An Approach to Asymmetric Multiple Traveling Salesman Problem with a Constant Number of Nodes

Bogusz Przybysławski

Institute of Industrial Engineering and Management, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, PL 50370 Wrocław

Abstract. The paper concerns a distribution network, where the fixed number of traveling salesmen start and end their route at the same point. Their purpose is to visit every point exactly once. We wish to find exactly k cities assigned to every salesman. It is assumed that both capacity of vehicles and the size of all packages are irrelevant. We seek a solution, which minimizes a total distance by a fleet. The presented method takes advantage of effective already constructed algorithms for sTSP. The paper recalls both the (n+m-1) transformation from TSP to MTSP and the 2n transformation from asymmetric to symmetric problem.

Keywords: multiple traveling salesman problem, heuristic, operations research.
PACS: 89.40.Bb, 89.20.Ff, 89.65.Gh.

INTRODUCTION

The traveling salesman problem (TSP) is one of the most famous operations research problem in the literature due to application for many real situations. For this reason, there were invented sophisticated, but very effective algorithms. In this paper we present that these algorithms could well be adopted to solve real distribution network problems, where the input data is asymmetric and there is a fixed number of salesmen. Namely, the special case of asymmetric multiple traveling salesman problem (aMTSP) is considered, where we wish to assign the same number of cities to every salesman.

The paper is proceed as follows: the following section contains linear programing formulations of single and multiple traveling salesman problem. Afterwards, we briefly describe proposed approach to solve a problem. An example is presented in the next section. Finally, we provide the conclusion and future remarks.

INTEGER LINEAR PROGRAMMING FORMULATIONS

In this section we provide mathematical formulations of concerned problems and give an overview of methods designed to solve them.

Traveling Salesman Problem

The symmetric TSP (first introduced in [1]) can be formally defined as follows: Let *G(V,A)* be a directed network defined by the set *N* of *n* nodes (cities) and the set *A* of arcs with an associated cost (travel time) c_{ij} from city *i* to city *j*. The problem is said to be symmetric when $c_{ij}=c_{ji}$, otherwise it is asymmetric. Let x_{ij} be a zero-one variable, indicating whether or not the salesman travels from city *i* to city *j*. The integer linear programming formulation of symmetric TSP can be given as follows:.

$$Min \sum_{i<j} c_{ij} x_{ij} \tag{1}$$

Subject to:

$$\sum_{i<k} x_{ik} + \sum_{k>j} x_{kj} = 2 \tag{2}$$

$$(S \subset V, 3 \le |S| \le n-3): \quad \sum_{i,j \in S} x_{ij} \le |S| - 1 \tag{3}$$

$$x_{ij} \in \{0,1\}, k \in V, (i,j) \in A \tag{4}$$

We seek a solution, which minimizes the cost of the tour (1). Constraint (2) states that each city has to be visited exactly once. The purpose of constrain (3) is to eliminate subtours. Note that the last constrain can be replaced by formulation presented in [2].

The sTSP is solvable by popular solvers such as CPLEX, when the size of input data is small. Another way is to use Concorde, which is software developed particularly for this problem and it is freely available at [3]. At the heart of the program there are algorithms for finding cutting planes, fully described in [4]. It is worth to mention that Concorde's TSP solver was

reported to obtain the optimal solutions to the instances having up to 85,900 cities.

One may consider other possibilities of solving the TSP such as heuristics. The software called LKH, which is available at [5], is implementation of improved Lin-Kerninghan Heuristic [6]. Although the algorithm is approximate, it is highly effective for large-scale instances with unknown optima.

