Yugoslav Journal of Operations Research
27 (2017), Number 4, 415–426
DOI: 10.2298/YJOR161012003E

# ON SOLVING TRAVELLING SALESMAN PROBLEM WITH VERTEX REQUISITIONS

Anton V. EREMEEV
*Sobolev Institute of Mathematics SB RAS,*
*Omsk State University n.a. F.M. Dostoevsky*
*eremeev@ofim.oscsbras.ru*

Yulia V. KOVALENKO
*Sobolev Institute of Mathematics SB RAS,*
*julia.kovalenko.ya@yandex.ru*

Received: October 2016 / Accepted: March 2017

**Abstract:** We consider the Travelling Salesman Problem with Vertex Requisitions where, for each position of the tour, at most two possible vertices are given. It is known that the problem is strongly NP-hard. The algorithm, we propose for this problem, has less time complexity compared to the previously known one. In particular, almost all feasible instances of the problem are solvable in $O(n)$ time using the new algorithm, where $n$ is the number of vertices. The developed approach also helps in fast enumeration of a neighborhood in the local search and yields an integer programming model with $O(n)$ binary variables for the problem.

**Keywords:** Combinatorial Optimization, System of Vertex Requisitions, Local Search, Integer Programming.

**MSC:** 90C59, 90C10.

## 1. INTRODUCTION

The Travelling Salesman Problem (TSP) is one of the well-known NP-hard combinatorial optimization problems [6]: given a complete arc-weighted digraph

This research of the authors is supported by the Russian Science Foundation Grant (project no. 15-11-10009).

with $n$ vertices, find a shortest travelling salesman tour (Hamiltonian circuit) in it.

The TSP with Vertex Requisitions (TSPVR) was formulated by A.I. Serdyukov in [12]: find a shortest travelling salesman tour, passing at $i$-th position a vertex from a given subset $X^i$, $i = 1, \dots, n$. A special case where $|X^i| = n$, $i = 1, \dots, n$, is equivalent to the TSP.

This problem can be interpreted in terms of scheduling theory. Consider a single machine that may perform a set of operations $X = \{x_1, \dots, x_n\}$. Each of the identical jobs requires processing all $n$ operations in such a sequence that the $i$-th operation belongs to a given subset $X^i \subseteq X$ for all $i = 1, \dots, n$. A setup time is needed to switch the machine from one operation of the sequence to another. Moreover, after execution of the last operation of the sequence, the machine requires a changeover to the first operation of the sequence to start processing of the next job. The problem is to find a feasible sequence of operations, minimizing the cycle time.

TSP with Vertex Requisitions where $|X^i| \leq k$, $i = 1, \dots, n$, was called $k$-TSP with Vertex Requisitions ($k$-TSPVR) in [12]. The complexity of $k$-TSPVR was studied in [12] for different values of $k$ on graphs with small vertex degrees. In [10], A.I. Serdyukov proved the NP-hardness of 2-TSPVR in the case of complete graph and showed that almost all feasible instances of the problem are solvable in $O(n^2)$ time. In this paper, we propose an algorithm for 2-TSPVR with time complexity $O(n)$ for almost all feasible problem instances. The developed approach also has some applications to local search and integer programming formulation of 2-TSPVR.

The paper has the following structure. In Section 2, a formal definition of 2-TSPVR is given. In Section 3, an algorithm for this problem is presented. In Section 4, a modification of the algorithm is proposed with an improved time complexity, and it is shown that almost all feasible instances of the problem are solvable in time $O(n)$. In Section 5, the developed approach is used to formulate and enumerate efficiently a neighborhood for the local search. In Section 6, this approach allows to formulate an integer programming model for 2-TSPVR using $O(n)$ binary variables. The last section contains the concluding remarks.

## 2. PROBLEM FORMULATION AND ITS HARDNESS

2-TSP with Vertex Requisitions is formulated as follows. Let $G = (X, U)$ be a complete arc-weighted digraph, where $X = \{x_1, \dots, x_n\}$ is the set of vertices, $U = \{(x, y) : x, y \in X, x \neq y\}$ is the set of arcs with non-negative arc weights $\rho(x, y)$, $(x, y) \in U$. Besides that, a system of vertex subsets (requisitions) $X^i \subseteq X$, $i = 1, \dots, n$, is given, such that $1 \leq |X^i| \leq 2$ for all $i = 1, \dots, n$.

Let $F$ denote the set of bijections from $X_n := \{1, \dots, n\}$ to $X$, such that $f(i) \in X^i$, $i = 1, \dots, n$, for all $f \in F$. The problem consists in finding such a mapping $f^* \in F$ that $\rho(f^*) = \min_{f \in F} \rho(f)$, where $\rho(f) = \sum_{i=1}^{n-1} \rho(f(i), f(i+1)) + \rho(f(n), f(1))$ for all $f \in F$. Later on, the symbol $I$ is used for the instances of this problem.

