Simulated annealing approach to verify vertex adjacencies in the traveling salesperson polytope

Anna Kozlova\(^1\) and Andrei Nikolaev\(^1\)

P. G. Demidov Yaroslavl State University
fyz95@mail.ru, andrei.v.nikolaev@gmail.com

Abstract. We consider 1-skeletons of the symmetric and asymmetric traveling salesperson polytopes whose vertices are all possible Hamiltonian tours in the complete directed or undirected graph, and the edges are geometric edges or one-dimensional faces of the polytope. It is known that the question whether two vertices of the symmetric or asymmetric traveling salesperson polytopes are nonadjacent is NP-complete. A sufficient condition for nonadjacency can be formulated as a combinatorial problem: if from the edges of two Hamiltonian tours we can construct two complementary Hamiltonian tours, then the corresponding vertices of the traveling salesperson polytope are not adjacent. We consider a heuristic simulated annealing approach to solve this problem. It is based on finding a vertex-disjoint cycle cover and a perfect matching. The algorithm has a one-sided error: the answer “not adjacent” is always correct, and was tested on random and pyramidal Hamiltonian tours.

Keywords: traveling salesperson problem · Hamiltonian tour · traveling salesperson polytope · 1-skeleton · vertex adjacency · simulated annealing · vertex-disjoint cycle cover · perfect matching.

1 Introduction

We consider a classical traveling salesperson problem on a complete directed or undirected graph.

**Symmetric traveling salesperson problem.** Given a complete weighted graph \( K_n = (V, E) \), it is required to find a Hamiltonian cycle of minimum weight.

**Asymmetric traveling salesperson problem.** Given a complete weighted digraph \( D_n = (V, A) \), it is required to find a Hamiltonian tour of minimum weight.

We denote by \( HC_n \) the set of all Hamiltonian cycles in \( K_n \) and by \( HT_n \) the set of all Hamiltonian tours in \( D_n \). With each Hamiltonian cycle \( x \in HC_n \) we associate a characteristic vector \( x^e \in \mathbb{R}^E \) by the following rule:

\[
  x^e_e = \begin{cases} 
    1, & \text{if the cycle } x \text{ contains an edge } e \in E, \\
    0, & \text{otherwise}. 
  \end{cases}
\]
With each Hamiltonian tour $y \in HT_n$ we associate a characteristic vector $y^v \in \mathbb{R}^A$ by the following rule:

$$y^v_a = \begin{cases} 1, & \text{if the tour } y \text{ contains an edge } a \in A, \\ 0, & \text{otherwise.} \end{cases}$$

The polytope

$$\text{TSP}(n) = \text{conv}\{x^v \mid x \in HC_n\}$$

is called the symmetric traveling salesperson polytope, and the polytope

$$\text{ATSP}(n) = \text{conv}\{y^v \mid y \in HT_n\}$$

is called the asymmetric traveling salesperson polytope.

The 1-skeleton of a polytope $P$ is the graph whose vertex set is the vertex set of $P$ (characteristic vectors $x^v$ for the traveling salesperson problem) and edge set is the set of geometric edges or one-dimensional faces of $P$. Many papers are devoted to the study of 1-skeletons associated with combinatorial problems. On the one hand, the vertex adjacency in 1-skeleton is of great interest for the development of algorithms to solve problems based on local search technique (when we choose the next solution as the best one among adjacent solutions). For example, various algorithms for perfect matching, set covering, independent set, ranking of objects, problems with fuzzy measures, and many others are based on this idea [2,3,10,11,15,21]. On the other hand, some characteristics of 1-skeletons, such as the diameter and the clique number, estimate the time complexity for different computation models and classes of algorithms [5,6,8,17].

Unfortunately, the classical result by Papadimitriou states that the construction of 1-skeleton of the traveling salesperson polytope is NP-complete for both directed and undirected graphs.

**Theorem 1 (Papadimitriou, [24]).** The question whether two vertices of the polytopes TSP$(n)$ or ATSP$(n)$ are nonadjacent is NP-complete.

However, the vertex adjacency test for TSP$(n)$ and ATSP$(n)$ is an interesting problem itself. Note that a geometric approach to the construction of 1-skeleton of the traveling salesperson polytope seems not very promising because both polytopes have superexponential number of vertices and faces [14,17].

