Local Sequence Method of Finding Solution to Traveling Salesman Problem

Igor Gritsuk, Dmytro Plechystyy, Andriy Morozov, Tamara Loktikova and Valentyna Shadura

September 23, 2021

Local Sequence Method of Finding Solution to Traveling Salesman Problem

Igor Gritsuk[1], Dmytro Plechystyy[2], Andriy Morozov[3], Tamara Loktikova[4], Valentyna Shadura[5]

[1] *Kherson State Maritime Academy, Ushakova ave., 20, Kherson, 73000, Ukraine*
[2,3,4,5] *Zhytomyr Polytechnic State University, Chudnivska st., 103, Zhytomyr, 10005, Ukraine*

Abstract
The article describes the method of local sequencing, addressed to traveling salesman problem (TSP) and its restricted versions in the form of polynomial algorithms of finding satisfactory exact solutions. The proposed method takes into account the features of efficiently solvable generalizations of assignment problem. The considerations that are the basis for the development of algorithms for solving TSP by local optimal sequence building are presented. The algorithm for building the bypass τ^o is presented. The proposed algorithm is characterized by low complexity of building a acceptable solution to τ^o. The time of its operation is estimated. The time complexity of the proposed algorithm is estimated with value $O(n^2)$. In practice, the algorithm works faster than the "go to the nearest" heuristics.

Keywords
Traveling salesman problem, permutation matrix, bypass, NP-hard.

1. Introduction

The problem of routing flows of raw materials, energy resources, information is one of the most pressing problems of modern industry, transport, public administration. Accelerating the pace of movement, increasing the density of traffic flows and transportation facilities lead to increased requirements for computers, software and algorithms that are responsible for automating traffic control processes.

Problems of efficient operation of transport systems in modern conditions can be successfully solved by methods of mathematical modeling, the importance of which is especially growing due to the large number of controlled objects and the need for real-time decision-making.

A wide range of tasks that model the processes of management and planning in transport networks, formally reduced to the tasks of the class of salesman, in which you need to define a cycle or chain with given conditions for a weighted graph [1-3]. The problem of the salesman, which is constantly supplemented by tasks of an applied nature, remains an active area of research aimed at the development and improvement of combinatorial optimization methods. It is represented by a class of NP-hard problems, among which the central place is occupied by the task of the salesman [4-8].

The task of the salesman is to find the route of a traveler-merchant who goes on a route from a certain city (for certainty we will consider it the first), aims to visit N cities, having visited each city exactly once and return to the first city. The route should have the shortest length. In the language of graph theory, the salesman's task is to find the shortest Hamiltonian cycle on a weighted graph. In [9] it was proved that the task of a salesman is NP-hard. In [10] it is shown that the Euclidean version of the salesman problem is also NP-hard. [11] provides an overview of the exact and approximate methods

International Conference "Information Control Systems and Technologies (ICST -2021)", September 23–25, 2021, Odessa, Ukraine
EMAIL: gritsuk_iv@ukr.net (I. Gritsuk); only.one.trogvar@gmail.com (D. Plechystyy); morozov@ztu.edu.ua (A. Morozov); tamlokt@ukr.net (T. Loktikova); valteach8@gmail.com (V. Shadura)
ORCID: 0000-0001-7065-6820 (I. Gritsuk); 0000-0002-0219-2603 (D. Plechystyy); 0000-0003-3167-0683 (A. Morozov); 0000-0002-3525-0179 (T. Loktikova); 0000-0003-2635-7211 (V. Shadura)

of solving the problem of a salesman with an analysis of the disadvantages and advantages of the methods. The article [12] considers the problem of several salesmen, provides options for practical application, describes the exact and heuristic methods of solving the problem [13-15].

The practical application of methods to solve the problem of the salesman includes not only in transport logistics, but also in cartographic and geographical systems [16], in robotics in the programming of mobile robots [17], in biology [18], etc. [19, 20].

2. Local sequence method in problem of finding the route of traveling salesman

Let $X = [x_{ij}]_n$ – be permutation matrix $\pi = (\pi[1], \pi[2], \ldots, \pi[n])$: $x_{i\pi[i]} = 1$, $i = \overline{1,n}$, $\pi[i] \in \{1,2,\ldots,n\}$; $x_{ij} = 0$ in all other cases. Each element $i \in \{1,2,\ldots,n\}$ is leveled to one-one conformity with element $\pi[i]$, having defined thereby the substitution

$$\begin{pmatrix} 1 & 2 & \ldots & n \\ \pi[1] & \pi[2] & \ldots & \pi[n] \end{pmatrix}.$$

All the substitutions of n degree will be divided into two classes. The first class includes all cyclic substitutions

$$\begin{pmatrix} 1 & 2 & \ldots & n \\ \tau[1] & \tau[2] & \ldots & \tau[n] \end{pmatrix},$$

they have only one length cycle in their cycle layout n, the second one includes all the rest [21]. A permutation that corresponds to the cyclic permutation will be denoted as $(\tau[1], \tau[2], \ldots, \tau[n])$ and called as cyclic permutation.