Any feasible solution uses only the arcs that start in a subset $X^i$ and end in $X^{i+1}$ for some $i \in \{1, \dots, n\}$ (we assume $n + 1 := 1$). Other arcs are irrelevant to the problem and we assume that they are not given in a problem input $I$.

2-TSPVR is strongly NP-hard [10]. The proof of this fact in [10] is based on a reduction of Clique problem to a family of instances of 2-TSPVR with integer input data, bounded by a polynomial in problem length. Therefore, in view of sufficient condition for non-existence of Fully Polynomial-Time Approximation Scheme (FPTAS) for strongly NP-hard problems [5], the result from [10] implies that 2-TSPVR does not admit an FPTAS, provided that P≠NP. The $k$-TSPVR with $k \geq 3$ cannot be approximated with any constant or polynomial factor of the optimum in polynomial time, unless P=NP, as follows from [11].

## 3. SOLUTION METHOD

Following the approach of A.I. Serdyukov [10], let us consider a bipartite graph $\bar{G} = (X_n, X, \bar{U})$ where the two subsets of vertices of bipartition $X_n, X$ have equal sizes and the set of edges is $\bar{U} = \{\{i, x\} : i \in X_n,\ x \in X^i\}$. Now there is a one-to-one correspondence between the set of perfect matchings $\mathcal{W}$ in the graph $\bar{G}$ and the set $F$ of feasible solutions to a problem instance $I$: Given a perfect matching $W \in \mathcal{W}$ of the form $\{\{1, x^1\}, \{2, x^2\}, \dots, \{n, x^n\}\}$, this mapping produces the tour $(x^1, x^2, \dots, x^n)$.

An edge $\{i, x\} \in \bar{U}$ is called *special* if $\{i, x\}$ belongs to all perfect matchings in the graph $\bar{G}$. Let us also call the vertices of the graph $\bar{G}$ *special* if they are incident with special edges.

Supposing that $\bar{G}$ is given by the lists of adjacent vertices, the special edges and edges that do not belong to any perfect matching in the graph $\bar{G}$ may be efficiently computed by the Algorithm 1, described below. After that, all edges, except for the special edges and those adjacent to them, are slit into cycles. Note that the method of finding all special edges and cycles in the graph $\bar{G}$ was not discussed in [10].

***Algorithm 1. Finding special edges in the graph*** $\bar{G}$

**Step 1** *(Initialization)***.** Assign $\bar{G}' := \bar{G}$.

**Step 2.** Repeat Steps 2.1-2.2 while it is possible:

**Step 2.1** *(Solvability test).* If the graph $\bar{G}'$ contains a vertex of degree 0 then, problem $I$ is infeasible, terminate.

**Step 2.2** *(Finding a special edge)***.** If the graph $\bar{G}'$ contains a vertex $z$ of degree 1, then store the corresponding edge $\{z, y\}$ as a special edge and remove its endpoints $y$ and $z$ from $\bar{G}'$.

Each edge of the graph $\bar{G}$ is visited and deleted at most once (which takes $O(1)$ time). The number of edges $|\bar{U}| \leq 2n$. So the time complexity of Algorithm 1 is $O(n)$.

Algorithm 1 identifies the case when problem $I$ is infeasible. Further, we consider only feasible instances of 2-TSPVR and bipartite graphs corresponding to them.

After the described preprocessing, the resulting graph $\bar{G}'$ is 2-regular (the degree of each vertex equals 2) and its components are even cycles. The cycles of the graph $\bar{G}'$ can be computed in $O(n)$ time using the Depth-First Search algorithm (see e.g. [2]). Note that there are no other edges in the perfect matchings of the graph $\bar{G}$, except for the special edges and edges of the cycles in $\bar{G}'$. In what follows, $q(\bar{G})$ denote the number of cycles in the graph $\bar{G}$ (and in the corresponding graph $\bar{G}'$).

Each cycle $j$, $j = 1, \dots, q(\bar{G})$, contains exactly two maximal (edge disjoint) perfect matchings, so it does not contain any special edges. Every perfect matching in $\bar{G}$ is uniquely defined by a combination of maximal matchings chosen in each of the cycles and the set of all special edges.

As an example consider an instance $I$ with the system of vertex requisitions:
$X^1 = \{x_1, x_2\}$, $X^2 = \{x_1, x_2\}$, $X^3 = \{x_3\}$, $X^4 = \{x_3, x_4\}$, $X^5 = \{x_5, x_6\}$, $X^6 = \{x_6, x_7\}$,
$X^7 = \{x_7, x_8\}$, $X^8 = \{x_6, x_8\}$.
The bipartite graph $\bar{G} = (X_8, X, \bar{U})$ corresponding to this problem is presented in Fig. 1. Here the edges drawn in bold define one maximal matching of a cycle, and the rest of the edges in the cycle define another one. The special edges are depicted by dotted lines. The edges depicted by dashed lines do not belong to any perfect matching. The feasible solutions of the instance $I$ are
$f^1 = (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$, $f^2 = (x_1, x_2, x_3, x_4, x_5, x_7, x_8, x_6)$,
$f^3 = (x_2, x_1, x_3, x_4, x_5, x_6, x_7, x_8)$, $f^4 = (x_2, x_1, x_3, x_4, x_5, x_7, x_8, x_6)$.