In [25] the sufficient condition for vertex adjacency in the traveling salesperson polytope was reformulated in a combinatorial form.

**Lemma 1 (Sufficient condition for nonadjacency).** If from the edges of two Hamiltonian tours $x$ and $y$ it is possible to construct two complementary Hamiltonian tours $z$ and $w$, then the corresponding vertices $x^v$ and $y^v$ of the polytope TSP$(n)$ (or ATSP$(n)$) are not adjacent.

From the geometric point of view, the Lemma 1 means that the segment connecting two vertices $x^v$ and $y^v$ intersects with the segment connecting two other vertices $z^v$ and $w^v$ of the polytope TSP$(n)$ (or ATSP$(n)$ correspondingly),
Fig. 1. Two complementary tours $z$ and $w$ are constructed from the edges of $x$ and $y$

thus it cannot be an edge in 1-skeleton. An example of a satisfied sufficient condition is shown in Fig. 1.

Let us formulate the sufficient condition for vertex nonadjacency of the traveling salesman polytope in the form of a combinatorial problem.

Instance. Let $x$ and $y$ be two Hamiltonian tours.

Question. Does the multigraph $x \cup y$ include a pair of Hamiltonian tours $z$ and $w$ different from $x$ and $y$ such that $z \cup w = x \cup y$ and $z \cap w = \emptyset$?

By $x \cup y$ we denote a multigraph that contains all edges of both tours $x$ and $y$ (Fig. 2).

In this formulation, the problem is close to the 2-peripatetic salesperson problem in which it is required to find two Hamiltonian tours of minimum weight without common edges. The 2-peripatetic salesperson is NP-complete even for 4-regular graphs [12]. Much attention was paid to the development of approximation algorithms for this problem (see, for example, [1,4,16]).

However, the combinatorial form of the sufficient condition for nonadjacency has a number of differences from the 2-peripatetic salesperson problem:

– this is a decision problem, not an optimization one;
– the graph is a 4-regular graph (or digraph) of a special form constructed as a union of two Hamiltonian tours;
– it is required to find two Hamiltonian tours different from $x$ and $y$.

Note also that if from the edges of two Hamiltonian tours $x$ and $y$ it is possible to construct another Hamiltonian tour $z$, then all the remaining edges $(x \cup y) \setminus z$ are almost certainly do not form a Hamiltonian tour (Fig. 2). Thus, instead of algorithms for a single Hamiltonian tour in the multigraph $x \cup y$, in this paper we...
Fig. 2. An example of $w = (z \cup y) \setminus z$ that is not a Hamiltonian tour

consider a heuristic simulated annealing approach to test vertex adjacencies in
the symmetric and asymmetric traveling salesperson polytopes based on finding
a vertex-disjoint cycle cover and a perfect matching.

2 Simulated annealing

The simulated annealing borrows the concept from annealing in metallurgy
where a metal material is repeatedly heated, kneaded, and cooled to enlarge
the size of its crystals to eliminate defects [19].

We consider a general scheme of the Algorithm 1. It is required to minimize
the energy function specified for the current system state. The algorithm starts
from a certain initial state: at each step a neighbor candidate state is generated
which energy is compared to the energy of the previous state. If the energy
decreases, the system transits to the new state, otherwise it may transit with a
certain probability (to prevent falling into the local minimum).

The algorithm receives input data in one of the following formats:

1. Two Hamiltonian tours $x = [a_1, \ldots, a_N]$ and $y = [b_1, \ldots, b_N]$, given as the
permutations of vertices in a complete graph (or digraph) $K_N$;

2. 2/4-regular graph (2 — for directed and 4 — for undirected graphs) of size
$N$, i.e. the union of two Hamiltonian tours, given as the adjacency list.