Let $[d_{ij}]_n$ – be square matrix of order n, where

$$d_{ij} = \begin{cases} d_{ij}, \text{if } i \neq j, \\ \infty \text{ otherwise,} \end{cases} \tag{1}$$

$d_{ij} \in Z_0^+$, Z_0^+ – is a set of non-negative integers.

The diagonal $\Pi = (d_{1\pi[1]}, d_{2\pi[2]}, \ldots, d_{n\pi[n]})$ is considered in matrix $[d_{ij}]_n$ that corresponds to a random permutation $\pi = (\pi[1], \pi[2], \ldots, \pi[n])$. Since π is a column number permutation of matrix $[d_{ij}]_n$, we will denote element $d_{i\pi[i]}$ of the diagonal Π with $d_{\pi[i]}$.

Following matrix $[d_{ij}]_n$ the complete oriented multigraph G is built with n vertices. Here every pair of vertices $\{i,j\}$, $i \neq j$, is connected with a pair of arcs (i,j) and (j,i) with weights or values d_{ij} or d_{ji}.

TSP formulation.

It is necessary to find a contour with the smallest total of arc weights included in it within the complete oriented multigraph G. It has to pass through each vertex exactly one time.

The sequence $\tau = (\tau[1], \tau[2], \ldots, \tau[n], \tau[1])$, which corresponds to the cyclic permutation $(\tau[1], \tau[2], \ldots, \tau[n])$, is named by us as a bypass.

It is obvious that the bypass $\tau = (\tau[1], \tau[2], \ldots, \tau[n], \tau[1])$ is an acceptable solution to TSP. Within multigraph it determines the closed route $(\tau[1], \tau[2], \ldots, \tau[n], \tau[1])$, where all the numbers $\tau[1], \tau[2], \ldots, \tau[n]$ of the set $\{1,2,\ldots,n\}$ are different. Let us determine the cost of bypass τ to G:

$$D(\tau) = d_{\tau[1]\tau[2]} + d_{\tau[2]\tau[3]} + \ldots + d_{\tau[n-1]\tau[n]} + d_{\tau[n]\tau[1]} = \sum_{i=1}^{n} d_{\tau[i]}.$$

Let us formulate TSP by analogy with assignment task formulation.

It is necessary to find cyclic permutation of the column numbers $(\tau^*[1], \tau^*[2], \dots, \tau^*[n])$ with the minimum total weight of the corresponding bypass τ^* in matrix $\left[d_{ij}\right]_n$. The matrix elements have to satisfy condition (1).

$$D(\tau^*) = \min_{\tau} \sum_{i=1}^{n} d_{\tau[i]}.$$

If to restrict the values of the matrix $\left[d_{ij}\right]_n$ with the condition $d_{ij} = d_{ji}$, then we will obtain a symmetric TSP. In this case the complete undirected graph with n vertices, where the edge $\{i,j\}$, $i \neq j$, $i,j = \overline{1,n}$, has weight d_{ij} and corresponds to matrix $\left[d_{ij}\right]_n$.

By representing the elements of matrix $\left[d_{ij}\right]_n$ as distances, it should be assumed that they satisfy the triangle inequality:

$$\begin{aligned} &d_{ij} = d_{ji} \text{ for all } i,j; \\ &d_{ij} + d_{jk} \geq d_{ik} \text{ for all } 1 \leq i,j,k \leq n. \end{aligned} \qquad (2)$$

In this case, the TSP is said to be restricted to matrix that satisfies the triangle inequality.

Thus, narrowing the range P of solutions $\pi = (\pi[1], \pi[2], \dots, \pi[n])$ to the assignment problem to subset P_τ of all cyclic permutations $(\tau[1], \tau[2], \dots, \tau[n])$ leads to TSP formulation as well as makes these problems different. It is expressed in the incomparability of complexity estimates for producing optimal solutions. The set where the optimal solution to assignment problem is being sought consists of $n!$ permutation of π. The size of the set P_τ that contains the required permutation $(\tau^*[1], \tau^*[2], \dots, \tau^*[n])$ of asymmetrical TSP is equal to $(n-1)!$ and the size of set solutions within symmetrical problem is twice smaller than $|P_\tau|$. At the same time assignment problem is known to be effectively solved for the period of time $O(n^3)$, while TSP with or without triangle inequality is NP - hard in a strong sense.

Let us outline the considerations that are the basis for the development of algorithms to be applied to solve TSP by the method of building local optimal sequences.

