Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 03, March-2021

Stochastic Column-Row Method for Travelling Salesman Problem: the Dhouib-Matrix-TSP2

Stochastic Column-Row Method: Dhouib-Matrix-TSP2

Souhail Dhouib
Department of Industrial Management,
Higher Institute of Industrial Management
Sfax University, Tunisia

***Abstract*—In this paper, a stochastic method, named Dhouib-Matrix-TSP2, is designed and developed to solve the travelling salesman problem. This method based on our last column-row method (Dhouib-Matrix-TSP1) is a stochastic process used as a tool to enrich the diversification phase.**

The result test shows that the diversification by the means of the stochastic process generates the shortest total transport distance.

Keywords—Combinatory Optimization; Metaheuristic; Column-Row Method; Stochastic

I. INTRODUCTION

Recently in (Dhouib, 2021), we developed a new constructive method named Dhouib-Matrix-TSP1 in order to quickly find the optimal or near optimal solution for the Travelling Salesman Problem (TSP). This method (Dhouib-Matrix-TSP1) is a deterministic algorithm. Whereas, in this paper, we present the Dhouib-Matrix-TSP2 which is a stochastic version of Dhouib-Matrix-TSP1. This stochastic technique is used as a way to diversify the research space. Hence, this process will allow to generate different realizable solutions.

The outlines of the paper are as follow. In section 2, the travelling salesman problem is described. In section 3, the Dhouib-Matrix-TSP1 is presented. In section 4, a new stochastic column-row algorithm (Dhouib-Matrix-TSP2) to find optimal or near optimal solution for the TSP is depicted. In section 5, the application of our proposed method on a 10x10 distance matrix is provided. Finally, concluding remarks are given in section 6.

II. TRAVELLING SALESMAN PROBLEM

In the travelling salesman problem, the aim is to find the shortest roundtrip. This latter is essentially a Hamiltonian tour, where a set of cities are visited only once except the starting city which will be also the ending one.

The travelling salesman problem is NP-hard. However, it is easily formulated as:

Optimize $$\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} \quad (1)$$

Subject to

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1,\dots,n$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1,\dots,n \quad . \quad (2)$$

$$x_{ij} = 0 \text{ or } 1, \quad i = 1,\dots,n \ , \quad j = 1,\dots,n$$

In the literature, there are several research works interested in solving the TSP. (Mehmet & Kalayci. 2021) applies the variable neighborhood search algorithm to optimize the cost-balanced TSP. (Saji & Barkatou, 2021) designs a bat metaheuristic to solve the TSP. (Krzysztof & Urszula, 2020) solves the asymmetric travelling salesman problem using the pheromone-based harmony search metaheuristic. (Akanksha & Sharma, 2019) uses the particle swarm optimization algorithm to solve the TSP. (Todosijevic et al., 2017) develops a variable neighborhood search metaheuristic to optimize the TSP.

III. DHOUIB-MATRIX-TSP1

In our previous research work (Dhouib, 2021), we have presented the deterministic Dhouib-Matrix-TSP1 method which is composed of four easy phases (see Fig 1.).

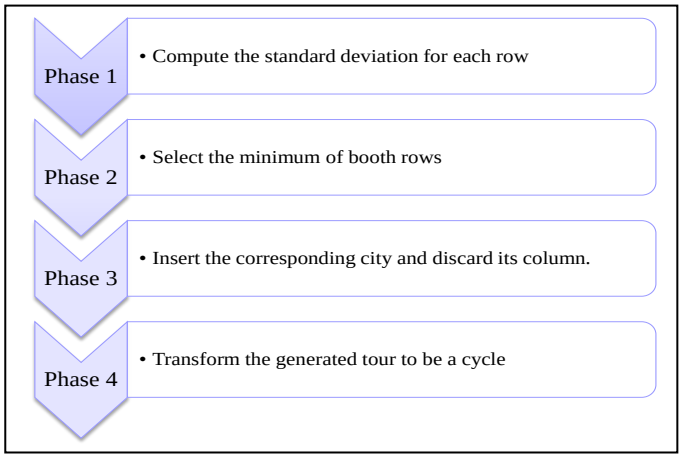

Fig. 1. Four phases for Dhouib-Matrix-TSP1

The four phases of Dhouib-Matrix-TSP1 are described as follow. At first, calculate the standard deviation for each row then find the minimal element existing in the position d_{xy} in the row of the smallest deviation. For more details see (Dhouib, 2021).

