BUSINESS PERSPECTIVES

Publisher

LLC "CPC "Business Perspectives"
Hryhorii Skovoroda lane, 10,
Sumy, 40022, Ukraine
www.businessperspectives.org

S. KUZNETS KHNUE

Founder

Simon Kuznets Kharkiv National University of Economics, Nauky avenue, 9-A, Kharkiv, 61166, Ukraine
http://www.hneu.edu.ua/

Received on: 2nd of March, 2020
Accepted on: 19th of March, 2020
Published on: 30th of June, 2020

Juraj Pekár (Slovakia), **Ivan Brezina** (Slovakia), **Jaroslav Kultan** (Slovakia), **Iryna Ushakova** (Ukraine), **Oleksandr Dorokhov** (Ukraine)

COMPUTER TOOLS FOR SOLVING THE TRAVELING SALESMAN PROBLEM

Abstract

The task of the traveling salesman, which is to find the shortest or least costly circular route, is one of the most common optimization problems that need to be solved in various fields of practice. The article analyzes and demonstrates various methods for solving this problem using a specific example: heuristic (the nearest neighbor method, the most profitable neighbor method), metaheuristic (evolutionary algorithm), methods of mathematical programming.

In addition to classic exact methods (which are difficult to use for large-scale tasks based on existing software) and heuristic methods, the article suggests using the innovative features of the commercially available MS Excel software using a meta-heuristic base. To find the optimal solution using exact methods, the Excel (Solver) software package was used, as well as the specialized GAMS software package.

Comparison of different approaches to solving the traveling salesman problem using a practical example showed that the use of traditional heuristic approaches (the nearest neighbor method or the most profitable neighbor method) is not difficult from a computational point of view, but does not provide solutions that would be acceptable in modern conditions.

The use of MS Excel for solving the problem using the methods of mathematical programming and metaheuristics enabled us to obtain an optimal solution, which led to the conclusion that modern tools are an appropriate addition to solving the traveling salesman problem while maintaining the quality of the solution.

Keywords traveling salesman problem, heuristic methods, metaheuristic methods, optimal solution, suboptimal solution, GAMS, Excel

JEL Classification C6, C61, C63

Ю. Пекар (Словаччина), **І. Брезіна** (Словаччина), **Я. Култан** (Словаччина), **І. О. Ушакова** (Україна), **О. В. Дорохов** (Україна)

КОМП'ЮТЕРНІ ІНСТРУМЕНТИ ДЛЯ ВИРІШЕННЯ ЗАВДАННЯ КОМІВОЯЖЕРА

Анотація

Завдання комівояжера, яка полягає у пошуку найкоротшого або найменш витратного кругового маршруту, є однією з найпоширеніших оптимізаційних задач, які потребують вирішення в різних сферах практичної діяльності. У статті проаналізовані і на конкретному прикладі продемонстровані різні методи вирішення даного завдання: евристичні (метод найближчого сусіда, метод найбільш вигідного сусіда), метаевристичні (еволюційний алгоритм), методи математичного програмування.

На додаток до класичних точних методів (які для великомасштабних завдань на основі наявного програмного забезпечення важко використовувати) і евристичним методам в статті запропоновано використовувати інноваційні можливості комерційно доступного програмного продукту MS Excel на основі метаеврістіческой бази. Для пошуку оптимального рішення точними методами був використаний пакет програм Excel (Solver), а також спеціалізований пакет програмного забезпечення GAMS.

Порівняння різних підходів до вирішення завдання комівояжера на практичному прикладі показало, що використання традиційних евристичних підходів (метод найближчого сусіда або метод найбільш вигідного сусіда) є не складним з обчислювальної точки зору, але не дає рішень, які були б прийнятні в сучасних умовах.

Застосування MS Excel для вирішення завдання методами математичного програмування і метаевристики дозволило отримати оптимальне рішення, що призвело до висновку про те, що сучасні інструменти є підходящим доповненням до вирішення завдання комівояжера при збереженні якості рішення.

Ключові слова задача комівояжера, евристичні методи, метаевристичні методи, оптимальне рішення, субоптимальное рішення, GAMS, Excel

Класифікація JEL C6, C61, C63

Juraj Pekár, Chief of Department of Operational Research and Econometrics, Faculty of Economic Informatics, University of Economics Bratislava, Slovakia.

Ivan Brezina, Dean Faculty of Economic Informatics, Department of Operational Research and Econometrics, University of Economics Bratislava, Slovakia.

Jaroslav Kultan, Department of Applied Informatics, Faculty of Economic Informatics, University of Economics Bratislava, Slovakia.

Iryna Ushakova, Candidate of Economic Sciences, Associate Professor, Chief of Department of Information Systems, Simon Kuznets Kharkiv National University of Economics, Ukraine.

Oleksandr Dorokhov, Professor, Associate Professor, Department of Information Systems, Simon Kuznets Kharkiv National University of Economics, Ukraine.

INTRODUCTION

Traveling Salesman Problem (TSP) has been in the spotlight since 1960. Its essence is to find the shortest or least costly circular route, which includes visiting a certain number of n points (for example, cities), with the starting and ending cities being identical, and each subsequent city included in the ring path once. Usually, a complete transport network is considered, that is, a complete schedule and the shortest distance between each pair d_{ij}, $i, j = 1,2, ..., n$).

This problem has a large number of practical applications, especially in the field of transport (for example, the disposal of household waste, the delivery of goods from a warehouse, the distribution of bakery products from bakeries to individual stores, the planning of school bus routes, the planning of services in companies, delivery services, drilling holes for printed circuit boards, computer systems, industrial robot control, circuit optimization, network design, and many others).

TSP research made a significant contribution to the development of the theory of computational complexity in the early 70s of the XX century. In the thirties of the XX century, this problem repeatedly became an object of interest for scientists at Princeton University. In the forties, this problem was studied by statisticians, especially in connection with the use in agriculture.

The term "business traveler, traveling salesman" from the point of view of mathematical optimization was first used in (Robinson, 1949), in which the shortest circular route of salesmen from Washington to the main cities of the other 49 states of the USA was found, and then returning to Washington. Currently, the optimal roundabout route through 85.900 nodes is best known (currently the largest TSP). In many other cases, with millions of nodes, non-optimal solutions have been found to ensure that the results are 2-3% of the optimal round trip.

Despite the simple statement of this problem, today there is no exact algorithm capable of solving such a large-scale problem since it belongs to a set of combinatorial problems known as NP - complex problems in which the computation time of the best available methods increases more than exponentially to the magnitude of the problem. There are two ways to solve NP - complex problems. You can use one of the general methods for constructing an exponential algorithm, which in most cases can behave like a polynomial (Ball, 1939; Menger, 1931). Well-known and well-developed exact algorithms can be used to solve TSP, which allows you to quickly and efficiently calculate the optimal solution.

The most commonly used methods are the branch and bound method, the decision tree method, dynamic programming, and others. However, they are difficult to apply to tasks with a large number of locations. Large-dimension TSP belongs to the class of decision-making problems for which it is difficult to obtain the optimal solution using real-time algorithms. Therefore, in practice, heuristic methods and metaheuristic approaches are often used.

