Transport and Telecommunication, 2016, volume 17, no. 4, 350–361
Transport and Telecommunication Institute, Lomonosova 1, Riga, LV-1019, Latvia
DOI 10.1515/ttj-2016-0031

APPLICATION OF FUZZY SETS FOR THE IMPROVEMENT OF ROUTING OPTIMIZATION HEURISTIC ALGORITHMS

Konstantinos Mattas, George Botzoris, Basil Papadopoulos

Democritus University of Thrace, Department of Civil Engineering
12, Vas. Sofias St., 67100 Xanthi – Greece
Ph.: +30.25410.79747, fax: +30.25410.79698
e-mail: kmattas@civil.duth.gr, gbotzori@civil.duth.gr, papadob@civil.duth.gr

The determination of the optimal circular path has become widely known for its difficulty in producing a solution and for the numerous applications in the scope of organization and management of passenger and freight transport. It is a mathematical combinatorial optimization problem for which several deterministic and heuristic models have been developed in recent years, applicable to route organization issues, passenger and freight transport, storage and distribution of goods, waste collection, supply and control of terminals, as well as human resource management. Scope of the present paper is the development, with the use of fuzzy sets, of a practical, comprehensible and speedy heuristic algorithm for the improvement of the ability of the classical deterministic algorithms to identify optimum, symmetrical or non-symmetrical, circular route. The proposed fuzzy heuristic algorithm is compared to the corresponding deterministic ones, with regard to the deviation of the proposed solution from the best known solution and the complexity of the calculations needed to obtain this solution. It is shown that the use of fuzzy sets reduced up to 35% the deviation of the solution identified by the classical deterministic algorithms from the best known solution.

Keywords: optimal circular path, travelling salesman, heuristic, algorithm, route improvement, fuzzy sets

1. Routing optimization – Definition and categorisation of the problem

Finding the shortest route to connect a specific number of nodes and return to the starting node is called the Travelling Salesman Problem (TSP). The TSP is a benchmark problem of NP complexity, so it can be solved by a non deterministic Touring's machine in polynomial time. It is also NP-hard, so any problem in NP can be reduced to TSP in polynomial time, making it NP-complete as it belongs to the intersection of NP and NP-hard (Dantzig *et al*., 1954; Garey and Johnson, 1979). There are multiple applications of TSP in numerous fields such as transportation and logistics were flexible, fast and easy to use problem solvers are needed (Bohács *et al*., 2013).

The TSP can be divided into subcategories of similar problems, by changing some of the restrictions or by changing the form of the input data. If the nodes are points in R^2 (or R^d generally) and the distance measured by the Euclidean metric, the problem is Euclidean. Moreover if the distances satisfy symmetry and the triangle inequality then the problem is metric. When at least symmetry applies, the problem is called symmetric (Padberg and Rinaldi, 1991). Accordingly when neither symmetry nor the triangle inequality is satisfied, the instance is asymmetric (Bai *et al*., 2012). Another category is the multiple travelling salesmen problem (mTSP) in which the number of vendors is greater than one (Junjie and Dingwei, 2006; Liu *et al*., 2009), therefore the nodes belong to different closed circular paths that can intersect at the origin.

2. Heuristic algorithms for routing optimization – Literature review

The travelling salesman problem is used as a benchmark problem for many algorithms and methods of solving optimization problems. Thus, the literature provides a significant number of different responses. Three main categories are the *exact*, the *approximate* and the *heuristic* algorithms. Exact algorithms are the ones that yield the best solution every time they are applied to a problem. Beyond the brutal search algorithm that seeks to identify and compare all possible solutions, more sophisticated algorithms of reduced time requirements have been developed, such as *branch and bound*, *branch and cut*, *cutting planes,* etc., (Laporte, 1992). However the required volume of calculations is still growing non-polynomially while the number of nodes increases (Papadimitriou, 1977). For large problems, even the most modern exact algorithms become practically unusable because of this characteristic of the TSP (Hansen, 2000).

This led to a completely different strategy to address the problem. Methods to identify a solution to the problem which is not necessarily the optimum, but is close enough to it have been developed. The advantage of these processes is the much shorter time requirements to solve the problem. The performance of such an algorithm is evaluated by comparing the approximate solution to the optimum S_{opt} (if it has been determined). In this regard an approximate solution S_H has quality p (%), where p is the percentage deviation of the solution from the optimum (Johnson *et al*., 2002):

$$p = 100 \cdot \left[\left(S_H - S_{opt} \right) \big/ S_{opt} \right] \tag{1}$$

There are some algorithms whose quality p has an upper bound, fixed or dependent on the size of the nodes. These algorithms are called *approximate*. Such algorithms are the *2-approximation* algorithm (Khuller and Vishkin, 1994), and the algorithm of *Christofides* (Christofides, 1976; Genova and Williamson, 2015; Xu *et al*., 2011). Algorithms that are able to produce a solution close to the optimum, without an upper bound to their quality are called *heuristic*.

The heuristic algorithms are divided into two main categories, the *path construction* algorithms (Reinelt, 1994) and the *route improvement* algorithms (Hwang *et al*., 1999). Path construction algorithms are procedures that make it possible to find a solution close to optimal one. The improvement algorithms have no such ability, however, given a possible solution and making corrections, they offer a reduced cost solution. Moreover, there are algorithms that combine the two categories, so path are constructed and also improved in the process.

The *nearest neighbour algorithm* is one of the simplest heuristics to approach the problem of travelling salesman. The route begins from the starting node and the nearest node is chosen as the next step of the route. It continues up to when all nodes have been added to the path and the final connection to the starting node. The results are not always satisfactory; nevertheless the advantage of this method is that it takes little time to find a solution. The algorithm of the nearest neighbour, with its variants, belongs to the class of *greedy* algorithms (Gutin *et al*., 2002).

