CYBERNETICS AND PHYSICS, VOL. 12, NO. 3, 2023, 194–200

PSEUDOGEOMETRIC VERSION OF THE TRAVELING SALESMAN PROBLEM: APPLICATION IN QUANTUM PHYSICS MODELS AND A HEURISTIC VARIANT OF POINT PLACEMENT

Boris Melnikov
Department of Computational Mathematics and Cybernetics, Shenzhen MSU – BIT Univ., China
bormel@smbu.edu.cn, bormel@mail.ru

Yulia Terentyeva
Department of Communication Network, Center for Information Technologies and Systems for Executive Authorities, Russia
terjul@mail.ru

Dmitrii Chaikovskii*
Department of Computational Mathematics and Cybernetics, Shenzhen MSU – BIT Univ., China
dmitriich@smbu.edu.cn, mitichya@yandex.ru

Article history:
Received 09.08.2023, Accepted 26.11.2023

Abstract

The geometric version of the traveling salesman problem (TSP) has been extensively studied, leading to the development of various approaches for solving its special cases. However, these algorithms often fall short when applied to problems beyond the geometric TSP. In this paper, we explore the pseudo-geometric TSP version, a generalization of the geometric TSP, and propose an adapted geometric algorithm for solving its specific instances. We leverage the knowledge of error bounds to estimate the reconstruction error of the TSP solution even when using geometric approaches for the pseudo-geometric TSP. This allows us to achieve reliable results despite uncertainties or noise in the data. We provide a concise description of our algorithmic adaptation and present the results of computational experiments to demonstrate its effectiveness.

Key words

Mathematical model, traveling salesman problem, geometric version, pseudo-geometric version, geometric approach, heuristic algorithms.

1 Introduction

The traveling salesman problem (TSP) is a well-known optimization problem whose goal is to find the shortest possible route that passes through a set of points (cities) and returns to the starting point. TSP can be formulated in different ways, depending on the problem instance.

Among the various formulation options, we note [Garey and Johnson, 1979; Gutin and Punnen, 1997; Hromkovič, 2003; Hromkovič, 2004]; of course, all these options are very similar. According to the classification given in [Hromkovič, 2004], the traveling salesman problem is an example of an optimization problem that belongs to the most complex class $\mathrm{NPO}(V)$. This class contains all optimization problems that, under an additional assumption such as $\mathrm{P} \neq \mathrm{NP}$, have a time complexity that cannot be bound by any polylogarithmic function for all possible polynomial algorithms. Let us describe the TSP according to [Hromkovič, 2003; Hromkovič, 2004]; in this paper, we shall not repeat the detailed description of the problem statement.

One of the most common variants of the TSP is the geometric TSP [Hromkovič, 2004], which is characterized by a set of points and their distances, where the distances are defined by the Euclidean distance between the points. In the case of the geometric TSP, we can use a number of algorithms to find an approximate solution to the problem. One of the most popular algorithms is the nearest neighbor algorithm, which starts from a random point and repeatedly selects the nearest unvisited point as the next point to visit. Another popular algorithm is the convex hull method, which uses the convex hull of the set of points to reduce the number of points that need to be considered. These algorithms can be used to find approximate solutions to the problem with a high degree of accuracy.

The TSP can be defined also as a random problem, if the points are not defined by their coordinates, but by a set of random numbers. In this case, it is not possible to

*Corresponding author.

use geometric algorithms to find solutions, and instead, more general optimization techniques must be used. For example, a common approach is to use the branch and bound method, when we systematically enumerate all possible solutions to a problem and determine which is the best based on some objective criteria. This is done by dividing the problem into smaller sub-problems (or branches) and solving each sub-problem separately. The solutions to these sub-problems are then compared, and the best solution is chosen as the overall best solution.