Other input parameters: the initial value of temperature $initT$, the maximum
number of iterations $iterN$, and the the size of a queue of fixed edges $fixEdgesN$. 
#### Algorithm 1. Simulated Annealing Algorithm

**Input**: Hamiltonian tours \(x\) and \(y\) (or 2/4-regular graph \(combG\)), initial temperature \(initT\), number of iterations \(iterN\), size of a queue of fixed edges \(fixEdgesN\)

**Output**: vertices \(x_v\) and \(y_v\) are adjacent or not adjacent, complementary Hamiltonian tours \(z\) and \(w\), if exist

**Procedure** `SimulatedAnnealing(x, y, combG, initT, fixEdgesN)`

1. \(T \leftarrow initT\)
2. \(z, w \leftarrow\) `GetInitialState(x, y, combG)`
3. For \(k \leftarrow 1\) to \(iterN\)
   - If \(z\) and \(w\) are Hamiltonian tours different from \(x\) and \(y\) Then
     - Return \(z\) and \(w\)
   - \(z\)Cand, \(w\)Cand \(\leftarrow\) `GenerateNeighbourCandidate(z, w, fixEdgesN)`
   - \(candE \leftarrow\) `CalculateEnergy(z\)Cand, w\)Cand)`
   - If \(candE < currE\) or `ShouldAcceptCandidate()` Then
     - \(z \leftarrow z\)Cand, \(w \leftarrow w\)Cand
   - End
   - \(T \leftarrow\) `CoolingSchedule(k)`
4. End
5. Return no complementary tours found;

**Procedure** `TestVertexAdjacency(x, y, combG, initT, fixEdgesN)`

1. \(z, w \leftarrow\) `SimulatedAnnealing(x, y, combG, initT, fixEdgesN)`
2. If \(z\) and \(w\) are not empty Then
   - Return vertices \(x_v\) and \(y_v\) are not adjacent
3. Else
   - Return vertices \(x_v\) and \(y_v\) are probably adjacent
4. End

The algorithm stops when the solution is found or when the number of iterations exceeds the value of the parameter \(iterN\). As an output, the algorithms returns two complementary Hamiltonian tours \(z\) and \(w\) constructed from the edges of \(x\) and \(y\). By the sufficient condition (Lemma 1), the corresponding vertices \(x_v\) and \(y_v\) of the traveling salesperson polytope are not adjacent. If the algorithm cannot find the complementary tours, then it returns that the corresponding vertices are probably adjacent. Thus, the algorithm has a one-sided error: the answer “not adjacent” is always correct, while the answer “probably adjacent” leaves the possibility that the vertices actually are not adjacent.

### 3 Generation of the initial state

To generate the initial system state and neighbor candidate states, we construct a vertex-disjoint cycle cover of the multigraph \(x \cup y\) (Fig. 3).
If $x$ and $y$ are undirected Hamiltonian cycles, then all vertices in the multigraph $x \cup y$ have degrees equal to 4. Let $z$ be a vertex-disjoint cycle cover of $x \cup y$, then all the remaining edges form a graph $w = (x \cup y) \setminus z$ with all vertex degrees being equal to 2. Thus, $w$ is also a vertex-disjoint cycle cover of $x \cup y$.

If $x$ and $y$ are directed Hamiltonian tours, then all vertices in the multigraph $x \cup y$ have both indegrees and outdegrees equal to 2. Let $z$ be a vertex-disjoint cycle cover of $x \cup y$, then in the digraph $w = (x \cup y) \setminus z$ all vertices have both indegrees and outdegrees equal to 1. Thus, $w$ is also a vertex-disjoint cycle cover of $x \cup y$.

Finding a vertex-disjoint cycle cover of both directed and undirected graphs can be performed in polynomial time by a reduction to perfect matching [27]. Let us recall that a perfect matching is a set of pairwise nonadjacent edges which matches all vertices of the graph. The procedures for directed and undirected graphs are somewhat different. We consider them separately.

Let $x$ and $y$ be undirected Hamiltonian cycles.

**Step 1.** From the multigraph $x \cup y = G = (V, E)$, we construct a new graph $G' = (V', E')$. With each vertex $v \in V$ we associate a gadget $G_v$ that is a complete bipartite subgraph $K_{4,2}$ (note that the degree of $v$ equals 4) as it is shown in Fig. 4:

- there are 4 vertices in the outer part ($v_a, v_b, v_c$ and $v_d$) that correspond to 4 edges incident to $v$ in $G$ (edges $A, B, C, D$); these vertices are connected with other gadgets;
- there are 2 vertices in the inner part ($v_1$ and $v_2$) that are connected only with the vertices of the outer part.