Let us consider two strictly increasing sequences $(i_1, i_2, \dots, i_{n_1})$ i $(j_1, j_2, \dots, j_{n_1})$, $n_1 \leq n$, one of which contains line numbers $i_s \in \{1,2,\dots,n\}$ and another one contains column numbers $j_t \in \{1,2,\dots,n\}$ of matrix $\left[d_{ij}\right]_n$. If $i_k = j_k$, $k = \overline{1,n_1}$, we will consider that all elements $d_{i_s j_t}$, $s = \overline{1,n_1}$, $t = \overline{1,n_1}$, form a square sub matrix of order n_1 on the main diagonal of matrix $\left[d_{ij}\right]_n$. Further it would be reasonable to consider sub matrices $\left[d_{ij}\right]_{n_1}$, where the numbers of rows and columns are represented by sequence $(1,2,\dots,n_1)$.

Let us consider sub matrix $\left[d_{ij}\right]_2$ on the main diagonal of matrix $\left[d_{ij}\right]_n$. Sub matrix $\left[d_{ij}\right]_2$ defines the only one cyclic permutation (2,1). The number of all permutations formed from the sub matrix $\left[d_{ij}\right]_3$ is equal to 6. Two of them are permutation (1,2,3) and (2,3,1). They are cyclic. The listed permutations form the set of all bypasses within TSP with cost matrix $\left[d_{ij}\right]_3$: (1,2,3,1), (1,3,2,1).

We correlate bipartite oriented graph (I,J,E), $|I| = |J| = 2$, $E = \{(1,2),(2,1)\}$ with permutation (2,1). It is easy to see that both permutations (1,2,3) and (2,3,1) are possible to be obtained from permutation (2,1). Permutation (1,2,3) is formed by removing arc (1,2) from graph (I,J,E) and adding arcs $(1,j)$, $j = 3$, and $(i,2)$, $i = 3$. Permutation (2,3,1) is formed by removing arc (2,1) and adding arcs $(2,j)$, $j = 3$, and $(i,1)$, $i = 3$ (fig. 1).

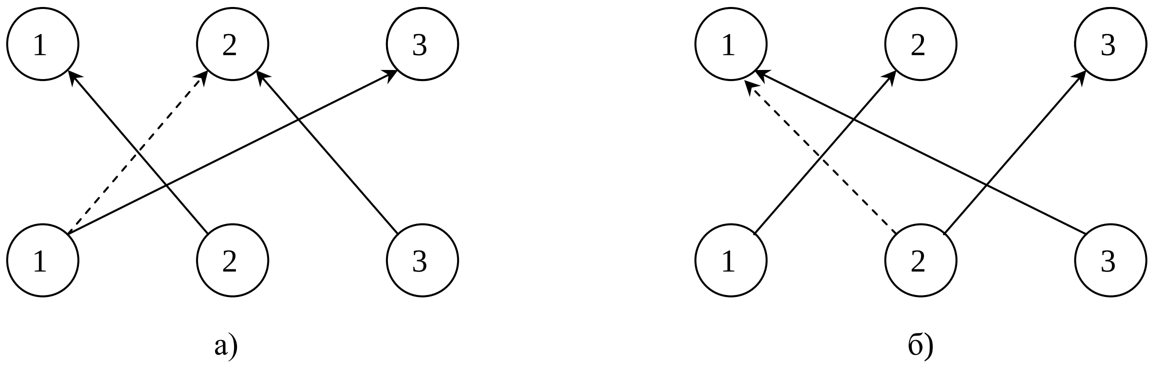

Figure 1: Transition to 2 permutations of dimension 3 from permutation of dimension 1

This remark causes the following generalization.

Let permutation of dimension $r+1$

$$p = \begin{pmatrix} 1 & \dots & l & \dots & k-1 & k & k+1 & \dots & s & \dots & r & r+1 \\ \tau[1] & \dots & \tau[l] & \dots & \tau[k-1] & k & \tau[k+1] & \dots & \tau[s] & \dots & \tau[r] & \tau[r+1] \end{pmatrix}$$

contain one cycle of length r and one cycle of length 1, i.e. it is represented by loop unrolling $p = p_1 p_2$, where

$$p_1 = \begin{pmatrix} 1 & \dots & l & \dots & k-1 & k+1 & \dots & s & \dots & r & r+1 \\ \tau[1] & \dots & \tau[l] & \dots & \tau[k-1] & \tau[k+1] & \dots & \tau[s] & \dots & \tau[r] & \tau[r+1] \end{pmatrix},$$
$$p_2 = \begin{pmatrix} k \\ k \end{pmatrix}.$$

Then any of r permutations of dimension $r+1$

$$p(l) = \begin{pmatrix} 1 & \dots & l & \dots & k-1 & k & k+1 & \dots & s & \dots & r & r+1 \\ \tau[1] & \dots & k & \dots & \tau[k-1] & \tau[l] & \tau[k+1] & \dots & \tau[s] & \dots & \tau[r] & \tau[r+1] \end{pmatrix},$$
$$l < k,$$

$$p(s) = \begin{pmatrix} 1 & \dots & l & \dots & k-1 & k & k+1 & \dots & s & \dots & r & r+1 \\ \tau[1] & \dots & \tau[l] & \dots & \tau[k-1] & \tau[s] & \tau[k+1] & \dots & k & \dots & \tau[r] & \tau[r+1] \end{pmatrix},$$
$$k < s,$$

consists of one cycle of length $r+1$.