However, in some cases, the problem instance may not be defined by the Euclidean distance between the points, but by some other metric that has some degree of randomness or noise built into it. This variant of the TSP is called the pseudo-geometric TSP [Melnikov, 2001], where the initial data are formed as follows:

- for a given geometric TSP, a special vector
$$R = (r_1, \ldots, r_m)$$
is added, where $m = |E|$;
- all r_i are independent identically distributed random variables with a normal distribution with $\mu = 1$ and a given "acceptable" value of the square deviation σ;
- in practice, we make sure that the random variable gets into the borders $[0, 2]$;
- each element of the cost matrix $(c(u, v)$ (we denote c_i for some $i \in \{1, \ldots, m\}$) is replaced for $c_i \cdot r_i$.

It is essential to highlight that the geometric TSP-version can be regarded as a special case of pseudo-geometric one with the deviation value $\sigma = 0$.

The approach to solve the pseudo-geometric variant of the traveling salesman problem, as discussed in this article aligns closely with the technique applied to the problem of reconstructing the distance matrix between DNA chains. Both problems utilize a pseudo-optimal configuration, as mentioned in [Melnikov et al., 2018; Melnikov et al., 2022]. Delving into more detail, both problems involve the placement of all absent elements (either zero or equivalent to infinity) into the depleted matrix according to certain rules. From a formal point of view, iterative algorithms similar to the rapid descent method can be utilized, however, with a few hundred variables, such methods are very impractical for implementation. Consequently, in both cases, we use heuristic methods which involve carefully composed descriptions of sequentially placed (or replaced) elements.

In the previous paragraph, the connection of two mentioned problems (the pseudo-geometric version of the TSP and the restoration of the DNA matrix) was noted *at the level of solution algorithms*. However, it is possible to observe the connection between these two problems *at the level of their statements*. Unfortunately, this thing is very little reflected in the literature so far; in addition to our works, we can give the only such reference [Korostensky and Gonnet, 2000].

Thus, heuristics methods yield practically highly successful results, and this article, along with its proposed follow-up, primarily concentrates on one of these problem.

The structure of this paper is outlined as follows. In Section 2, we provide a motivation for considering such problems, in particular, its use in some quantum physics models. In Section 3, we provide a theoretical basis for our approach. In Section 4, we describe in detail the method developed for solving the pseudo-geometric TSP, including the techniques for classifying input data and pseudo-optimal point placement. Finally, in the conclusion (Section 5), we provide a detailed outline for the proposed continuation.

2 Motivation. Hamilton cycles in quantum physics models

In this section, we provide a motivation for considering such problems, in particular, its use in some quantum physics models.

The "ecological niche" of the geometric version of the TSP is well-studied, with many approaches to solving its particular cases, developed over the years, [Gutin and Punnen, 1997] etc. Some examples of algorithms belonging to this group are the "onion-peeling" [Sing, 2012] and the "elastic network" [Somhom, 1999] algorithms, which have a relatively low computational complexity and can provide good solutions for special cases with millions of points that are practically optimal. However, these algorithms are usually not applicable outside of the geometric TSP-version, so this article is aimed at exploring the possibility of applying similar algorithms to other TSP-versions. Therefore, our goal is to create a robust heuristic approach *to solving pseudo-geometric TSP using techniques from geometric TSP*. By doing so, we can better understand the complexities of this challenging optimization problem and gain insight into more efficient ways to solve it. Additionally, this research can provide a platform for developing more efficient algorithms in the future.

Mathematical models and algorithms, originally developed to solve the traveling salesman problem, were also used to solve other applied problems, such as:

- to build optimal transport routes and select the optimal trajectory;
- for sequencing nucleotide sequences of biopolymers, [Korostensky and Gonnet, 1999];
- for the calculation of string similarity [Melnikov and Panin, 2012; Makarkin et al., 2013];
- for the development of practical algorithms for the study of specially defined infinite grammatical structures [Melnikov, 1996; Melnikov and Kashlakova, 2000];
- for the construction of evolutionary trees [Korostensky and Gonnet, 2000].

In addition, the TSP has been applied for the DNA analysis; we shall return to this issue in this section. After sequencing, a set of small-length strings is obtained, and an oriented graph is created with weights that represent the degree of similarity between them. In this setting, the longest common superstring has the maximum cost. This problem is close to the pseudo-geometric TSP-version, as the length of overlapping sections can be arbitrary (within certain boundaries). Thus, there is potential for further work in this area.