Heuristic methods are a sequence of steps that allows you to find a real suboptimal solution (does not guarantee complete optimality). These methods, unlike exact methods, do not guarantee to find the optimal solution, but, on the other hand, they allow you to calculate a "good" acceptable solution in a relatively short time even in complex problems called suboptimal solutions.

Heuristics, by its nature, offers some special steps to solve a specific problem (Flood, 1956). Metaheuristics are a higher-level structure aimed at solving common problems, which form a set of principles and recommendations for developing heuristic optimization algorithms.

Therefore, if the exact algorithm cannot be used due to the dimension of the problem, an alternative is to create an approximation algorithm or heuristic that will be polynomial. However, we repeat, it does not always lead to an optimal solution, that is, it provides a suboptimal solution.

Today there are a large number of heuristic algorithms (nearest neighbor, profitable neighbor, gradual increase algorithm, Clarke-Wright algorithm, Lin-Kernighan algorithm), and so-called metaheuristic algorithms (evolutionary strategies, memetic algorithms, self-organizing migration algorithm, differential evolution, and others).

1. LITERATURE REVIEW

As noted above, for almost 50 years, TSP has been the focus of the attention of researchers due to its computational complexity for practical use. Its application includes logistics tasks, code theory, control of industrial robots, design of electrical distribution, various practical tasks related to the construction of optimal networks, and others. So, the classical problem formulated in (Robinson, 1949) is solved using linear programming. Another task, described in (Dantzig, Fulkerson and Johnson, 1954), can be considered as the primary basis for further research on the traveling salesman problem.

As early as the sixties in (Bellman, 1960), TSP was used as an example of a combinatorial problem that can be solved by dynamic programming. Also, TSP was formulated as an integer programming problem (Miller, Tucker and Zemlin, 1960). During this period, another study (Hellmich, 1960) emphasized the close relationship between the TSP and the linear problem. This article discusses the problem of assigning and creating basic loops to obtain a circular route. An important step in solving TSP was the branch and bound method, which was described in 1963 in (Little, Murty, Sweeney and Karel, 1963). Then, in the 1970s, was developed a heuristic algorithm to solve TSP (Lin and Kernighan, 1973).

However, to this day, the Salesman Problem has been in the interest of many experts in the field of operations research, mathematics, and artificial intelligence. The TSP solution is often used to test the effectiveness of evolving algorithms (Čičková, Brezina and Pekár, 2013).

With the development of computer technology, the problem of finding the optimal solution for large dimensions is becoming more realistic and achievable. So, in 1978, six symmetric TSPs with 80 nodes were solved (Miliotis, 1978). In 1987, in the paper (Padberg and Rinaldi, 1987) was solved TSP by the method of branches and sections with 532 nodes. Finally, in 2001, a network of 110 Alpha 500 MHz processors was built with a route connecting 15112 points in Germany.

In May 2004, the so-called Swedish TSP was created (24 978 points in Sweden). In the calculation, it was used with 96 Intel Xeon 2.8 GHz, and it was shown that for calculation using only one computer with the same CPU, it will take 84.8 years. Therefore, it remains quite obvious that to solve large-scale tasks on one computer, it is still necessary to rely only on suboptimal solutions. Today, this is the maximum task of 85,900 nodes solved by the Concorde computer system in 2006.

2. AIMS

In the case of the task of finding the shortest circular path, it is necessary to determine in what order it is necessary to go through all the vertices once so that the initial (starting) vertex is simultaneously final and the total length of the route traveled is minimal.

This problem is mentioned in the literature, for example, in (Brezina, Čičková and Gežík, 2012; Čičková, Brezina and Pekár, 2008) as the task of finding the shortest circular route. Examples of practical applications of such tasks include the disposal of household waste, the delivery of goods from a warehouse, the delivery of bakery products to stores, the planning of bus trips, the planning of services in companies, the delivery services, the planning of machine work in warehouses, and many others (Pekár, Brezina, Čičková, 2017).

In other words, the goal of finding the shortest circular route is to find the shortest path from the starting point to the other upper points of the diagram and return to the starting point, where each vertex can be visited only once.

It is known that the number of vertices at distances between all vertices is d_{ij} *(i, j = 1,2, ... n)*, and the shortest distance matrix between all vertices is *D = {*d_{ij}*} (i, j = 1,2, .. .n).*

The shortest circular path search problem can be formulated as a model of a mathematical programming problem with bivalent variables $x_{ij} \in \{0,1\}$, *i, j = 1,2, ... n*, where if the path is implemented, the value of the variable is *1*, otherwise, the value of the variable is *0*.

The best-known model for finding the shortest circular path uses the Tucker formula, which introduces additional conditions to avoid the appearance of cycles:

$$\min f(\mathbf{X},\mathbf{y}) = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij}, \qquad (1)$$

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j = 1,2,\ldots n,\ i \neq j, \qquad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i = 1,2,\ldots n,\ i \neq j, \qquad (3)$$

$$y_i - y_j + nx_{ij} \le n-1, \qquad i = 1,2,\ldots n,\ \ j = 2,3,\ldots n,\ \ i \neq j, \qquad (4)$$

$$x_{ij} \in \{0,1\}, \quad i, j = 1,2,\ldots n. \qquad (5)$$

where y_i and y_j - any numbers (real numbers assigned to the vertex u_i or u_j).

Conditions (4) prevent the occurrence of cycles. It is clear from the wording that each model will contain *n·n* binary variables and *n+n+(n–1)·(n–1)* boundary conditions.

3. METHODS

To solve problems with a large number of vertices, the classical solution to the mathematical programming problem is not applicable, because it is necessary to check, sort out, an excessively large number of options. Therefore, a large number of heuristic methods have been developed for finding the shortest closed path, which provides at least a good solution in an acceptable time.

As already mentioned, the following methods can be used to solve research problems: heuristic (nearest neighbor method, most preferred neighbor method), mathematical programming methods, metaheuristics methods (evolutionary algorithm and the like).

4. RESULTS

Considering the solution of the traveling salesman problem by heuristic methods, we note that one of the relatively simple to use and fairly easily programmed heuristic methods for solving the traveling salesman problem is the nearest neighbor method and its modification, the most favorable neighbor method.

Solving the traveling salesman problem using the nearest neighbor algorithm and, accordingly, the essence of the algorithm is to find the nearest place (neighbor) to the selected location i_1, among the possible neighbors, i_s and the achievement of which will be connected at this moment with the route with the smallest possible matrix count $D=\{d_{ij}\}$.

This method works with locations included in routes $i_1, i_2, \ldots, i_l$, denoting many places lying on the traveling salesman route (places included in the circular route), such as $I=\{1, 2, \ldots, n\}$, and unassigned locations $J=I-\{i_1, i_2, \ldots, i_l\}$.

http://dx.doi.org/10.21511/dm.18(1).2020.03

A short record of the nearest neighbor algorithm is as follows:

- step 1. Let be $k=1$, $I=\{1, 2, ..., n\}$, position u_1 is starting point, then $i=i_1=1$ and $J=I-\{i_1\}=\{2, 3, ..., n\}$;
- step 2. If $k=n$, then go to step 4; otherwise, if k < n, in i line of matrix D; in the set J one should find the smallest element $\min\limits_{q\in J}\left\{d_{iq}\right\}$ and denote it as q;
- step 3. Let be $k=k+1$ and $i=i_k=q$, then $J=J-\{q\}$ and you must go to step 2;
- step 4. Accept $i_{n+1}=1$ and complete the calculations.