Multiple Traveling Salesman Problem

We first recall the exact method, which is based on computing the mixed integer programming model by CPLEX solver. The formulation of the problem, which consider the same number of cities assigned to each salesman, was originally introduced in [2]. Let *G(V,A)* be the undirected graph with a set *V* of *n* nodes (cities). The set *A* denotes arcs *(i,j)* and c_{ij} represents a distance between each pair of cities. The distance does not depend upon which salesman is taken under consideration. Here, there are *m* salesmen – all of them start and end their route in the "base city". In this model one binary variable x_{ij} is introduced, if $x_{ij}=1$ then arc *(i,j)* lays on a tour.

$$Min \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij} \tag{5}$$

Subject to:

$$\sum_{j=2}^{n} x_{1j} = m \tag{6}$$

$$\sum_{j=2}^{n} x_{1j} = m \tag{7}$$

$$(\forall j \in V : j > 1) : \sum_{i=1}^{n} x_{ij} = 1 \tag{8}$$

$$(\forall j \in V : j > 1) : \sum_{i=1}^{n} x_{ij} = 1 \tag{9}$$

$$(\forall i, j \in V : 2 \le i \ne j \le n) : \quad u_i + u_j + k x_{ij} \le k - 1 \tag{10}$$

$$u_1 = 1, x \in \{0,1\} \tag{11}$$

The purpose of this model (5) is to find exactly *m* Hamiltonian cycles with the minimal cost. Every traveler starts and ends at the node 1 (6), (7). Each city is visited exactly once (8), (9). Constrain (10) eliminates subtours and *k* denotes the maximum number of nodes which have to be visited by every salesman.

Of course, solving the problem is **NP**-hard, so this approach is efficient only when the problem size is very small. In this paper we propose the new approach for solving this problem by making use of already implemented algorithms for TSP. There are different approaches to solve similar problems. We refer reader to detailed study in [7], [8] for further information.

A SOLUTION PROCEDURE

In this section we present the approximate method to solve the problem described previously. Firstly, we describe input data transformations, which may easily be adopted to provide data for Concorde's TSP solver. Thus, the relaxed version of a problem is obtained. Finally, we introduce the heuristic, which assign exactly *k* number of nodes to each traveler.

The input data

At the beginning the data (the distances or travelling time between nodes) have to be collected. As far as the distribution network is concerned it may be taken from i.e. standard or internet maps. As a result we get the *C* matrix of distances $c_{ij} \in C:(i,j) \in A$. According to various traffic regulations etc. the data is always asymmetric. The number of salesmen *m* is determined and form a set $M=\{0,1...,m-1\}$. There are no fixed costs of assigning each salesman. Below, we present the similar extension of graph *G(V,A)* to *G'(V',A')* as it is described in [9].

The set *V'* of nodes is equal to $M \cup V=\{-(m-1),-\{m-2\},\ldots 0,1,2,\ldots,n\}$. The distance matrix *C'* is formed from *C* in the following way:

$$i, j \in \{1,...,n\} : c'_{ij} = c_{ij} \tag{12}$$

$$j \in \{1,...,n\}, i \in \{-(m-1),...,0\} : \quad c'_{ij} = c_{0j}; c'_{ji} = c_{j0} \tag{13}$$

Next, the problem have to be converted from asymmetric to symmetric in the way like it is presented in [10]. Let *n* be an updated number of nodes including previous transformation, so the set $V''=\{1,\ldots,2n\}$ of *2n* nodes may be introduced. For further consideration we donate *max* as the maximal value in the matrix of costs ($\forall c_{ij} \in C: c_{ij} \le max$). Now, we construct distance matrix *C''* from *C'* in the succeeding way:

$$i \in \{1,...,n\}, j \in \{1,...,n\} : \quad c''_{ij} = \max \tag{14}$$

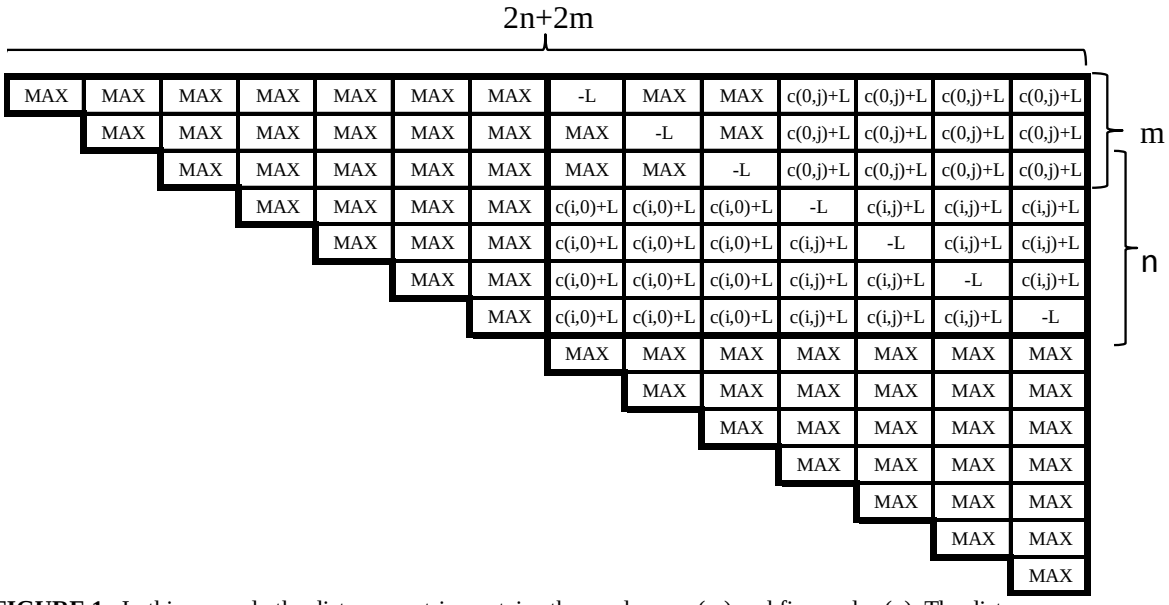
$$i \in \{1,...,n\},\ j \in \{n+1,...,2n\}, i \neq j: \quad c''_{ij} = c'_{i,j-n} + L \tag{15}$$

$$i \in \{1,...,n\},\ j \in \{n+1,...,2n\}, i = j: \quad c''_{ij} = -L \tag{16}$$

$$i \in \{n+1,...,2n\},\ j \in \{n+1,...,2n\}: \quad c''_{ij} = \max \tag{17}$$

An example of these transformations is presented in Fig. 1. Note that L is constant value, which does not influence the final result.

From now on it is possible to solve the asymmetric multiple traveling salesman problem by any software implemented for symmetric traveling salesman problem.



MAX	MAX	MAX	MAX	MAX	MAX	MAX	-L	MAX	MAX	c(0,j)+L	c(0,j)+L	c(0,j)+L	c(0,j)+L
	MAX	MAX	MAX	MAX	MAX	MAX	MAX	-L	MAX	c(0,j)+L	c(0,j)+L	c(0,j)+L	c(0,j)+L
		MAX	MAX	MAX	MAX	MAX	MAX	MAX	-L	c(0,j)+L	c(0,j)+L	c(0,j)+L	c(0,j)+L
			MAX	MAX	MAX	MAX	c(i,0)+L	c(i,0)+L	c(i,0)+L	-L	c(i,j)+L	c(i,j)+L	c(i,j)+L
				MAX	MAX	MAX	c(i,0)+L	c(i,0)+L	c(i,0)+L	c(i,j)+L	-L	c(i,j)+L	c(i,j)+L
					MAX	MAX	c(i,0)+L	c(i,0)+L	c(i,0)+L	c(i,j)+L	c(i,j)+L	-L	c(i,j)+L
						MAX	c(i,0)+L	c(i,0)+L	c(i,0)+L	c(i,j)+L	c(i,j)+L	c(i,j)+L	-L
							MAX	MAX	MAX	MAX	MAX	MAX	MAX
								MAX	MAX	MAX	MAX	MAX	MAX
									MAX	MAX	MAX	MAX	MAX
										MAX	MAX	MAX	MAX
											MAX	MAX	MAX
												MAX	MAX
													MAX

FIGURE 1. In this example the distance matrix contains three salesmen (m) and five nodes (n). The distances are asymmetric.