The pseudo-geometric approach can be used to solve other tasks, such as finding the optimal route for manipulating the state of one or more photons in a *quantum system*. In this approach, the *quantum states* are represented by points in a geometric space, [Aharonov et al., 1993] etc. However, the distance between two states is not determined by the distance between their corresponding points in the space. Instead, the distance is defined using a metric that captures the similarity or dissimilarity between the states, [Trugenberger, 2002; Tomilin and Il'ichov, 2023] etc. For example, the TSP can be used with points corresponding to different states of photons and in order to find the optimal route for manipulating the state of the photons. In addition, it is possible to use *quantum computers* for solutions of such problems, with the usage of technologies such as quantum graph walking and *quantum annealing*.

The *quantum graph walking* (QWG) is a quantum algorithm that uses quantum states to find the shortest path in a graph, while quantum annealing is used to control quantum systems through various influences such as electric and magnetic fields, [Childs et al., 2002; Childs, 2009] etc. By combining quantum graph walking and quantum annealing, we can solve the traveling salesman problem in a quantum system. This approach can be used to find the optimal path of a *quantum particle* in quantum mechanics, and then optimize and control the trajectory using quantum annealing. These techniques can be used to manipulate the state of photons in a quantum system, [Childs et al., 2002; Sergeenko et al., 2020] etc., and implement various quantum operations such as encryption and decryption in quantum cryptography and measurements in quantum metrology.

Quantum graphs have already been mentioned in the previous paragraph, but, apparently, the information given there is sufficient to describe the motivation for their study within the framework of the problems we are considering. However, of course, for further presentation, at least a brief description of the variants of quantum graphs is useful, we shall bring them close to [Freedman et al., 2007], and also use some material from the Internet.

Thus, a quantum graph is a graph in which each edge is assigned a length and a differential equation is given on each edge. An important example is an electrical network consisting of wires (edges) connected in transformer substations (vertices). In this case, the differential equations describe the voltage on the wires, and the boundary conditions at the vertices provide a zero sum of current on all incoming and outgoing edges of each vertex.

In addition to the subject areas we have mentioned, they have found wide application in other subject areas related to physics:

- in models of quantum chaos systems;
- in the study of wave guides;
- for modeling photonic crystals;
- in the so-called mesoscopic physics, where quantum graphs are used to theoretically substantiate the concepts considered there.

In addition to solving differential equations on a quantum graph for specific applications, issues of controllability are studied (what input effect ensures the transition of the system to the desired state, for example, to ensure sufficient electrical power at all substations) and identification of systems (how and where it is necessary to measure any value in order to obtain the necessary information about the state of the system, for example, pressure measurement in the water supply system to detect water leakage).

3 The theoretical substantiation for the application of a heuristic method

Thus, we use the geometric approaches for the pseudo-geometric TSP. In this section, we provide a theoretical basis for it.

In the current paper, we propose to use some of the geometric approaches for solution of the pseudo-geometric TSP, in which the distance between two points is defined by a function that takes into account both the Euclidean distance between the points and a random variable, which is referred to as the "dispersion". The dispersion represents the degree of randomness or noise in the problem, and it is used to introduce a level of uncertainty into the problem, which makes it more difficult to solve.

For the TSP in 2D, we need to reconstruct the noisy path and estimate the error bound for the total distance. However, this is an ill-posed problem and even a small error in the initial data can result in a significant error in the calculated derivatives. Let us consider a set of points

$$P = (x_i, y_i) \text{ with } i = 1, \dots, N$$

in the unit square, where the noisy coordinates are (x_i^δ, y_i^δ). Moreover, we assume the following:

$$\max\left(|y(i) - y_i^\delta|, |x(i) - x_i^\delta|\right) \leqslant \delta,$$

where δ is a known level of noise or dispersion in the data. As we know, the TSP requires finding the shortest closed path that visits each point in P exactly once.