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 03, March-2021

Next, insert *x* and *y* cities in the list *Cycle_List* and discard the columns of *x* and *y*.

At second, compute the minimal element for row *x* and row *y* and select the smallest between them named *z*.

At third, add *z* in the *Cycle_List* and discard its column. Then, return to the last phase if there are still non discarded columns in the matrix, unless, go to the next phase.

Finally, transform the generated route in *Cycle_List* to be a roundtrip: translate the cities in the beginning of the roundtrip at the end until the starting city will be at the first position. In addition, insert the starting city at the last position in the roundtrip.

IV. DHOUIB-MATRIX-TSP2 DESIGN

This section will describe the proposed method: the Dhouib-Matrix-TSP2. Actually, it is a stochastic version of the Dhouib-Matrix-TSP1. Whereas, the selection of the minimal element is replaced by the random selection from *k* minimal elements: this will enhance the diversification of the research space.

The pseudo-code of the proposed approach, Dhouib-Matrix-TSP2, is presented as follow (see Fig 2).

Algorithm of Dhouib-Matrix-TSP2

Input: Distance matrix
Output: Optimal or near optimal path

1. Initialize i=1 and S^*=∞
2. Repeat
3. Set *Cycle_List* = {}
4. Compute the standard deviation for each row
5. Select randomly one element (row) from the k smallest standard deviations. Then, for this selected row, choose randomly one element from the k minimal elements: let say that it is at the position d_{xy}
6. *Cycle_List* = {x,y}
7. Discard column x and column y
8. For i=1 to number of cities
9. Select one element from the k minimal elements d_{xa} in the row x
10. Select one element from the k minimal elements d_{yb} in the row y
11. If ($d_{xa} < d_{yb}$) then
12. Insert a before x in the list *Cycle_List*
13. $x = a$
14. Else
15. Insert b after y in the list *Cycle_List*
16. $y = b$
17. End
18. Compute S' from *Cycle_List*
19. If ($S^* > S'$) then
20. $S^* = S'$
21. $i = i + 1$
22. Until (i > maximum number of iterations)
23. Return S^*

Fig. 2. Dhouib-Matrix-TSP2

V. EXPERIMENTS AND RESULTS ANALYSIS

The proposed methods, namely Dhouib-Matrix-TSP1 and Dhouib-Matrix-TSP2, are developed using Python language. A numerical example from (Mandziuk, 1996), 10x10 matrix (see Fig 3.), is used to prove the good convergence and the efficiency of Dhouib-Matrix-TSP2.

0	0.629	0.408	0.564	0.117	0.620	0.327	0.814	0.799	0.370
0.629	0	0.228	0.212	0.712	0.738	0.461	0.304	0.626	0.260
0.407	0.228	0	0.265	0.500	0.672	0.327	0.480	0.668	0.064
0.564	0.212	0.264	0	0.612	0.530	0.304	0.250	0.427	0.242
0.117	0.712	0.500	0.617	0	0.569	0.340	0.865	0.789	0.454
0.620	0.738	0.672	0.530	0.569	0	0.362	0.663	0.323	0.610
0.326	0.461	0.326	0.304	0.340	0.362	0	0.539	0.474	0.262
0.814	0.304	0.480	0.250	0.865	0.663	0.539	0	0.430	0.476
0.798	0.626	0.668	0.428	0.789	0.323	0.474	0.430	0	0.622
0.370	0.260	0.064	0.242	0.454	0.610	0.262	0.476	0.622	0

Fig. 3. Distance Matrix

Well, to quickly and effectively search the optimal path for the ten cities of the travelling salesman problem, we first apply the Dhouib-Matrix-TSP1 which needs only 10 iterations.

Step 1: Compute the standard deviation for each row. Then, select the minimal value (0.1399) which is in row 7.