Therefore, 2-TSPVR is solvable by the following algorithm.

***Algorithm 2. Solving 2-TSPVR***

**Step 1.** Build the bipartite graph $\bar{G}$, identify the set of special edges and cycles and find all maximal matchings in cycles.

**Step 2.** Enumerate all perfect matchings $W \in \mathcal{W}$ of $\bar{G}$ by combining the maximal matchings of cycles and joining them with special edges.

**Step 3.** Assign the corresponding solution $f \in F$ to each $W \in \mathcal{W}$ and compute $\rho(f)$.

**Step 4.** Output the result $f^* \in F$, such that $\rho(f^*) = \min_{f \in F} \rho(f)$.

To evaluate the Algorithm 2, first note that maximal matchings in cycles are found easily in $O(n)$ time. Now, $|F| = |\mathcal{W}| = 2^{q(\bar{G})}$ so the time complexity of Algorithm 2 of solving 2-TSPVR is $O(n2^{q(\bar{G})})$, where $q(\bar{G}) \leq \lfloor \frac{n}{2} \rfloor$ and the last inequality is tight.

## 4. IMPROVED ALGORITHM

In [10], it was shown that almost all feasible instances of 2-TSPVR have not more than $n$ feasible solutions and may be solved in quadratic time. To describe this

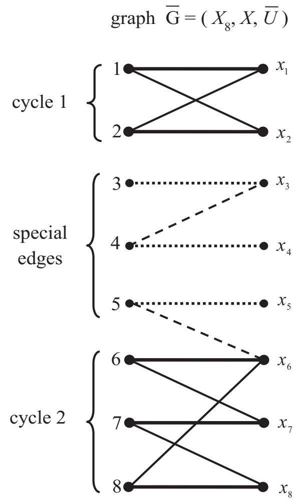


Figure 1: Example of a graph $\bar{G} = (X_8, X, \bar{U})$ with three special edges and two cycles.

result precisely, let us give the following

**Definition 4.1.** [10] *A graph $\bar{G} = (X_n, X, \bar{U})$ is called "good" if it satisfies the inequality $q(\bar{G}) \leq 1.1\ln(n)$.*

Note that any problem instance $I$, which corresponds to a "good" graph $\bar{G}$, has at most $2^{1.1\ln n} < n^{0.77}$ feasible solutions.

Let $\bar{\chi}_n$ denote the set of "good" bipartite graphs $\bar{G} = (X_n, X, \bar{U})$ , and let $\chi_n$ be the set of *all* bipartite graphs $\bar{G} = (X_n, X, \bar{U})$. The results of A.I. Serdyukov from [10] imply

**Theorem 4.2.** $|\bar{\chi}_n|/|\chi_n| \longrightarrow 1$ *as $n \to \infty$.*

The proof of Theorem 4.2 from [10] is provided in the appendix for the sake of completeness. According to the frequently used terminology (see e.g. [1]), this theorem means that almost all feasible instances $I$ have at most $n^{0.77}$ feasible solutions and thus, they are solvable in $O(n^{1.77})$ time by Algorithm 2.

Using the approach from [3], we will now modify Algorithm 2 for solving 2-TSPVR in $O\left(q(\bar{G})2^{q(\bar{G})} + n\right)$ time. Let us carry out some preliminary computations before enumerating all possible combinations of maximal matchings in cycles in order to speed up the evaluation of objective function. We will call a *contact between cycle $j$ and cycle $j' \neq j$ (or between cycle $j$ and a special edge)* the pair of vertices $(i, i+1)$ (we assume $n+1 := 1$) in the left-hand part of the graph $\bar{G}$, such that one of the vertices belongs to the cycle $j$ and the other one belongs to the cycle $j'$ (or the special edge). A *contact inside a cycle* will denote a pair of vertices in the left-hand part of a cycle, if their indices differ exactly by one, or these vertices are $(n, 1)$.

Consider a cycle $j$. If a contact $(i, i+1)$ is present inside this cycle, then each of the two maximal matchings $w^{0,j}$ and $w^{1,j}$ in this cycle determines the $i$-th arc of a tour in the graph $G$. Also, if the cycle $j$ has a contact $(i, i+1)$ to a special edge, each of the two maximal matchings $w^{0,j}$ and $w^{1,j}$ also determines the $i$-th arc of a tour in the graph $G$. For each of the matchings $w^{k,j}$, $k = 0, 1$, let the sum of the weights of arcs determined by the contacts inside the cycle $j$ and the contacts to special edges be denoted by $P_j^k$.

