Research Journal of Recent Sciences ISSN 2277-2502
Vol. **10(3),** 18-20, July **(2021)** *Res. J. Recent Sci.*

Haar dhouib-matrix-TSP1 method to solve triangular fuzzy travelling salesman problem

Souhail Dhouib
Department of Industrial Management, Higher Institute of Industrial Management, Sfax University, Tunisia
souhail.dhouib@gmail.com

Available online at: www.isca.in, www.isca.me
Received 21th March 2021, revised 7th May 2021, accepted 30th June 2021

Abstract

This paper aims to propose a constructive method named the Dhouib-Matrix-TSP1 in order to optimize the triangular fuzzy sets for the travelling salesman problem. First, we change the triangular fuzzy numbers into Haar sets using the Haar ranking function. Second, we use the Dhouib-Matrix-TSP1 method to solve the travelling salesman problem. The detailed new method is discussed and illustrated by a numerical example.

Keywords: Travelling salesman problem, linear integer programming, fuzzy set, haar ranking function, approximation method.

Introduction

The Travelling Salesman Problem (TSP) is associated with daily activities in uncertain parameters (distance, cost, time). This uncertainty (the parameters are not always exactly known and stable) is modalized mathematically as the fuzzy set theory which is widely used in TSP. There are many methods to solve the travelling salesman problem as well as taxonomy and classification[1]. The Hungarian method is used to solve the intuitionistic fuzzy approach to the travelling salesman problem[2]. The based population algorithm (Genetic Algorithm) is designed to optimize the multi-objective travelling salesman problem with fuzziness[3].

The colony algorithm is developed to find the optimalroute for mobile robot[4]. The fuzzy stage dependent TSP can be formulated as integer linear programming model[5]. The large scale of travelling salesman problems is solved using a hybrid evolutionary algorithm[6]. An inspired method based on physarumpolycephalum model is designed to solve the fuzzy travelling salesman problem[7]. The Hungarian method is developed to solve the triangular fuzzy travelling salesman problem[8].

Recently, we design a new method namely Dhouib-Matrix-TSP1 in order to solve the TSP[9]. Whereas, in this study we adapt the Dhouib-Matrix-TSP1 to solve the triangular fuzzy TSP through the Haar ranking function. This paper is presented as follows. Prefaces of fuzzy basic concepts, operations of fuzzy set theory and techniques for Haar method for ranking fuzzy numbers have been reviewed in section 2. In section 3, Dhouib-Matrix-TSP1 approximation method has been proposed for triangular fuzzy transport salesman problem. A numerical example is illustrated and comparison with other methods is depicted in section 4.

Preliminaries

A fuzzy number is defined by mathematically affecting, to each element in the universe of discourse, a continuous value between zero and one, describing its grade of membership[10]. If the membership function $\mu_{\tilde{X}}(x)$ of a fuzzy number $\tilde{X}$ are normal, continuous in pieces and convex; then it will be a fuzzy set.

The fuzzy number $\tilde{X} = (f_1, f_2, f_3; z)$, Figure-1, with membership function in the form described in equation 1 is entitled triangular fuzzy number.

$$\mu_{\tilde{X}}(x)\begin{cases} 0, & x < f_1 \\ \dfrac{x-f_1}{f_2-f_1} & f_1 \le x \le f_2 \\ 1 & x = f_2 \\ \dfrac{f_3-x}{f_3-f_2} & f_2 \le x \le f_3 \\ 0 & x \ge f_3 \end{cases} \qquad (1)$$

Figure-1: The triangular fuzzy number.

The fuzzy addition is given by:
$$(f_1, f_2, f_3) + (e_1, e_2, e_3) = (f_1 + e_1, f_2 + e_2, f_3 + e_3)$$

The fuzzy substruction is given by:
$$(f_1, f_2, f_3) - (e_1, e_2, e_3) = (f_1 - e_3, f_2 - e_2, f_3 - e_1)$$

Proposed Algorithm for solving TSP by means of Dhouib-Matrix-TSP1 Methods: In this paper, we first use the Haar ranking function in order to convert all the triangular fuzzy distances into Haar tuples. Then, we consider the Dhouib-Matrix-TSP1 method to solve the travelling salesman problem. Well, for a given triangular fuzzy number, it can be written as four tuples by f_1,f_2,f_2,f_3. Then, the scaling and wavelet coefficients using unnormalized Haar wavelet can be calculated using equation (2) where this tuple is considered as $R(\tilde{X}) = (\alpha,\beta,\gamma,\delta)$:

$$\begin{aligned}
\alpha &= (z*(f_1+f_2+f_2+f_3))/4 \\
\beta &= (z*((f_1+f_2)-(f_2+f_3)))/4 \\
\gamma &= (z*(f_1-f_2))/2 \\
\delta &= (z*(f_2-f_3))/4
\end{aligned} \qquad (2)$$

Using: i. $\tilde{X} \prec \tilde{Y}$ only when the initial component of $R(\tilde{X})$ is less than its of $R(\tilde{Y})$. ii. $\tilde{X} \succ \tilde{Y}$ only when the initial component of $R(\tilde{X})$ is greater than its of $R(\tilde{Y})$. iii. $\tilde{X} \approx \tilde{Y}$ when all components of both $R(\tilde{X})$ and $R(\tilde{Y})$ are equal $\alpha_1 = \alpha_2, \beta_1 = \beta_2, \gamma_1 = \gamma_2, \delta_1 = \delta_2$.