We can represent the true path as a sequence of indices $s = (s_1, s_2, \dots, s_N)$, where the index s_i refers to the i-th point in the true path. This sequence of indices can be

found by techniques such as *nearest neighbors* or *greedy algorithms*. The noisy path will be given by a sequence of noisy coordinates $(x_{(}s_i)^\delta, y_{(}s_i)^\delta)$. We shall consider the problem in a piece-wise manner using smoothing cubic splines.

In some applications, considering a smooth path using splines over nearest neighbor or linear interpolation can be beneficial for several reasons. Smooth paths are important when the path is meant to be followed by a vehicle or a robot, as they ensure feasible and efficient movement without challenging or impossible-to-navigate sharp turns or discontinuities. Additionally, spline interpolation can provide a more accurate approximation of the true underlying path by reducing the impact of noise on the final result, whereas linear interpolation may be more sensitive to noise and nearest neighbor interpolation may not even create a continuous path. Smooth paths also tend to be more visually appealing and easier to understand when visualizing the TSP results. Lastly, if the true underlying path is smooth, using splines can yield a better approximation than linear interpolation or nearest neighbor methods.

To construct a piece-wise cubic spline for the 2D TSP-path, we can consider a parametric representation of the path. Let t be the parameter that varies from 0 to 1, with t_i corresponding to the point (x_{s_i}, y_{s_i}).

For each consecutive pair of points (x_{s_i}, y_{s_i}) and $(x_{s_{i+1}}, y_{s_{i+1}})$, we can define two cubic functions

$$X_i(t) \text{ and } Y_i(t) \quad \text{for } t \in [t_i, t_{i+1}] \equiv \Omega_i$$

in the following way.

$$\begin{aligned} &X_i(t) = \\ &\quad a_{x,i} + b_{x,i}(t-t_i) + c_{x,i}(t-t_i)^2 + d_{x,i}(t-t_i)^3, \\ &Y_i(t) = \\ &\quad a_{y,i} + b_{y,i}(t-t_i) + c_{y,i}(t-t_i)^2 + d_{y,i}(t-t_i)^3. \end{aligned}$$

The coefficients $a_{x,i}$, $b_{x,i}$, $c_{x,i}$, $d_{x,i}$, $a_{y,i}$, $b_{y,i}$, $c_{y,i}$, and $d_{y,i}$ can be determined by imposing the following conditions.

1. The cubic functions pass through the points:

$$\begin{aligned} X_i(t_i) &= x_{s_i}, \\ X_i(t_{i+1}) &= x_{s_{i+1}}, \\ Y_i(t_i) &= y_{s_i}, \\ Y_i(t_{i+1}) &= y_{s_{i+1}}. \end{aligned}$$

2. The first and second derivatives are continuous at the junctions:

$$\begin{aligned} X_i'(t_{i+1}) &= X_{i+1}'(t_{i+1}), \\ X_i''(t_{i+1}) &= X_{i+1}''(t_{i+1}), \\ Y_i'(t_{i+1}) &= Y_{i+1}'(t_{i+1}), \\ Y_i''(t_{i+1}) &= Y_{i+1}''(t_{i+1}). \end{aligned}$$

We denote the sets of piece-wise functions as:

$$X(t) = \begin{cases} X_1(t), \text{ for } t \in [t_1, t_2], \\ X_2(t), \text{ for } t \in [t_2, t_3], \\ \dots, \\ X_N(t), \text{ for } t \in [t_N, t_{N+1}]. \end{cases}$$

and

$$Y(t) = \begin{cases} Y_1(t), \text{ for } t \in [t_1, t_2], \\ Y_2(t), \text{ for } t \in [t_2, t_3], \\ \dots, \\ Y_N(t), \text{ for } t \in [t_N, t_{N+1}]. \end{cases}$$