**Step 2.** A perfect matching in $G'$ corresponds to a vertex-disjoint cycle cover in the original graph $G$. Indeed, a perfect matching has to cover both inner vertices $v_1$ and $v_2$. Therefore, it includes exactly one edge of $\{(v_1, v_a), (v_1, v_b), (v_1, v_c), (v_1, v_d)\}$ and exactly one edge of $\{(v_2, v_a), (v_2, v_b), (v_2, v_c), (v_2, v_d)\}$. Both of these edges cover exactly two vertices of $\{v_a, v_b, v_c, v_d\}$. The other two vertices has to be covered by the edges that correspond to the edges of
Fig. 4. Construction of the graph \( G' \) for the symmetric problem

\[
G = (V, E) \quad \Rightarrow \quad G' = (V', E')
\]

Fig. 5. A perfect matching in \( G' \) and a vertex-disjoint cycle cover in \( G \)

\[
G' = (V', E')
\]

\[
G = (V, E)
\]

\[
G' = (V', E')
\]

A perfect matching in a general undirected graph can be found by Edmond’s algorithm \([13]\) in \( O(V^2E) \) time or using Micali-Vazirani matching algorithm \([22]\) in \( O(\sqrt{VE}) \) time. We have chosen Edmond’s algorithm as a more simple one to implement. Note that replacing it with a more efficient Micali-Vazirani algorithm does not require changing the rest of the algorithm.

Let \( x \) and \( y \) be directed Hamiltonian tours.

**Step 1.** From the directed multigraph \( x \cup y = D = (V, A) \), we construct a bipartite graph \( D' = (L, R, E) \). With each vertex \( v \in V \) we associate a pair of vertices \( v_L \in L \) and \( v_R \in R \), and with each edge \( (u, v) \in A \) we associate a new edge \((u_L, v_R)\) in the bipartite graph \( D' \) (Fig. 6).
Fig. 6. Construction of the bipatite graph $D'$ for the asymmetric problem

Fig. 7. A perfect matching in $D'$ corresponds to a vertex-disjoint cycle cover of $D$

Step 2. A perfect matching in the bipartite graph $D'$ corresponds to a vertex-disjoint directed cycle cover in the original graph $D$. Indeed, every vertex of $D$ is a head of exactly one edge and a tail of exactly one edge of a perfect matching in $D'$ (Fig. 7).

A perfect matching in a bipartite graph can be found by Hopcroft-Karp algorithm [18] in $O(\sqrt{V}E)$ time.

4 Generation of a neighbor candidate state

A process of constructing a neighbor candidate state is shown in Procedure 2.

The algorithm receives as input the current state as the vertex-disjoint cycle covers $z$ and $w$, and the parameter fixEdgesN that set the size of a queue of edges that are fixed in the graph $z$ and the corresponding perfect matching. When this limit is exceeded, the first edge of the queue is deleted.
Algorithm 2. Constructing a neighbor candidate state

Procedure GenerateNeighbourCandidate\( (z, w, f_{\text{ixEdgesN}}) \)
\[ \text{UpdateFixedEdgesQueue}(z, w, f_{\text{ixEdgesN}}) \]
\[ \text{If tours } z \text{ and } w \text{ are directed Then} \]
\[ \quad \text{RunHopcroftKarpAlgorithm()} \]
\[ \text{Else} \]
\[ \quad \text{RunEdmondsAlgorithm()} \]
\[ \text{End} \]
\[ \text{Return } z, w \]
\[ \text{End} \]

Fig. 8. Generation of a neighbor candidate state

In order to find a neighbor candidate state we chose an edge of \( w \) with endpoints in two different connected components of \( z \) and add it to the queue of fixed edges (Fig. 8, fixed edges of \( z \) are dashed, an edge of \( w \) that is added to the queue is dashed dotted). Such edge always exists due to the connectivity of the multigraph \( x \cup y \). If \( z \) contains exactly one connected component, then the graphs \( z \) and \( w \) can be swapped. The idea of this procedure is to reduce the number of connected components in \( z \) and \( w \). The neighbor candidate state is constructed by the perfect matching algorithms with fixed edges forming the initial matching.

