.

Review process

Single-blind peer review process.

Copyright © Acta Logistica, www.actalogistica.eu