The improvement algorithms, depending on the strategy by which the improvements are made, can be divided into *local search* and *global search* (Schonlau *et al*., 1998). The principle of local search algorithms is to make small interventions to a given path, so new routes of reduced cost are created. Such small moves are *k-opt*, *node exchange* and the *node insertion* (Stattenberger *et al*., 2007). If an exchange is applied, two nodes of an existing route are selected and the benefits or costs of exchanging their positions are examined. If there is benefit, then the exchange is applied, thus resulting in to a new, improved solution. In the same way the node insertion method can be applied. A node and an edge are selected. The edge is no longer used and the nodes that it has been connecting are reunited with the selected node as an intermediate, if the move results to an improved route.

The 3-opt method is one of the k-opt methods. In the course of 3-opt, three edges are selected, which are broken and joined together in a different order. It is one of the best small changes that can be made to improve a route, together with the 2-opt. However, 2-opt unlike 3-opt, changes the direction of the salesman for other edges, so that in asymmetric problems all these changes must be examined, rendering the method too cumbersome. Out of all the possible ways of route reunification on 3-opt only one does not change the direction of any edges, making it suitable for asymmetric problems (Lin, 1965). All changes that may occur with the node insertion can also arise from the 3-opt, but not the other way around.

These small moves (k-opt, exchange, node insertion) have been studied in terms of effectiveness and it has emerged that the 2-opt and 3-opt produce higher quality solutions in symmetrical problems, in which they can be used (Stattenberger *et al*., 2007). However, the use of only one of them may lead to a local optimum, a route which cannot be improved using the same alternation, but it is not the optimum solution. As a result, algorithms have been created that combine these moves or establish criteria for acceptance or not of any change, to produce better solutions (Mattas *et al*., 2015).

3. Fuzzy sets for the improvement of route optimization heuristic algorithms

3.1. The concept of fuzzy sets

In the process of walking a TSP path with n number of nodes, the salesman makes n different steps, choosing each time to move from a node i to a node j. In each step an edge $i \rightarrow j$ is added to the path, cancelling essentially all other edges starting from node i or destined to node j. Therefore, the edge $i \rightarrow j$ should be compared to all the edges from which it will be preferred. To accomplish this comparison, the linguistic variable *"cheap edge"* will be described with the use of fuzzy sets.

First step of the algorithm is to organize the data needed, which are the number of nodes n and the cost of any edge connecting them. A square matrix n×n, *Table A*, is filled and each element a_{ij} of the table is equal to the transition cost from node *i* to node *j* (Fig. 1, step 1). This is the only input data of the method.

Then, each individual connection is compared to the connections it excludes, namely with all the connections that have a common starting point as well as the connections with common destination. The comparison with the first and the second is accomplished separately. To quantify the comparison, linguistic variables are created with the names *"cheap edge compared to edges with starting point i"* and *"cheap edge compared to edges with endpoint j"*. The process is repeated for each node and 2×n linguistic variables are created. Two square tables are created, *Table B* and *Table C*, containing the membership value of each edge in each of the fuzzy sets (Fig. 1, step 2).

After this computation, any edge in the network has a membership value to determine how cheap it is compared to all the edges with the same starting point and a second one to determine how cheap it is compared to all with the same ending point. The next step is to produce the fuzzy set *"cheap edge"*. In this set every edge has a membership value determining how cheap it is compared to every edge that is not viable if that edge is included to the solution. The membership value is computed out of the two previous membership values for every edge.

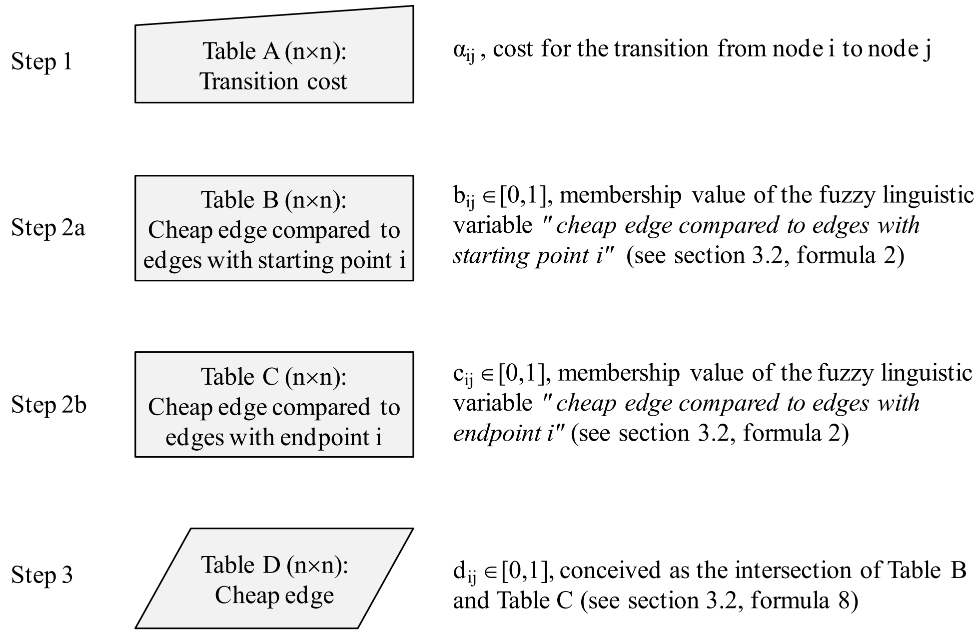

Figure 1. The consecutive steps of the path construction algorithm with the use of fuzzy sets

Computing the membership value d_{ij} from b_{ij} and c_{ij}, is one of the main concerns of the paper. The simplest way is the membership value d_{ij} to be the average of the membership values b_{ij} and c_{ij}. However, the way in which the nodes are sited provides important information that can be quantified and affect the assessment of the fuzzy set *"cheap edge"*. For example, a cheap edge to a remote node and another one to a node located close to several others should not be evaluated the same. Thus, statistical factors for every node and adjusted edges can be used in the computation of d_{ij}, as a weighted average of b_{ij} and c_{ij}. Moreover, since for an edge to be chosen, it needs to be cheap in comparison to edges with starting point *i* and cheap compared to edges with endpoint *j*, *"cheap edge"* can be conceived as the intersection of the two sets, thus available t-norms can be utilized (Fig. 1, step 3).