Indeed, the contour of length $r+1$ that corresponds to permutation $p(l)$, can be obtained from p by the only one way, namely, by removing arc $(l, \tau[l])$ and adding arcs (l, k) and $(k, \tau[l])$, $l < k$, within the contour that corresponds to permutation p. By analogy, the contour of length $r+1$ that corresponds to permutation $p(s)$, is formed by removing arc $(s, \tau[s])$ and adding arcs (s, k) and $(k, \tau[s])$, $k < s$, from contour that corresponds to permutation p.

Thus, we have a simple way to build a subset of cyclic permutations of length n for matrix $\left[d_{ij}\right]_n$. The following steps have to be performed. We form $n-2$ sub matrices $\left[d_{ij}\right]_r$ on the main diagonal of matrix $\left[d_{ij}\right]_n$ from $\left[d_{ij}\right]_n$. Each sub matrix $\left[d_{ij}\right]_r$ has the same sequence $(1,2,\dots,r)$, $r = \overline{2, n-1}$, for the numbers of rows and columns. Further, r cyclic permutations of length $r+1$ are formed for each cyclic permutation of length r obtained from matrix $\left[d_{ij}\right]_r$. It is performed by following the mentioned steps on removing and adding the elements of matrix $\left[d_{ij}\right]_{r+1}$.

Such a method of forming a set of available solutions within sub matrices $\left[d_{ij}\right]_r$ of the given matrix of costs $\left[d_{ij}\right]_n$, $r = \overline{2, n-1}$, is well embedded in flowchart of building local sequences. It made possible to efficiently solve some generalizations of assignment task [22, 23]. Let us demonstrate how to apply the flowchart for building for polynomial time of bypass τ^o which is acceptable in accuracy for real input data.

At the initial moment of building τ^o let us assume that cyclic permutation (2,1) is given. The corresponding to it bypass $\tau_{o2} = (2,1,2)$ has cost $D(\tau_{o2}) = d_{12} + d_{21}$. Let $k = 2$, $j_1 = 2$, $j_k = 1$. We obtain two cyclic permutations for $l = 3$ in matrix $\left[d_{ij}\right]_3$ from permutation $\sigma_{ok} = (2,1)$. Its row and column numbers are presented with sequence (1,2,3): $(3, j_1, j_2)$, $(j_1, 3, j_2)$ (fig. 2). Let us define the costs of bypasses $\tau_{31} = (3, j_1, j_2, 3)$, $\tau_{32} = (j_1, 3, j_2, j_1)$:

$$D(\tau_{31}) = D(\tau_{o2}) + d_{j_2 3} + d_{3 j_1} - d_{j_2 j_1},$$

$$D(\tau_{32}) = D(\tau_{o2}) + d_{j_1 3} + d_{3 j_2} - d_{j_1 j_2}.$$

Bypass τ_{o3}, that gives $\min\{D(\tau_{31}), D(\tau_{32})\}$, allows building bypass τ_{o4} from four arcs after repetition of actions similar to performed ones for $l = 3$.

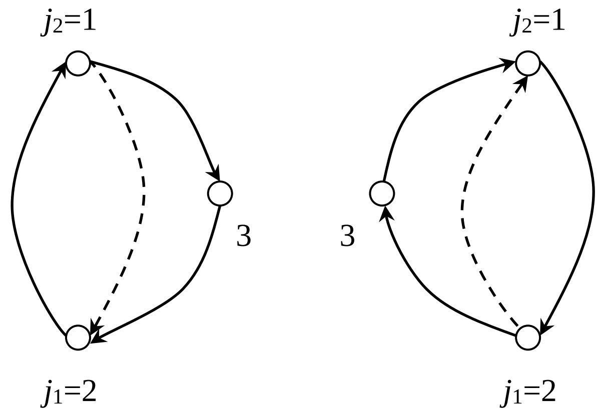

Figure 2: Formation of two bypasses of dimension 3 from bypass of dimension 2

The algorithm for building bypass τ^o is presented as following.

S0. The algorithm for finding valid rout τ^o by flowchart of building local sequences; $\left[d_{ij}\right]_n$– matrix of costs of order n in TSP, where $d_{ii} = \infty$, $i = \overline{1,n}$, $d_{ij} \in Z_o^+, i \neq j$; $j_1 = 2$, $k = 2$, $j_k = 1$, $\sigma_{ok} = (j_1, j_k)$, $D(\tau_{ok}) = d_{j_1 j_k} + d_{j_k j_1}$.