In this case, the decision is determined by the sequence of places i_1, i_2, ..., i_n, i_{n+1}.

The full trajectory T of salesman traveler is determined by calculating the rank of the route along with T.

The multiple nearest neighbor algorithm is based on n fold replication of the nearest neighbor algorithm. In this case, each subsequent location in the set I is sequentially selected.

The algorithm substantially eliminates the most significant drawback of the nearest neighbor algorithm, namely, the problem of connecting the final and initial points of the desired route.

A specific implementation of the nearest neighbor algorithm can be represented in the following example.

Suppose that the task is to visit eight cities of Slovakia along the minimum route: Banska Bystrica, Bratislava, Kosice, Nitra, Presov, Trencin, Trnava, Zilina.

The starting point can be any of them, and after a single visit to each (all), you need to return to the starting point (city). In this case, the shortest distances between the individual peaks (cities) d_{ij} are given in Table 1.

Table 1. Distance matrix D

Source: Initial data in the author's statement of the problem.

Matrix D distance (km)	B.Bystrica	Bratislava	Košice	Nitra	Prešov	Trenčín	Trnava	Žilina
Banská Bystrica	0	208	213	119	195	142	166	89
Bratislava	208	0	391	88	403	127	47	200
Košice	213	391	0	302	35	303	349	230
Nitra	119	88	302	0	315	85	46	140
Prešov	195	403	35	315	0	293	361	221
Trenčín	142	127	303	85	293	0	78	73
Trnava	166	47	349	46	361	78	0	151
Žilina	89	200	230	140	221	73	151	0

If the route starts, for example, in Banska Bystrica, the result will be: Banska Bystrica - Zilina - Trencin - Trnava - Nitra - Bratislava - Kosice - Presov - Banska Bystrica with a total length of 995 km, as shown in Figure 1.

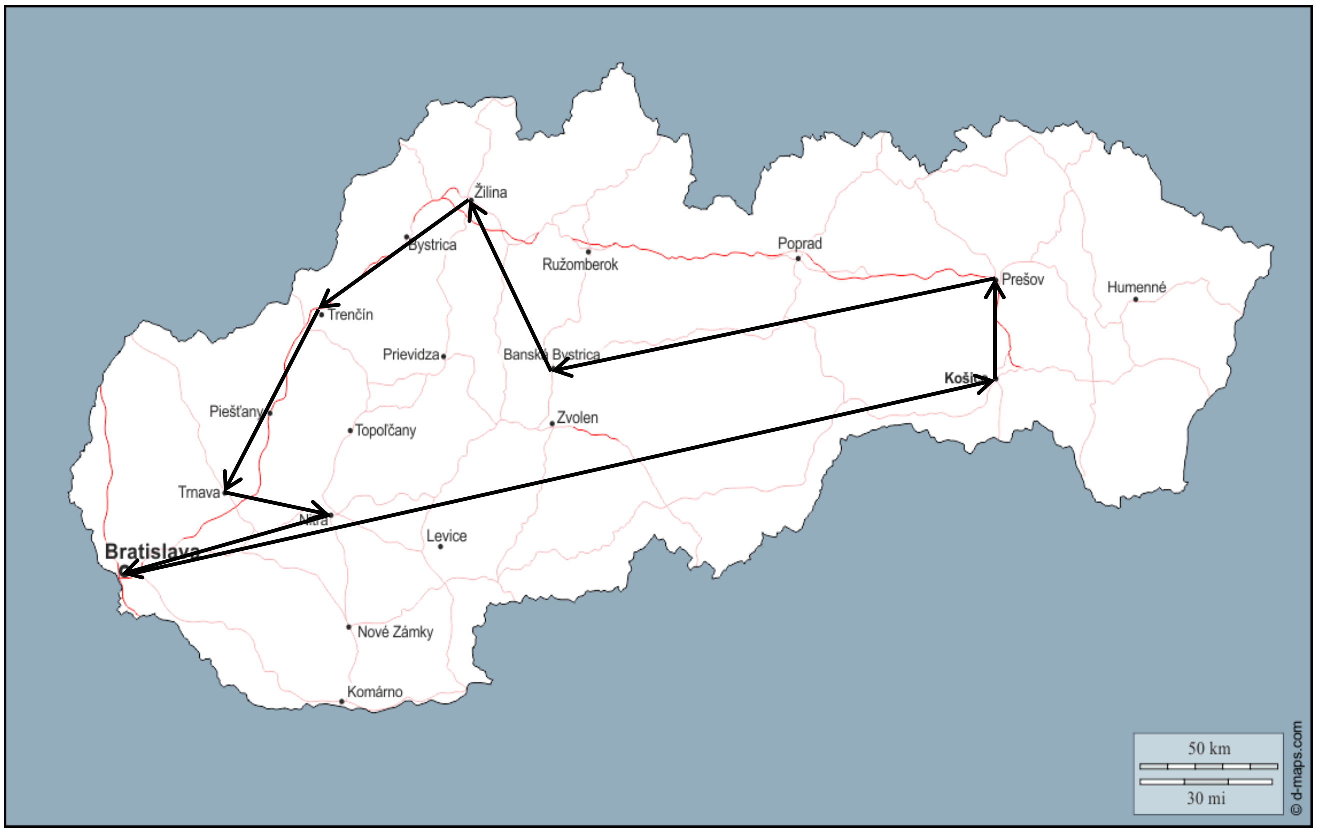
Another solution to the traveling salesman problem can be obtained using the algorithm of the most profitable (preferred) neighbor.

The steps of the most preferred neighbor algorithm are the same as in the nearest neighbor algorithm, but instead of matrix D, the frequency matrix E is used, the elements of which form a complete asymmetric graph are determined by the relation:

$$e_{ij}=(n-1).d_{ij}-\sum_{k=1}^{n}d_{ik}-\sum_{q=1}^{n}d_{qj}-d_{ji}\,;\ k\neq j\,,\ q\neq i. \qquad (6)$$

And, accordingly, for a complete symmetric graph:

$$e_{ij}=(n-2).\ d_{ij}-\sum_{k=1}^{n}d_{ik}-\sum_{q=1}^{n}d_{qj}\,;\ k\neq j\,,\ q\neq i. \qquad (7)$$



Source: Route developed by the authors.

Figure 1. An example of solving the traveling salesman problem using the nearest neighbor algorithm

The meaning of the coefficients e_{ij} is that instead of the direct distances between the points u_i and u_j of the matrix $D = \{d_{ij}\}$, we consider the "environment" of this pair of places.

If e_{ij} is high, it is preferable to use other edges starting with u_j or ending with u_j to be included in the circular path. Conversely, in the case of a low value of e_{ij}, it is preferable to include h_{ij} in the formed circular path.