Furthermore after the user of the algorithm creates the fuzzy set "cheap edge", it can be useful for another fuzzy set to be created that can be called "preferable edge". This is because many problems of transportation planning and organization are affected by multiple criteria, and there are many ways to combine those multiple criteria with fuzzy sets (Chernov *et al*., 2012). For example a trip organizer can plan the best possible route for a tourist bus, taking into account the cost of the tour and how picturesque the tour will be. In this case the linguistic variable "picturesque edge" can be described as a fuzzy set, and from it and the fuzzy set "cheap edge" the fuzzy set "preferable edge" can be created. The rest of the solution will continue using "preferable edge" instead of "cheap edge".

3.2. Formation of the fuzzy sets

The fuzzy sets *"cheap edge compared to edges with starting point i"* and *"cheap edge compared to edges with endpoint j"* are triangular fuzzy numbers, defined from the minimum and maximum cost, as shown in Figure 2. They are called *Set B* and *Set C*, have domain *R* and take values in the closed interval [0, 1]. The triangular form is chosen for two reasons: i) only the cheapest edge has membership value 1 and only the most expensive edge membership value 0, and ii) the computations required to evaluate each set are easy and rather fast even for very large instances.

Therefore, for the set *"cheap edge compared to edges with starting point i"*, *k* is the minimum cost of all edges starting from the node and *l* is the maximum cost (Fig. 2).

The evaluation of *Table B*, which contains the membership values per connection, is as follows:

$$b_{ij} = \frac{l - a_{ij}}{l - k} \tag{2}$$

Similarly, the format for the set *"cheap edge compared to edges with endpoint j"* is the same and the calculation of *Table C* which contains the membership values for each connection is as follows:

$$c_{ij} = \frac{l - a_{ij}}{l - k} \tag{3}$$

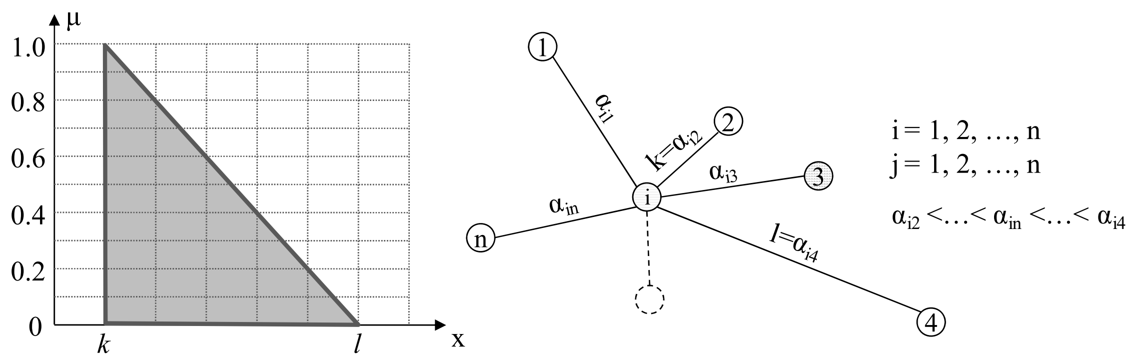

Figure 2. The format of the fuzzy Set A and fuzzy Set B

Statistical data will be used as weights for the weighted average, to generate the fuzzy set *"cheap edge"* and the membership values d_{ij} for every edge. Useful coefficients are:

- The average distance to and from the node (in the calculation of average, variance and standard deviation, the data of *Table A* that refer to an internal path of a node, i.e. a_{11}, a_{22}, ..., a_{nn}, are not taken into account):

$$\overline{x}_a = \frac{\sum_{i=1}^{n-1} x_i}{n-1} \tag{4}$$

- The variance and the standard deviation of distances to and from the node *i*:

$$s_a^2 = \frac{\sum_{i=1}^{n-1} \left(\overline{x}_a - x_i\right)^2}{n-2} \tag{5}$$

- The mean and the variance (and hence the standard deviation) of the values of *Table B* and *Table C* for rows and columns respectively which can be proved to be:

$$\overline{x}_b = \frac{l - \overline{x}_a}{l - k}, \ \overline{x}_c = \frac{l - \overline{x}_a}{l - k} \tag{6}$$

$$\sigma_b^2 = \frac{\sigma_a^2}{(l-k)^2},\ \sigma_c^2 = \frac{\sigma_a^2}{(l-k)^2} \tag{7}$$

All edges of a given instance belong to the set *"cheap edge"* with a membership value; therefore the domain of its characteristic function is the set of all the connections, as opposed to other fuzzy sets of the algorithm defined in *R*. Also, it is recognized that while two connections may have the same distance, it is possible to have different membership value to set *"cheap edge"*, as they have been compared with different connections.

