Hybrid Local Search Algorithm for Optimization Route of Travelling Salesman Problem

Muhammad Khahfi Zuhanda[1], Noriszura Ismail[2], Rezzy Eko Caraka[3], Rahmad Syah[4], Prana Ugiana Gio[5]
Informatics Engineering Study Program, Faculty of Engineering, Universitas Medan Area, Medan, Indonesia[1]
Department of Mathematical Sciences- Faculty of Science and Technology, Universiti Kebangsaan Malaysia, Selangor, Malaysia[2]
Research Organization for Electronics and Informatics-National Research and Innovation Agency (BRIN), Bandung, Indonesia[3]
Informatics Engineering Study Program-Faculty of Engineering, Universitas Medan Area, Medan, Indonesia[4]
Department of Mathematics-Universitas Sumatera Utara, Medan, Indonesia[5]

***Abstract*—This study explores the Traveling Salesman Problem (TSP) in Medan City, North Sumatra, Indonesia, analyzing 100 geographical locations for the shortest route determination. Four heuristic algorithms—Nearest Neighbor (NN), Repetitive Nearest Neighbor (RNN), Hybrid NN, and Hybrid RNN—are investigated using RStudio software and benchmarked against various problem instances and TSPLIB data. The results reveal that algorithm performance is contingent on problem size and complexity, with hybrid methods showing promise in producing superior solutions. Statistical analysis confirms the significance of the differences between non-hybrid and hybrid methods, emphasizing the potential for hybridization to enhance solution quality. This research advances our understanding of heuristic algorithm performance in TSP problem-solving and underscores the transformative potential of hybridization strategies in optimization.**

Keywords*—Travelling Salesman Problem; heuristic algorithms; hybridization techniques algorithm performance; route optimization

I. INTRODUCTION

Irish and British mathematicians first introduced the Traveling Salesman Problem (TSP). They were William Rowan Hamilton and Thomas Penyngton in 1800. The Traveling Salesman Problem (TSP) involves a salesman who has to travel to several points. Each point is visited only once, and the salesperson must return to the starting point again by trying the minimum path. For every *n* number of points, the number of routes to be traveled is *n*!. This problem causes no optimal solution except to calculate every possibility. So, the discussion of TSP is growing exponentially.

In life, many TSP problems are found to solve passenger delivery and pickup problems [1]–[3], drone and truck combination trips in improving customer service [4], [5], land logistics delivery problems[6]–[9], air logistics delivery [10], [11], picking up trash cars [12], automating systems on mobile robotics [13], [14] and others. The traveling salesperson problem has become one of the most studied problems in combinatorial optimization. The search for TSP solutions offers many algorithms that are fast in their calculations and produce optimal solutions.

Many scientists have tried to solve the TSP problem. Various methods are used to obtain more optimal and faster results in the calculations [15], [16]. Zhang et al. [17] presented a variable neighborhood discrete whale optimization algorithm for TSP. Teng and Li [18] proposed a discrete firefly algorithm combining genetic algorithms for solving TSP. Several other algorithms are offered in solving TSP, such as construction tour techniques based on the Convex-hull heuristic and Nearest Neighbor (CH-NN) [19], galaxy-based search algorithm (GbSA), and embedding new ideas called clockwise search processes and operations cluster crossover [20], optimal heuristic algorithm (2-opt) with Nearest Neighbor (NN) [21]–[24], tour construction Ant Colony Algorithm [25]–[27], and Cuckoo Search Algorithm [28].

Traveling Salesman Problem (TSP) research has witnessed significant advancements over the years, marked by the development of sophisticated heuristic algorithms and the establishing of benchmark datasets like TSPLIB. These algorithmic improvements, such as Nearest Neighbor (NN), Repetitive Nearest Neighbor (RNN), and hybrid approaches, have greatly enhanced our ability to find near-optimal solutions for TSP instances. However, several challenges and unresolved issues persist in this field. Scaling complexity remains a formidable challenge, particularly when dealing with large-scale TSP instances, where the problem's exponential nature poses computational hurdles. Furthermore, despite their efficiency, heuristic algorithms do not guarantee the global optimum, leaving room for further exploration to bridge the gap between heuristic and true optimal solutions. Additionally, dynamic TSP scenarios, where cities and distances change over time, demand adaptive heuristic approaches.

Given these challenges, the presented study focusing on TSP in Medan City, North Sumatra, Indonesia, assumes particular importance and novelty. Medan City's unique geographical layout and urban complexities present a real-world context that offers practical relevance for logistics, transportation, and urban planning. The study goes beyond theoretical analysis by rigorously evaluating four heuristic algorithms—NN, RNN, Hybrid NN, and Hybrid RNN—across various problem instances specific to Medan City. This empirical examination provides valuable insights into the algorithms' performance under the city's unique conditions, aiding decision-makers in optimizing routes efficiently. Moreover, the study introduces the novel concept of hybridization techniques within the context of TSP in Medan City. The successful application of hybrid algorithms

demonstrates their potential to yield high-quality solutions, contributing to the broader knowledge of computational optimization. In essence, this study not only addresses a complex optimization problem in a real-world setting but also pioneers innovative approaches, making it a noteworthy addition to the field of TSP research.