Python language has been used to develop the Dhouib-Matrix-TSP1 and to compute the detailed coefficients using discrete wavelet transform through Haar wavelet. The Dhouib-Matrix-TSP1 is composed of four simple phases repeated in only *n* iterations where *n* is the number of cities:

Phase-1: Compute the minimum value for each row (you can change the metric sum etc.), then select the smallest. This smallest element will define the row to be activated and for which we select the minimal element[9]. Thus, this is the first cities to be added to *Cycle_List* (insert them by order as noted by matrix element position) and discarded their columns (a stepwise example will be given in the next section).

Phase 2: Select the minimum of each row for the first and last cities in the *Cycle_List*. Then select the smallest between them.

Phase 3: Insert the corresponding city for the smallest element in the *Cycle_List* (at left, if the corresponding city is linked to the first element in the *Cycle_List*, or at right, if the corresponding city is linked to the last element of the *Cycle_List*) and finally discard its column.

Phase 4: Transform the *Cycle_List*to be a cycle by translating the first element to the last position until the starting city will be at the first position. Then, add the starting city to the end of *Cycle_List*(we finch the cycle by the starting city).

Numerical application

To validate the proposed method Dhouib-Matrix-TSP1, we will compare its results with those founded by Hungarian one[9]. Here is an example of four cities (Figure-2), our method Dhouib-Matrix-TSP1 will find the optimal solution only in *n* (*n*=4 cities) iterations.

$$\begin{bmatrix} 0 & (1,2,3) & (8,9,10) & (9,10,11) \\ (0,1,2) & 0 & (5,6,7) & (3,4,5) \\ (14,15,16) & (6,7,8) & 0 & (7,8,9) \\ (5,6,7) & (2,3,4) & (11,12,13) & 0 \end{bmatrix}$$

Figure-2: Fuzzy distance Matrix.

First, we need to convert the triangular fuzzy distance to Haar tuples (Figure-3) using equation (2).

$$\begin{bmatrix} 0 & (2,-0.5,-0.5,-0.5) & (9,-0.5,-0.5,-0.5) & (10,-0.5,-0.5,-0.5) \\ (1,-0.5,-0.5,-0.5) & 0 & (6,-0.5,-0.5,-0.5) & (4,-0.5,-0.5,-0.5) \\ (15,-0.5,-0.5,-0.5) & (7,-0.5,-0.5,-0.5) & 0 & (8,-0.5,-0.5,-0.5) \\ (6,-0.5,-0.5,-0.5) & (3,-0.5,-0.5,-0.5) & (12,-0.5,-0.5,-0.5) & 0 \end{bmatrix}$$

Figure-3: Haar tuples Matrix.

Iteration number 1: Select the minimal element of each row (blue elements in Figure-4). Then, select the smallest (1, -0.5, -0.5, -0.5) which is in the row number 2 (in the position d_{21}).

$$\begin{bmatrix} 0 & \color{blue}{(2,-0.5,-0.5,-0.5)} & (9,-0.5,-0.5,-0.5) & (10,-0.5,-0.5,-0.5) \\ \color{blue}{(1,-0.5,-0.5,-0.5)} & 0 & (6,-0.5,-0.5,-0.5) & (4,-0.5,-0.5,-0.5) \\ (15,-0.5,-0.5,-0.5) & \color{blue}{(7,-0.5,-0.5,-0.5)} & 0 & (8,-0.5,-0.5,-0.5) \\ (6,-0.5,-0.5,-0.5) & \color{blue}{(3,-0.5,-0.5,-0.5)} & (12,-0.5,-0.5,-0.5) & 0 \end{bmatrix}$$

Figure-4: Select the minimum of each row.

Next, insert the city 2 and city 1 in the *Cycle_List* {2-1} and discard their columns (red columns in Figure-5). Note that, the insertion of the cities in the *Cycle_List* should be in their corresponding order, as their index in the matrix (insert 2-1 and not 1-2).

$$\begin{bmatrix} \color{red}{0} & \color{red}{(2,-0.5,-0.5,-0.5)} & (9,-0.5,-0.5,-0.5) & (10,-0.5,-0.5,-0.5) \\ \color{red}{(1,-0.5,-0.5,-0.5)} & \color{red}{0} & (6,-0.5,-0.5,-0.5) & (4,-0.5,-0.5,-0.5) \\ \color{red}{(15,-0.5,-0.5,-0.5)} & \color{red}{(7,-0.5,-0.5,-0.5)} & 0 & (8,-0.5,-0.5,-0.5) \\ \color{red}{(6,-0.5,-0.5,-0.5)} & \color{red}{(3,-0.5,-0.5,-0.5)} & (12,-0.5,-0.5,-0.5) & 0 \end{bmatrix}$$

Figure-5: Discard columns 2 and 1.

