Abstract — The Mixed Chinese Postman Problem (MCPP) is to find a minimum shortest tour of given graph or multigraph traversed each edge or arc at least once. The problem is NP-hard. However, mixed case of the problem has many potentially useful applications, including delivering of something, robot exploration, web site usability, etc. In this article, we propose to solve the problem using the graph transformation and solving well-known Asymmetric Travelling Salesman Problem (ATSP). The algorithm for transforming the MCPP in multigraph into ATSP is pointed out.

Keywords — Mixed Chinese Postman Problem, Arc Routing Problem, graph transformation, Travelling Salesman Problem.

I. INTRODUCTION

In the Mixed Chinese Postman Problem in multigraph, we a given a set of vertices (cities) and arcs and edges, that connect the vertices and define non-negative costs between them. The problem was originally studied by the Chinese mathematician Kwan Mei-Ko in 1962 [1]. A problem is called the Chinese Postman Problem (CPP) after him [2].

Apart from the traditional application of the CPP to solving the routing problems such as path planning of snowplows or serving teams, there is a wide range of applications including robot exploration, testing web site usability and finding broken links [3].

There are many various classifications of CPP. Let is consider some of them.

A. Classification based on graph type

Depending on the graph type, the CPP can be following:

- Undirected Chinese Postman Problem (UCPP, UPP). The problem is defined on undirected graph $G = (V, E, C)$, where $V$ is the set of vertices, $E$ is the set of edges, $C: E \to \mathbb{R}_+$— cost function giving non-negative weights of arcs and edges between vertices. Each edge can be traversed in both directions. The problem can be solved exactly in polynomial time. Notice that between a single pair of vertices can be more than one arc. The exact polynomial solution was provided [4].

- Directed Chinese Postman Problem (DCPP, DPP). In this case, the directed graph $G = (V, A, C)$ is given, where $E$ is the set of edges. Each arc has a direction and can be traversed in one way. The problem also may be determined on multigraph, where between a single pair of vertices can be more than one arc. The exact polynomial solution was provided [3].

- Mixed Chinese Postman Problem (MCPP, MPP). The initial graph $G = (V, E \cup A, C)$ consists of directed and undirected edges, simultaneously [5]. The MCPP is a simply-stated problem, which has many potentially useful applications, however, has no exact algorithms [6].

B. Classification based on tour type

According to type of tour, the CPP can be following:

- Closed CPP. The solution tour should be closed (start and end vertex are equal). It means that postman should beginning and ending at the same point [3].

- Open CPP. The postman should not return to the starting point [3].

C. Classification associated with arcs and edges

There are many modifications of CPP based on arcs and/or edges’ parameters variation. Below some of them are presented.

- Windy Chinese Postman Problem. The cost between to vertices $v_i$ and $v_j$ depends on traversal direction ($c_{ij} \neq c_{ji}$) [7].

- Rural Chinese Postman Problem. The set of edges $E$ in undirected graph, the set of arcs $A$ in directed graph or the set $\{A \cup E\}$ in mixed graph consists from two subsets: required and non-required. Required arcs/edges should appear in solution at least once, non-required edges can absent in tour solution [8].

- The Hierarchical Postman Problem. The set of edges $E$ in undirected graph, the set of arcs $A$ in directed graph or the set $\{A \cup E\}$ in mixed graph partitioned into clusters and on each clusters the priority of the traversing is determined [9].

Above the main classifications and typical representatives of the problem are represented. However, the list of variations of the problem is not exhaustive.

In this paper, we are talking about the MCPP in multigraph. Despite the fact that the problem has many applications, there is no exact solution. There are many

Manuscript received April 20, 2017.

Maria K. Gordenko is with the Department of Software Engineering, Faculty of Computer Science, National Research University Higher School of Economics, Moscow, Russia (mkgordenko@edu.hse.ru).

Sergey M. Avdoshin is with the Department of Software Engineering, Faculty of Computer Science, National Research University Higher School of Economics, Moscow, Russia (savdoshin@hse.ru).
heuristic algorithms for solving the problem in a mixed graph [6]. However, it was not possible to find an algorithm for solving the problem in a mixed multigraph. To obtain a solution in the mixed multigraph, the methods of transforming the problem into equivalent arc routing problem Asymmetric Travelling Salesman Problem (ATSP) throw the transformation of initial problem into the Asymmetric Generalized Travelling Salesman Problem (GTSP) were examined [10–12].