Since it has not been possible to define the most effective way to calculate *Table D* a priori, a variety of ways were implied. Formula (8) of the weighted average was used in fourteen different forms (formulas 8.1 to 8.14) so statistical evidence can be evaluated (Guh *et al*., 2008). Also, besides the average, *Table D* can be conceived as the intersection of the previous *Tables B* and *C*, and it has been calculated as a product (formula 8.15), minimum (formula 8.16) and Łukasiewicz t-norm (formula 8.17), (Botzoris and Papadopoulos, 2015; Mattas *et al*., 2015):

$$d_{ij} = \left(\frac{w_i \cdot b_{ij}^{\rho} + w_i \cdot c_{ij}^{\rho}}{w_i + w_j} \right)^{\frac{1}{\rho}} \tag{8}$$

where:

1) $w_i = 1 \qquad w_j = 1 \qquad \rho = 1$ (8.1)

2) $w_i = \overline{x}_i \qquad w_j = \overline{x}_j \qquad \rho = 1$ (8.2)

3) $w_i = \frac{1}{\sigma_i^2} \qquad w_j = \frac{1}{\sigma_j^2} \qquad \rho = 1$ (8.3)

4) $w_i = \sigma_i^2 \qquad w_j = \sigma_j^2 \qquad \rho = 1$ (8.4)

5) $w_i = \frac{1}{\sigma_i} \qquad w_j = \frac{1}{\sigma_j} \qquad \rho = 1$ (8.5)

6) $w_i = \sigma_i \qquad w_j = \sigma_j \qquad \rho = 1$ (8.6)

7) $w_i = \overline{x}_{\beta i} \qquad w_j = \overline{x}_{\gamma j} \qquad \rho = 1$ (8.7)

8) $w_i = \frac{1}{\sigma_{\beta i}^2} \qquad w_j = \frac{1}{\sigma_{\gamma j}^2} \qquad \rho = 1$ (8.8)

9) $w_i = \sigma_{\beta i}^2 \qquad w_j = \sigma_{\gamma j}^2 \qquad \rho = 1$ (8.9)

10) $w_i = \frac{1}{\sigma_{\beta i}} \qquad w_j = \frac{1}{\sigma_{\gamma j}} \qquad \rho = 1$ (8.10)

11) $w_i = \overline{x}_{\beta i} \qquad w_j = \overline{x}_{\gamma j} \qquad \rho = \sigma_{\beta,\gamma i}^2$ (8.11)

12) $w_i = \frac{1}{\sigma_i^2} \qquad w_j = \frac{1}{\sigma_j^2} \qquad \rho = \beta_{ij} + \gamma_{ij} - \overline{x}_{\beta i} - \overline{x}_{\gamma i}$ (8.12)

13) $w_i = \frac{1}{\sigma_{\beta i}^2} \qquad w_j = \frac{1}{\sigma_{\gamma j}^2} \qquad \rho = \beta_{ij} + \gamma_{ij} - \overline{x}_{\beta i} - \overline{x}_{\gamma i}$ (8.13)

14) $w_i = 1 \qquad w_j = 1 \qquad \rho = \beta_{ij} + \gamma_{ij} - \overline{x}_{\beta i} - \overline{x}_{\gamma i}$ (8.14)

15) $d_{ij} = \beta_{ij} \cdot \gamma_{ij}$ (8.15)

16) $d_{ij} = \min(\beta_{ij}, \gamma_{ij})$ (8.16)

17) $d_{ij} = \max(0,\ 1 - \beta_{ij} - \gamma_{ij})$ (8.17)

4. Application of fuzzy sets for the improvement of route optimization heuristic algorithms

4.1. Fuzzy sets for path construction

The nearest neighbour algorithm was used to determine the effect of the implementation of fuzzy sets on heuristic path construction. For the application of nearest neighbour's fuzzy method, beginning from the starting point, the edge of the highest membership value is chosen instead of the nearest. Similarly, the process continues until each node in the path is included, when the connection of the last node to the starting point is made.

4.2. Fuzzy sets for route improvement

The solutions initially determined by the nearest neighbour method, are improved by using the methods of node exchange, node insertion, and 3-opt as control. Subsequently, these methods were applied using fuzzy sets.

The strategy used for improving the route was as follows: The first connection after the starting point is selected and all possible changes are evaluated. If no beneficial change emerges, the following edge is checked. Whenever a change occurs, the process starts again. If for one edge more than one beneficial change is found, the change of maximum profit is selected. The process is completed when all connections are checked and there is no profitable change, therefore the route is a local optimum for the method used.

A fuzzy set strategy of route improvement was also implemented to the routes obtained by the fuzzy nearest neighbour method. In this procedure, all connections included in the considered route are compared in terms of the membership value to the set *"cheap edge"*. The edge of the lowest membership value is selected and checked for all the possible changes. Whenever there is no profitable change, the next edge to be inspected for any change is the one of the lowest membership value that hasn't been inspected yet, and each time a change is made the process starts from the beginning. The process ends when all edges have been inspected. The use of fuzzy sets in route improvement was applied in all three methods of improving described previously (node exchange, node insertion, and 3-opt).

5. Results

For the application of the proposed methods to test their efficiency, Python programming language has been used and the necessary functions have been developed. The problems on which the fuzzy sets were implemented were eighteen (18) benchmark asymmetric problems of TSPLIB (Travelling Salesman Problem LIBrary), (Reinelt, 1991). The asymmetric class of problems was chosen so the problems better resemble real life transportation situations were neither the triangular inequality nor symmetry have to apply. Also the asymmetric class is a superset of symmetric metric and Euclidean classes, so other problems of these classes can be solved although the solver should make use of their attributes to save time. Any of these problems could be solved to a heuristic solution without the use of fuzzy sets. In addition fuzzy sets could be implemented on the path construction algorithm, and the resulting route can be improved using improvement algorithms with or without the use of fuzzy sets.

Firstly paths have been constructed and evaluated for their cost, using the nearest neighbour algorithm (Table 2). Then the nearest neighbour with fuzzy sets was implemented to every one of the eighteen asymmetric problems of TSPLIB, to compare the two cases; without and with the use of fuzzy sets. Of all the seventeen ways to create the fuzzy set "cheap edge", the best value obtained is shown in the Table 1 (min value of the formulas 8.1 to 8.17), along with the average value (average value of the formulas 8.1 to 8.17). The best known solution of the problems is provided by the TSPLIB.