II. Mathematical Model

In this study, the mathematical model of the TSP aims to minimize travel distances. x_{ij} is the decision variable on which vertex to traverse. The set decision variable is {1,0}. The decision is worth 1 if $arc\ (i,j)$ is passed and 0 if arc (i,j) is not passed. The w_{ij} variable is the distance between points that are traversed using point distance calculations so that it can be written down like Eq. (1).

Parameter:

n : The number of points that the salesman needs to visit.

w_{ij} : The distance or cost between two points, calculated using the Euclidean distance formula.

x_{ij} : A binary decision variable that indicates whether the salesman travels from point i to point j ($x_{ij} = 1$) or not ($x_{ij} = 0$).

$|V|$: The number of elements in the subset.

(a_i, b_i) : The position of a point in terms of its horizontal (a_i) and vertical (b_i) coordinates.

Formula:

$$w_{ij} = \left(\left(a_i - a_j\right)^2 + \left(b_i - b_j\right)^2\right)^{1/2} \quad (1)$$

In general, the mathematical model for the TSP problem can be seen in Eq. (2) to Eq. (5).

$$\min \Sigma_{i=1}^{n} \Sigma_{j\neq i, j=1}^{n} w_{ij} x_{ij} \quad (2)$$

$$\Sigma_{i=1, i\neq j}^{n} x_{ij} = 1,\ j = 1, 2, 3, \dots, n \quad (3)$$

$$\Sigma_{j=1, j\neq i}^{n} x_{ij} = 1,\ i = 1, 2, 3, \dots, n \quad (4)$$

$$\Sigma_{i\in V}^{n} \Sigma_{j\neq i, j\in V}^{n} x_{ij} \leq |V| - 1, \forall V \subseteq \{1,2,3 \dots n\}, |V| \geq 2 \quad (5)$$

Eq. (2) is the objective function of TSP to minimize costs. Eq. (3) and Eq. (4) guarantee that each point is traversed exactly once and returns to the starting point of departure. Eq. (5) ensures that the number of traversed vertices is not more than or equal to the specified number of vertices minus one or can be written as $|V| - 1$.

III. RNN Algorithm

RNN and Nearest Neighbor NN are commonly used algorithms for solving the shortest route problem on a map with many points. The NN algorithm works by starting from a random point and looking for the nearest neighbour to proceed to the next point. This process is repeated until all points are connected in a closed route. The NN algorithm is simple and fast but only sometimes produces the best solution.

The RNN algorithm is a variation of the NN algorithm that works by finding the nearest neighbour at each stage but with some modifications. This algorithm looks for the nearest neighbours for the first point, returns to the starting point and looks for the nearest neighbours yet to be connected to the route. This algorithm is repeated until all points are connected in a closed route. The RNN algorithm is generally better at finding better solutions than the NN algorithm but requires a longer computation time.

Both are heuristic algorithms that can quickly solve the shortest route problem with many points. However, keep in mind that the solution provided by the heuristic algorithm is only sometimes optimal, especially in cases with many points or in cases with many constraints. The step-by-step construction of the RNN algorithm can be read in the following step sequence:

Step 1: Suppose $V = \{v_1, v_2, v_3, \dots, v_n\}$ is the point of n locations and $d(v_i, v_j)$ is the distance between location v_i and v_j, the notation v_i is the notation of the i^{th} site.

In this case, the search engine formulates n sub-routes and has one location in each. The set of sub-routes can be denoted as follows:

$$P_1 = \{v_i; i = 1, 2, \dots, n\} \quad (6)$$

Step 2: In the next step, each sub-route obtained in the previous step adds a network by finding the closest location that differs from the remaining spots. So this model can be formulated based on Eq. (7). The notation $d(v_i, v_j)$ is the Euclidean distance between locations v_i and v_j. This distance is calculated in the formula in Eq. 1.

$$P_2 = \{v_i, v_j\}; \forall\ v_i \in R_1;$$

$$min\ d(v_i, v_j), \forall\ v_j \in V;\ i \neq j;\ j = 1,2, \dots, n \quad (7)$$

Step 3: In the last step, each sub-route from 2 locations in R_2 is expanded by finding unvisited paths. The third sub-route built is notated in Eq. 8.

$$P_3 = \{v_i, v_j\}, v_k; \forall\ v_i, v_j \in R_1;$$

$$min\left\{d\left(v_j, v_k\right)\right\}, \forall\ v_k \in V\ ;\ j \neq k;\ j = 1, 2, \dots, n \quad (8)$$

where $|P_3| = n$ is the total number of routes in the set R_3. The process of building the path is continued until every location is visited. Furthermore, when the nth step has been completed, a set of routes P_n will be obtained, where $|P_n| = n$ and each path from R_n contains n locations. The algorithm added the initial site to each path's position $(n + 1)$ in P_n to obtain a feasible TSP route solution. The steps are continued until the best route is obtained from a set of n possible TSP routes [29].

