

Smart Routes: Hybrid Metaheuristics for Efficient Vehicle Routing Problem

Yehor Kovalenko*, Andrei Pivavaru[†] and Joanna Ochelska-Mierzejewska[‡]

Technical University of Lodz

116 Zeromskiego Street, 90-924 Lodz, Poland

*ORCID: 0009-0003-3744-3294

[†]ORCID: 0009-0008-7730-6063

[‡]ORCID: 0000-0002-9295-3962

Email: joanna.ochelska-mierzejewska@p.lodz.pl

Abstract—This article presents a hybrid algorithm developed to solve the Vehicle Routing Problem with Time Windows (VRPTW), which involves finding optimal routes for a fleet of vehicles serving a set of geographically dispersed customers within specified time intervals. The proposed solution combines Ant Colony Optimization (ACO) as the primary method for global solution construction, with the 2-opt local search technique used for route refinement, and a Tabu Search strategy to escape local optima and further improve solution quality.

The algorithm dynamically adapts pheromone levels to favor both spatial and temporal proximity between customers, enhancing decision making during route construction. Experimental results demonstrate that the hybrid approach yields high-quality solutions, significantly improving known results by up to 30% in some cases, while maintaining reasonable computation times. This makes the algorithm well-suited for real-time logistics scenarios where time efficiency and solution accuracy are both critical.

I. INTRODUCTION

FOR MANY years, researchers have explored a wide range of challenges in the field of combinatorial optimization. One of the most thoroughly studied problems in this area is the Traveling Salesman Problem (TSP). Although initially mentioned in the 19-th century by Kirkman and Hamilton, it was not formally defined until the 1930s by Schirjver [1]. The basic version of the TSP involves determining the shortest possible route that visits n cities exactly once and returns to the starting point. As an NP-hard problem, it is computationally infeasible to exhaustively evaluate all permutations for larger instances. Nevertheless, due to its straightforward formulation and relevance to real-world applications, the TSP has become a standard benchmark for evaluating the performance of optimization algorithms [2].

The TSP can be seen as a specific instance of a broader class of problems that focus on finding an optimal route through a set of locations using a single vehicle. This concept is generalized by the Vehicle Routing Problem (VRP), which was formally introduced in 1959 by Dantzig and Ramser. Their pioneering work not only proposed a mathematical model but also applied it to a practical problem involving fuel distribution [3].

VRP is one of the key issues in logistics and supply chain management, having a direct impact on operating costs,

customer service quality and overall business efficiency. In practice, it involves the process of making decisions about when, how and what to use to deliver goods to end users in the most efficient way possible while meeting a number of constraints, such as transport capacity, delivery time windows, driver availability and road conditions [4]. Problems of this type are extremely important not only for transport and logistics companies, but also for commercial, manufacturing and e-commerce organizations.

The VRP seeks to generate a set of vehicle routes that begin and end at a central depot, such that customer demands are met, all operational constraints are satisfied, and the total transportation cost is minimized. Depending on the configuration of the problem - such as the layout of the road network, customer requirements, depot locations, and available vehicles - different variants of the VRP can be defined, each corresponding to specific research or operational challenges [5].

In most formulations, the transportation network is modeled as a graph, where vertices represent depots and customers, and edges represent roads (see Fig. 1). Edges may be directed (for one-way streets) or undirected (for two-way roads), and each edge has an associated cost metric, such as distance, travel time, or monetary cost. Customers are typically described by the following attributes [5]:

- the vertex in the network they are assigned to;
- the amount of goods to be delivered or picked up;
- the time window during which service must occur;
- the service duration (loading or unloading time);
- the set of vehicles eligible to serve them;
- penalties for non-service.

In the standard VRP, only the first two attributes are considered, while the others are included in extended versions of the problem.

To summarize, a solution to the VRP involves determining routes for a fleet of vehicles that start and end at one or more depots (represented as vertices in the graph). These depots are defined not only by their location but also by the number and types of vehicles they host, and the volume of goods available for dispatch. The fleet itself may be homogeneous (identical vehicles) or heterogeneous (vehicles with varying capacities and capabilities), and the fleet size may be fixed

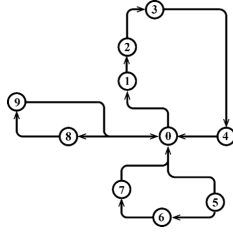


Fig. 1. Graphic representation of VRP

or dynamically adjusted based on customer demand. Vehicles can also have specific features, such as:

- their assigned depot (start and end location);
- capacity constraints (e.g., maximum weight, volume, or number of units);
- compartmentalization for transporting multiple types of goods;
- restricted access to certain parts of the road network;
- operational costs per distance unit, time unit, or network edge.

Section *Literature review* presents a review of the literature on VRP solutions discussed in this article. Section *A Novel Hybrid Algorithm for Solving VRP* discusses the proposal of a new hybrid approach to solving VRP combining the ant algorithm, Tabu Search and 2-opt. Section *Experimental Evaluation and Results* presents the research methodology, discusses the conducted experiments. Section *Evaluation Against Standard Benchmarks* compares the obtained results with the published results and indicates the improvement in the effectiveness of the proposed new hybrid method.