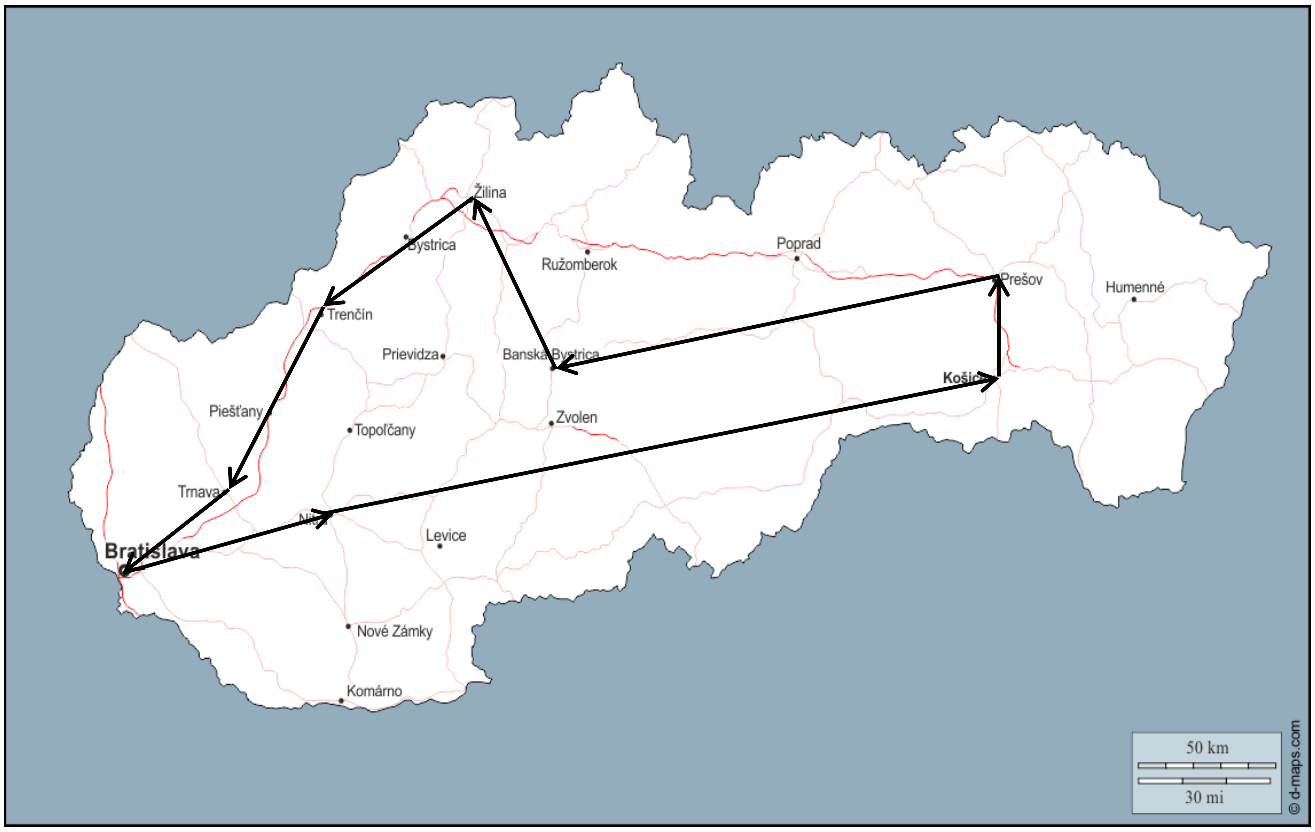
The algorithm of the most preferred neighbor is identical to the principle of the repeated nearest neighbor algorithm, but the frequency matrix E is used instead of the D matrix.

The algorithm of the most profitable neighbor can be presented for our example with the corresponding form of the matrix E, presented in Table 2.

Table 2. Frequency matrix E

Source: Calculation results in a model developed by the authors.

Matrix E	B. Bystrica	Bratislava	Košice	Nitra	Prešov	Trenčín	Trnava	Žilina
Banská Bystrica	0	–932	–1.251	–1.275	–1.395	–1.097	–1.002	–1.524
Bratislava	–932	0	–159	–1.855	–63	–1.549	–2.286	–968
Košice	–1.251	–159	0	–502	–3.366	–500	–229	–1.087
Nitra	–1.275	–1.855	–502	0	–398	–1.516	–1.925	–1.079
Prešov	–1.395	–63	–3.366	–398	0	–580	–133	–1.159
Trenčín	–1.097	–1.549	–500	–1.516	–580	0	–1.675	–1.621
Trnava	–1.002	–2.286	–229	–1.925	–133	–1.675	0	–1.094
Žilina	–1.524	–968	–1.087	–1.079	–1.159	–1.621	–1.094	0



Source: Route developed by the authors.

Figure 2. An example of solving the TSP problem using the most profitable neighbor algorithm

If the path starts again in Banska Bystrica, the result will be the following sequence: Banska Bystrica - Zilina - Trencin - Trnava - Bratislava - Nitra - Kosice - Presov - Banska Bystrica, with its other total length, this time amounting to 907 km, which Figure 2 illustrates.

Another widely used approach is to solve the traveling salesman problem with mathematical programming methods. The traveling salesman problem can be formulated as a mathematical programming problem, and then it can be solved using, for example, MS Excel.

Next, we consider two ways to solve the traveling salesman problem, namely, using the Solver add-in in MS Excel and specialized software GAMS for solving mathematical programming problems. Both methods are practically implemented based on the mathematical programming problem.

Both approaches will be described using the previous route as an example.

To solve the TSP problem in the MS Excel spreadsheet processor, you must first prepare the source data. Table 3 shows a method to determine the source data for the illustrative example that we are considering in the MS Excel.

Table 3. Source data and auxiliary calculation results in MS Excel

Source: Model developed by the authors.

City (visited point)	B. Bystrica	Bratislava	Košice	Nitra	Prešov	Trenčín	Trnava	Žilina	Of
Banská Bystrica	0	208	213	119	195	142	166	89	
Bratislava	208	0	391	88	403	127	47	200	
Košice	213	391	0	302	35	303	349	230	
Nitra	119	88	302	0	315	85	46	140	865
Prešov	195	403	35	315	0	293	361	221	
Trenčín	142	127	303	85	293	0	78	73	
Trnava	166	47	349	46	361	78	0	151	
Žilina	89	200	230	140	221	73	151	0	
Y	0	6	2	7	0	4	5	3	1
X	0	0	0	0	1	0	0	0	1
	0	0	0	1	0	0	0	0	1
	0	0	0	0	0	0	0	1	1
	1	0	0	0	0	0	0	0	1
	0	0	1	0	0	0	0	0	1
	0	0	0	0	0	0	1	0	1
	0	1	0	0	0	0	0	0	1
	0	0	0	0	0	1	0	0	1
	1	1	1	1	1	1	1	1	
	0	4	7	6	2	1	3		
	–4	0	–5	2	–2	–3	7		
	1	5	0	7	3	2	4		
	–6	6	–7	0	–4	–5	–3		
	–2	2	–3	4	0	7	1		
	7	3	–2	5	1	0	2		
	–3	1	–4	3	7	–2	0		

In our case (for described example), the task, which is mathematically described before by formulas 1-5, has 72 decision variables, of which 64 variables represent all existing paths on the graph, and an additional 8 variables prevent looping.