IV. Heuristics 2-opt

2-Opt is a straightforward local search method. Croes first introduced this method to solve the TSP in 1958. However, this algorithm is used to improve the shortcomings of other algorithms, which require a long computation time. 2-Opt heuristics is a heuristic technique to solve the shortest route problem by improving existing routes. This technique is named "2-Opt" because it considers two points on a route and tries to swap the path connecting those two points with an alternative path that may be more efficient. The 2-Opt

algorithm takes an initial route consisting of several points and then considers all possible combinations of two points on the route. After that, the algorithm tries to rotate the part of the route between the two points to produce a new route. Then, the algorithm compares the total initial route distance with the resulting new total route distance and chooses the route that has the shortest total distance. Fig. 1 is an illustration of the 2-Opt algorithm.

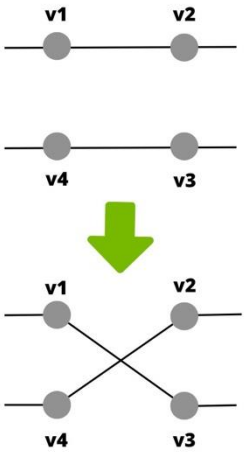


Fig. 1. An illustration of how the 2-Opt algorithm works.

In Fig. 2, there are four location points symbolized by v_1, v_2, v_3, and v_4. Initially, some edges intersect, namely v_1 to v_2 and v_3 to v_4. This algorithm checks the distance between other adjacent points without adding new edges. The other closest sides to be checked are (v_1, v_3) and $d(v_2, v_4)$. The criteria for selecting the ideal side can be calculated by comparing the Euclidean distance-like Eq. (9). If the conditions in Eq. 9 are met, then the two sides (v_1, v_2) and (v_3, v_4) will be replaced with two new sides (v_1, v_3) and $d(v_2, v_4)$.

$$d(v_1, v_3) + d(v_2, v_4) < d(v_1, v_2) + d(v_3, v_4) \quad (9)$$

V. Hybrid Heuristics

Hybrid NN is a heuristic algorithm which is a combination of two heuristic techniques, namely NN and 2-Opt. Hybrid RNN is a heuristic algorithm which is a combination of two heuristic techniques, namely RNN and 2-Opt. This algorithm is designed to solve the shortest route problem by using the advantages of both heuristic techniques. In the Hybrid Nearest Neighbor-2Opt approach, you will generate an initial solution using the NN method. Then, you'll refine this solution using the 2-opt method. This combination can be effective because the NN method provides a good starting point for the 2-opt method, which can refine the solution. The hybrid RNN-2-Opt algorithm begins by building an initial route using the RNN algorithm. After the initial route is built, the 2-Opt technique is applied to improve the route by finding a point on the route that can be exchanged so that the total route distance becomes shorter. This process is repeated until there are no more points that can be exchanged to shorten the route distance.

```
function RNN-2Opt(cities):
  best_tour = None
  best_distance = infinity
  for each city in cities:
    current_tour = NearestNeighborTour(starting from city)
    improved_tour = True
    while improved_tour:
      improved_tour = False
      for i=1 to number of cities in the tour - 1:
        for j=i+1 to number of cities in the tour:
          new_tour = 2OptSwap(current_tour, i, j)
          if distance of new_tour < distance of current_tour:
            current_tour = new_tour
            improved_tour = True
    if distance of current_tour < best_distance:
      best_tour = current_tour
      best_distance = distance of current_tour
  return best_tour
function NearestNeighborTour(starting city):
  create an empty tour
  add the starting city to the tour
  while there are unvisited cities:
    find the nearest unvisited city to the last city in the tour
    add the nearest city to the tour
  return the tour
function 2OptSwap(tour, i, j):
  return the tour up to i-1, followed by the section from i to j reversed, followed by the rest of the tour
```

Fig. 2. Pseudocode hybrid RNN.

This pseudocode follows the Hybrid RNN algorithm as follows [29]:

1) It starts from each city and generates a tour using the Nearest Neighbor (NN) heuristic.

2) Then it tries to improve this tour using the 2-Opt heuristic, making 2-Opt swaps as long as they improve the tour.

3) It keeps track of the best tour found so far, and once all cities have been used as starting points, it returns the best tour found.

4) The NearestNeighborTour function implements the Nearest Neighbor heuristic: starting from a given city, it repeatedly visits the nearest unvisited city.

VI. Experimental Results

In the discussion in this study, we present 100 locations in Medan city, North Sumatra, Indonesia. The locations will be analyzed for the shortest route based on location coordinates. The location to be calculated can be seen in Fig. 3. Customer location according to geographical coordinates can be seen in Fig. 3 (a), and Fig. 3(b) is the location transformation to the Cartesian point plane. This transformation makes it easy to illustrate Euclidean distance calculations.