Then we define the functional $\Phi[X, Y]$ as follows:

$$\begin{aligned} \Phi[X, Y] \equiv \frac{1}{N} \cdot \sum_{i=1}^{N} \Big(\big(x_{s_i}^\delta - X(t_i)\big)^2 + \big(y_{s_i}^\delta - Y(t_i)\big)^2 \Big) \\ + \alpha \cdot \Big(\big|X''(t)\big|^2_{L^2(\Omega)} + \big|Y''(t)\big|^2_{L^2(\Omega)}, \Big); \end{aligned} \quad (1)$$

here, the domain Ω is the union of all the segments' domains, i.e.,

$$\Omega = \bigcup_{i=1}^{N} \Omega_i.$$

Let α be such that the minimizing elements $X_\alpha(t)$ and $Y_\alpha(t)$ of $\Phi[X, Y]$ satisfy:

$$\frac{1}{N} \cdot \sum_{i=1}^{N} \Big(\big(x_{s_i}^\delta - X_\alpha(t_i)\big)^2 + \big(y_{s_i}^\delta - Y_\alpha(t_i)\big)^2 \Big) = \delta^2.$$

We find the minimizing elements $X_\alpha(t)$ and $Y_\alpha(t)$ of $\Phi[X, Y]$:

$$(X_\alpha(t), Y_\alpha(t)) = \arg\min_{X,Y} \Phi[X, Y]. \quad (2)$$

These minimizing cubic spline functions $X_\alpha(t)$ and $Y_\alpha(t)$ will provide the best approximation of the true path, given the noisy data and the chosen smoothness parameter α.

The total distance can be approximated as follows:

$$D_{TSP} \approx \int_{t_1}^{t_N} \sqrt{(X_\alpha'(t))^2 + (Y_\alpha'(t))^2}\, dt\,.$$

Remark that we used piece-wise cubic splines.

With the piece-wise cubic spline, it is possible obtain a smooth approximation of the TSP path, that is less sensitive to the noise in the data points, leading to a more accurate estimation of the total distance.

Then according to [Hanke and Scherzer, 2001], the total error bound for the TSP path can be expressed in the following way:

$$\begin{aligned}&\left\|(X_\alpha(t),Y_\alpha(t))-(X_{s_i}(t),Y_{s_i}(t))\right\|_{H^1(\Omega)}\leqslant\\&\leqslant\sqrt{8}\Big(h\big(\|X''_\alpha(t)\|_{L^2(\Omega)}+Y''_\alpha(t)\|_{L^2(\Omega)}\big)+\\&+\sqrt{\delta}\big(\|X''_\alpha(t)\|^{1/2}_{L^2(\Omega)}+Y''_\alpha(t)\|^{1/2}_{L^2(\Omega)}\big)\Big).\end{aligned}\tag{3}$$

This error bound depends on the smoothness of the true path segments and the noisy level δ. If

$$\begin{aligned}\mathtt{distance}_h=\max_{1\leqslant i<n}\big((x_{(}s_{(}i+1))-x_{(}s_i))^2+\\+(y_{(}s_{(}i+1))-y_{(}s_i))^2\big)^{1/2}\end{aligned}$$

and δ are sufficiently small, then the reconstruction error will be minimal, resulting in an accurate approximation of the total distance for the TSP.

It is important to note that the error bound provided here depends on the assumption that we know the correct order of points in the TSP path. In practice, solving the TSP involves finding the optimal order of points, which is a computationally challenging problem. The error bound given here can be used as an estimate of the error for the TSP when the correct order of points is known or assumed.

If we have knowledge of the error bounds, it enables us to estimate the reconstruction error of the TSP solution, even when applying geometric approaches to a pseudo-geometric version of the TSP problem. This means that, by understanding the limits of our approximation, we can still achieve reliable results when addressing the TSP instances that possess inherent geometric properties, *despite any uncertainties or noise in the data*.

4 Geometric approach: pseudo-optimum placement of the points

In this section, we describe in detail the method developed for solving the pseudo-geometric TSP, including the techniques for classifying input data and pseudo-optimal point placement.

The problem of pseudo-recovery the original coordinates of a set of points is more significant than the problem of classifying them (i.e., determining their class based on the known cost matrix in the case of the TSP).