Iteration number 2: Find the minimal element for row 2 and row 1, which are respectively (4, -0.5, -0.5, -0.5) for the position d_{24} and (9, -0.5, -0.5, -0.5) for the position d_{13}. The smallest element is in position d_{24}, then insert city 4 at the first position of *Cycle_List* {4-2-1}: at the first position (linked to city 2) and not at the last position. Finally, discard the column of city 3 (Figure-6).

$$\begin{bmatrix} 0 & (2,-0.5,-0.5,-0.5) & (9,-0.5,-0.5,-0.5) & (10,-0.5,-0.5,-0.5) \\ (1,-0.5,-0.5,-0.5) & 0 & (6,-0.5,-0.5,-0.5) & (4,-0.5,-0.5,-0.5) \\ (15,-0.5,-0.5,-0.5) & (7,-0.5,-0.5,-0.5) & 0 & (8,-0.5,-0.5,-0.5) \\ (6,-0.5,-0.5,-0.5) & (3,-0.5,-0.5,-0.5) & (12,-0.5,-0.5,-0.5) & 0 \end{bmatrix}$$

Figure-6: Discard column 4.

Iteration number 3: Find the minimal element for row 4 and row 1 and select the smallest which is in the position d_{13}. Then, insert the city 3 in the *Cycle_List* {4-2-1-3} and discard column 3 (Figure-7).

$$\begin{bmatrix} 0 & (2,-0.5,-0.5,-0.5) & (9,-0.5,-0.5,-0.5) & (10,-0.5,-0.5,-0.5) \\ (1,-0.5,-0.5,-0.5) & 0 & (6,-0.5,-0.5,-0.5) & (4,-0.5,-0.5,-0.5) \\ (15,-0.5,-0.5,-0.5) & (7,-0.5,-0.5,-0.5) & 0 & (8,-0.5,-0.5,-0.5) \\ (6,-0.5,-0.5,-0.5) & (3,-0.5,-0.5,-0.5) & (12,-0.5,-0.5,-0.5) & 0 \end{bmatrix}$$

Figure-7: Discard column 3.

Iteration number 4: To generate the feasible solution (in this example, it is the optimal one), we need to generate a cycle from the *Cycle_List* {4-2-1-3}: start from city 1, visit all cities then return to city 1. For that, we translate the position of the first element (4) to the last position {2-1-3-4}.Similarly, we change the position of city 2 to the last position {1-3-4-2}. Finally, we add the starting city at the end of the list and now, we obtain the optimal cycle {1-3-4-2-1}.

Thus, our method Dhouib-Matrix-TSP1 finds easily the optimal solution in just 4 iterations: the Haar optimal cost is (21, -20, -20, -20) and the matching optimal fuzzy cost is (17, 21, 25). Although, Hungarian method finds the optimal solution but in more complicated iterations[8].

Conclusion

In this study, we present the Dhouib-Matrix-TSP1 method and we prove its performance to easily optimize the triangular fuzzy travelling salesman problem by the means of the Haar ranking function. The proposed method is effective and very easy to apply compared with the other existing methods, our method finds the optimal solution in just n iterations, where n is the number of cities.

References

1. Cheikhrouhoua O. and Khoufi I., (2021). A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy. *Computer Science Review*, 40, 100369.
2. Traneva V. and Tranev S. (2019). An Intuitionistic Fuzzy Approach to the Travelling Salesman Problem, International Conference on Large-Scale Scientific Computing. 11958, 530-539.
3. Changdar C., Maiti M.K. and Maiti M., (2013). A constrained solid TSP in fuzzy environment: two heuristic approaches. *Ranian Journal of Fuzzy Systems*, 10(1), 1-28.
4. Yen C. T. and Cheng M. F. (2018). A study of fuzzy control with ant colony algorithm used in mobile robot for shortest path planning and obstacle avoidance. *Microsystem Technologies*, 24(1), 125-135.
5. Kuchta D. (2016). Fuzzy Stage Dependent Travelling Salesman Problem with Networks as Nodes, Information Systems Architecture and Technology. Proceedings of 36th International Conference on Information Systems Architecture and Technology, Part I, 89-100.
6. Feng H. M. and Liao K. L., (2014). Hybrid evolutionary fuzzy learning scheme in the applications of traveling salesman problems. *Information Sciences*, 270, 204-225.
7. Pezhhan E. and Mansoori E., (2014). A Biologically Inspired Solution for Fuzzy Travelling Salesman Problem, International Symposium on Artificial Intelligence and Signal Processing. *Artificial Intelligence and Signal Processing*, 427, 277-287.
8. Dhanasekar S., Dash K.S. and Hariharan S. (2018). Hungarian Algorithm using Haar Tuples to Solve Fuzzy Travelling Salesman Problem. *International Journal of Engineering & Technology*, 7, 380-382.
9. Dhouib S., (2021). A New Column-Row Method for Travelling Salesman Problem: The Dhouib-Matrix-TSP1. *International Journal of Recent Engineering Science*, 8(1), 6-10.
10. Zadeh L. A., (1965). Fuzzy Sets, Information and Control, 8(3), 338-353.