If cycle $j$ contacts to cycle $j'$, $j' \neq j$, then each combination of the maximal matchings of these cycles determines the $i$-th arc of a tour in the graph $G$ for any contact $(i, i+1)$ between the cycles. If a maximal matching is chosen in each of the cycles, one can sum up the weights of the arcs in $G$ determined by all contacts between cycles $j$ and $j'$. This yields four values which we denote by $P_{jj'}^{(0,0)}$, $P_{jj'}^{(0,1)}$, $P_{jj'}^{(1,0)}$, and $P_{jj'}^{(1,1)}$, where the superscripts identify the matchings chosen in each of the cycles $j$ and $j'$, respectively.

Parameters $P_j^0, P_j^1, P_{jj'}^{(0,0)}, P_{jj'}^{(0,1)}, P_{jj'}^{(1,0)}$, and $P_{jj'}^{(1,1)}$ can be found as follows. Suppose that intermediate values of $P_j^0, P_j^1$ for $j = 1, \dots, q(\bar{G})$ are stored in one-dimensional arrays of size $q(\bar{G})$, and intermediate values of $P_{jj'}^{(0,0)}$, $P_{jj'}^{(0,1)}$, $P_{jj'}^{(1,0)}$, and $P_{jj'}^{(1,1)}$ for $j,\ j' = 1, \dots, q(\bar{G})$ are stored in two-dimensional arrays of size $q(\bar{G}) \times q(\bar{G})$. Initially, all of these values are assumed to be zero, and they are computed in an iterative way by the consecutive enumeration of pairs of vertices $(i, i+1)$, $i = 1, \dots, n-1$, and $(n, 1)$ in the left-hand part of the graph $\bar{G}$. When we consider a pair of vertices $(i, i+1)$ or $(n, 1)$, at most four parameters (partial sums) are updated depending on whether the vertices belong to different cycles or to the same cycle, or one of the vertices is special. So, the overall time complexity of the preprocessing procedure is $O(q^2(\bar{G}) + n)$.

Now all possible combinations of the maximal matchings in cycles may be enumerated using a Grey code (see e.g. [8]) so that the next combination differs from the previous one by altering a maximal matching only in one of the cycles. Let the binary vector $\delta = (\delta_1, \dots, \delta_{q(\bar{G})})$ define assignments of the maximal matchings in cycles. Namely, $\delta_j = 0$, if the matching $w^{0,j}$ is chosen in the cycle $j$; otherwise (if the matching $w^{1,j}$ is chosen in the cycle $j$), we have $\delta_j = 1$. This way every vector $\delta$ is bijectively mapped into a feasible solution $f_\delta$ to 2-TSPVR.

In the process of enumeration, a step from the current vector $\bar{\delta}$ to the next vector $\delta$ changes the maximal matching in one of the cycles $j$. The new value of objective function $\rho(f_\delta)$ may be computed via the current value $\rho(f_{\bar{\delta}})$ by the formula $\rho(f_\delta) = \rho(f_{\bar{\delta}}) - P_j^{\bar{\delta}_j} + P_j^{\delta_j} - \sum_{j' \in A(j)} P_{jj'}^{(\bar{\delta}_j, \bar{\delta}_{j'})} + \sum_{j' \in A(j)} P_{jj'}^{(\delta_j, \delta_{j'})}$, where $A(j)$ is the set of cycles contacting to the cycle $j$. Obviously, $|A(j)| \leq q(\bar{G})$, so updating the objective function value for the next solution requires $O(q(\bar{G}))$ time, and the overall time complexity of the modified algorithm for solving 2-TSPVR is $O\left(q(\bar{G})2^{q(\bar{G})} + n\right)$.

In view of Theorem 4.2 we conclude that using this modification of Algorithm 2, almost all feasible instances of 2-TSPVR are solvable in $O(n^{0.77} \ln n + n) = O(n)$ time.

## 5. LOCAL SEARCH

A local search algorithm starts from an initial feasible solution. It moves iteratively from one solution to a better neighboring solution and terminates at a local optimum. The number of steps of the algorithm, the time complexity of one step, and the value of the local optimum depend essentially on the neighborhood. Note that neighborhoods, often used for the classical TSP (e.g. k-Opt, city-swap, Lin-Kernighan [7]), will contain many infeasible neighboring solutions if applied to 2-TSPVR because of the vertex requisition constraints.

A local search method with a specific neighborhood for 2-TSPVR may be constructed using the relationship between the perfect matchings in the graph $\bar{G}$ and the feasible solutions. The main idea of the algorithm consists in building a neighborhood of a feasible solution to 2-TSPVR on the basis of a Flip neighborhood of the perfect matching, represented by the maximal matchings in cycles and the special edges.