There are four solutions proposed to be observed, namely NN, RNN, hybrid NN, and hybrid RNN. This study uses the Rstudio software with the TSP package and tspmeta for calculating heuristic solutions. A comparison of the total mileage obtained can be seen in Table I. In Table I, RNN and hybrid RNN methods have the best solution in the case of a combination of 100 location points in Medan city, North Sumatra Province, Indonesia. The table provides information about the heuristics used to solve the shortest route problem at

100 different points and the total distance travelled in kilometres. Four types of heuristics are used: NN, RNN, hybrid NN, and hybrid RNN. NN and RNN have different total distances travelled, where RNN produces a shorter total distance than NN (385.83 km compared to 435.91 km). The hybrid NN also produces a shorter total distance than the NN (414.09 km compared to 435.91 km), although it is still longer than the RNN. RNN-2Opt has the same total distance as RNN (385.83 km), so RNN-2Opt is the most effective heuristic in solving the shortest route problem at those 100 points. The results of data processing with Rstudio are visualized in Fig. 4. Fig. 4 is a travel route using the local search heuristic algorithm: NN, RNN, hybrid NN-2Opt, and hybrid NN-2Opt.

In the next stage, we tested several location points totaling 25, 40, 50, 75, 100, 150, 200, and 300 randomly generated around Medan city, North Sumatra Province. From the simulation results given, the total distance traveled in kilometers can be seen in Table II. The Table II has six problem instances, namely n25mdn, n40mdn, n50mdn, n75mdn, n100mdn, and n150mdn.

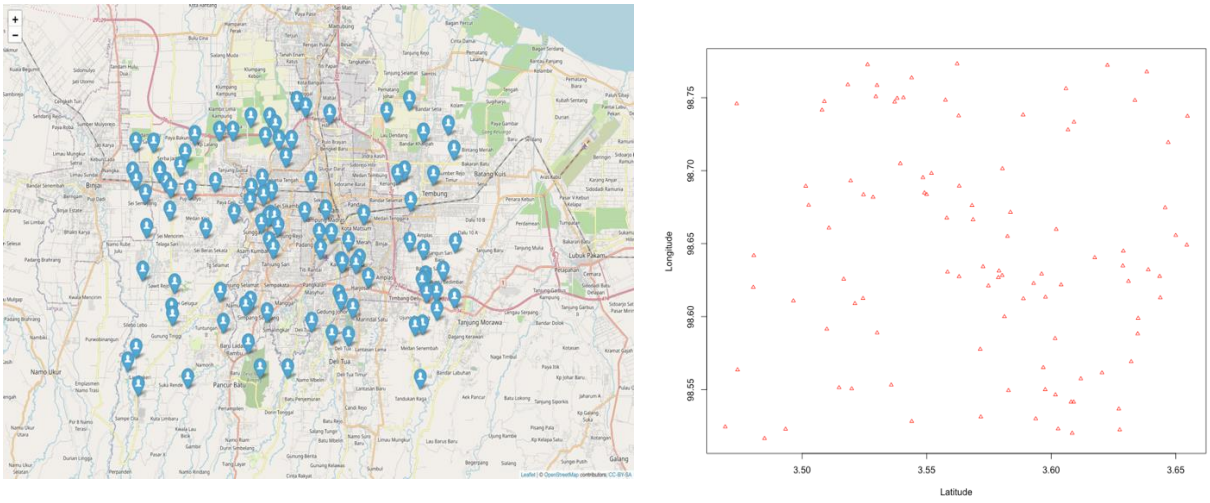
The total distance travelled by the four different algorithms is given for each problem instance. From the table, the algorithm's performance varies depending on the size of the problem instance. In the n25mdn problem instance, the hybrid NN algorithm produces the shortest total distance (202.28 km) compared to the other three algorithms. However, in the case of the n50mdn problem, the hybrid NN algorithm is no longer the best choice, while the hybrid RNN algorithm produces the shortest total distance (309.25 km). In more significant problem instances such as n100mdn, RNN and hybrid RNN beat the other two algorithms, producing the same total distance (385.83 km). At the same time, hybrid NN could only produce a total distance of 414.10 km, and NN produced a longer total distance of 435.92 km. However, remember that the algorithm's performance depends on the particular problem instance, so choosing an algorithm based on the characteristics and size of the problem instance to be solved is better. To find out the capabilities of this method to quickly calculate the minimum mileage. So we tested the problem with some data from TSPLIB and tried to compare the best-known solutions with our proposed method. The results of the simulation carried out can be seen in Table III.

TABLE I. TOUR LENGTH

Heuristics	n	Total Distance (km)
NN	100	435.91
RNN	100	385.83
hybrid NN	100	414.09
hybrid RNN	100	385.83