The paper is organized as follows. First, the mathematical formulation of the problem is pointed out. The next section is dedicated to related works. In the following parts the description of implemented algorithm and methods of the research are presented. Then already obtained results are revealed. In the final part, the future directions of research are described.

II. MATHEMATICAL FORMULATION OF THE MCPP

The weight strongly connected mixed multigraph \( G = < V, E \cup A, C > \) is given, where \( V \) is the set of multigraph's vertices, \( E \) is the multiset of edges, \( A \) is the multiset of arcs, \( C : E \cup A \rightarrow R_+ \) cost function giving non-negative weights of arcs and edges between vertices [6].

Let \( I = \{1, 2, \ldots, |E + A|\} \), \( L = \{1, 2, \ldots, |V|\} \).

Indexation on the set of vertices \( V \) is defined as \( inv: V \rightarrow L \), \( \forall \nu_i \in V \ \forall \nu_j \in V \ \nu_i \neq \nu_j \Rightarrow i \neq j \), \( i = inv(\nu_i) \). On the multiset \( E \cup A \) indexation is defined as \( inea: E \cup A \rightarrow I \), \( \forall e_i \in (E \cup A) \ \forall e_j \in (E \cup A) \ \ e_i \neq e_j \Rightarrow i \neq j \), \( i = inea(e_i) \).

Route \( (1) \) is a solution of the MCPP that satisfies the following properties.

\( v^{-}(e_{p_1}) = v^{+}(e_{p_k}) \), \( \forall i \in \{1, 2, \ldots, k-1\} \ v^{+}(e_{p_i}) = v^{-}(e_{p_{i+1}}) \), where \( v^{-} \) is the start vertex of arc or edge \( e \), \( v^{+} \) is the end vertices of arc or edge \( e \).

\( E \cup A \) \( \{e_{p_1}, e_{p_2}, \ldots, e_{p_k}\} = \emptyset \).

Let \( C(\mu) = \sum_{i=1}^{k} C(e_i) \) is the cost of the route.

Let \( \mathcal{M} \) be a set of routes of the MCPP. It is necessary to find a route \( \mu_\mathcal{M} \in \mathcal{M} \) that satisfies the following property \( \forall \mu \in \mathcal{M} \ C(\mu_\mathcal{M}) \leq C(\mu) \) or \( C(\mu_\mathcal{M}) = \min_{\mu \in \mathcal{M}} C(\mu) \).

III. RELATED WORKS

Number of articles and the research on the CPP issue have been found.

The most popular classifications and variations of the CPP was found in the paper by D. Mohammaditabar [10].

The mathematical formulations of problems can be found in the Optimization and Operation Research book. It shows that the UCPP and DCPP has exact solution and can be solved in polynomial time. The MCPP is NP-hard [11].

In the work of Harold Thimbleby from University College London, the polynomial algorithm and executable code on Java for DCPP in multigraph were represented. The article reveals possible applications CPP. Great attention is paid to the application of the solution of the problem to testing web-site, broken links and etc. [3].

In many researches, the algorithm for solving UCPP is presented. The algorithm is based on making the Eulerian graph and finding the Eulerian cycle in it.

In the paper by G. Laporte the exact algorithm for an undirected and directed graph is covered and the formulation of the problem in the mixed graph is shown [12]. The T. Ralph's paper contains formulation of CPP on the mixed graph in terms of integer linear programming [6].

For the Mixed case, the many heuristics are existed. However, for the MCPP in multigraph finding the algorithms description was failed [6].

It was shown that several arc routing problems (ARP) can be converted into equivalent ARP. It opens the possibility to solve ARP, including the MCPP, as equivalent of another arc routing problem [11–13].

The paper given grounds to assume the MCPP can be converted into Asymmetric Travelling Salesman Problem (ATSP).