S1. $l = k + 1$; to build sequence $\sigma_l = (j_1, j_2, \ldots, j_s, \ldots, j_k, l)$ from permutation σ_{ok}; to form k cyclic permutations σ_{ls}:$\sigma_{l1} = (l, j_1, j_2, \ldots, j_s, \ldots, j_k)$,$\sigma_{l2} = (j_1, l, j_2, \ldots, j_s, \ldots, j_k)$,…, $\sigma_{ls} = (j_1, j_2, \ldots, j_{s-1}, l, j_s, \ldots, j_k)$, …, $\sigma_{lk} = (j_1, j_2, \ldots, j_s, \ldots, j_{k-1}, l, j_k)$ from σ_l; to calculate costs of bypasses that correspond to already built permutations σ_{ls}:

$$D(\tau_{l1}) = D(\tau_{ok}) + d_{j_k l} + d_{l j_1} - d_{j_k j_1},$$

$$D(\tau_{l2}) = D(\tau_{ok}) + d_{j_1 l} + d_{l j_2} - d_{j_1 j_2},$$

…

$$D(\tau_{ls}) = D(\tau_{ok}) + d_{j_{s-1} l} + d_{l j_s} - d_{j_{s-1} j_s},\ 2 \leq s \leq k-1,$$

$$D(\tau_{lk}) = D(\tau_{ok}) + d_{j_{k-1} l} + d_{l j_k} - d_{j_{k-1} j_k};$$

to find such bypass τ_{lj}, that

$$D\left(\tau_{lj}\right) = \min\{D(\tau_{ls}) | 1 \leq s \leq k\};$$

to update $k = l, \tau_{ok} = \tau_{lj}$;

S2. If $k = n$, then the end is $\tau^o = \tau_{ok}$, otherwise to define cyclic permutation σ_{ok} that corresponds to bypass τ_{ok} by removing the final element from it, go to step S1.

The proposed algorithm is characterized by low complexity of building a acceptable solution to τ^o. Let us estimate the time of its work. The algorithm executes $n - 2$ of steps S1. Every step S1 builds k permutations $\sigma_{(k+1)s}$, $s = \overline{1,k}$, $k = \overline{2,n-1}$, as a result of inserting element l before elements $j_1, j_2, \ldots, j_k$ of permutation σ_l, $l = k + 1$. Calculating k values $D(\tau_{ls})$ requires $3k$ operations of addition and subtraction and the search for the minimum number within unordered array of numbers $D(\tau_{ls})$ will be completed after comparison operations $k - 1$ have been performed. Thus, step S1 performs $5k - 1$ of elementary operations such as insertion, addition, subtraction and comparison. The building of transition τ^o finishes after execution of

$$\sum_{k=2}^{n-1}(5k-1) = 5\sum_{k=1}^{n-2} k = 5(n-1)(n-2)/2$$

elementary operations. Respectively, time complexity of the proposed algorithm is estimated with value $O(n^2)$.

Example 1. Let us build bypass τ^o for matrix

$$[d_{ij}]_6 = \begin{bmatrix} \infty & \boxed{3} & 1 & 7 & 7 & 1 \\ 3 & \infty & 7 & 7 & \boxed{9} & 9 \\ \boxed{1} & 10 & \infty & 7 & 9 & 1 \\ 8 & 7 & 9 & \infty & 6 & \boxed{6} \\ 9 & 7 & 8 & \boxed{1} & \infty & 1 \\ 1 & 9 & \boxed{1} & 1 & 6 & \infty \end{bmatrix}.$$

We update $j_1 = 2$, $k = 2$, $j_1 = 1$. The building of τ^o starts with permutation $\sigma_{o2} = (2,1)$. The cost of bypass $\tau_{o2} = (2,1,2)$ is equal to $D(\tau_{o2}) = d_{12} + d_{21} = 3 + 3 = 6$.

We update $l = 3$ and build permutation $\sigma_3 = (2,1,3)$, which produces two permutations $\sigma_{31} = (3,2,1)$, $\sigma_{32} = (2,3,1)$ and corresponding to them bypasses $\tau_{31} = (3,2,1,3)$, $\tau_{32} = (2,3,1,2)$.

We calculate $D(\tau_{31}) = D(\tau_{o2}) + d_{13} + d_{32} - d_{12} = 6 + 1 + 10 - 3 = 14$, $(\tau_{32}) = D(\tau_{o2}) + d_{23} + d_{31} - d_{21} = 6 + 7 + 1 - 3 = 11$. Since $\min\{D(\tau_{31}), D(\tau_{32})\} = D(\tau_{32})$, we set $\tau_{o3} = (2,3,1,2)$.

We obtain cyclic permutation $\sigma_{o3} = (2,3,1)$ at step S2; we go to perform step S1.

We update $l = 4$, $\sigma_l = (2,3,1,4)$, we form permutations $\sigma_{41} = (4,2,3,1)$, $\sigma_{42} = (2,4,3,1)$, $\sigma_{43} = (2,3,4,1)$. Let us define the cost of bypasses τ_{41}, τ_{42}, τ_{43}:

$$D(\tau_{41}) = D(\tau_{o3}) + d_{14} + d_{42} - d_{12} = 11 + 7 + 7 - 3 = 22,$$

$$D(\tau_{42}) = D(\tau_{o3}) + d_{24} + d_{43} - d_{23} = 11 + 7 + 9 - 7 = 20,$$

$$D(\tau_{43}) = D(\tau_{o3}) + d_{34} + d_{41} - d_{31} = 11 + 7 + 8 - 1 = 25.$$

We select bypass $\tau_{42} = \tau_{o4}$ and cyclic permutation $\sigma_{o4} = (2,4,3,1)$.