5 Cooling schedule

If a neighbor candidate state has two complementary Hamiltonian tours different from \( x \) and \( y \) (or any two complementary Hamiltonian tours, if the input is a 2/4-regular graph), then the algorithm successfully stops and returns a solution.
Otherwise, the energy function is calculated for a neighbor candidate state. We have chosen the following function (Procedure 3): (number of connected components in $z$) + (number of connected components in $w$).

**Algorithm 3. Energy function**

| Procedure CalculateEnergy($z, w$) |
|-----------------|
| Return $\text{CountComponents}(z) + \text{CountComponents}(w)$ |

At each step of the algorithm the candidate states correspond to two vertex-disjoint cycle covers $z$ and $w$. Therefore, if the total number of connected components in $z$ and $w$ is equal to 2, then $z$ and $w$ are Hamiltonian tours.

If the energy function has decreased compared to the previous state, then we accept a transition to the neighbor candidate state.

If the energy function has not decreased, then we make a transition with probability

$$P = e^{-\frac{\text{curr}E - \text{cand}E}{T}},$$

where $T$ is the current temperature, $\text{curr}E$ is the current energy value, and $\text{cand}E$ is the energy of the considered neighbor candidate state. Such transition is necessary to avoid the problem of falling into the local minimum.

The current temperature gradually decreases from $\text{init}T$ to 0, and its function depends on the initial temperature $\text{init}T$ and the index of the current iteration $k$ (Procedure 4).

**Algorithm 4. Cooling schedule**

| Procedure CoolingSchedule($k$) |
|-----------------|
| Return $\text{init}T / k$ |

6 Experiments

The algorithm to test vertex adjacencies in the polytopes TSP($n$) and ATSP($n$) was implemented as a console application with different input parameters. Some of them are described below:

--N — number of vertices in the input graph/tour;
--times — number of times to run the algorithm;
SA approach to verify vertex adjacencies in the TSP polytope

--iterN — number of iterations in the simulated annealing algorithm;
--stateCandidate=random—match — how to generate a neighbor candidate state:
random : random exchange of edges between tours
match : constructing a vertex-disjoint cycle cover and a perfect matching;
--exEdgesN — number of edges to randomly exchange between tours (used only for --stateCandidate=random);
--ansN — multistart: number of repeatedly runs of the algorithm (used only for --stateCandidate=random);
--fixEdgesN — the size of a queue of edges that can be fixed in the initial matching (used only for --stateCandidate=match).

We tested the algorithm on random directed and undirected Hamiltonian tours, and also on directed and undirected pyramidal tours.
A Hamiltonian tour
\[ \tau = (1, i_1, i_2, \ldots, i_r, j_1, j_2, \ldots, j_{n-r-2}) \]
is called pyramidal if
\[ i_1 < i_2 < \ldots < i_r \text{ and } j_1 > j_2 \ldots > j_{n-r-2}. \]

We chose pyramidal tours for experiments since for them the vertex adjacencies in the corresponding polytopes can be easily verified. In particular, the following problem: given two pyramidal tours \( x \) and \( y \), is it possible to construct two complementary tours \( z \) and \( w \) from the edges of \( x \) and \( y \), can be solved in linear time [8,9]. Thus, we run the algorithm on pyramidal tours, for which it is known that the sufficient condition of Lemma 1 is satisfied. This allows us to estimate the error percentage when the algorithm could not find complementary tours that are guaranteed to exist.

The results of the tests for undirected pyramidal tours are presented in Table 1 (Edmond’s algorithm is used). The algorithm was run with the number of iterations \( \text{iterN} = 8000 \) and the number of fixed edges \( \text{fixEdgesN} = \lfloor N/3 \rfloor \). In the previous version of the program [20] a different method to generate neighbor candidate state was implemented — the exchange of random edges between two subgraphs \( z \) and \( w \). Its results are also shown in the table for comparison. For the exchange of random edges the following input parameters were used: the number of iterations \( \text{iterN} = 50000 \), the number of multistart attempts \( \text{ansN} = 5 \) and the number of edges to exchange \( \text{exEdgesN} = 3 \).

Note that compared to the exchange of random edges, both the accuracy of the algorithm and the size of solved problems have increased. The accuracy can also be adjusted by increasing the number of iterations or changing the maximum number of fixed edges in the queue.