IV. TRANSFORMATION THE MCPP INTO ASYMMETRIC GTSP

The MCPP can be transformed into an equivalent arc routing problem. When problem defined in directed multigraph (DCPP), it can be transformed into ATSP. When problem defined in mixed or undirected multigraph, it can be at first transformed into an asymmetric GTSP, and after that into ATSP [11–12]. The proof of correctness of this algorithm was given by M. Blais [13].

A. Description of transformation algorithm

The process of transformation the MCPP to GTSP is to transform the original graph \( G = < V, E \cup A, C > \) into equivalent problem on complete graph \( \tilde{G} = < \tilde{V}, \tilde{A}, \tilde{C} > \).

Each arc \( a^k_{ij} \in A \) between to vertices \( \nu_i, \nu_j \in V \) is represented as vertex \( \nu_{ij}^k \in \tilde{V} \), which must be used in the solution at least once, where \( k \) is the serial number of parallel arc.

Each edge \( e^k_{ij} \in \tilde{E} \) between to vertices \( \nu_i, \nu_j \in V \) is represented as two vertices \( \nu_{ij}^{k_1}, \nu_{ij}^{k_2} \in \tilde{V} \), one of which must be used in the solution, another may not be used, where \( k \) is a serial number of parallel edge, \( k_1, k_2 \) are the serial numbers of parallel arcs between two vertices.

After replacing the arcs and edges in vertices, the cost from each pair of vertex \( \nu_{ab}, \nu_{cd} \in \tilde{V} \) graph \( \tilde{G} \) compute, as

\[ \tilde{c}_{ij} = d_{bc} + c_{cd} \] (2)

where \( d_{bc} \) is the shortest distance between vertices \( \nu_b, \nu_c \in V \) in original multigraph \( G \).

Then, the compete graph \( \tilde{G} \) is partitioned into clusters as follows: each arc and each edge is separate clusters. The number of clusters is equals to \( |A \cup E| \). The graph partitioned into clusters because edge can be traverse in two ways, for solving the MCPP any way is appropriate and the problem transforms into asymmetric GTSP [13].

The GTSP is a variation of the Traveling Salesman Problem in which all vertices are divided into clusters, and solution consist from one vertices from each cluster.
TABLE I. FORMULAS FOR COMPUTING ARC COSTS OF ASYMMETRIC GTSP

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$v_{12}$</td>
<td>$v_{13}$</td>
<td>$v_{14}$</td>
<td>$v_{23}$</td>
<td>$v_{24}$</td>
<td>$v_{34}$</td>
<td>$v_{12}$</td>
</tr>
<tr>
<td>2</td>
<td>$v_{12}$</td>
<td>-</td>
<td>$s_{21} + c_{12}$</td>
<td>$s_{21} + c_{21}$</td>
<td>$s_{22} + c_{23}$</td>
<td>$s_{22} + c_{24}$</td>
<td>$s_{23} + c_{24}$</td>
</tr>
<tr>
<td>3</td>
<td>$v_{13}$</td>
<td>$s_{31} + c_{12}$</td>
<td>-</td>
<td>$s_{31} + c_{31}$</td>
<td>$s_{32} + c_{33}$</td>
<td>$s_{32} + c_{34}$</td>
<td>$s_{33} + c_{34}$</td>
</tr>
<tr>
<td>4</td>
<td>$v_{14}$</td>
<td>$s_{41} + c_{12}$</td>
<td>$s_{41} + c_{31}$</td>
<td>-</td>
<td>$s_{42} + c_{33}$</td>
<td>$s_{42} + c_{34}$</td>
<td>$s_{43} + c_{34}$</td>
</tr>
<tr>
<td>5</td>
<td>$v_{23}$</td>
<td>$s_{21} + c_{12}$</td>
<td>$s_{21} + c_{31}$</td>
<td>$s_{22} + c_{23}$</td>
<td>-</td>
<td>$s_{22} + c_{24}$</td>
<td>$s_{23} + c_{24}$</td>
</tr>
<tr>
<td>6</td>
<td>$v_{24}$</td>
<td>$s_{21} + c_{12}$</td>
<td>$s_{21} + c_{31}$</td>
<td>$s_{22} + c_{23}$</td>
<td>$s_{22} + c_{24}$</td>
<td>-</td>
<td>$s_{23} + c_{24}$</td>
</tr>
<tr>
<td>7</td>
<td>$v_{34}$</td>
<td>$s_{31} + c_{12}$</td>
<td>$s_{31} + c_{31}$</td>
<td>$s_{32} + c_{33}$</td>
<td>$s_{32} + c_{34}$</td>
<td>$s_{33} + c_{34}$</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>$v_{42}$</td>
<td>$s_{41} + c_{12}$</td>
<td>$s_{41} + c_{31}$</td>
<td>$s_{42} + c_{33}$</td>
<td>$s_{42} + c_{34}$</td>
<td>$s_{43} + c_{34}$</td>
<td>$s_{44} + c_{34}$</td>
</tr>
</tbody>
</table>