In the same manner Tables 2, 3 and 4 were designed, to show the cost of the resulting paths when route improving algorithms were implemented. The tables are the results of the node exchange, node insertion and 3-opt algorithm respectively. In every table the best known solution is given and next to it the solution of the improvement algorithm implemented on the route created by the nearest neighbour heuristic, without any use of fuzzy sets. For the remaining columns the path was always created by the nearest neighbour with fuzzy sets and for the last three of them, fuzzy sets were used also during the route improvement.

Table 1. Implementation of the nearest neighbour algorithm for path construction, with and without the use of fuzzy sets

TSPLIB problem code	Best known solution (provided by the TSPLIB)	Nearest neighbour without fuzzy sets	Nearest neighbour with fuzzy sets (min value of the formulas 8.1 to 8.17)	Nearest neighbour with fuzzy sets (average value of the formulas 8.1 to 8.17)
br17	39	92	56	89
ft53	6,905	9,514	8,665	9,369
ft70	38,673	43,186	42,366	43,027
ftv33	1,286	1,683	1,450	1,635
ftv35	1,473	1,791	1,651	1,678
ftv38	1,530	1,956	1,731	1,757
ftv44	1,613	2,147	1,795	1,843
ftv47	1,776	2,575	2,180	2,233
ftv55	1,608	2,023	2,069	2,085
ftv64	1,839	2,416	2,381	2,491
ftv70	1,950	2,543	2,391	2,414
ftv170	2,755	3,991	3,745	3,824
kro124	36,230	47,506	42,841	43,181
rbg323	1,326	1,760	1,718	1,765
rbg358	1,163	1,648	1,594	1,619
rbg403	2,465	3,304	3,288	3,332
rbg443	2,720	3,830	3,796	3,835
ry48p	14,422	16,757	16,999	17,751
Deviation from the best known solution (%)		**37.38%**	**24.61%**	**32.75%**

Table 2. Route improvement using the node exchange method, with and without the use of fuzzy sets

TSPLIB problem code	Best known solution (provided by the TSPLIB)	Nearest neighbour without fuzzy sets → Improved using node exchange without fuzzy sets	Nearest neighbour with fuzzy sets → Improved using node exchange without fuzzy sets (min value of the formulas 8.1 to 8.17)	Nearest neighbour with fuzzy sets → Improved using node exchange without fuzzy sets (average value of the formulas 8.1 to 8.17)	Nearest neighbour with fuzzy sets → Improved using node exchange with fuzzy sets (min value of the formulas 8.1 to 8.17)	Nearest neighbour with fuzzy sets → Improved using node exchange with fuzzy sets (average value of the formulas 8.1 to 8.17)
br17	39	42	40	42	40	42
ft53	6,905	8,919	8,406	8,847	8,351	8,785
ft70	38,673	41,803	41,400	41,915	41,400	41,853
ftv33	1,286	1,590	1,450	1,531	1,450	1,531
ftv35	1,473	1,788	1,590	1,609	1,590	1,609
ftv38	1,530	1,820	1,661	1,676	1,661	1,671
ftv44	1,613	2,035	1,784	1,805	1,784	1,805
ftv47	1,776	2,270	1,991	1,999	1,974	1,990
ftv55	1,608	1,911	1,972	2,037	1,972	2,037
ftv64	1,839	2,346	2,324	2,347	2,327	2,343
ftv70	1,950	2,359	2,377	2,402	2,375	2,408
ftv170	2,755	3,853	3,645	3,713	3,647	3,717
kro124	36,230	45,506	40,342	40,433	40,342	40,423
rbg323	1,326	1,521	1,453	1,478	1,430	1,473
rbg358	1,163	1,330	1,324	1,346	1,302	1,337
rbg403	2,465	2,574	2,585	2,611	2,535	2,590
rbg443	2,720	2,936	2,858	2,901	2,837	2,886
ry48p	14,422	16,615	16,502	17,137	16,230	16,972
Deviation from the best known solution (%)		**19.58%**	**13.65%**	**16.02%**	**13.10%**	**15.72%**

Table 3. Route improvement using the node insertion method, with and without the use of fuzzy sets

TSPLIB problem code	**Best known solution (provided by the TSPLIB)**	**Nearest neighbour without fuzzy sets → Improved using node insertion without fuzzy sets**	**Nearest neighbour with fuzzy sets → Improved using node insertion without fuzzy sets (min value of the formulas 8.1 to 8.17)**	**Nearest neighbour with fuzzy sets → Improved using node insertion without fuzzy sets (average value of the formulas 8.1 to 8.17)**	**Nearest neighbour with fuzzy sets → Improved using node insertion with fuzzy sets (min value of the formulas 8.1 to 8.17)**	**Nearest neighbour with fuzzy sets → Improved using node insertion with fuzzy sets (average value of the formulas 8.1 to 8.17)**
br17	39	42	40	42	40	42
ft53	6,905	8,186	7,665	8,006	7,784	8,507
ft70	38,673	41,253	40,785	41,382	40,938	41,320
ftv33	1,286	1,452	1,391	1,401	1,435	1,441
ftv35	1,473	1,652	1,512	1,541	1,512	1,541
ftv38	1,530	1,649	1,608	1,620	1,608	1,620
ftv44	1,613	2,003	1,778	1,788	1,788	1,807
ftv47	1,776	2,016	1,963	1,965	1,912	2,000
ftv55	1,608	1,918	1,941	1,965	1,960	1,988
ftv64	1,839	2,257	2,212	2,227	2,267	2,277
ftv70	1,950	2,268	2,231	2,277	2,245	2,293
ftv170	2,755	3,722	3,298	3,486	3,507	3,549
kro124	36,230	41,019	38,294	39,712	40,016	40,044
rbg323	1,326	1,463	1,437	1,451	1,429	1,460
rbg358	1,163	1,262	1,245	1,283	1,260	1,291
rbg403	2,465	2,519	2,499	2,519	2,507	2,525
rbg443	2,720	2,788	2,758	2,789	2,759	2,778
ry48p	14,422	15,725	14,874	15,267	14,816	15,418
Deviation from the best known solution (%)		**13.48%**	**8.77%**	**10.97%**	**9.94%**	**12.29%**