The results of the tests for directed pyramidal tours and the Hopcroft-Karp algorithm are presented in Table 2. The input parameters are similar to the case with undirected tours. Here, for almost all considered sizes of test graphs, the algorithm works with the accuracy of 100%, which is a very good result.
Table 1. Results for undirected pyramidal tours with number of tests *times* = 50

| N  | Exchange of random edges | Reconstructing with perfect matching |
|----|--------------------------|-------------------------------------|
|    | Tours are not found, *T*<sub>NF_avg</sub> | Tours are not found, *T*<sub>F_avg</sub> | Average accuracy, %, *Acc* | Tours are not found, *T*<sub>NF_avg</sub> | Tours are not found, *T*<sub>F_avg</sub> | Average accuracy, %, *Acc* |
| 8  | 9.84                     | 9.84                                 | 100                         | 5.57                              | 5.57                                  | 100 |
| 16 | 5066.57                  | 382.79                               | 851.17, 90                  | 22.08                             | 22.08                                 | 100 |
| 24 | 6549.75                  | 1403.32                              | 5005.82, 30                 | 9096.65                           | 33.34                                 | 98  |
| 32 | 7832.24                  | 1330.37                              | 7312.09, 8                  | 224.93                            | 224.93                                | 100 |
| 40 | 10035.64                 | 2351.53                              | 9728.28, 4                  | 18656.05                          | 610.26                                | 971.18, 98 |
| 48 | 13455.39                 | 13455.39                             | 0                            | 27695.76                          | 978.18                                | 3649.94, 90 |
| 64 | 16243.99                 | 108.64                               | 15921.28, 2                | 48377.16                          | 988.05                                | 12361.43, 76 |
| 96 | 98480.5                  | -                                    |                              | 98480.5                           | 12293.68                              | 53629.27, 52 |
| 128| -                        | 188485.17                            | 21982.49                    | 129264.4                          | 28                                     |
| 192| -                        | 334841.38                            | 26165.54                    | 297800.28                         | 12                                     |

Finally, Table 3 shows the test results for random directed and undirected Hamiltonian tours. From the table it can be concluded that for random tours the algorithm works even faster than for pyramidal tours. Note that for undirected graphs the algorithm finds complementary tours more often than for directed graphs. This is due to the fact that 1-skeleton of the asymmetric traveling salesman polytope is generally much more dense than 1-skeleton of the symmetric polytope. For example, the diameter of 1-skeleton of ATSP(*n*) is 2 [23], while the best known upper bound for the diameter of 1-skeleton of TSP(*n*) is 4 [26]. Besides, for undirected cycles, the algorithm was able to find a solution for almost all cases. We can conclude that for the symmetric traveling salesman polytope TSP(*n*) two random vertices are not adjacent with a very high probability.

The largest instance that was solved by the algorithm had random Hamiltonian tours on 4096 vertices and required 1 530 682 ms. However, due to the long waiting time for several tests, we limited the presented experiments to tours of size under 200 vertices.

### 7 Conclusion

The construction and study of 1-skeletons of the polytopes associated with intractable problems is of interest for the development and analysis of combinatorial algorithms. However, for such problems as the traveling salesperson even determining whether two vertices are adjacent or not is an NP-complete problem. This paper proposes an original heuristic approach based on simulated annealing to verify vertex adjacencies in 1-skeleton of the traveling salesperson polytope. The algorithm has a one-sided error: the answer “not adjacent” is always correct, while the answer “probably adjacent” leaves the possibility that the vertices actually are not adjacent. The algorithm showed good practical results during the experiments.
Table 2. Results for directed pyramidal tours (Hopcroft-Karp algorithm), with the number of tests $times = 50$ and number of fixed edges $fixEdgesN = \lfloor N/3 \rfloor$

| N   | Tours are not found, $T_{NF_{avg}}$ | Tours are found, $TF_{avg}$ | Average time, $T_{avg}$ | Accuracy, %, $Acc$ |
|-----|-----------------------------------|----------------------------|-------------------------|---------------------|
| 8   | 2.27                             | 2.27                       | 100                     |                     |
| 16  | 5.03                             | 5.03                       | 100                     |                     |
| 24  | 19.75                            | 19.75                      | 100                     |                     |
| 32  | 19.14                            | 19.14                      | 100                     |                     |
| 40  | 40.89                            | 40.89                      | 100                     |                     |
| 48  | 95.38                            | 95.38                      | 100                     |                     |
| 64  | 689.20                           | 689.20                     | 100                     |                     |
| 96  | 330.21                           | 330.21                     | 100                     |                     |
| 128 | 4514.36                          | 9515.07                    | 96                      |                     |
| 192 | 15783.70                         | 70190.93                   | 76                      |                     |