At step S1 we obtain bypass $\tau_{o5} = \tau_{52} = (2,5,4,3,1,2)$, $D(\tau_{o5}) = 23$ for $l = 5$.

The algorithm completes calculations at $k = 6$ by building bypass $\tau^o = \tau_{64} = (2,5,4,6,3,1,2)$, where $D(\tau_{64}) = 9 + 1 + 6 + 1 + 1 + 3 = 21$ (fig. 3).

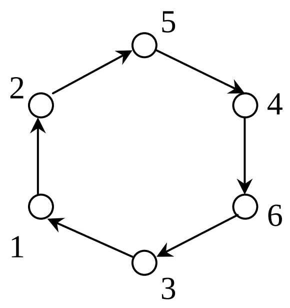

Figure 3: The built bypass-solution

Example 2. The proposed algorithm builds optimum solution $\tau^o = (2,4,1,3,2)$, $D(\tau^o) = 8d$ by using "uncomfortable" example of symmetrical TSP presented with matrix $[d_{ij}]_4$ and corresponding to it graph (fig. 4), whereas procedure NN "go to the nearest", which is characterized with the same performance, finds bypass $(2,3,4,1,2)$ with the maximum cost that equals $13d$.

$$[d_{ij}]_4 = \begin{bmatrix} \infty & 2d & 2d & 8d \\ 2d & \infty & d & 2d \\ 2d & d & \infty & 2d \\ 8d & 2d & 2d & \infty \end{bmatrix}$$

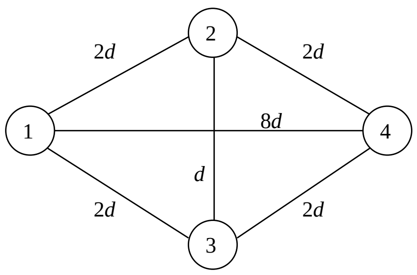

Figure 4: Cost matrix and corresponding graph

In practice, the algorithm works faster than the "go to the nearest" heuristics, since its performance estimation cn^2 has coefficient c with less value.

Based on the results of the computational experiment, the proposed algorithm is more precise than NN procedure. In any case, a large number of numerical examples of TSP has not got any where bypass τ_{NN}, formed by the rule "go to the nearest" could have less cost than the cost of bypass τ^o.

The optimal solutions τ^* and corresponding to them values $D(\tau^*)$ are obtained implementing brunch and bound method using programming language C++ for the cases of TSP with matrices $[d_{ij}]_n$ of limited dimension ($n \le 40$). At $n \le 40$ bypass of τ^* minimum cost $D(\tau^*)$ are built for acceptable time (~40-80 sec). As it was expected, with increasing n of value $D(\tau^o)/D(\tau^*)$ demonstrates the unlimited growth in TSP without any additional restrictions for value d_{ij}.

It is interesting to observe the behavior of the algorithm of building τ^o for TSP that is restricted on matrix $[d_{ij}]_n$ which satisfies the triangle inequality (2).

Let us note that it is possible to build a case of TSP with matrix $[\overline{d_{ij}}]_n$. It has to satisfy restrictions in the form of (2) for the time $O(n^3)$ following the case of TSP with randomly generated values of matrix $[d_{ij}]_n$. Matrix $[\overline{d_{ij}}]_n$ is called the closure $[d_{ij}]_n$, if $\overline{d_{ij}}$ – is the length of the nearest path from i to j in complete graph with n vertices $\{1,2,\dots,n\}$, where the edge length $\{i,j\}$ is equal to d_{ij}.

The transformation of matrix $[d_{ij}]_n$ into matrix $[\overline{d_{ij}}]_n$ is implemented by using a well-known Floyd-Warshall algorithm:

```
S0. Floyd-Warshall algorithm: matrix $[\overline{d_{ij}}]_n$, where $\overline{d_{ij}}$ – is the nearest distance with the given distances $[d_{ij}]_n$ is for matrix $[d_{ij}]_n$ with integral elements.
S1. For all $i \ne j$ to perform $\overline{d_{ij}} = d_{ij}$.
S2. For all $i = 1,2,\dots,n$ to perform $\overline{d_{ii}} = \infty$.
S3. For all $j = 1,2,\dots,n$ to perform
      for all $i = 1,2,\dots,n$, $i \ne j$, to perform
```

for all $k = 1,2,\ldots,n$, $k \neq j$, to perform

$$\overline{d_{ik}} = \min\{\overline{d_{ik}}, \overline{d_{ij}} + \overline{d_{jk}}\}.$$

The flowchart used to experimentally estimate the accuracy of algorithm building τ^o for the matrix of costs $[\overline{d_{ij}}]_n$, that satisfies triangle inequality, is presented in fig. 5.