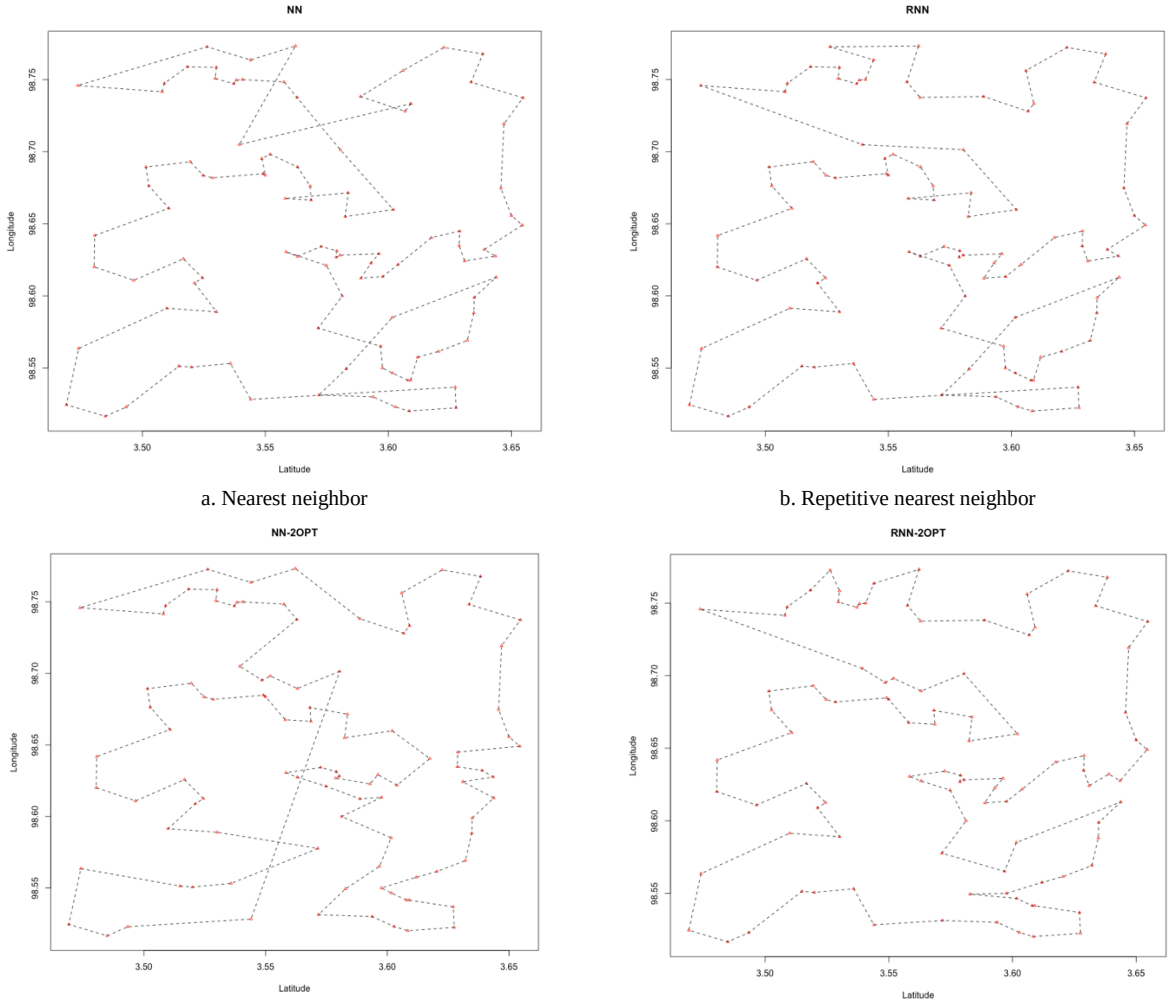
From the Table III, it can be seen that the performance of the algorithm varies depending on the particular problem instance. In Berlin52, the Hybrid NN and RNN Algorithm, algorithms have the best performance, producing a total value of the shortest distance of 8182.19. However, in Ch150, the Hybrid RNN algorithm produced the best total shortest distance, 6695.24, while the NN Algorithm produced the longest total distance (8025.45). In larger problem instances such as pr299, the Hybrid NN and RNN Algorithm, algorithms can beat the other two algorithms, where both produce a better total distance value compared to the NN Algorithm and Hybrid RNN. However, these methods still need to be revised to the best-known solutions. To see a comparison of the methods is presented in Fig. 5. In Fig. 5, it can be seen that the error weight is calculated by Eq. (10). The hybrid method can reduce the error rate.

$$Error\ (\%) = \frac{Proposed\ Method - Best\ Known}{Best\ Known}\ x\ 100\% \qquad (10)$$



(a) (b)

Fig. 3. One hundred location points that must be visited in Medan city, North Sumatra Province.



a. Nearest neighbor

b. Repetitive nearest neighbor

c. Hybrid nearest neighbor

d. Hybrid repetitive nearest neighbor

Fig. 4. Comparison of travel routes using the NN, RNN, hybrid NN-2Opt, and hybrid NN-2Opt algorithms.