0	0.629	0.408	0.564	0.117	0.620	0.327	0.814	0.799	0.370	0.2574
0.629	0	0.228	0.212	0.712	0.738	0.461	0.304	0.626	0.260	0.2388
0.407	0.228	0	0.265	0.500	0.672	0.327	0.480	0.668	0.064	0.2175
0.564	0.212	0.264	0	0.612	0.530	0.304	0.250	0.427	0.242	0.1809
0.117	0.712	0.500	0.617	0	0.569	0.340	0.865	0.789	0.454	0.2654
0.620	0.738	0.672	0.530	0.569	0	0.362	0.663	0.323	0.610	0.2108
0.326	0.461	0.326	0.304	0.340	0.362	0	0.539	0.474	0.262	0.1399
0.814	0.304	0.480	0.250	0.865	0.663	0.539	0	0.430	0.476	0.2476
0.798	0.626	0.668	0.428	0.789	0.323	0.474	0.430	0	0.622	0.2283
0.370	0.260	0.064	0.242	0.454	0.610	0.262	0.476	0.622	0	0.1997

Fig. 4. Compute the standard deviation for each row

Step 2: Select the minimal element in row 7 which is 0.262 in position $d_{7,10}$. Thus, insert cities 7 and 10 in the list and discard their columns (See Fig 5.).

0	0.629	0.408	0.564	0.117	0.620	0.327	0.814	0.799	0.370
0.629	0	0.228	0.212	0.712	0.738	0.461	0.304	0.626	0.260
0.407	0.228	0	0.265	0.500	0.672	0.327	0.480	0.668	0.064
0.564	0.212	0.264	0	0.612	0.530	0.304	0.250	0.427	0.242
0.117	0.712	0.500	0.617	0	0.569	0.340	0.865	0.789	0.454
0.620	0.738	0.672	0.530	0.569	0	0.362	0.663	0.323	0.610
0.326	0.461	0.326	0.304	0.340	0.362	0	0.539	0.474	0.262
0.814	0.304	0.480	0.250	0.865	0.663	0.539	0	0.430	0.476
0.798	0.626	0.668	0.428	0.789	0.323	0.474	0.430	0	0.622
0.370	0.260	0.064	0.242	0.454	0.610	0.262	0.476	0.622	0

Fig. 5. Discard columns 7 and 10

Step 3: Select the minimal element in row 7 and row 10 and select the smallest one which is 0.064 in position $d_{10,3}$. Thus, insert city 3 after city 10 and discard its column (See Fig 6.).

0	0.629	0.408	0.564	0.117	0.620	0.327	0.814	0.799	0.370
0.629	0	0.228	0.212	0.712	0.738	0.461	0.304	0.626	0.260
0.407	0.228	0	0.265	0.500	0.672	0.327	0.480	0.668	0.064
0.564	0.212	0.264	0	0.612	0.530	0.304	0.250	0.427	0.242
0.117	0.712	0.500	0.617	0	0.569	0.340	0.865	0.789	0.454
0.620	0.738	0.672	0.530	0.569	0	0.362	0.663	0.323	0.610
0.326	0.461	0.326	0.304	0.340	0.362	0	0.539	0.474	0.262
0.814	0.304	0.480	0.250	0.865	0.663	0.539	0	0.430	0.476
0.798	0.626	0.668	0.428	0.789	0.323	0.474	0.430	0	0.622
0.370	0.260	0.064	0.242	0.454	0.610	0.262	0.476	0.622	0

Fig. 6. Discard column 3

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 03, March-2021

Step 4: Compute the minimal element in row 7 and row 3 and select the smallest one which is 0.228 in position $d_{3,2}$. Thus, insert citiy 2 after city 3 and discard its column (See Fig 7.).