Our algorithm for pseudo-recovery involves a simpler problem: restoring the location of points in a geometric TSP-version using the distance matrix defined with the function

$$c : E \to \mathbb{N}_0.\tag{4}$$

This problem can be easily solved using the following method.

- Select two arbitrary points v_1 and v_2 from the set of vertices of the graph V.
- Place v_1 at the origin, and v_2 at the point $(0, c(v_1, v_2))$.
- Find the coordinates of each of the following points (let it be u with the coordinates (x, y)), choose two points (let v and w) that have already been placed and solve the system of equations:

$$\begin{cases}(x-x_v)^2+(y-y_v)^2=(c(u,v))^2\\(x-x_w)^2+(y-y_w)^2=(c(u,w))^2.\end{cases}\tag{5}$$

However, when this algorithm is applied to the distance matrix (4) in the pseudo-geometric TSP-version, it generally does not result in the original location of the points in the geometric TSP-version. Additionally, due to the possible violation of the triangle inequality in the given distance matrix, the algorithm may not be applicable.

As we already said, the main focus of this paper and its prospective continuation is a modified algorithm proposed by the authors for solving a particular case of the TSP which involves the use of the above algorithm for restoring the location of points in the case of a geometric TSP-version. Additionally, it is worth noting that this algorithm can be applied to any particular case of the TSP, but its use is hardly advisable in most "random" or different from pseudo-geometric special cases of the TSP.

Thus, despite the fact that the application of algorithms similar to the above is, generally speaking, impossible for the pseudo-geometric TSP-version, we are trying the same algorithms to solve the problem of the location of cities, actually solving the minimization problem for the following specially computed discrepancy ("badness")

$$\sqrt{\frac{2}{n\cdot(n-1)}\cdot\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\big(c(u_i,u_j)-\tilde{c}(u_i,u_j)\big)^2},\tag{6}$$

where:

- u_i $(i = 1, \ldots, n)$ are the points; we shall denote coordinates of the i-th point by (x_i, y_i);
- $c(u_i, u_j)$ is the (i, j)-th element of the *given* cost matrix;
- $\tilde{c}(u_i, u_j)$ is the (i, j)-th element of the *obtained* cost matrix.

Note that in the degenerate case (i.e., when $\sigma = 0$), and when considering an ideal solution, the value 0 for the badness should be obtained.

As the heuristic algorithm for this problem, the authors propose the following one. *In fact, we are solving the same minimization problem, not paying attention to the fact that the points are not located according to a geometric law.*

Algorithm for pseudo-optimal placement of points
Input. Matrix $c : E \to \mathbb{N}_0$ (the matrix of weights of edges of a complete weighted graph with vertices $V = u_1, \ldots, u_n$); the value $N \in \mathbb{N}$.
N is the number of selectable pairs from the already allocated points. These pairs are chosen to accommodate each new point, starting with the 4-th one.

Step 1. $x_1 := 0$; $y_1 := 0$; $x_2 := 0$; $y_2 := c(u_1, u_2)$.

Step 2. For each point $k = 3, \ldots, n$ (we shall also denote the current k-th point under consideration simply by u), compute the coordinates (x_k, y_k) by the way of steps 3–7 below.

Step 3. $M := \min\left(\frac{(k-1)\cdot(k-2)}{2}, N\right)$.

Step 4. For each value $l = 1, \ldots, M$, implement steps 5 and 6 below.

Step 5. Randomly select uniformly distributed values $i, j \in \{1, \ldots, k-1\}$, where $i \neq j$. We shall also denote the i-th point by v, and the j-th point by w, and, additionally, denote $c_1 = c(u, v)$, $c_2 = c(u, w)$.

Step 6. If $c_1 + c_2 \leq c(v, w)$ (i.e., for the points u, v and w, the triangle inequality is violated), then we select

$$x_u = \frac{c_1 x_w + c_2 x_v}{c_1 + c_2}, \qquad y_u = \frac{c_1 y_w + c_2 y_v}{c_1 + c_2}.$$

In the event of a degenerate situation, where both c_1 and c_2 are equal to 0, the coordinates of the point are chosen to be in the middle of the segment between v and w. The coordinates are selected proportionally to the elements of the cost matrix.