Table 3. Results for random Hamiltonian tours with the number of tests $times = 50$

| N   | Tours are not found, $T_{NF_{avg}}$ | Tours are found, $TF_{avg}$ | Average time, $T_{avg}$ | Percentage of found tours, % |
|-----|-----------------------------------|----------------------------|-------------------------|------------------------------|
| 8   | 1094.66                           | 491.91                     | 30                      | 3118.99                      |
| 16  | 13.76                             | 13.756                     | 100                     | 6799.17                      |
| 24  | 28.52                             | 28.52                      | 100                     | 10594.29                     |
| 32  | 46.64                             | 46.64                      | 100                     | 14764.29                     |
| 40  | 75.27                             | 75.27                      | 100                     | 19184.31                     |
| 48  | 87.86                             | 87.86                      | 100                     | 25214.48                     |
| 64  | 235.37                            | 235.37                     | 100                     | 38886.56                     |
| 96  | 481.37                            | 481.37                     | 100                     | 74654.25                     |
| 128 | 827.42                            | 827.42                     | 100                     | 121790.32                    |
| 192 | 4064.84                           | 4064.84                    | 100                     | 252321.67                    |

Acknowledgments. The research is supported by the grant of the President of the Russian Federation MK-2620.2018.1.

References

1. Ageev, A.A., Pyatkin, A.V.: A 2-approximation algorithm for the metric 2-peripatetic salesman problem. In: 5th International Workshop on Approximation and Online Algorithms, WAOA 2007, LNCS 4927, 103–115, 2016. https://doi.org/10.1007/978-3-540-77918-6_9.

2. Aguilera, N.E., Katz, R.D., Tolomei, P.B.: Vertex adjacencies in the set covering polyhedron. Discrete Appl. Math. 218, 40–56 (2017). https://doi.org/10.1016/j.dam.2016.10.024.
3. Balinski, M.L.: Signature methods for the assignment problem. Oper. Res. 33 (3), 527–536 (1985). https://doi.org/10.1287/opre.33.3.527.

4. Baburin, A.E., Della Croce, F., Gimadi, E.K., Glazkov, Y.V., Paschos, V.Th.: Approximation algorithms for the 2-peripatetic salesman problem with edge weights 1 and 2. Discrete Appl. Math. 157, 1988–1992 (2009). https://doi.org/10.1016/j.dam.2008.06.025.

5. Bondarenko, V.A.: Nonpolynomial lower bounds for the complexity of the traveling salesman problem in a class of algorithms. Autom. Rem. Contr. 44, 1137–1142 (1983).

6. Bondarenko, V.A., Maksimenko, A.N.: Geometricheskie konstruktsii i slozhnost’ v kombinatornoy optimizatsii (Geometric constructions and complexity in combinatorial optimization), LKI, Moscow, 2008 [Russian].

7. Bondarenko, V.A., Nikolaev, A.V.: On Graphs of the Cone Decompositions for the Min-Cut and Max-Cut Problems, Int. J. Math. Sci., 2016, Article ID 7863650, 6 p (2016). https://doi.org/10.1155/2016/7863650.

8. Bondarenko, V.A., Nikolaev, A.V.: Some properties of the skeleton of the pyramidal tours polytope. Electron. Notes Discrete Math. 61, 131–137 (2017). https://doi.org/10.1016/j.endm.2017.06.030.

9. Bondarenko, V.A., Nikolaev, A.V.: On the Skeleton of the Polytope of Pyramidal Tours. J. Appl. Ind. Math. 12, 9-18 (2018). https://doi.org/10.1134/S1990478918010027.