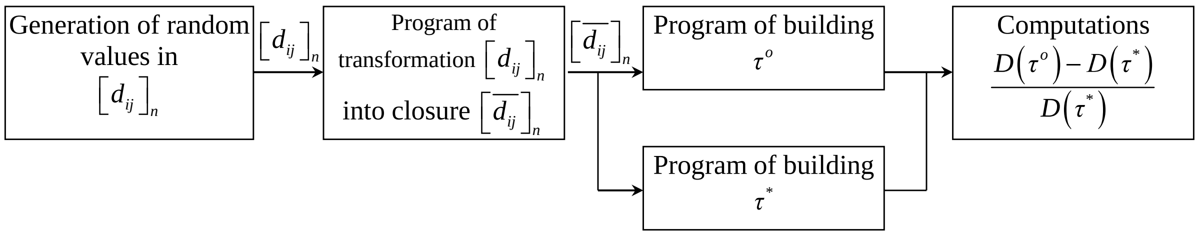
Generation of random values in $[d_{ij}]_n$
$[d_{ij}]_n$
Program of transformation $[d_{ij}]_n$ into closure $[\overline{d_{ij}}]_n$
$[\overline{d_{ij}}]_n$
Program of building τ^o
Program of building τ^*
Computations $\frac{D(\tau^o) - D(\tau^*)}{D(\tau^*)}$

Figure 5: The flowchart of algorithm accuracy estimation

The experiment, performed by following the given flowchart, demonstrates that the increase of the order of matrix $[\overline{d_{ij}}]_n$ goes alongside with the increase of relative error $\left(D(\tau^o) - D(\tau^*)\right)/D(\tau^*)$. The most promising result of the experiment is the absence of a case with the cost of bypass τ^o. It could exceed the cost of optimal bypass τ^* in more than one and a half time.

It is important to mention, that the TSP with the triangle inequality belongs to the class of NP - hard problems solved by using ε- approximate polynomial algorithms. To put it mildly, such algorithms are known to have analytically expressed relative error that depends on the dimension of task input and is restricted with $\varepsilon > 0$. The highest achievement in study of TSP with triangle inequality is 1/2-approximate algorithm suggested by Christofides. Christofides algorithm is known to have the relative error restricted with a half of optimal solution cost and updated as a result of performing actions with such objects as bipartite, minimal spinning tree, Eulerian multigraph and Eulerian path. Computational operations with the listed objects have high level of performance that is why, time estimation of Christofides algorithms that is equal to $O(n^4)$, is relatively high. Experimentally obtained characteristics of the proposed algorithm open the prospect of its use in numerous applications of the TSP and at the same time raise a number of questions related mainly to the search for resources to reduce the relative error of acceptable solutions. There is one more important question. How to use the idea of converting cyclic permutation of length r into cyclic permutation of length $r + 1$ in order to find an error that does not depend on n in TSP with triangle inequality? The proposed algorithm proves to find solution to the task with the same error as Christofides algorithm has, but for less time $O(n^2)$ and it could become a notable achievement in the field of combinatorial optimization.

3. Acknowledgements

The authors are deeply grateful to the management of their higher education institutions − Zhytomyr Polytechnic State University and Kherson State Maritime Academy − for providing proper working conditions, which allowed them to perform this work.

We are very grateful to the organizers of the International Scientific and Practical Conference "Information Control Systems and Technologies" for advice, assistance and support.

We should also thank our families for their understanding, patience, care and love.

And, finally, a huge thank you to our constant assistant Serhiy Fomenko.

4. References

[1] D. R. Fulkerson, O. A. Gross, Incidence and interval graphs, Pacific J. Math., volume 15, 3 (1965) 835–855.

[2] M. L. Fisher, G. L. Nemhauser, L. A. Wolsey, An analysis of approximations for finding a maximum weight Hamiltonian circuit, Oper. Res., volume 27, 6 (1979) 799–809.

[3] Yu. G. Stoyan, S. V. Yakovlev, Matematicheskie modeli i optimizatsionnyie metodyi geometricheskogo proektirovaniya, Naukova dumka, Kiev, 1986.

[4] E. H. Gimadi, A. I. Serdyukov, O nekotoryih rezultatah dlya zadachi kommivoyazhera na maksimum, Diskretnyiy analiz i issledovanie operatsiy, seriya 2, t. 8, 1 (2001) 22–39.

[5] A. E. Baburin, E. H. Gimadi, Ob asimptoticheskoy tochnosti odnogo algoritma resheniya zadachi kommivoyazhera na maksimum, Diskretnyiy analiz i issledovanie operatsiy, seriya 1, t. 9, 4 (2002) 23–32.

[6] I. H. Sigal, Algoritmyi dlya resheniya bikriterialnoy zadachi kommivoyazhera bolshoy razmernosti, ZhVM i MF, t. 34, 1 (1994) 44–57.