TABLE II. THE DISTANCE TRAVELED WITH FOUR HEURISTIC METHODS

Problem	n	NN	Hybrid NN	RNN	Hybrid RNN
n25mdn	25	258.74	202.28	216.79	216.05
n40mdn	40	303.03	251.06	258.68	235.44
n50mdn	50	341.50	286.67	341.50	309.25
n75mdn	75	447.97	378.39	406.00	361.03
n100mdn	100	435.92	414.10	385.83	385.83
n150mdn	150	544.63	548.17	535.47	488.00
n200mdn	200	637.70	599.51	628.23	565.69
n300mdn	300	844.22	691.48	767.99	692.26

TABLE III. SIMULATION RESULTS FOR CALCULATING TOUR LENGTH USING TSPLIB DATA

Problem Name	Num. of City	Best Known	NN	Hybrid NN	RNN	Hybrid RNN
Berlin52	52	7542	8182.19	7713.03	8182.19	8182.19
Ch150	150	6528	8025.45	7656.96	7078.44	6695.24
pr299	299	48191	57901.3	49555.9	56199.22	50566.20
pr264	264	49135	54491.5	52084	54124.53	53431.19
pcb442	442	50778	58953	53044.4	59947.47	53898.70
bier127	127	118282	133971	122072	127708.80	125030.50

Fig. 5 states that the percentage errors of the NN, RNN, Hybrid NN, and Hybrid RNN algorithms in solving some shortest route problems using several datasets. The percentage error indicates how far the results produced by the algorithm are from the known best (best-known solution) for each problem. The lower the error percentage, the better the results produced by the algorithm. Based on the Fig. 5, it can be seen that hybrid NN and hybrid RNN tend to give a lower error percentage than NN and RNN. Hybrid NN even produces the lowest error value on the Berlin52 dataset, and hybrid RNN produces the lowest error value on the Ch150 dataset. This shows that using hybridization techniques can improve the quality of the solutions produced by the algorithm.

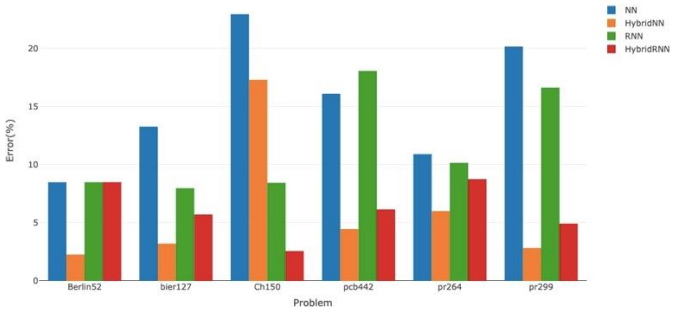


Fig. 5. Comparison of the heuristic method errors offered.

In aggregate, there is an average difference between non-hybrid and hybrid methods. The simulation results using RStudio using the t.test() function obtained a value of $t = 2.8056$, $df = 27$, and $p - value = 0.0092$ at a 95% confidence level. The t-value indicates the magnitude of the difference between the means of the two groups, in this case, the non-hybrid and hybrid methods. A larger t-value indicates a greater difference between the means. The degrees of freedom (df) represent the independent observations to estimate a parameter. In this case, it indicates the number of observations minus the number of parameters estimated, which is equal to 27. The p-value is the probability of obtaining a t-value as extreme as or more extreme than the observed t-value, assuming the null hypothesis is true. In this case, the null hypothesis is that there is no significant difference between the means of the non-hybrid and hybrid methods. A p-value of 0.0092 suggests that the probability of obtaining such an extreme t-value under the null hypothesis is less than 0.01. Therefore, the result is statistically significant at a 95% confidence level. In summary, the simulation results suggest a significant average difference between non-hybrid and hybrid methods in terms of their performance, with the hybrid methods outperforming the non-hybrid methods on average.

VII. DISCUSSION

In this study, the authors delve into the challenging Traveling Salesman Problem (TSP) using a variety of heuristic algorithms. Their research focuses on 100 locations in Medan city, North Sumatra, Indonesia, and aims to find the shortest route based on geographical coordinates. We transform geographical coordinates into Cartesian points to facilitate this analysis, making Euclidean distance calculations more accessible. Four heuristic solutions are proposed for examination: Nearest Neighbor (NN), Repetitive Nearest Neighbor (RNN), Hybrid NN, and Hybrid RNN. RStudio software, coupled with the TSP package and tspmeta, is employed for heuristic solution calculations. The study sheds light on the performance of these algorithms, particularly focusing on the total mileage obtained, which is compared across different problem instances and known benchmarks.

The results of this study reveal that the performance of heuristic algorithms is contingent upon the specific problem instance under consideration. In smaller problem instances, such as n25mdn, the Hybrid NN algorithm proves to be the most efficient in producing the shortest total distance. Nevertheless, as problem instances grow in complexity, like n50mdn and n100mdn, the RNN and Hybrid RNN consistently outperform the other algorithms, yielding identical total distances in the case of n100mdn. The choice of the algorithm should be tailored to the problem's characteristics and scale, highlighting the nuanced nature of TSP problem-solving.

The study evaluates the proposed heuristic methods with known best solutions from the TSPLIB data. This comparative analysis offers valuable insights into algorithmic performance under standardized benchmarks. For instance, the Berlin52 dataset demonstrates that Hybrid NN and RNN algorithms attain the best results, aligning with the best-known solution. Conversely, in the Ch150 dataset, the Hybrid RNN algorithm emerges as the frontrunner, boasting the best total shortest distance. This highlights the potential of hybridization techniques, like combining NN and RNN with 2-Opt, to yield superior-quality solutions and reduce error rates vis-à-vis non-hybrid methods.