10. Chegireddy, C.R., Hamacher, H.W.: Algorithms for finding K-best perfect matchings. Discrete Appl. Math. 18, 155–165 (1987). https://doi.org/10.1016/0166-218X(87)90017-5.

11. Combarro, E.F., Miranda, P.: Adjacency on the order polytope with applications to the theory of fuzzy measures. Fuzzy Set. Syst. 161, 619–641 (2010). https://doi.org/10.1016/j.fss.2009.05.004.

12. De Kort, J.B.J.M.: Bounds for the Symmetric 2-Peripatetic Salesman Problem. Optim. 23, 357–367 (1992). https://doi.org/10.1080/02331939208843770.

13. Edmonds, J.: Paths, trees, and flowers. Can. J. Math. 17, 449–467 (1965). https://doi.org/10.4153/CJM-1965-045-4.

14. Fiorini, S., Massar, S., Pokutta, S., Tiwary, H.R., De Wolf, R.: Exponential Lower Bounds for Polytopes in Combinatorial Optimization. J. ACM 62, Article No. 17 (2015). https://doi.org/10.1145/2716307.

15. Gabow, H.N.: Two algorithms for generating weighted spanning trees in order. SIAM J. Comput. 6, 139–150 (1977). https://doi.org/10.1137/0206011.

16. Glebov, A.N., Zambalaeva, D.Z.: A polynomial algorithm with approximation ratio 7/9 for the maximum two peripatetic salesmen problem. J. Appl. Ind. Math. 6, 69–89 (2012). https://doi.org/10.1134/S1990478912010005.

17. Grötschel, M.; Padberg, M.: Polyhedral theory. In: The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization, E. Lawler, J.K. Lenstra, A. Rinnooy Kan, and D. Shmoys, eds., pp. 251–305, John Wiley, Chichester, 1985.

18. Hopcroft, J.E., Karp, R.M.: An n5/2 algorithm for maximum matchings in bipartite graphs. SIAM J. Comp. 2 (4), 225–231 (1973). https://doi.org/10.1137/0202019.

19. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science. 220 (4598), 671–680 (1983). https://doi.org/10.1126/science.220.4598.671.

20. Kozlova, A.P., Nikolaev, A.V.: Proverka smezhnosti ver shin mnogogrannika zadachi kommivoyazhyora (Verification of vertex adjacency in the traveling salesman polytope). Zametki po informatike i matematike (Notes on computer science and mathematics) 10, 51–58 (2018).
21. Matsui, T., Tamura, S.: Adjacency on combinatorial polyhedra. *Discrete Appl. Math.* **56**, 311–321 (1995).

22. Micali, S., Vazirani, V.V.: An $O(\sqrt{|V||E|})$ algorithm for finding maximum matching in general graphs. In: *Proc. 21st IEEE Symp. Foundations of Computer Science*, 17–27 (1980). [https://doi.org/10.1109/SFCS.1980.12](https://doi.org/10.1109/SFCS.1980.12).

23. Padberg, M.W., Rao, M.R.: The travelling salesman problem and a class of polyhedra of diameter two. *Math. Program.*, **7**(1), 32–45 (1974). [https://doi.org/10.1007/BF01585502](https://doi.org/10.1007/BF01585502).

24. Papadimitriou, C.H.: The adjacency relation on the traveling salesman polytope is NP-Complete. *Math. Program.*, **14**, 312–324 (1978). [https://doi.org/10.1007/BF01588973](https://doi.org/10.1007/BF01588973).

25. Rao, M.R.: Adjacency of the traveling salesman tours and 0-1 vertices. *SIAM J. Appl. Math.* **30**, 191–198 (1976). [https://doi.org/10.1137/0130021](https://doi.org/10.1137/0130021).

26. Rispoli, F.J., Cosares, S.: A Bound of 4 for the Diameter of the Symmetric Traveling Salesman Polytope. *SIAM J. Discrete Math.*, **11**(3), 373–380 (1998). [https://doi.org/10.1137/S0895480196312462](https://doi.org/10.1137/S0895480196312462).

27. Tutte, W.T.: A short proof of the factor theorem for finite graphs. *Can. J. Math.*, **6**, 347–352 (1954). [https://doi.org/10.4153/CJM-1954-033-3](https://doi.org/10.4153/CJM-1954-033-3).