Block *B*11:*I*19 is reserved for these variables. Each variable is assigned an initial value of 0, while in the block of variables *B*12:*I*19, the unit indicates that the route is located on a circular path. Also are needed additional formulas to find the optimal solution.

To be able to introduce separate restrictions, it is necessary to express the functions of the left side and compare them with the coefficients of the right side. Table 4 below lists the constraints and the corresponding formula for calculating them for the left side of the equation.

Table 4. Restrictive conditions

Source: Model developed by the authors.

Limiting the conditions under which a route from a node can be used once	Cell	Formula
x12 + x13 + x14+ x15+ x16+ x17+ x18	J12	=SUM(B12:I12)–B12
x21 + x23 + x24+ x25+ x26+ x27+ x28	J13	=SUM(B13:I13)–C13
. . .		
x81 + x82 + x83+ x84+ x85+ x86+ x87	J19	=SUM(B19:I19)–I19
x21 + x31 + x41 + x51 + x61 + x71+ x81	B20	=SUM(B12:B19)–B12
x12 + x32 + x42 + x52 + x62 + x72+ x82	C20	=SUM(C12:C19)–C13
. . .		
x18 + x28 + x38 + x48 + x58 + x68+ x78	I20	=SUM(I12:I19)–I19

In our case, we need to calculate the sum of the variables, except for the variables with the same index, so the SUM function was used, and then the variable with the same index was subtracted.

In addition to the above limiting conditions, cyclic limiting conditions are also used.

To determine them, cells C22:I28 were calculated; for this, the limit values of the left side were calculated, obtained as the difference between variables yi, yj and xij, multiplied to the number of vertices.

For example, cells *C*22=*C*11-*C$*11+8·*C*13; *D*22=*$C$*11-*D$*11+8·*D*13, ..., *I*28=*I*11-*I$*11+8·*I*19.

To simplify this task in Solver, diagonal elements of loop avoidance conditions were included in the limit conditions. However, these restrictions do not cause any changes to the solution.

The final step in preparing the source data is to define an optimization criterion. This criterion should be written in the form of formula and placed in the appropriate cell.

In our an illustrative example, the Formula=SUMPRODUCT (B2:I9; B12:I19) is located in cell J2 (see Table 3 above), and the optimization criterion itself is expressed as the scalar multiplication of the distance matrix (block B2:I9) and the variable matrix X (block B12:I19).

Once the initial data has been entered, the Solver optimization module can be activated. After starting DataSolver, the user can specify the SolverParameters to indicate the model parameters. Initial, individual Solver parameters that require preliminary adjustment, shown in Figure 4, are described in detail below, based on how they are used in the problem being solved.

Set Objective - in a given task is the optimization criterion in the cell *J*2.

TO - minimization criterion (*Min*).

By Changing Variable Cells - *B*11:*I*19, these cells contain the resulting values at the end of the calculation.

Subject to the Constraints – a set of necessary solution constraints.

Cell Reference - cells *B*20:*I*20 and *J*12:*J*19, set of left limits.

Relation - type of restriction for all borders is "=" (equal).

Constraint - values are equal to 1.

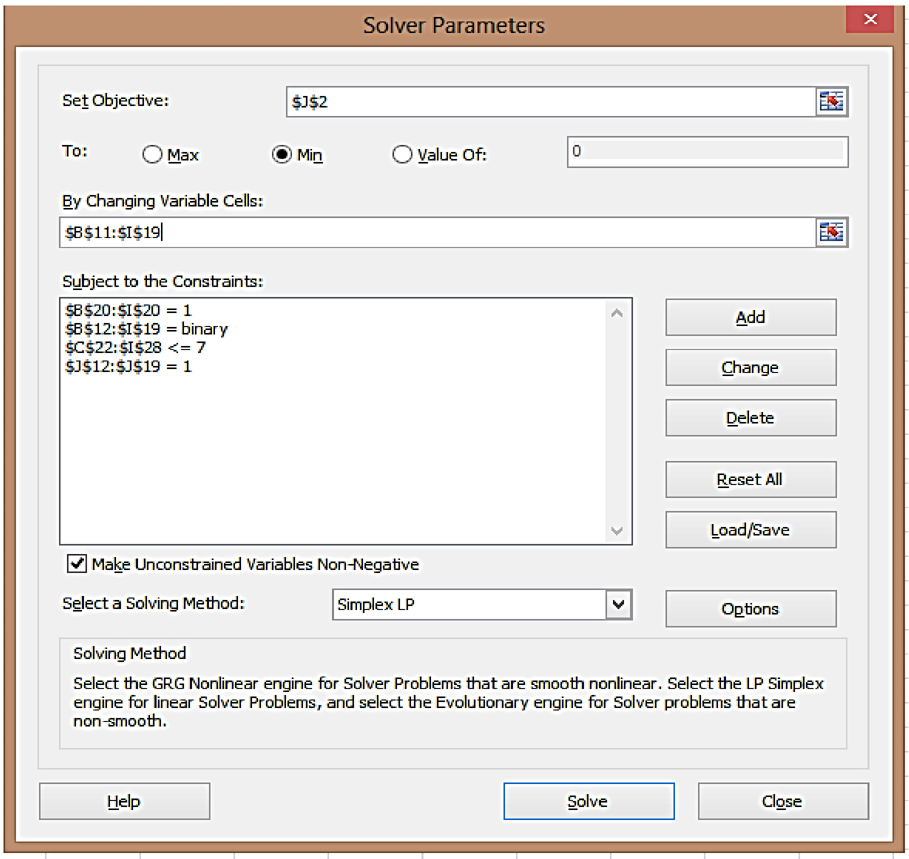
Cell Reference - cells *C*22:*I*28, also representing left constraint.

Relation - a type of restriction for all borders is "≤" (less or equal).

Constraint - values of restrictions are equal to 7 (number of nodes minus one).

In addition to the above conditions, we determine that the values of *B*12:*I*19 are binary. The full form of restrictive conditions for the given example is illustrated in Figure 3.

The optimal solution found in the sequence of driving around Banska Bystrica - Presov - Kosice - Zilina - Trencin - Trnava - Bratislava - Nitra - Banska Bystrica. The total length of the route shown in Figure 4 is 865 km.