K-points heuristic

The algorithm has been coded in ANSI C on computer with Ubuntu OS. As it was mentioned to solve relaxed version of MTSP we use Concorde's TSP Solver and to set exactly k points to every traveler we propose the k-points heuristic. We provide additional elements:

- $t(C'')$ – a tour computed for C'' matrix by Concorde, which is an optimal solution for aMTSP, where a number of points assigned to every traveler is unrestricted.
- M – set of m salesmen.
- K_m – set of k points for m-th traveler.
- t'_m – a route contained exactly k nodes assigned for m-th traveler.
- t''- a single tour in t with the largest amount of nodes.

The k-points heuristics can be summarized as follows:

Step 1. Compute *k*.

First of all we get k from equation:

$$k = \frac{n}{m} \tag{18}$$

If it is an odd number it is increased by one.

Step 2. For all *i*∈*M*

Step 3. Compute *t(C''$_i$)*

Here, the tour is split, so as we are able to compare the list of all points assigned to every salesman and choose the one with the largest number of points - $t''(C''_i)$.

Step 4. For all *j*∈*K*

Step 5. t'$_i$←t''(C''$_i$)

Only first k elements of t'' is considered.

Step 6. End for

Step 7. Construct *C''$_{i+1}$←C''$_i$∩t'$_i$*

Generate new distance matrix, which do not contain nodes assigned to t'_i.

Step 8. End for

Step 9. Compute total distance

At the end the final result is calculated:

```
NAME:                 example
TYPE:                 TSP
DIMENSION:              28
EDGE_WEIGHT_TYPE:     EXPLICIT
EDGE_WEIGHT_FORMAT:   FULL_MATRIX
EDGE_WEIGHT_SECTION
  999  999  999  999  999  999  999  999  999  999  999  999  999  999  -10  999  999   29   27   27   28   26   27   24   13   14   14   28
  999  999  999  999  999  999  999  999  999  999  999  999  999  999  999  -10  999   29   27   27   28   26   27   24   13   14   14   28
  999  999  999  999  999  999  999  999  999  999  999  999  999  999  999  999  -10   29   27   27   28   26   27   24   13   14   14   28
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   30   30   30  -10   14   18   19   14   14   17   27   29   29   19
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   27   27   27   14  -10   15   16   11   11   13   24   26   26   16
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   27   27   27   18   14  -10   11   15   15   14   24   26   26   11
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   28   28   28   19   15   11  -10   16   16   14   25   27   27   10
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   27   27   27   14   11   15   16  -10   12   13   24   26   26   16
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   27   27   27   14   11   15   15   11  -10   14   24   26   26   16
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   25   25   25   17   13   14   14   13   13  -10   22   24   24   15
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   13   13   13   26   24   24   25   23   24   21  -10   12   12   25
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   14   14   14   28   26   26   27   25   26   23   12  -10   10   27
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   14   14   14   28   26   26   27   25   26   23   12   10  -10   27
  999  999  999  999  999  999  999  999  999  999  999  999  999  999   28   28   28   20   15   11   11   16   16   15   25   27   27  -10
  -10  999  999   30   27   27   28   27   27   25   13   14   14   28  999  999  999  999  999  999  999  999  999  999  999  999  999  999
  999  -10  999   30   27   27   28   27   27   25   13   14   14   28  999  999  999  999  999  999  999  999  999  999  999  999  999  999
  999  999  -10   30   27   27   28   27   27   25   13   14   14   28  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   29   29   29  -10   14   18   19   14   14   17   26   28   28   20  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   27   27   27   14  -10   14   15   11   11   13   24   26   26   15  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   27   27   27   18   15  -10   11   15   15   14   24   26   26   11  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   28   28   28   19   16   11  -10   16   15   14   25   27   27   11  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   26   26   26   14   11   15   16  -10   11   13   23   25   25   16  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   27   27   27   14   11   15   16   12  -10   13   24   26   26   16  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   24   24   24   17   13   14   14   13   14  -10   21   23   23   15  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   13   13   13   27   24   24   25   24   24   22  -10   12   12   25  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   14   14   14   29   26   26   27   26   26   24   12  -10   10   27  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   14   14   14   29   26   26   27   26   26   24   12   10  -10   27  999  999  999  999  999  999  999  999  999  999  999  999  999  999
   28   28   28   19   16   11   10   16   16   15   25   27   27  -10  999  999  999  999  999  999  999  999  999  999  999  999  999  999
EOF
```