Let the binary vector $\delta = (\delta_1, \ldots, \delta_{q(\bar{G})})$ denote the assignment of the maximal matchings to cycles, as in Section 4. The set of $2^{q(\bar{G})}$ vectors $\delta$ corresponds to the set of feasible solutions by a one-to-one mapping $f_\delta$. We assume that a solution $f_{\delta'}$ belongs to the *Exchange* neighborhood of solution $f_\delta$ iff the vector $\delta'$ is within Hamming distance 1 from $\delta$, i.e. $\delta'$ belongs to the Flip neighborhood of vector $\delta$.

Enumeration of the Exchange neighborhood takes $O(q^2(\bar{G}))$ time if the preprocessing described in Section 4 is carried out before the start of the local search (without the preprocessing it takes $O(nq(\bar{G}))$ operations). Therefore, for almost all feasible instances $I$, the Exchange neighborhood may be enumerated in $O(\ln^2(n))$ time.

## 6. MIXED INTEGER LINEAR PROGRAMMING MODEL

The one-to-one mapping between the maximal matchings in cycles of the graph $\bar{G}$ and feasible solutions to 2-TSPVR may be also exploited in formulation of a mixed integer linear programming model.

Recall that $P^0_j$ ($P^1_j$) is the sum of weights of all arcs of the graph $G$ determined by the contacts inside the cycle $j$ and the contacts of the cycle $j$ with special edges, when the maximal matching $w^{0,j}$ ($w^{1,j}$) is chosen in the cycle $j$, $j = 1, \ldots, q(\bar{G})$. Furthermore, $P^{(k,l)}_{jj'}$ is the sum of weights of arcs in the graph $G$ determined by the contacts between cycles $j$ and $j'$, if the maximal matchings $w^{k,j}$ and $w^{l,j'}$ are chosen in the cycles $j$ and $j'$, respectively, $k, l = 0, 1$, $j = 1, \dots, q(\bar{G}) - 1$, $j' = j + 1, \dots, q(\bar{G})$. These values are computable in $O(n + q^2(\bar{G}))$ time as shown in Section 4.

Let us introduce the following Boolean variables:

$$d_j = \begin{cases} 0, & \text{if matching } w^{0,j} \text{ is chosen in cycle } j, \\ 1, & \text{if matching } w^{1,j} \text{ is chosen in cycle } j, \end{cases} \quad j = 1, \dots, q(\bar{G}).$$

The objective function combines the pre-computed arc weights for all cycles, depending on the choice of matchings in $d = (d_1, \dots, d_{q(\bar{G})})$:

$$\sum_{j=1}^{q(\bar{G})-1} \sum_{j'=j+1}^{q(\bar{G})} \left( P_{jj'}^{(0,0)}(1 - d_j)(1 - d_{j'}) + P_{jj'}^{(0,1)}(1 - d_j) d_{j'} \right)$$
$$+ \sum_{j=1}^{q(\bar{G})-1} \sum_{j'=j+1}^{q(\bar{G})} \left( P_{jj'}^{(1,0)} d_j (1 - d_{j'}) + P_{jj'}^{(1,1)} d_j d_{j'} \right) \qquad (1)$$
$$+ \sum_{j=1}^{q(\bar{G})} \left( P_j^0 (1 - d_j) + P_j^1 d_j \right) \to \min,$$

$$d_j \in \{0, 1\}, \; j = 1, \dots, q(\bar{G}). \qquad (2)$$

Let us define supplementary real variables in order to remove non-linearity of the objective function: for $k \in \{0, 1\}$, we assume that $p_j^{(k)} \geq 0$ is an upper bound on the sum of weights of arcs in the graph $\bar{G}$, determined by the contacts of the cycle $j$, if matching $w^{k,j}$ is chosen in this cycle, i.e. $d_j = k$, $j = 1, \dots, q(\bar{G}) - 1$.

Then the mixed integer linear programming model has the following form:

$$\sum_{j=1}^{q(\bar{G})-1} \left( p_j^{(0)} + p_j^{(1)} \right) + \sum_{j=1}^{q(\bar{G})} \left( P_j^0 (1 - d_j) + P_j^1 d_j \right) \to \min, \qquad (3)$$

$$p_j^{(0)} \geq \sum_{j'=j+1}^{q(\bar{G})} P_{jj'}^{(0,0)} \left( 1 - d_j - d_{j'} \right) + \sum_{j'=j+1}^{q(\bar{G})} P_{jj'}^{(0,1)} \left( d_{j'} - d_j \right),$$
$$j = 1, \dots, q(\bar{G}) - 1, \qquad (4)$$

$$p_j^{(1)} \geq \sum_{j'=j+1}^{q(\bar{G})} P_{jj'}^{(1,0)} \left( d_j - d_{j'} \right) + \sum_{j'=j+1}^{q(\bar{G})} P_{jj'}^{(1,1)} \left( d_j + d_{j'} - 1 \right),$$
$$j = 1, \dots, q(\bar{G}) - 1, \qquad (5)$$