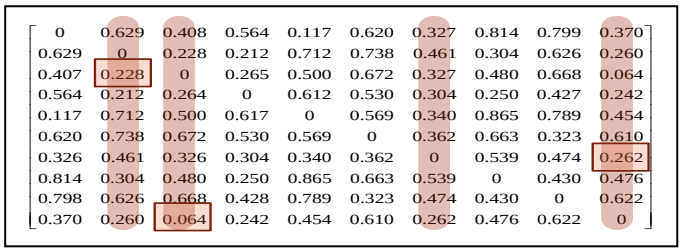

Fig. 7. Discard column 2

Step 5: Select the minimal element in row 7 and row 2 and select the smallest one which is 0.212 in position $d_{2,4}$. Thus, insert city 4 after city 2 and discard its column (See Fig 8.).

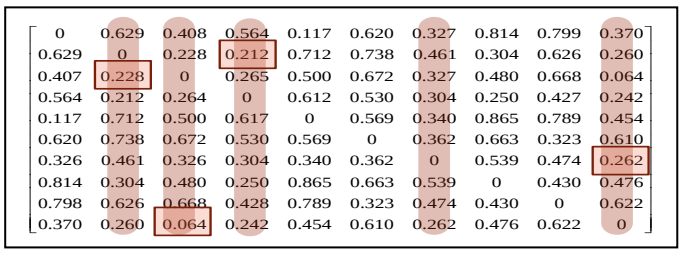

Fig. 8. Discard column 4

Step 6: Select the minimal element in row 7 and row 4 and select the smallest one which is 0.250 in position $d_{4,8}$. Thus, insert city 8 after city 4 and discard its column (See Fig 9.).

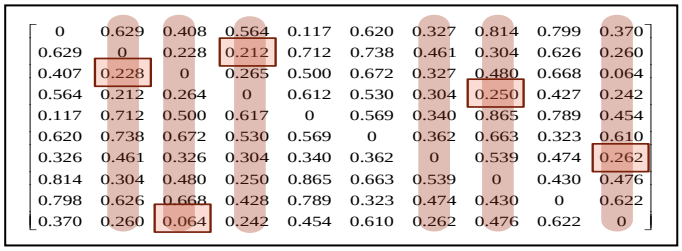

Fig. 9. Discard column 8

Step 7: Select the minimal element in row 7 and row 8 and select the smallest one which is 0.326 in position $d_{1,7}$. Thus, insert city 1 before city 7 and discard its column (See Fig 10.).

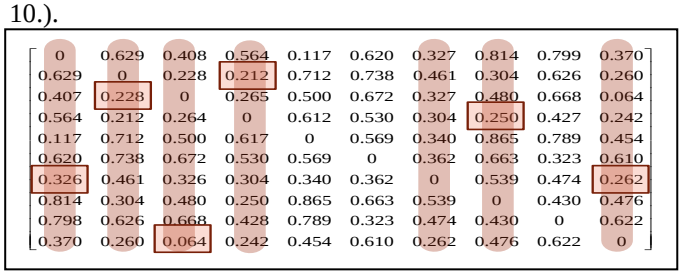

Fig. 10. Discard column 1

Step 8: Select the minimal element in row 1 and row 8 and select the smallest one ($d_{1,5}$). Thus, insert city 5 before city 1 and discard its column (See Fig 11.).

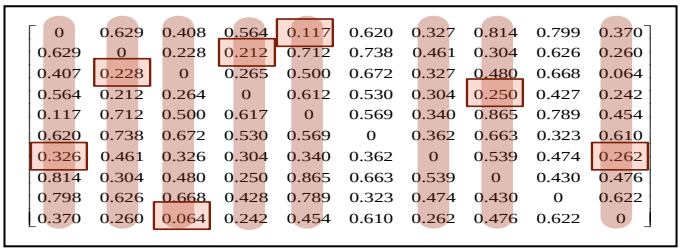

Fig. 11. Discard column 5

Step 9: Select the minimal element in rows 5 and 8. Thus, insert city 9 after city 8 and discard its column (See Fig 12.).

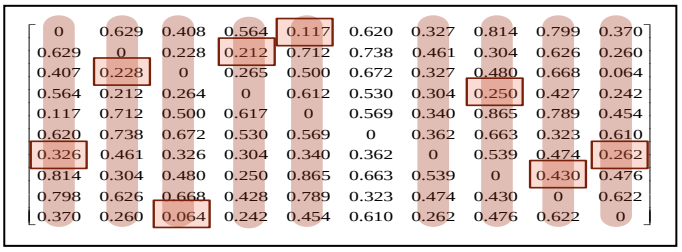

Fig. 12. Discard column 9

Step 10: Select the minimal element in rows 5 and 9. Thus, insert city 6 after city 9 and discard its column (See Fig 13.).