However, if $c_1 + c_2 > c(v, w)$ (this option can be called a natural one), then two variants of pairs x_k and y_k are calculated according to equation (5). If $k = 3$, an arbitrary pair of coordinates is selected, otherwise, a pair is chosen according to one randomly chosen point among the numbers $1, \ldots, k-1$, that are not i and j. The selected pair of coordinates is added to the collection of pairs.

Step 7. Calculate the final values x_k and y_k as the arithmetic mean of the values l of the corresponding coordinates of the generated collection.

Output. Coordinates $(x_1, y_1), \ldots, (x_n, y_n)$.

End of the algorithm description.

The goal of the above-mentioned algorithm is to apply a simple method of searching through all possible solutions, by sequentially placing the points $u_1, \ldots, u_n$ while minimizing the value

$$\sum_{i=1}^{k-2} \sum_{j=i+1}^{k-1} \left(c(u_i, u_j) - \tilde{c}(u_i, u_j)\right)^2,$$

included in equation (6). Once all the points have been placed, standard methods for solving the geometric TSP version are used (such as the "onion-peeling" algorithm), as explained in more detail in [Melnikov, 2001].

5 Conclusion

In this section, we provide a detailed outline of the proposed continuation paper.

Despite the frequent use of the the "onion-peeling" algorithm, including as an auxiliary in other algorithms, the authors have not been able to find specific references (aside from the one mentioned above [Sing, 2012]). Consequently, in the follow-up article, we shall thoroughly describe a particular version of this algorithm that we use, which, as suggested by the content of this paper, will also be utilized by us as an auxiliary.

We shall present the results of numerical experiments using our method and summarize the results. As concrete results of computational experiments, we are going to cite the following. Firstly, for the "onion-peeling" algorithm we describe, we shall count the number of its layers, when the number of points (cities) changes from 30 to 10 million (in increments of about 3 times). Further, for "normal" dimensions (those where it makes sense to consider the pseudo-geometric TSP, i.e. no more than 100), we consider the values of badness for three variants of pseudo-placement of points:

- a simple algorithm (we randomly select several pairs, and the new point is the first one that comes along);
- a complicated algorithm (we choose the new point farthest from the middle of the already selected ones);
- a most complex algorithm (by setting a new point, we correct the previous ones according to it: randomly select 3 of the previous ones and recalculate their coordinates, and with such a recalculation, with a probability of about 1/3 of one of the points of the pair, we select this new point).

More important, however, are the following calculated values (we repeat that they will considered also for different "natural" dimensions):

- we calculate the badness using the usual "onion-peeling" algorithm; with the value $\sigma = 0$, the obtained value should be exactly 0; in general, we average the "percentages" of deterioration here and below;
- we calculate the badness by making the placement using a location known to us in advance, i.e., found "by the onion-peeling" for the original (not pseudo-geometric) matrix;
- we calculate the badness by making *our* placement; we make it without knowing the location of the found "onion-peeling" for the original (not pseudo-geometric) matrix.

Let us also repeat one of the directions of further work, already briefly described in Introduction. The connection of two mentioned problems (the pseudo-geometric

version of the TSP and the restoration of the DNA matrix) was already noted *at the level of solution algorithms*. However, it is possible to observe the connection between two mentioned problems *at the level of their statements*. Unfortunately, this thing is very little reflected in the literature so far; we are going to consider such a connection in future publications.

Acknowledgement

Two authors of this paper were partially supported by a grant from the scientific program of Chinese universities "Higher Education Stability Support Program" (chapter "Shenzhen 2022 – Science, Technology and Innovation Commission of Shenzhen Municipality").

References

Aharonov, Y., Davidovich, L., and Zagury, N. (1993). Quantum random walks. *Physical Review A, 1687*.