$$p_j^{(k)} \geq 0, \; k = 0, 1, \; j = 1, \dots, q(\bar{G}) - 1, \qquad (6)$$

$$d_j \in \{0, 1\}, \ j = 1, \ldots, q(\bar{G}). \quad (7)$$

Note that if matching $w^{0,j}$ is chosen for the cycle $j$ in an optimal solution of problem (3)-(7), then inequality (4) holds for $p_j^{(0)}$ as equality and $p_j^{(1)} = 0$. Analogously, if matching $w^{1,j}$ is chosen for the cycle $j$, then inequality (5) holds for $p_j^{(1)}$ as equality and $p_j^{(0)} = 0$. Therefore, problems (1)-(2) and (3)-(7) are equivalent because a feasible solution of one problem corresponds to a feasible solution of another problem, and their optimal solutions correspond as well.

The number of real variables in model ((3-7) is $(2q(\bar{G}) - 2)$, the number of Boolean variables is $q(\bar{G})$. The number of constraints is $O(q(\bar{G}))$, where $q(\bar{G}) \leq \lfloor \frac{n}{2} \rfloor$. The proposed model may be used for computing lower bound of the objective function, or in branch-and-bound algorithms, even if the graph $\bar{G}$ is not "good".

Note that there are a number of integer linear programming models in the literature on the classical TSP, involving $O(n^2)$ Boolean variables. Model (3-7) for 2-TSPVR has at most $\lfloor \frac{n}{2} \rfloor$ Boolean variables and, for almost all feasible instances, the number of Boolean variables is $O(\ln(n))$.

## 7. CONCLUSION

We presented an algorithm for solving 2-TSP WITH VERTEX REQUISITIONS that reduces the time complexity bound formulated in [10]. It is easy to see that the same approach is applicable to the problem 2-HAMILTONIAN PATH OF MINIMUM WEIGHT WITH VERTEX REQUISITIONS, which asks for a Hamiltonian Path of Minimum Weight in the graph $G$, assuming the same system of vertex requisitions as in 2-TSP WITH VERTEX REQUISITIONS.

Using the connection to perfect matchings in a supplementary bipartite graph and some preprocessing, we constructed a MIP model with $O(n)$ binary variables and a new efficiently searchable *Exchange* neighborhood for the problem under consideration.

Further research might address the existence of approximation algorithms with constant approximation ratio for 2-TSP WITH VERTEX REQUISITIONS.

## REFERENCES

[1] Chvatal, V., "Probabilistic methods in graph theory", *Annals of Operations Research*, 1 (1984) 171-182.

[2] Cormen, T.H., Leiserson, C.E., Rivest, R.L., and Stein, C., *Introduction to Algorithms*, 2nd edition, MIT Press, 2001.

[3] Eremeev, A., and Kovalenko, J., "Optimal recombination in genetic algorithms for combinatorial optimization problems: Part II", *Yugoslav Journal of Operations Research*, 24 (2) (2014) 165-186.

[4] Feller, W., *An Introduction to Probability Theory and Its Aplications*, Vol. 1, John Wiley & Sons, New York, NY, 1968.

[5] Garey, M.R., and Johnson, D.S., "Strong NP-completeness results: Motivation, examples, and implications", *Journal of the ACM*, 25 (1978) 499-508.

[6] Garey, M.R., and Johnson, D.S., *Computers and intractability. A guide to the theory of NP-completeness*, W.H. Freeman and Company, San Francisco, CA, 1979.

[7] Kochetov, Yu. A., "Computational bounds for local search in combinatorial optimization", *Computational Mathematics and Mathematical Physics*, 48 (5) (2008) 747-763.

[8] Reingold, E.M., Nievergelt, J., and Deo, N., *Combinatorial algorithms: Theory and Practice,* Englewood Cliffs, Prentice-Hall, 1977.
[9] Riordan, J., *An Introduction to Combinatorial Analysis*, John Wiley & Sons, New York, NY, 1958.
[10] Serdyukov, A.I., "On travelling salesman problem with prohibitions", *Upravlaemye systemi,* 17 (1978) 80-86. (In Russian)
[11] Serdyukov, A.I., "On finding Hamilton cycle (circuit) problem with prohibitions", *Upravlaemye systemi,* 19 (1979) 57-64. (In Russian)
[12] Serdyukov, A.I., "Complexity of solving the travelling salesman problem with requisitions on graphs with small degree of vertices", *Upravlaemye systemi,* 26 (1985) 73-82. (In Russian)

## APPENDIX

Note that A.I. Serdyukov in [10] used the term *block* instead of term *cycle,* employed in Section 3 of the present paper. A *block* was defined in [10] as a maximal (by inclusion) 2-connected subgraph of graph $\bar{G}$ with at least two edges. However, in each block of the graph $\bar{G}$, the degree of every vertex equals 2 (otherwise $F = \emptyset$ because the vertices of degree 1 do not belong to blocks and the vertex degrees are at most 2 in the right-hand part of $\bar{G}$). So, the notions *block* and *cycle* are equivalent in the case of considered bipartite graph $\bar{G}$. We use the term *block* in the proof of Theorem 4.2 below, as in the original paper [10], in order to avoid a confusion of cycles in $\bar{G}$ with cycles in permutations of the set $\{1, \dots, n\}$.