Solver Parameters

Set Objective: $J$2

To: ◯ Max [✓ Min] ◯ Value Of: 0

By Changing Variable Cells:
$B$11:$I$19

Subject to the Constraints:
```
$B$20:$I$20 = 1
$B$12:$I$19 = binary
$C$22:$I$28 <= 7
$J$12:$J$19 = 1
```

[✓] Make Unconstrained Variables Non-Negative

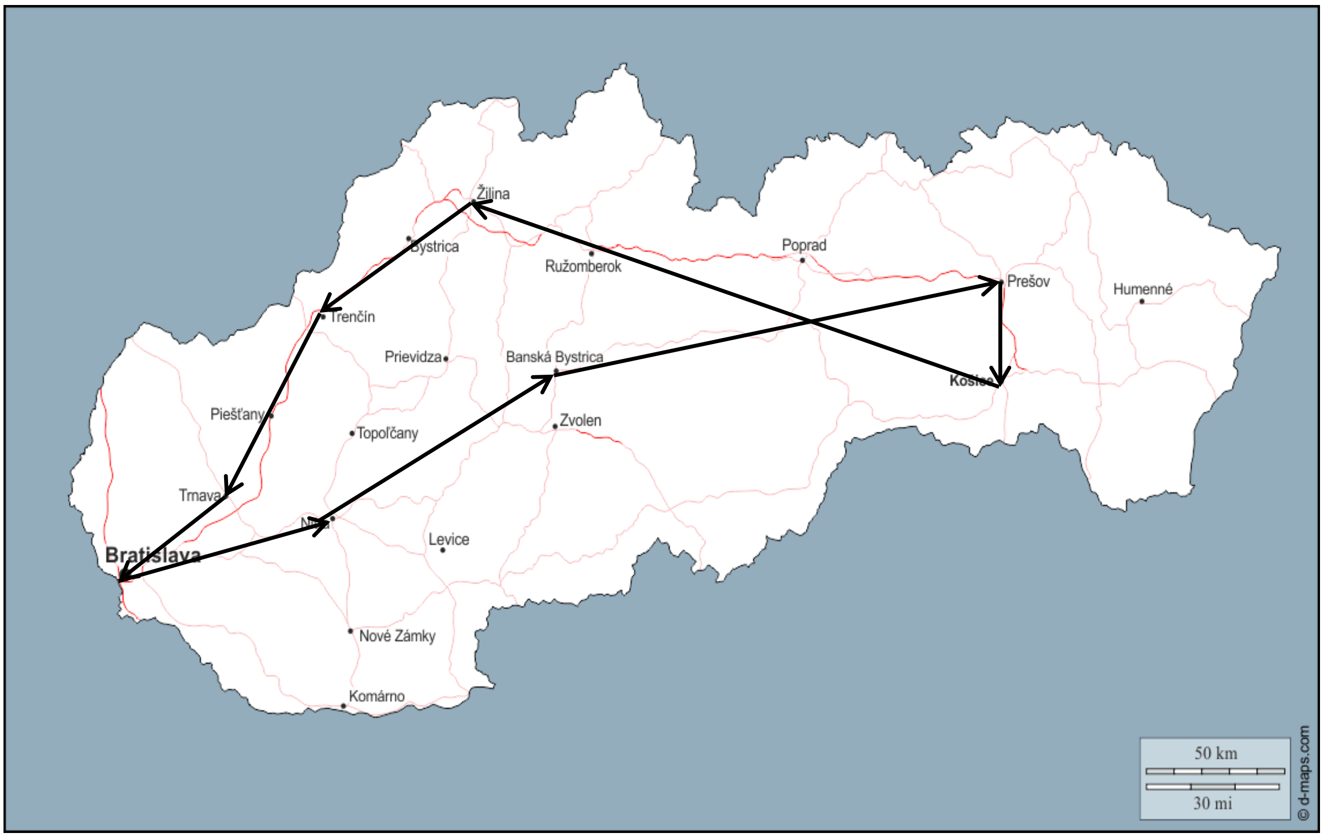
Select a Solving Method: Simplex LP

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

Source: Model developed by the authors.

Figure 3. Filled form of restrictive conditions for TSP



Source: Route developed by the authors.

Figure 4. General route sequence

Another possibility of solving the TSP problem is to use specialized software products for solving sets of optimization problems. We, in particular, used the GAMS package (Čičková, Brezina, Pekár, 2016).

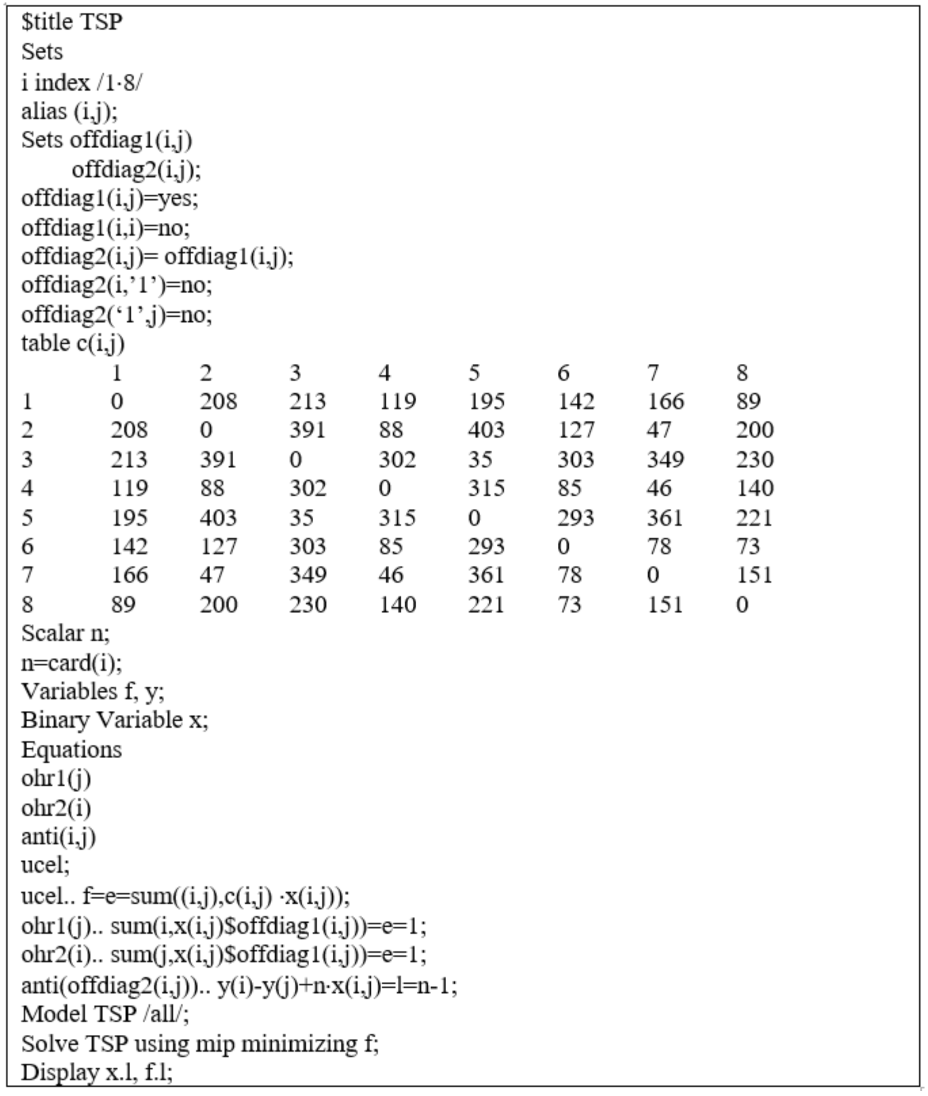
To describe the problem from the solved example in GAMS, we use the compact form of the mathematical model represented by formulas 1-5.

The corresponding program recording code in GAMS is shown in Figure 5.

As a result of the decision in GAMS, we obtain the following sequence of Banska Bystrica - Presov - Kosice - Zilina - Trencin - Trnava - Bratislava - Nitra - Banska Bystrica. Moreover, the total length of the route is also 865 km and remains the same as was previously shown in Figure 5 above.