B. The example of transformation

The example of original multigraph is shown on Fig. 1. Each arc and edge has the cost of traverse. Each vertex has a serial number.

Fig. 1. The original MCPP problem in multigraph

We replace each edge by a pair of two oppositely directed arcs and specify the numbering of parallel arcs between each pair of vertices (see Fig. 2). In multigraph only one arc $a_{12}^{1}$ or $a_{23}^{2}$ is required, because these arcs represent one edge. The same applies to arc $a_{23}^{2}$ or $a_{32}^{3}$.

Fig. 2. The results of numbering each parallel arc

After that, should replace each arc and edge as vertex. We received new graph $G$ with 8 vertices. The $\bar{V}$ can be calculates as formula:

$$|\bar{V}| = |\bar{A}| + 2|\bar{E}|$$

The cost from each pair of vertices is calculated by formulas (see Table 1, see Table 2). The vertices represent the edge are marked with a color in the table (different colors for different edges).

TABLE II. THE COST MATRIX

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$v_{12}$</td>
<td>-</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>$v_{13}$</td>
<td>5</td>
<td>10</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>$v_{14}$</td>
<td>5</td>
<td>9</td>
<td>-</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>$v_{23}$</td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>-</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>$v_{24}$</td>
<td>5</td>
<td>9</td>
<td>10</td>
<td>4</td>
<td>-</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>$v_{32}$</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>$v_{34}$</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

Then vertices from $V$ are partitioned into clusters. Fig. 3 depicts the vertices and clusters of transformed graph.

Fig. 3. The vertices and clusters of transformed problems

The transformation of the MCPP into Asymmetric GTSP is received. After transformation of the original MCPP into GTSP, the existing algorithm for GTSP can be applied. However, we try to transform received Asymmetric GTSP into ATSP.

V. TRANSFORMATION THE ASYMMETRIC GTSP INTO ATSP

The Asymmetric GTSP can be transformed into ATSP with the same number of vertices. Proof that the algorithm is applicable to any GTSP is given in the paper by G. Laporte [13]. Below the description of transformation algorithm is described [12, 16].

A. Description of transformation algorithm

The steps of transformation:

- Define the set of vertices $\bar{V}$, which is equal $\bar{V}$.
- Within each cluster, set the cost of the arcs between each pair of vertices in the cluster equal to 0. $c_{ij} = 0$,
For each dimension, there is 24 different tasks. In Corberán’s database the input data for graph are presented. Different task s. In Corberán’s database the input data for graph are presented. For each dimension, there is 25 different tasks. In subsection A.

To test all algorithms, two test databases were used. For solving ATSP the library LKH by Keld Helsgaun was used as a black box [17]. The code for graph transformations was written on C++ in MS Visual Studio 2015. For solving ATSP the library LKH by Keld Helsgaun was used as a black box [17].

The based idea of this transformation that salesman, firstly visit all vertices into clusters and then go into another cluster. The GTSP tour can be easily find from TSP tour, by extracting first vertex from each cluster from TSP tour [14].

If two vertices \( v_i, v_j \in \mathcal{V} \) are in different clusters, the cost calculate as:

\[
\overline{c}_{ij} = \overline{c}_{ij}^{t}
\]

where \( \overline{c}_{ij} \) is another vertex from cluster with \( v_i \).