[10] $|\bar{\chi}_n|/|\chi_n| \longrightarrow 1$ as $n \to \infty$.

**Proof.** Let $\mathcal{S}_n$ be the set of all permutations of the set $\{1, \dots, n\}$. Consider a random permutation $s$ from $\mathcal{S}_n$. By $\xi(s)$ denote the number of cycles in permutation $s$. It is known (see e.g. [4]) that the expectation $E[\xi(s)]$ of random variable $\xi(s)$ is equal to $\sum_{i=1}^{n} \frac{1}{i}$ and the variance $Var[\xi(s)]$ equals $\sum_{i=1}^{n} \frac{i-1}{i^2}$. Let $\bar{\mathcal{S}}_n$ denote the set of permutations from $\mathcal{S}_n$, where the number of cycles is at most $1.1 \ln(l)$. Then, using Chebychev's inequality [4], we get

$$|\bar{\mathcal{S}}_n|/|\mathcal{S}_n| \longrightarrow 1 \text{ as } n \to \infty. \tag{8}$$

Now let $\mathcal{S}'_n$ denote the set of permutations from $\mathcal{S}_n$, which do not contain the cycles of length 1, and let $\mathcal{S}_n^{(i)}$ be the set of permutations from $\mathcal{S}_n$, which contain a cycle with element $i$, $i = 1, \dots, n$. Using the principle of inclusion and exclusion [9], we obtain

$$\left|\mathcal{S}_n \setminus \mathcal{S}'_n\right| = \left|\bigcup_{1 \le i \le n} \mathcal{S}_n^{(i)}\right| = \sum_{i=1}^{n} \left|\mathcal{S}_n^{(i)}\right| - \sum_{1 \le i \ne j \le n} \left|\mathcal{S}_n^{(i)} \bigcap \mathcal{S}_n^{(j)}\right| +$$

$$\sum_{1 \le i \ne j \ne k \le n} \left|\mathcal{S}_n^{(i)} \bigcap \mathcal{S}_n^{(j)} \bigcap \mathcal{S}_n^{(k)}\right| - \cdots = n! - C_n^2(n-2)! + C_n^3(n-3)! - \cdots \le$$

$$\frac{n!}{2} + \frac{n!}{6} = \frac{2}{3}n! = \frac{2}{3}|\mathcal{S}_n|.$$

Therefore,

$$|\mathcal{S}'_n| \geq \frac{1}{3}|\mathcal{S}_n|. \tag{9}$$

Combining (8) and (9), we get

$$|\bar{\mathcal{S}}'_n|/|\mathcal{S}'_n| = 1 - \frac{|\mathcal{S}'_n \backslash \bar{\mathcal{S}}'_n|}{|\mathcal{S}'_n|} \geq 1 - \frac{3|\mathcal{S}'_n \backslash \bar{\mathcal{S}}'_n|}{|\mathcal{S}_n|} \geq 1 - \frac{3|\mathcal{S}_n \backslash \bar{\mathcal{S}}_n|}{|\mathcal{S}_n|} \underset{n\to+\infty}{\longrightarrow} 1, \tag{10}$$

where $\bar{\mathcal{S}}'_n = \mathcal{S}'_n \cap \bar{\mathcal{S}}_n$.

The values $|\bar{\chi}_n|$ and $|\chi_n \backslash \bar{\chi}_n|$ may be bounded, using the following approach. We assign any permutation $s \in \mathcal{S}'_l$, $l \leq n$, a set of bipartite graphs $\chi_n(s) \subset \chi_n$ as follows. First of all let us assign an arbitrary set of $n - l$ edges to be special. Then the non-special vertices $\{i_1, i_2, \ldots, i_l\} \subset X_n$ of the left-hand part, where $i_j < i_{j+1}$, $j = 1, \ldots, l-1$, are now partitioned into $\xi(s)$ blocks, where $\xi(s)$ is the number of cycles in permutation $s$. Every cycle $(t_1, t_2, \ldots, t_r)$ in permutation $s$ corresponds to some sequence of vertices with indices $\{i_{t_1}, i_{t_2}, \ldots, i_{t_r}\}$ belonging to the block associated with this cycle. Finally, it is ensured that for each pair of vertices $\{i_{t_j}, i_{t_{j+1}}\}$, $j = 1, \ldots, r-1$, as well as for the pair $\{i_{t_r}, i_{t_1}\}$, there exists a vertex in the right-hand part $X$ which is adjacent to both vertices of the pair. Except for special edges and blocks additional edges are allowed in graphs from class $\chi_n(s)$. These edges are adjacent to the special vertices of the left-hand part such that the degree of any vertex of the left-hand part is not greater than two. Moreover, additional edges should not lead to creating new blocks.