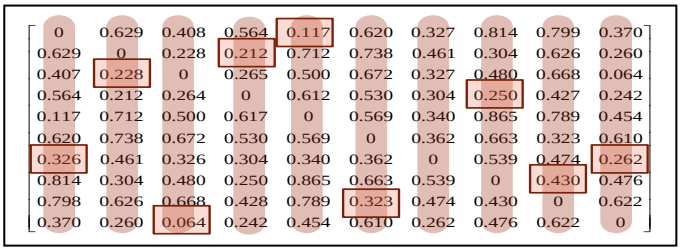

Fig. 13. Discard column 6

Finally, we need to transform the generated tour {5-1-7-10-3-2-4-8-9-6} to a roundtrip by translating the cities in the first position to the end until the starting city will be at the first position and add the starting city at the last position {1-7-10-3-2-4-8-9-6-5-1}.

So, using the Dhouib-Matrix-TSP1, we found 2.78211 (deviation of 3.17%) in only 10 iterations using the standard deviation metric (see Fig 14.).

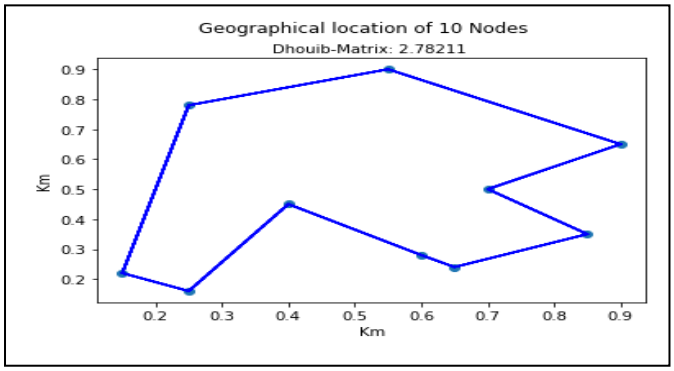

Fig. 14. The solution found by Dhouib-Matrix-TSP1

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 10 Issue 03, March-2021

When we insert the Dhouib-Matrix-TSP2 in an iterative structure, it can easily find the optimal solution (2.69646). In Fig 15., we can see the step-by-step generation of the optimal solution using Dhouib-Matrix-TSP2.