FIGURE 2. An example of an input file in TSPLIB format.

$$\sum_{i=1}^{m} t_i'(C_i'') \qquad (19)$$

Note that, for the last two iterations it is obligatory to check if $k*m>n$ to properly generate a next distance matrix. If the condition is satisfied we propose to set $k_1=n-k*m$ for a penultimate iteration. Moreover, k_1 may well be an odd number, then it is necessarily to increase it and compute $k_2=n-k*m-k_1$. Here, k_2 is always even since all of expressions on the right side are even.

EXAMPLE

First, we construct distance matrix containing 12 nodes and 3 salesman (Fig. 2) in TSPLIB format [11]. We get the following result:

Salesman 1: 1-9-7-14-6
Salesman 2: 2-10-8-4-5
Salesman 3: 3-12-13-11
Total distance: 154.

CONCLUSION

This paper presents an approach to solve the asymmetric multiple traveling salesman problem. It describes in detail a double transformation of data from aTSP into sTSP with *m* salesmen. Here, the *k*-points heuristic is proposed to assign the same number of nodes to every traveler. The proposed algorithm could conceivably be improved and should be computationally tested.

REFERENCES

1. G. Dantzig, R. Fulkerson and S. Johnson, (1954), "Solution of a large-scale traveling-salesman problem", *Journal of the Operations Research Society of America*, pp. 393-410.
2. C.E. Miller, A.W. Tucker and R.A. Zemlin, (1960), "Integer Programming Formulation of Traveling Salesman Problems", *J. ACM* 7, pp. 326-329.
3. <http://www.tsp.gatech.edu/index.html>, 28-04-2013.
4. D. L. Applegate, R. E. Bixby, V. Chvatal and W. J. Cook, (2011). "The traveling salesman problem: a computational study", *Princeton University Press*.
5. <http://www.akira.ruc.dk/~keld/research/LKH/> 28-04-2013.
6. K. Helsgaun, (2000), "An effective implementation of the lin-kernighan traveling salesman heuristic", *European Journal of Operational Research*, 126(1).
7. Bektas Tolga, (2006), "The multiple traveling salesman problem: an overview of formulations and solution procedures", *Omega* 34.3, pp. 209-219.
8. Matai Rajesh, S. Singh and Murari Lal Mittal, (2010), "Traveling Salesman Problem: an Overview of Applications, Formulations, and Solution Approaches." *Traveling Salesman Problem, Theory and Applications, InTech*.
9. M. Bellmore and S. Hong, (1974), "Transformation of multisalesman problem to the standard traveling salesman problem", *Journal of the ACM (JACM)*, 21(3), pp. 500-504.
10. R. Jonker and T. Volgenant (1983), "Transforming asymmetric into symmetric travelling salesman problems", *Operations Research Letters*, nr 2(4), pp. 161–163.
11. G. Reinelt, (1991), "TSPLIB—A traveling salesman problem library", *ORSA Journal on Computing*, 3(4), pp. 376-384.