Table 4. Route improvement using the 3-opt method, with and without the use of fuzzy sets

TSPLIB problem code	**Best known solution (provided by the TSPLIB)**	**Nearest neighbour without fuzzy sets → Improved using 3-opt without fuzzy sets**	**Nearest neighbour with fuzzy sets → Improved using 3-opt without fuzzy sets (min value of the formulas 8.1 to 8.17)**	**Nearest neighbour with fuzzy sets → Improved using 3-opt without fuzzy sets (average value of the formulas 8.1 to 8.17)**	**Nearest neighbour with fuzzy sets → Improved using 3-opt with fuzzy sets (min value of the formulas 8.1 to 8.17)**	**Nearest neighbour with fuzzy sets → Improved using 3-opt with fuzzy sets (average value of the formulas 8.1 to 8.17)**
br17	39	**39**	**39**	**39**	**39**	**39**
ft53	6,905	7,882	7,081	7,374	7,187	7,423
ft70	38,673	40,511	39,657	39,906	39,630	40,264
ftv33	1,286	1,340	1,357	1,359	1,340	1,351
ftv35	1,473	1,490	1,479	1,508	1,479	1,509
ftv38	1,530	1,604	1,536	1,580	1,536	1,582
ftv44	1,613	1,741	1,623	1,642	**1,613**	1,633
ftv47	1,776	1,895	1,871	1,876	1,912	1,914
ftv55	1,608	1,650	1,723	1,785	1,676	1,736
ftv64	1,839	1,957	1,989	1,992	2,061	2,061
ftv70	1,950	2,086	2,070	2,084	2,070	2,084
ftv170	2,755	3,179	3,055	3,165	3,017	3,075
kro124	36,230	37,916	38,625	38,623	38,341	38,807
rbg323	1,326	1,331	1,328	1,331	1,328	1,332
rbg358	1,163	1,171	1,168	1,172	1,165	1,172
rbg403	2,465	2,468	2,467	2,471	2,467	2,471
rbg443	2,720	2,725	**2,720**	2,722	**2,720**	2,722
ry48p	14,422	15,214	14,650	14,967	14,831	15,197
Deviation from the best known solution (%)		**4.81%**	**3.25%**	**4.52%**	**3.35%**	**4.67%**

6. Conclusions and further research

The results of the tests run without any use of fuzzy sets were a confirmation of the literature review; nearest neighbour algorithm provides quite poor results but very quickly and easily. The implementation of route improving on the solutions provided, increased the running time but improved the quality of the solutions. Among the three different route improvement algorithms that was used, the best solutions were provided, as expected, by the 3-opt algorithm, secondly by the node insertion method and the worst solutions were result of the node exchange method.

Comparing the results provided without the use of fuzzy sets to those provided with the use of fuzzy sets (Fig. 3), either solely on path construction or both on path construction and route improvement, it is clear that the use of fuzzy sets is beneficiary on almost every case. Although for some specific cases the proposed use of fuzzy sets did not provide better solutions, on most cases the solutions were much better, and even the best solutions for some problems were provided when this was not possible without them. Specifically the deviation of the solution produced by the nearest neighbour with fuzzy sets was 35% better, from 37.4% to 24.6%. When the resulting route was improved with one of the local search algorithms the benefit was 33% (from 19.6% to 13.1%), 35% (from 13.5% to 8.8%) and 32% (from 4.8% to 3.2%) for the exchange, node insertion and 3-opt algorithms respectively (Fig. 3). Also chosen any of the proposed ways to create the fuzzy set "cheap edge" the average deviation from the best solution provided, was always less than the respective deviation without fuzzy sets.

The next comparison that can be made is between improvement with fuzzy sets and improvement without fuzzy sets on paths that were created with the use of fuzzy sets. Those results were not definitive. While fuzzy sets were beneficiary when implied on the exchange move, the results were worse in case of the node insertion move. In the case of the 3-opt algorithm the results were really close. The strategy in which those fuzzy sets were used can change in many ways to produce different results. For example instead of changing the worst edge in the node insertion move, the worst connected edge can be the first candidate to change. In any case the expected result could not be much different because all algorithms would finish when a local optimum had been reached. The advantage of the use of fuzzy sets on these cases is the number of changes needed to obtain that local optimum. It has been determined that when using fuzzy sets less changes are needed, as shown in Table 5, because the proposed method can easily find the worst connections and aim to change those first, rather than search blindly for improvements. The disadvantage of the proposed algorithm is the use of fuzzy sets has not been yet tested on more sophisticated algorithms that can overcome local optimum solutions, and this will be the next step of our research.