The paper augments its findings with statistical rigour, employing a t-test to substantiate the significance of differences between non-hybrid and hybrid methods. The results confirm that, on average, hybrid methods surpass their non-hybrid counterparts, underscoring their potential for achieving more optimal solutions.

This study carries substantial implications for addressing TSP and akin optimization challenges. Algorithm selection should be a nuanced process tailored to the problem's size and intricacies. The fusion of heuristic algorithms, as demonstrated in hybridization techniques, holds promise for elevating solution quality. Benchmarking against established datasets like TSPLIB serves as a litmus test for algorithmic reliability. Importantly, while heuristic methods may not always secure the global optimum, their swiftness renders them indispensable for tackling real-world routing and scheduling problems. This research advances the understanding of heuristic algorithm performance within the TSP domain and underscores the transformative potential of hybridization strategies in optimization problem-solving.

VIII. Conclusion

TSP is a complex combinatorial problem in calculations to find the best combination. Because it is challenging to calculate the final solution with the shortest tour length globally, however, the heuristic approach method can be used. This heuristic method is a method that can calculate a solution quickly with a pretty good solution. However, it is still inferior to modern and exact metaheuristic methods in finding solutions, but the speed of finding solutions cannot be doubted because they can calculate quickly. Therefore this method is still widely used in various applications in the real world. The results of this study indicate that the offered hybrid method can correct errors from the usual heuristic methods. The combination of NN and RNN algorithms with 2-Opt provides a better solution. The hybrid method minimized the percentage of errors compared to the non-hybrid method. The 2-Opt technique can be used in combination with other heuristic algorithms such as NN or RNN to improve the quality of the resulting solutions. However, although the 2-Opt technique can increase the efficiency and quality of solutions, this technique does not guarantee that the given solution will always be optimal.

References

[1] J. G. Lopes Filho, M. C. Goldbarg, E. F. Gouvea Goldbarg, and V. A. Petch, "Traveling salesman problem with optional bonus collection, pickup time and passengers," *Revista de Informatica Teorica e Aplicada*, vol. 27, no. 1, 2020, doi: 10.22456/2175-2745.93733.

[2] Z. Lyu, D. Pons, J. Chen, and Y. Zhang, "Developing a Stochastic Two-Tier Architecture for Modelling Last-Mile Delivery and Implementing in Discrete-Event Simulation," *Systems*, vol. 10, no. 6, 2022, doi: 10.3390/systems10060214.

[3] Z. Zhang, H. Liu, M. C. Zhou, and J. Wang, "Solving Dynamic Traveling Salesman Problems With Deep Reinforcement Learning," *IEEE Trans Neural Netw Learn Syst*, 2021, doi: 10.1109/TNNLS.2021.3105905.

[4] R. Roberti and M. Ruthmair, "Exact methods for the traveling salesman problem with drone," *Transportation Science*, vol. 55, no. 2, 2021, doi: 10.1287/TRSC.2020.1017.

[5] M. Dell'Amico, R. Montemanni, and S. Novellani, "Matheuristic algorithms for the parallel drone scheduling traveling salesman problem," *Ann Oper Res*, vol. 289, no. 2, 2020, doi: 10.1007/s10479-020-03562-3.

[6] P. Baniasadi, M. Foumani, K. Smith-Miles, and V. Ejov, "A transformation technique for the clustered generalized traveling salesman problem with applications to logistics," *Eur J Oper Res*, vol. 285, no. 2, 2020, doi: 10.1016/j.ejor.2020.01.053.

[7] M. K. Zuhanda, S. Suwilo, O. S. Sitompul, and M. Mardiningsih, "A combination k-means clustering and 2-opt algorithm for solving the two echelon e-commerce logistic distribution," *Logforum*, vol. 18, no. 2, pp. 213–225, Jun. 2022, doi: 10.17270/J.LOG.2022.734.

[8] M. K. Zuhanda *et al.*, "Optimization of Vehicle Routing Problem in the Context of E-commerce Logistics Distribution," Mar. 2023.

[9] M. K. Zuhanda *et al.*, "Supply chain strategy during the COVID-19 terms: sentiment analysis and knowledge discovery through text mining," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 30, no. 2, p. 1120, May 2023, doi: 10.11591/ijeecs.v30.i2.pp1120-1127.

[10] Muren, J. Wu, L. Zhou, Z. Du, and Y. Lv, "Mixed steepest descent algorithm for the traveling salesman problem and application in air logistics," *Transp Res E Logist Transp Rev*, vol. 126, 2019, doi: 10.1016/j.tre.2019.04.004.

[11] F. Kong, J. Li, B. Jiang, H. Wang, and H. Song, "Trajectory Optimization for Drone Logistics Delivery via Attention-Based Pointer Network," *IEEE Transactions on Intelligent Transportation Systems*, 2022, doi: 10.1109/TITS.2022.3168987.

[12] C. X. Lou, J. Shuai, L. Luo, and H. Li, "Optimal transportation planning of classified domestic garbage based on map distance," *J Environ Manage*, vol. 254, 2020, doi: 10.1016/j.jenvman.2019.109781.