Childs, A. M., Farhi, E., and Gutmann, S. (2002). An Example of the Difference Between Quantum and Classical Random Walks. *Quantum Information Processing*, pp. 35–43.

Childs, A. M. (2009) Universal Computation by Quantum Walk. *Phys. Rev. Lett. 102, 180501*.

Freedman, M., Lovász, L., and Schrijver, A. (2007). Reflection positivity, rank connectivity, and homomorphism of graphs. *Journal of the American Mathematical Society, 20(01)*, pp. 37–52.

Garey, M. and Johnson, D. (1979). Computers and Intractability. W. H. Freeman and Company, USA.

Gutin, G. and Punnen, A. (editors). (1997). The Traveling Salesman problem. Kluwer Academic Publishers, Boston.

Hanke M. and Scherzer O. (2001). Inverse problems light: Numerical differentiation. *The American Mathematical Monthly, vol. 108*, pp. 512–521.

Hromkovič, J. (2003). Theoretical Computer Science. An Introduction to Automata, Computability, Complexity, Algorithmics, Randomization, Communication, and Cryptography. Springer, Berlin.

Hromkovič, J. (2004). Algorithmics for Hard Problems. Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics. Springer, Berlin.

Korostensky, C., and Gonnet, G. (1999). Near optimal multiple sequence alignments using a travelling salesman problem approach. *In: Proceedings of String Processing and Information Retrieval Symposium & International Workshop on Groupware*, pp. 105–114.

Korostensky, C., and Gonnet, G. (2000). Using traveling salesman problem algorithms for evolutionary tree construction. *Bioinformatics, vol. 16*, pp. 619–627.

Liew Sing (2012). Introducing convex layers to the Traveling Salesman Problem. *Preprint: arXiv:1204.2348* (Internet resource).

Makarkin, S., Melnikov, B., and Panin, A. (2013): On the metaheuristics approach to the problem of genetic sequence comparison and its parallel implementation. *Applied Mathematics, 4(10A)*, pp. 35–39.

Melnikov, B. (1996). An algorithm for proving the equivalence of the infinite iteration of finite languages. *Vestnik of Moscow University. Series 15: Computing mathematics and cybernetics, no. 4*, pp. 49–53 (in Russian).

Melnikov, B. and Kashlakova, E. (2000). Some grammatical structures of programming languages as simple bracketed languages. *Informatica (Lithuania), 11(4)*, pp. 441–453.

Melnikov, B. and Romanov, N. (2001). Once again about heuristics for the traveling salesman problem. *Theoretical Problems of Informatics and its expansions, vol. 4*, pp. 81–92 (in Russian).

Melnikov, B. and Panin, A. (2012). On a parallel implementation of the multi-heuristic approach in the problem of comparison of genetic sequences. *In: Vektor Nauki of Togliatti State University, 22(4)*, pp. 83–86 (in Russian).

Melnikov, B. F., Melnikova, E. A., Pivneva, S. V., Dudnikov, V. A., and Davydova, E. V. (2018). A multiheuristic algorithmic skeleton for hard combinatorial optimization problems. *In: CEUR Workshop Proceedings, 2212*, pp. 312–321.

Melnikov, B., Zhang, Y., and Chaikovskii, D. (2022). An inverse problem for matrix processing: an improved algorithm for restoring the distance matrix for DNA chains. *Cybernetics and Physics, 11(4)*, pp. 217–226.

Sergeenko A., Granichin O., and Yakunina M. (2020). Hamiltonian path problem: the time consumption comparison of DNA computing and branch and bound method. *Cybernetics and Physics, 9(1)*, pp. 121–127.

Somhom, S., Modares, A., Enkawa, T. (1999). Competition-based neural network for the multiple travelling salesmen problem with minimax objective. *Computers & Operations Research, 26(4)*, pp. 395–407.

Tomilin V., Il'ichov L. (2023) Trainable unravelling for quantum state discrimination. *Cybernetics and Physics, 12(2)*.

Trugenberger, C. A. (2002) Phase Transitions in Quantum Pattern Recognition. *Physical Review Letters, Vol. 89, 0277903*.