The measurements for each input data set were carried out 10 times. The results of computational time were obtained as the average of 10 runs of the program:

\[
T_{av} = \frac{T_{1} + \cdots + T_{10}}{10}
\]

At first, the MCPP was transformed into the GTSP, then the GTSP was converted into ATSP. The error and time performance of the LKH [16, 17] algorithm was measured on the converted data, as follows:

- Test converted data of ATSP were loaded in console program;
- The measurements for each input data set were carried out 10 times. The results of time performance were obtained as the average of 10 runs of the program (according to the formula (5));
- Error rate of the algorithms was evaluated according to the formula:

\[
Error = \frac{C(\mu) - C(\mu_0)}{C(\mu_0)}
\]

where \( C(\mu) \) is the resulting length of the route of the MCPP, \( C(\mu_0) \) is the optimum length of the route of the MCPP given in input data.

All test provides on Mac Book Pro 13 retina 2014 (Intel Core i5, 2.6 GHz).

### VII. Obtained Results

The test results are divided into three subsections. At first, the time performance of the MCPP into the GTSP transformation was received. Second, the testing of transformation the GTSP into ATSP was conducted. Finally, the LKH [18, 19] algorithm was applied to the transformed data and its time performance and error rate were measured.

#### A. Results of testing transformation the MCPP into the GTSP

The time performance of algorithm’s work most depends on number of vertex \(|V|\) in transformed problems. The graphic for \(|V| = 500\) is presented on Fig.4. The results for all test data are presented in Table 6.

![Fig. 4. The time performance of transforming the MCPP into the GTSP, \(|V| = 500\).](image)

Relatively long time of work is because that during the calculation of the transformed weights for the original matrix, it is necessary to find the matrix of shortest paths between all pairs of vertices (using the Floyd Warshall Algorithm, computational complexity \(-O(V^3)\)).
TABLE VI. THE RESULT OF TESTING TRANSFORMING THE MCPP INTO THE GTSP

<table>
<thead>
<tr>
<th></th>
<th>Av.</th>
<th>Time Standart Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Av. time, sec</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
|400|180|50|300|200|100|50
|0.005|0.007|0.012|0.017|0.021|0.026|0.031|

B. Results of testing transformation the GTSP into the ATSP

The time of algorithm’s work most depends on number of vertex |V| in transformed problems. The graphic for |V|=500 is presented on Fig.5. The results for all test data are presented in Table 7.

![Fig. 5. The time performance of transforming the GTSP into the ATSP, |V|=500](image)

TABLE VII. THE RESULT OF TESTING TRANSFORMING THE GTSP INTO THE ATSP

<table>
<thead>
<tr>
<th></th>
<th>Av.</th>
<th>Time Standart Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Av. time, sec</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
|400|180|50|300|200|100|50
|0.005|0.007|0.012|0.017|0.021|0.026|0.031|

It can be seen from the table that the transformation takes place quickly.

C. Results of applying the LKH to solving the MCPP

For transforming data set the LKH was applied. We did not make out the principle of the algorithm, but applied it as a black box with standard perimeters, suggested by the author [17].

The results of average computing time and error rate is represented in Table 8. The applying algorithm for solving the MCPP as ATSP throw the transformations opens many opportunities for research. As seen from Table 8, the results of such approach are good. Using confidence intervals, the accuracy of the time estimate was determined, which was at least 87%.

TABLE VIII. THE RESULT OF TESTING LKH TO SOLVING THE TRANSFORMED MCPP

<table>
<thead>
<tr>
<th></th>
<th>Av.</th>
<th>Time Standart Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Av. time, sec</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
|400|180|50|300|200|100|50
|0.005|0.007|0.012|0.017|0.021|0.026|0.031|

In our future work, we are going to fine-tune parameters of LKH methods using genetic search optimization algorithms. Further, it is possible to apply and investigated other existing algorithms for ATSP. Next, we are going to develop (prepare) test data for the mixed multigraph.

VIII. CONCLUSION

The way of solving the MCPP in multigraph were proposed. Previously, the MCPP were solved only in a mixed graph.

The algorithm for transforming the MCPP in multigraph into an equivalent arc routing problem the GTSP, and then in the ATSP was implemented in C++ and tested.

The approach to transforming the MCPP in mixed multigraph into equivalent ARP was evaluated on test data from the open databases. Testing has shown that this approach can be applied to solve the MCPP in multigraph and should be further investigated with various algorithms for solving the ATSP.

REFERENCES