[7] C. H. Papadimitriu, M. Yannakakis, The traveling salesman problem with distance one and two, Math. Oper. Res., volume 18, 1 (1993) 1–11.

[8] Yu. G. Stoyan, O. O. Emets, Teoriya i metodi evklidovoyi kombinatornoyi optimizatsiyi, ISDO, Kiev, 1993.

[9] M. R. Garey, D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman & Co., New York, NY, 1979.

[10] C. H. Papadimitriou, The Euclidean travelling salesman problem is NP-complete, Theor. Comput. Sci. 4 (1977) 237–244.

[11] G. Laporte, The traveling salesman problem: An overview of exact and approximate algorithms, European Journal of Operational Research, volume 59, issue 2, 1992, pp. 231−247. doi:10.1016/0377-2217(92)90138-Y.

[12] T. Bektas, The multiple traveling salesman problem: an overview of formulations and solution procedures, Omega, volume 34, 3 (2006) pp. 209–219.

[13] J. K. Lenstra, Clustering a data array and the traveling salesman problem, Oper. Res., volume 22, 2 (1974) 413–414.

[14] J. D. C. Little, K. G. Murty, D. W. Sweeny, C. Karel, An Algorithm for the Traveling Salesman Problem, Oper. Res. 11 (1963) 972–989.

[15] M. Held, R. M. Karp, The Traveling Salesman Problem and Minimum Spanning Trees, Oper. Res. 18 (1970) 1138–1162.

[16] J. L. Santos, A. Oliveira, Traveling Salesman Problem in a Geographic Information Management System, in: M. Cruz, C. Parés, P. Quintela, (Eds.), Progress in Industrial Mathematics: Success Stories, volume 5 of SEMA SIMAI Springer Series, Springer, Cham, 2021. doi:10.1007/978-3-030-61844-5_8.

[17] Y. Veksler, E. D. Rimon, Evasive Navigation of an Autonomous Mobile Robot in Hostile Unknown Environments, in: S. M. LaValle, M. Lin, T. Ojala, D. Shell, J. Yu, (Eds.), Algorithmic Foundations of Robotics XIV, volume 17 of Springer Proceedings in Advanced Robotics, Springer, Cham, 2021. doi:10.1007/978-3-030-66723-8_4.

[18] K-C Ying, S-W Lin, Maximizing cohesion and separation for detecting protein functional modules in protein-protein interaction networks, PLoS ONE 15(10):e0240628, 2020. doi:10.1371/journal.pone.0240628.

[19] D. G. Kendall, Incidence matrices, interval graphs and seriation in archaeology, Pacific J. Math., volume 28, 2 (1969) 565–570.

[20] G. Laporte, The seriation problem and the traveling salesman problem, J. Comput. and Appl. Math., volume 4, 4 (1978) 259–268.

[21] A. V. Panishev, D. D. Plechystyy, Do pytannia pobudovy marshrutu komivoiazhera, Visnyk Zhytomyrskoho inzhenerno-tekhnolohichnoho instytutu 20 (2002) 120–124.

[22] A. V. Panishev, O. A. Podolyaka, E. V. Skakalina, Effektivnyiy algoritm rasparallelivaniya rabot na neidentichnyih mashinah, in: Sb. nauch. tr. Aviatsionno-kosmicheskaya tehnika i tehnologiya, vyip. 13, Gos. Aerokosm, Harkov, 1999, s. 136–146.

[23] A. V. Panishev, I. V. Skripina, O. V. Skakalina, Effektivnoe postroenie optimalnyih resheniy v zadache o naznachenii transportnogo tipa, in: Sb. nauch. tr. Avtomobilnyiy transport, vyip. 4, HTADTU, Harkov, 2000, s. 63–65.

[24] H. Papadimitriu, K. Stayglits, Kombinatornaya optimizatsiya. Algoritmy i slozhnost, Mir, Moskva, 1985.

[25] C. H. Coombs, J. E. K. Smith, On the detection of structure in attitudes and developmental processes, Psychological Review, volume 80, 5 (1973) 337–351.

[26] S. R. Kosaraju, I. K. Park, C. Stein, Long tours and shot superstrings, in: Proceedings of the 35th. Annual IEEE Symposium on Foundation of Computer Science, Comput. Soc. Press, Los Alamitos, SA, 1994, pp. 166–177.

[27] A. I. Barvinok, Two algorithmic results for traveling salesman problem, Math. Oper. Res., volume 21, 1 (1996) 65–84.

[28] R. Hassin, S. Rubinstein, An approximation algorithm for the maximum traveling salesman problem, Inform. Process. Lett., volume 67, 3 (1998) 125–130.

[29] S. Axsaeter, On scheduling in a semi-ordered flow shop without intermediate queues, AIIE Trans., volume 14, 2 (1982) 128–130.

[30] I. Graham, Object-oriented methods: principles & practice, 3rd. ed., Addison-Wesley, 2001.