There are $n!$ ways to associate vertices of the left-hand part to vertices of the right-hand part, therefore the number of different graphs from class $\chi_n(s)$, $s \in \mathcal{S}'_l$, $l \leq n$, is $|\chi_n(s)| = C_n^l \frac{n!}{2^{\xi_1(s)}} h(n,l)$, where function $h(n,l)$ depends only on $n$ and $l$, and $\xi_1(s)$ is the number of cycles of length two in permutation $s$. Division by $2^{\xi_1(s)}$ is here due to the fact that for each block that corresponds to a cycle of length two in $s$, there are two equivalent ways to number the vertices in its right-hand part.

Let $s = c_1 c_2 \ldots c_{\xi(s)}$ be a permutation from set $\mathcal{S}'_l$, represented by cycles $c_i$, $i = 1, \ldots, \xi(s)$, and let $c_j$ be an arbitrary cycle of permutation $s$ of length at least three, $j = 1, \ldots, \xi(s)$. Permutation $s$ may be transformed into permutation $s^1$,

$$s^1 = c_1 c_2 \ldots c_{j-1} c_j^{-1} c_{j+1} \ldots c_{\xi(s),} \tag{11}$$

by reversing the cycle $c_j$. Clearly, permutation $s^1$ induces the same subset of bipartite graphs in class $\chi_n$ as the permutation $s$ does. Thus any two permutations $s^1$ and $s^2$ from set $\mathcal{S}'_l$, $l \leq n$, induce the same subset of graphs in $\chi_n$, if one of these permutations may be obtained from the other one by several transformations of the form (11). Otherwise the two induced subsets of graphs do not intersect. Besides that $\chi_n(s^1) \cap \chi_n(s^2) = \emptyset$ if $s^1 \in \mathcal{S}'_{l_1}$, $s^2 \in \mathcal{S}'_{l_2}$, $l_1 \neq l_2$.

On one hand, if $s \in \bar{\mathcal{S}}'_l$, $l \le n$, then $\chi_n(s) \subseteq \bar{\chi}_n$. On the other hand, if $s \in \tilde{\mathcal{S}}'_l := \mathcal{S}'_l \backslash \bar{\mathcal{S}}'_l$, $l < n$, then either $\chi_n(s) \subseteq \bar{\chi}_n$ or, alternatively, $\chi_n(s) \subseteq \chi_n \backslash \bar{\chi}_n$ may hold. Therefore,

$$|\bar{\chi}_n| \ge \sum_{l=2}^{n} \sum_{s \in \bar{\mathcal{S}}'_l} C_n^l \frac{n!}{2^{\xi_1(s)} 2^{\xi(s)-\xi_1(s)}} h(n,l) = \sum_{l=2}^{n} \sum_{s \in \bar{\mathcal{S}}'_l} C_n^l \frac{n!}{2^{\xi(s)}} h(n,l) \ge \tag{12}$$

$$\ge \sum_{l=2}^{n} |\bar{\mathcal{S}}'_l| \cdot C_n^l \frac{n!}{2^{1.1\ln(l)}} h(n,l) \ge \sum_{l=\lfloor 1.1\ln(n) \rfloor}^{n} |\bar{\mathcal{S}}'_l| \cdot C_n^l \frac{n!}{2^{1.1\ln(l)}} h(n,l),$$

$$|\chi_n \backslash \bar{\chi}_n| \le \sum_{l=\lfloor 1.1\ln(n) \rfloor}^{n} \sum_{s \in \tilde{\mathcal{S}}'_l} C_n^l \frac{n!}{2^{\xi(s)}} \le \sum_{l=\lfloor 1.1\ln(n) \rfloor}^{n} |\tilde{\mathcal{S}}'_l| \cdot C_n^l \frac{n!}{2^{1.1\ln(l)}} h(n,l). \tag{13}$$

Now assuming $\psi(n) = \max_{l=\lfloor 1.1\ln(n) \rfloor,\dots,n} |\tilde{\mathcal{S}}'_l| / |\bar{\mathcal{S}}'_l|$ and taking into account (12), (13) and (10), we obtain

$$\frac{|\chi_n \backslash \bar{\chi}_n|}{|\bar{\chi}_n|} \le \psi(n) \to 0 \text{ as } n \to +\infty. \tag{14}$$

Finally, the statement of the theorem follows from (14). Q.E.D.