Finally, we consider the solution of the traveling salesman problem using metaheuristics (evolutionary algorithm).

Recent versions of MS Excel provide another way to solve the problem, starting with finding solutions that represent different combinations of numbers from 1 to n. Since the goal of the salesman is to find a route between all the peaks, you can use this tool to find a path where the total route is minimal.



```
$title TSP
Sets
i index /1·8/
alias (i,j);
Sets offdiag1(i,j)
     offdiag2(i,j);
offdiag1(i,j)=yes;
offdiag1(i,i)=no;
offdiag2(i,j)= offdiag1(i,j);
offdiag2(i,'1')=no;
offdiag2('1',j)=no;
table c(i,j)
        1       2       3       4       5       6       7       8
1       0       208     213     119     195     142     166     89
2       208     0       391     88      403     127     47      200
3       213     391     0       302     35      303     349     230
4       119     88      302     0       315     85      46      140
5       195     403     35      315     0       293     361     221
6       142     127     303     85      293     0       78      73
7       166     47      349     46      361     78      0       151
8       89      200     230     140     221     73      151     0
Scalar n;
n=card(i);
Variables f, y;
Binary Variable x;
Equations
ohr1(j)
ohr2(i)
anti(i,j)
ucel;
ucel.. f=e=sum((i,j),c(i,j) ·x(i,j));
ohr1(j).. sum(i,x(i,j)$offdiag1(i,j))=e=1;
ohr2(i).. sum(j,x(i,j)$offdiag1(i,j))=e=1;
anti(offdiag2(i,j)).. y(i)-y(j)+n·x(i,j)=l=n-1;
Model TSP /all/;
Solve TSP using mip minimizing f;
Display x.l, f.l;
```

Source: Algorithm and program developed by the authors.

Figure 5. Program code in the GAMS software system for TSP solving

The representation of the source data and the results are shown in Table 5.

Cells *K*2:*K*9 reflects the sequence of vertices on the route. The INDEX function is used to calculate the value of the objective function from the values from the specified row and column for the selected area. Thus, the initial parameters are the data area and the row and column number. In solving the problem, we use this function to find the distance between the two vertices of the route.

Table 5. Initial data and the results of solving the traveling salesman problem

Source: Model developed by the authors.

City (visited point)	B. Bystrica	Bratislava	Košice	Nitra	Prešov	Trenčín	Trnava	Žilina		Route	Summed distance
Banská Bystrica	0	208	213	119	195	142	166	89		8	230
Bratislava	208	0	391	88	403	127	47	200		6	73
Košice	213	391	0	302	35	303	349	230		7	78
Nitra	119	88	302	0	315	85	46	140		2	47
Prešov	195	403	35	315	0	293	361	221		4	88
Trenčín	142	127	303	85	293	0	78	73		1	119
Trnava	166	47	349	46	361	78	0	151		5	195
Žilina	89	200	230	140	221	73	151	0		3	35
Total											865

Cells *L*2-*L*9 represent the distance to the vertex indicated in the cell on the left, that is, the distance between *L*2 cells to vertex 5 from the previous vertex on the route (vertex 7), cell *L*3 - the distance to vertex 3 from vertex 5, and so on.

Further, in cell *L*2, the INDEX function is used with the parameters of area *B*2:*I*9, the row number in cell *K*9, and the column number in cell *K*2. After that, similarly, in cell *L*3, we determine the INDEX function with domain *B*2:*I*9, row number in cell *K*2, column number in cell *K*3, and so on.

Finally, we define the target function in cell *L*10, which is the sum of the individual paths along a circular path, i.e. it is (=*SUM*(*L*2:*L*9)).

After completing the preparation of the initial data, the optimization module Solver plugin can be activated. After its launch (Data-Solver), the user is prompted to set the Solver Parameters dialog to indicate the parameters of the task being solved. The individual initial parameters of the Solution are described in detail below, showed in Figure 6, which are used in solving the problem.

Set Objective - in a given task is the criterion optimization in cell *L*10.

To - in the above problem, the minimization criterion (*Min*).

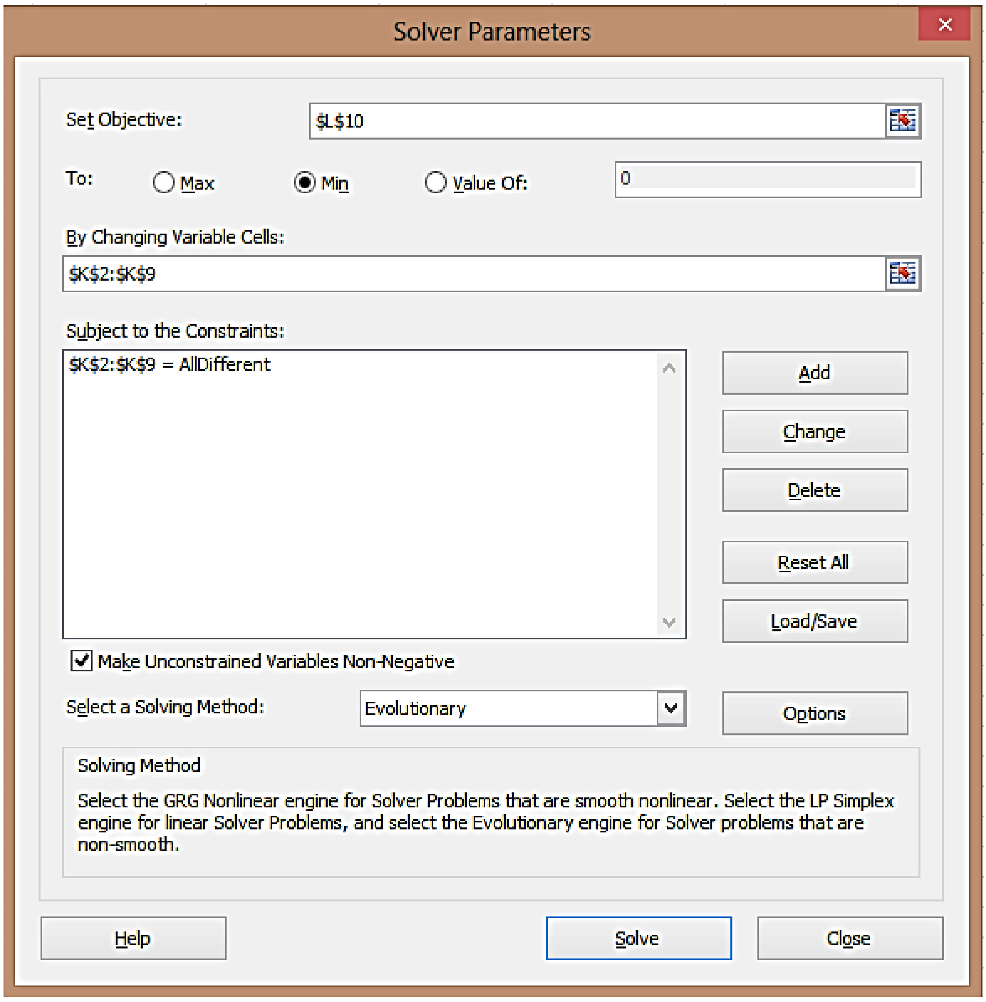
By Changing Variable Cells - cell areas *K*2:*K*9, which contains the final values at the end of the calculation.