Table 5. Average number of changes required for reaching the local optimum

TSPLIB problem code	**Nearest neighbour with fuzzy sets → Improved using exchange without fuzzy sets**	**Nearest neighbour with fuzzy sets → Improved using exchange with fuzzy sets**	**Nearest neighbour with fuzzy sets → Improved using node insertion without fuzzy sets**	**Nearest neighbour with fuzzy sets → Improved using node insertion with fuzzy sets**	**Nearest neighbour with fuzzy sets → Improved using 3-opt without fuzzy sets**	**Nearest neighbour with fuzzy sets → Improved using 3-opt with fuzzy sets**
br17	5.67	1.88	1.89	1.88	2.89	2.29
ft53	8.50	8.12	23.06	17.29	20.17	19.59
ft70	10.78	10.59	16.67	13.12	25.17	22.76
ftv33	6.72	6.88	11.17	7.12	10.89	5.76
ftv35	1.61	1.41	5.67	5.12	6.56	5.29
ftv38	3.06	2.71	6.94	6.00	7.67	5.82
ftv44	1.61	1.71	3.50	3.41	8.44	5.24
ftv47	6.50	5.18	10.56	11.00	10.17	5.71
ftv55	2.72	2.29	7.44	4.65	15.06	10.53
ftv64	4.94	3.71	10.94	6.82	16.22	9.59
ftv70	0.83	0.47	7.06	6.35	12.94	7.82
ftv170	6.50	5.59	16.50	11.53	26.50	21.18
kro124	18.39	18.12	23.22	25.00	27.94	18.88
rbg323	82.06	72.53	83.44	80.82	91.39	90.71
rbg358	79.11	69.88	93.50	80.59	88.94	81.82
rbg403	158.44	172.35	171.94	192.29	125.78	118.88
rbg443	198.22	208.29	217.00	236.35	152.94	148.47
ry48p	7.72	9.24	24.28	19.06	17.61	12.94

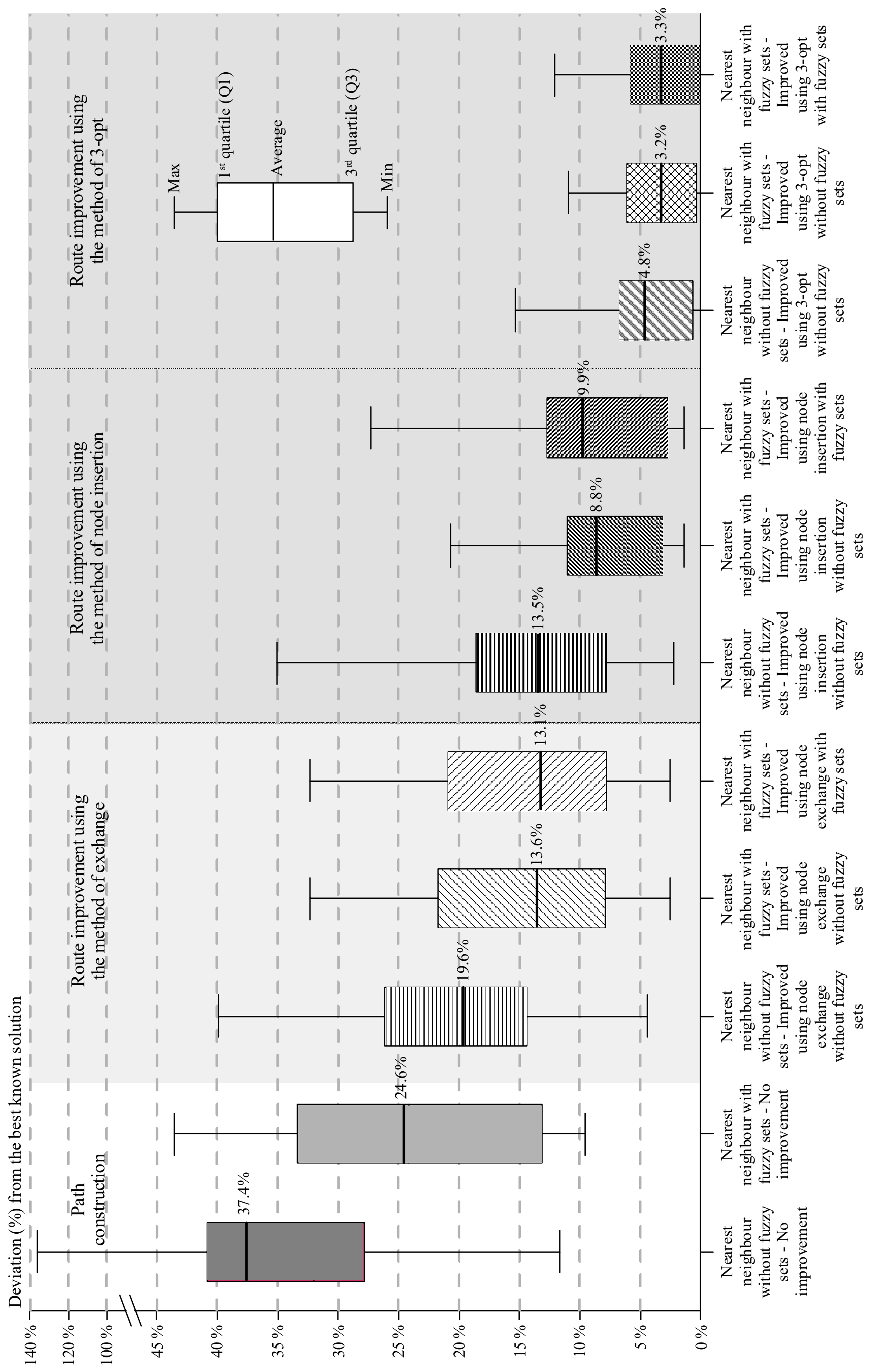

Figure 3. Deviation (%) from the best known solution for the various path construction and route improvement algorithms