[13] K. Hernandez, B. Bacca, and B. Posso, "Multi-goal path planning autonomous system for picking up and delivery tasks in mobile robotics," *IEEE Latin America Transactions*, vol. 15, no. 2, 2017, doi: 10.1109/TLA.2017.7854617.

[14] S. Piao, Z. Ba, L. Su, D. Koutsonikolas, S. Li, and K. Ren, "Automating CSI Measurement with UAVs: From Problem Formulation to Energy-Optimal Solution," in *Proceedings - IEEE INFOCOM*, 2019. doi: 10.1109/INFOCOM.2019.8737613.

[15] A. P. U. Siahaan *et al.*, "Comparative study of prim and genetic algorithms in minimum spanning tree and Travelling Salesman Problem," *International Journal of Engineering and Technology(UAE)*, vol. 7, no. 4, 2018, doi: 10.14419/ijet.v7i4.20606.

[16] M. K. Zuhanda, H. Mawengkang, S. Suwilo, Mardiningsih, and O. S. Sitompul, "Logistics distribution supply chain optimization model with VRP in the context of E-commerce," in *AIP Conference Proceedings*, 2023. doi: 10.1063/5.0128465.

[17] J. Zhang, L. Hong, and Q. Liu, "An improved whale optimization algorithm for the traveling salesman problem," *Symmetry (Basel)*, vol. 13, no. 1, 2021, doi: 10.3390/sym13010048.

[18] L. Teng and H. Li, "Modified discrete Firefly Algorithm combining genetic algorithm for traveling salesman problem," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 16, no. 1, 2018, doi: 10.12928/TELKOMNIKA.V16I1.4752.

[19] E. O. Asani, A. E. Okeyinka, and A. A. Adebiyi, "A Construction Tour Technique for Solving the Travelling Salesman Problem Based on Convex Hull and Nearest Neighbour Heuristics," in *2020 International Conference in Mathematics, Computer Engineering and Computer Science, ICMCECS 2020*, 2020. doi: 10.1109/ICMCECS47690.2020.240847.

[20] A. P. Ang and D. Jitkongchuen, "The cluster crossover operation for the symmetric Travelling Salesman Problem," *ECTI Transactions on Computer and Information Technology*, vol. 12, no. 2, 2018, doi: 10.37936/ecti-cit.2018122.132018.

[21] D. R. Singh, M. K. Singh, and T. Singh, "A hybrid heuristic algorithm for the Euclidean traveling salesman problem," in *International Conference on Computing, Communication and Automation, ICCCA 2015*, 2015. doi: 10.1109/CCAA.2015.7148514.

[22] S. Klootwijk, B. Manthey, and S. K. Visser, "Probabilistic analysis of optimization problems on generalized random shortest path metrics," *Theor Comput Sci*, vol. 866, 2021, doi: 10.1016/j.tcs.2021.03.016.

[23] V. Ilin, D. Simić, S. D. Simić, and S. Simić, "Hybrid Genetic Algorithms and Tour Construction and Improvement Algorithms Used for Optimizing the Traveling Salesman Problem," in *Advances in Intelligent Systems and Computing*, 2021. doi: 10.1007/978-3-030-57802-2_51.

[24] A. Agrawal, N. Ghune, S. Prakash, and M. Ramteke, "Evolutionary algorithm hybridized with local search and intelligent seeding for solving multi-objective Euclidian TSP," *Expert Syst Appl*, vol. 181, 2021, doi: 10.1016/j.eswa.2021.115192.

[25] M. Mavrovouniotis, S. Yang, M. Van, C. Li, and M. Polycarpou, "Ant colony optimization algorithms for dynamic optimization: A case study of the dynamic travelling salesperson problem [Research Frontier]," *IEEE Comput Intell Mag*, vol. 15, no. 1, 2020, doi: 10.1109/MCI.2019.2954644.

[26] M. Mavrovouniotis, F. M. Muller, and S. Yang, "Ant Colony Optimization with Local Search for Dynamic Traveling Salesman Problems," *IEEE Trans Cybern*, vol. 47, no. 7, 2017, doi: 10.1109/TCYB.2016.2556742.

[27] M. Mavrovouniotis, F. M. Müller, and S. Yang, "An ant colony optimization based memetic algorithm for the dynamic Travelling Salesman Problem," in *GECCO 2015 - Proceedings of the 2015 Genetic and Evolutionary Computation Conference*, 2015. doi: 10.1145/2739480.2754651.

[28] V. Bhavana, V. Ramesh, and M. Sivagami, "Implementing discrete cuckoo search algorithm for TSP using MPI and beowulf cluster," *International Journal of Innovative Technology and Exploring Engineering*, vol. 8, no. 8, 2019.

[29] Md. A. Rahman and H. Parvez, "Repetitive Nearest Neighbor Based Simulated Annealing Search Optimization Algorithm for Traveling Salesman Problem," *OAlib*, vol. 08, no. 06, 2021, doi: 10.4236/oalib.1107520.