Subject to the Constraints - this task does not contain a structural boundary, but it is necessary to determine the property of the variables so that they all differ from each other.

Cell Reference - cells *K*2:*K*9 containing variable parameters of the task.

Relation - the type of restriction, diff.

Constraint - value not defined.



Solver Parameters

Set Objective: $L$10

To: Max [✓ Min] Value Of: 0

By Changing Variable Cells: $K$2:$K$9

Subject to the Constraints:

$K$2:$K$9 = AllDifferent

[✓] Make Unconstrained Variables Non-Negative

Select a Solving Method: Evolutionary

Solving Method

Select the GRG Nonlinear engine for Solver Problems that are smooth nonlinear. Select the LP Simplex engine for linear Solver Problems, and select the Evolutionary engine for Solver problems that are non-smooth.

Source: Model developed by the authors.

Figure 6. Initial Settings for Configuring Solutions

It is necessary to set the Evolutionary option in the Select and Solving Method menu since the solution to the problem in this example with the diff parameter can be obtained with this value.

As a result, the same solution to the problem was obtained using exact methods. It is set by the sequence of cities Zilina, Trencin, Trnava, Bratislava, Nitra, Banska Bystrica, Presov, Kosice. The total length of the salesman route also remained the same. Recall that it is calculated by calculating the distances between the tops of the route and is the same 865 km.

5. DISCUSSION

A generalized comparison of the results of solving the traveling salesman problem by the various methods considered by us above is given in Table 6.

Table 6. Results of solving the traveling salesman problem by various considered methods

Source: Developed by the authors.

Researched and used solution method	Solution	Route length, km
Excel – precise method	Optimal	865
GAMS - precise method	Optimal	865
Excel – metaheuristic method	Optimal	865
Nearest neighbor – heuristic method	Suboptimal	995
The most profitable neighbor - heuristic method.	Suboptimal	907

The presented methods were also applied to practical, real-life tasks of large sizes, using the example of 32 cities in Slovakia.

The total length of the optimal route calculated by exact methods (Excel, GAMS) was 1.370 km.

The nearest neighbor method provided a suboptimal solution of 1.975 km, which is 605 km (44.1%) more.

The most profitable neighbor method also showed a suboptimal solution of 1.526 km, which is 156 km (11.4%) more than with the optimal solution to the problem.

CONCLUSION

Summing up the above, we can conclude that given the computational complexity of the traveling salesman problem, its solution by classical methods of optimization in real-time is not rational, therefore, it is advisable to use heuristic approaches to solve large-scale problems.

At the same time, the development of new alternative approaches can be achieved by solving sequential tasks that are more suitable for practice in terms of complexity, time, and quality of the solution.

A comparison of different approaches to solving the traveling salesman problem using a practical example showed that the use of traditional heuristic approaches (the nearest neighbor method or the most profitable neighbor method) is not computationally difficult, but does not provide solutions that would be acceptable in modern conditions.

The use of MS Excel to solve the problem by mathematical programming and metaheuristics allowed us to obtain the optimal solution, which led to the conclusion that modern tools are a suitable complement to solving the traveling salesman problem while maintaining the quality of the solution.

AUTHORS CONTRIBUTIONS

Conceptualization: Juraj Pekár, Jaroslav Kultan.
Formal analysis: Ivan Brezina, Iryna Ushakova.
Investigation: Ivan Brezina, Jaroslav Kultan.
Methodology: Juraj Pekár.
Project administration: Oleksandr Dorokhov.
Software: Jaroslav Kultan.
Supervision: Juraj Pekár, Ivan Brezina.
Validation: Iryna Ushakova.
Writing – original draft: Oleksandr Dorokhov, Jaroslav Kultan.
Writing – review & editing: Oleksandr Dorokhov, Iryna Ushakova.

REFERENCES

1. Ball, W. (1939). *Mathematical recreations and essays*. New York.
2. Bellman, R. (1960). C*ombinatorial processes and dynamic programming, Combinatorial Analysis*. American Mathematical Society.
3. Brezina, I., Čičková, Z., & Gežík, P. (2012). *Sieťová analýza*. Bratislava: EKONÓM.
4. Čičková, Z., Brezina, I., & Pekár, J. (2008). An alternative method for solving traveling salesman problem by evolutionary algorithm. *Management Information Systems, 3*(1), 17-22. Retrieved from http://www.ef.uns.ac.rs/mis/archive-pdf/2008%20-%20No1/MIS2008_1_3.pdf
5. Čičková, Z., Brezina, I., & Pekár, J. (2013). Open traveling salesman problem with time windows. *Logistics International, 13*, 40-43. Retrieved from http://logic.sf.bg.ac.rs/wp-content/uploads/Papers/ID-8.pdf
6. Čičková, Z., Brezina, I., & Pekár, J. (2016). Solving the routing problems with time windows. In D. Davendra, I. Zelinka (Ed.), *Self-organizing migrating algorithm: methodology and implementation* (pp. 207-236). Springer. Retrieved from https://link.springer.com/chapter/10.1007/978-3-319-28161-2_10
7. Dantzig, G., Fulkerson, R., & Johnson, S. (1954). Solution of a large-scale traveling salesman problem. *Journal of the Operations Research Society of America, 2*(4), 393-410. Retrieved from https://www.jstor.org/stable/166695?seq=1
8. Flood, M. (1956). The traveling-salesman problem. *Operations Research, 4*(1), 61-75. Retrieved from https://www.jstor.org/stable/167517?seq=1
9. Hellmich, K. (1960). *Die Reiseroute kűrzester Weglänge*. Dauer.
10. Lin, S., & Kernighan, B. (1973). An efficient heuristic for the traveling salesman problem. *Operations Research, 21*(2), 17-28.
11. Little, J., Murty, K., Sweeney, D., & Karel, C. (1963). An Algorithm for the Traveling Salesman Problem. *Operations Research, 11*(6), 863-1025. https://doi.org/10.1287/opre.11.6.972
12. Menger, K. (1931). Bericht über ein mathematisches Kolloquium 1929/30*. Monatshefte für Mathematik und Physik*, 38, 17-38. Retrieved from https://link.springer.com/article/10.1007%2FBF01700678
13. Miliotis, K. (1978). Using cutting planes to solve the symmetric travelling salesman problem. *Mathematical Programming, 15*, 177-188. Retrieved from https://link.springer.com/article/10.1007/BF01609016
14. Miller, C., Tucker, A., & Zemlin, R. (1960). Integer Programming Formulation of Traveling Salesman Problems. *Journal of the ACM, 7*, 42-49. https://doi.org/10.1145/321043.321046
15. Padberg, M., & Rinaldi, G. (1987). Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters, 6*(1), 1-7. https://doi.org/10.1016/0167-6377(87)90002-2
16. Pekár, J., Brezina, I., & Čičková, Z. (2017). Synchronization of capacitated vehicle routing problem among periods. *Ekonomický časopis, 65*, 66-78.
17. Robinson, J. (1949). *On the Hamiltonian game and traveling-salesman problem*. RAND Research.