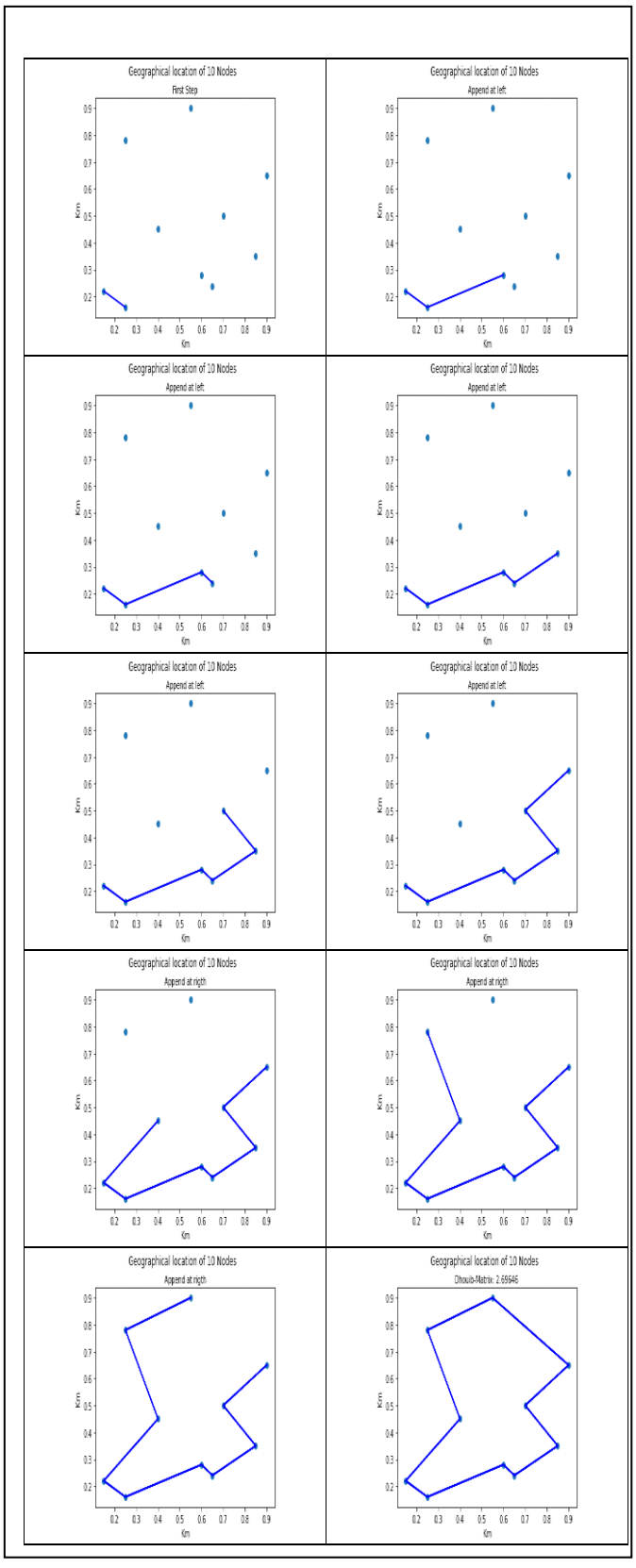

Fig. 15. The optimal solution found by Dhouib-Matrix-TP2

VI. CONCLUSION

This paper aims to insert a stochastic process in a deterministic optimization method (Dhouib-Matrix-TSP1), in order to generate the new method: Dhouib-Matrix-TSP2. The distinguishing feature of this new technique resides in the random selection of the *n* smallest elements which forced the fetch of new solutions, not simply sought the smallest value as in Dhouib-Matrix-TSP1.

This new stochastic method is developed using Python language and the final test result proves its effectiveness and good convergence. In our future research works, larger scale TSP will be explored through our new method Dhouib-Matrix-TSP2 to search the optimal path.

REFERENCES

[1] Dhouib S., "A New Column-Row Method for Traveling Salesman Problem: The Dhouib-Matrix-TSP1," International Journal of Recent Engineering Science, vol. 8, Issue 1, pp.6-10, 2021.

[2] Mandziuk J., "Solving the Travelling Salesman Problem with a Hopfield - type neural network," Demonstratio Mathematica, vol. 29, issue 1,·1996.

[3] Glover F., "Multi-wave algorithms for metaheuristic optimization," Journal of Heuristics, vol. 22, pp. 331-358, 2016.

[4] Saji Y. and Barkatou M., "A discrete bat algorithm based on Lévy flights for Euclidean traveling salesman problem," Expert Systems with Applications, vol.172(15), pp.114639, 2021.

[5] Todosijevic R., Mjirda A., Mladenovic M., Hanafi S., and Gendron, B., "A general variable neighborhood search variants for the travelling salesman problem with draft limits," Optimization Letters, vol.11(6), pp.1047-1056, 2017.

[6] Krzysztof S. and Urszula B., "The Pheromone-Based Harmony Search Algorithm for the Asymmetric Traveling Salesman Problem," Applied. Sciences, vol.10(18), pp. 6422, 2020.

[7] Akanksha S. & Sharma R. S., "Design of Semi-chaotic Integration-Based Particle Swarm Optimization Algorithm and Also Solving Travelling Salesman Problem Using It," Soft Computing for Problem Solving, pp. 695-709, 2019.

[8] Mehmet A. A. and Kalayci C. B., "A Variable Neighborhood Search Algorithm for Cost-Balanced Travelling Salesman Problem," Metaheuristics for Combinatorial Optimization, pp. 23-36, 2021.