References

1. Applegate, D.L., Bixby, R.E., Chvátal, V. and Cook, W.J. (2007) *The Traveling Salesman Problem: A Computational Study*. Princeton (NJ): Princeton University Press.
2. Bai, J., Yang, G.K., Chenb, Y.W., Hu, L.S. and Pan, C.C. (2012) A model induced max-min ant colony optimization for asymmetric travelling salesman problem. *Applied Soft Computing*, 13(3), 1365–1375. DOI: doi:10.1016/j.asoc.2012.04.008.
3. Bohács, G., Frikker, I. and Kovács, G. (2013) Intermodal logistics processes supported by electronic freight and warehouse exchanges. *Transport and Telecommunication*, 14(3), 206–213. DOI:10.2478/ttj-2013-0017.
4. Botzoris, G. and Papadopoulos, B. (2015) *Fuzzy Sets: Applications for the Design and Operation of Civil Engineering Projects* (in Greek). Thessaloniki: Sofia Editions.
5. Chernov, V., Dorokhov, O. and Malyaretz, L. (2012) Construction of estimates in the choice of alternative solutions by using the fuzzy utilities. *Transport and Telecommunication*, 13(1), 11–17. DOI: 10.2478/v10244-012-0002-z.
6. Christofides, N. (1976) *Worst-case Analysis of a New Heuristic for the Travelling Salesman Problem*. Pittsburgh (PA): Carnegie Mellon University – Graduate School of Industrial Administration. (Technical Report 388, ADA025602).
7. Dantzig, G., Fulkerson, R. and Johnson, S. (1954) Solution of a large-scale travelling-salesman problem. *Operations Research*, 2(4), 393–410. DOI: 10.1287/opre.2.4.393.
8. Garey, M.R. and Johnson, D.S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York (NY): W.H. Freeman & Co.
9. Genova, K. and Williamson, D.P. (2015) An experimental evaluation of the best-of-many Christofides algorithm for the traveling salesman problem. In: *Algorithms - ESA 2015*, (Eds.) N. Bansal, and I. Finocchi, Berlin, Heidelberg: Springer-Verlag, pp. 570-581.
10. Guh, Y.Y., Po, R.W. and Stanley Lee, E. (2008) The fuzzy weighted average within a generalized means function. *Computers and Mathematics with Applications,* 55(12), 2699–2706. DOI:10.1016/j.camwa.2007.09.009.
11. Gutin, G., Yeo, A. and Zverovich, A. (2002) Travelling salesman should not be greedy: Domination analysis of greedy-type heuristics for the TSP. *Discrete Applied Mathematics*, 117(1-3), 81–86. DOI:10.1016/S0166-218X(01)00195-0.
12. Hansen, M. (2000) Use of substitute scalarizing functions to guide a local search based heuristics: The case of MOTSP. *Journal of Heuristics*, 6(3), 419–431. DOI: 10.1023/A:1009690717521.
13. Hwang, C.P., Alidaee, B. and Johnson, J.D. (1999) A tour construction heuristic for the travelling salesman problem. *The Journal of the Operational Research Society*, 50(8), 797–809. DOI: 10.2307/3010339.
14. Johnson, D.S., Gutin, G., McGeoch, L.A., Yeo, A., Zhang, W. and Zverovitch, A. (2002) Experimental analysis of heuristics for the ATSP. In: *The Travelling Salesman Problem and its Variations*, (Eds.) G. Gutin and A.P. Punnen, Dordrecht: Kluwer Academic Publishers, pp. 485-488.
15. Junjie, P. and Dingwei, W. (2006) An ant colony optimization algorithm for multiple travelling salesman problem. In: *Proceedings of the 1st International Conference on Innovative Computing, Information and Control*, Beijing, August 2006, pp. 210-213.
16. Khuller, S. and Vishkin, U. (1994) Biconnectivity approximations and graph carvings. *Journal of the Association for Computing Machinery*, 41(2), 214–235. DOI:10.1145/174652.174654.
17. Laporte, G. (1992) The travelling salesman problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(2), 231–247. DOI:10.1016/0377-2217(92)90138-Y.
18. Lin, S. (1965) Computer solutions of the travelling salesman problem. *The Bell System Technical Journal*, 44(10), 2245–2269. DOI:10.1002/j.1538-7305.1965.tb04146.x.
19. Liu, W., Li, S., Zhao, F. and Zheng, A. (2009) An ant colony optimization algorithm for the multiple traveling salesmen problem. In: *Proceeding of the 4th IEEE Conference on Industrial Electronics and Applications*, Xi'an, May 2009, pp. 1533-1537.
20. Mattas, K., Botzoris, G. and Papadopoulos, B. (2015) Improvement of route optimization algorithmic methods with application of fuzzy sets. In: *Proceeding of the 7th International Congress on Transportation Research,* Athens, November 2015.
21. Padberg, M. and Rinaldi, G. (1991) A branch-and-cut algorithm for the resolution of large-scale symmetric travelling salesman problems. *Society for Industrial and Applied Mathematics,* 33(1), 60–100. DOI:10.1137/1033004.

22. Papadimitriou, C.H. (1977) The Euclidean travelling salesman problem is np-complete. *Theoretical Computer Science*, 4(3), 237–244. DOI:10.1016/0304-3975(77)90012-3.
23. Reinelt, G. (1991) TSPLIB – A travelling salesman problem library. *ORSA Journal on Computing*, 3(4), 376–384. DOI: 10.1287/ijoc.3.4.376.
24. Reinelt, G. (1994) *The Traveling Salesman: Computational Solutions for TSP Applications*. Berlin, Heidelberg: Springer-Verlag.
25. Schonlau, M., Welch, W.J. and Jones, D.R. (1998) Global versus local search in constrained optimization of computer models. *Institute of Mathematical Statistics: Lecture Notes-Monograph Series,* 34, 11–25. doi:10.1214/lnms/1215456182.
26. Stattenberger, G., Dankesreiter, M., Baumgartner, F. and Schneider, J. (2007) On the neighbourhood structure of the travelling salesman problem generated by local search moves. *Journal of Statistical Physics*, 129(4), 623–648. DOI: 10.1007/s10955-007-9382-1.
27. Xu, Z., Xu, L., and Rodrigues, B. (2011) An analysis of the extended Christofides heuristic for the k-depot TSP. *Operations Research Letters*, 39(3), 218–223. DOI:10.1016/j.orl.2011.03.002.