II. LITERATURE REVIEW

For solving VRP, one of the simplest and most commonly used heuristics for local search is the 2-opt algorithm, originally proposed by [6] for the traveling salesman problem (TSP). This method consists in iteratively removing two edges from the route and replacing them with others in such a way as to reduce the cycle length [7]. Despite its simplicity, 2-opt is an effective tool for local improvement of the solution and is often an element of local refinement in more complex approaches. In the context of VRP, 2-opt can be used in both classical and extended versions – e.g. 2-opt* for multiple routes [8]. The method is often combined with other metaheuristic algorithms as part of the intensification phase [9].

Ant Colony Optimization (ACO) is a bioinspired metaheuristic proposed by Dorigo in the 1990s [10]. et al., 1996]. Modeled on the behavior of real ant colonies, the method relies on the simulation of artificial agents (ants) that construct solutions based on probabilistic choices driven by pheromones and local heuristics. Pheromones are reinforced on edges leading to good solutions, which leads to a gradual convergence of the population towards optimal or near-optimal routes.

ACO has been successfully applied to VRP and its numerous variants, including VRPTW (with time windows) and VRP with limited fleet. An example is the work of [11],

who applied a modified ACO to transportation problems in a dynamic environment, but also in [12]. Modifications to ACO include dynamic pheromone evaporation, local search (e.g. 2-opt as an improvement component), and stagnation avoidance mechanisms.

Tabu Search (TS) is a metaheuristic developed by [13] whose main goal is to efficiently search the solution space by avoiding cycles and local minima. TS uses a so-called tabu list — a data structure that stores information about illegal moves for a certain time – to prevent backtracking to recent solutions.

In the context of VRP, Tabu Search has proven to be extremely effective, as evidenced by numerous successful implementations in the literature [15], [14], [16]. This method allows for the flexible consideration of complex constraints such as time windows, payload constraints, or service zone divisions. Many variants of TS for VRP implement advanced aspiration strategies, restart procedures, and adaptive changes to tabu list parameters [17], [18].

In recent years, there has been a growing interest in hybrid approaches that combine the advantages of different methods. An example is the integration of ACO with a local search based on 2-opt, which allows for an increase in the quality of the final solutions [19]. Other works combine ACO with Tabu Search, where ACO generates initial solutions and TS is responsible for their intensification and exploration of new areas of the solution space. Hybrid approaches often achieve better results than single algorithms, especially for large-scale and multi-criteria problems [20], [21], [22], [23].

III. A NOVEL HYBRID ALGORITHM FOR SOLVING VRP

The central component responsible for solving VRP algorithm that initiates the solution search using a hybrid algorithm but also collects statistical data required for performance analysis—such as the globally best-found routes, shortest total distances, and average or worst distances per iteration is proposed in this paper.

The algorithm proceeds as follows:

1) Initialization Phase:

- Set initial best distance and routes.
- Prepare containers for collecting statistics across iterations.

2) Iterative Optimization Loop:

- For each iteration:
 - a) Ant Colony Construction:
 - A colony of artificial ants is instantiated. Each ant constructs a set of feasible routes, attempting to serve all customers starting and ending at the depot.
 - b) Local Improvement via 2-opt:
 - Each route generated by the ants undergoes local optimization using the 2-opt algorithm to reduce path crossings and improve solution quality.
 - c) Best Ant Selection:

- The ant with the shortest total distance in the current iteration is selected. If it improves the global best solution, it is recorded.
- d) Tabu Search Activation:
 - If no improvement is detected for a defined number of iterations (e.g., 10), the algorithm switches to a Tabu Search procedure applied to the best known routes. This mechanism is used to escape local minima.
- e) Pheromone Update:
 - A pheromone evaporation process is applied to all edges.
 - Each ant deposits pheromone proportional to the inverse of its total distance.
 - An additional pheromone bonus is granted to the elite (best) ant.
- 3) Statistics Collection:
 - At the end of each iteration, the best, worst, and average distances are stored to support further analysis.
- 4) Termination and Output:
 - After completing all iterations, the algorithm returns the best-found solution along with its performance metrics.

To improve decision-making during the constructive phase of the solution, the pheromone initialization strategy was enhanced to incorporate problem-specific heuristics. Specifically, the initial pheromone levels between each pair of customers are adjusted to favor geographically closer customers and those with overlapping or adjacent time windows. This is achieved by combining a distance-based component—computed as the inverse of the Euclidean distance—with a time window similarity component—calculated as the inverse of the absolute difference between the customers' ready times. As a result, the algorithm introduces a meaningful prior bias in the search space, guiding ants toward more promising initial configurations while still allowing stochastic exploration..

Once initial solutions are constructed by the ants, each route undergoes refinement using a 2-opt local search procedure. This well-known heuristic systematically explores pairs of edges within a route and checks whether reversing the intermediate segment reduces the total traveled distance. The process is repeated iteratively until no further improvement can be achieved, thereby ensuring that each solution reaches a local minimum with respect to intra-route edge exchanges. Although 2-opt does not guarantee a globally optimal solution, its application significantly reduces path crossings and unnecessary detours, particularly in randomly scattered customer instances.

The underlying algorithm is a hybrid metaheuristic composed of three synergistic components: (1) Ant Colony Optimization (ACO) serves as the base method for generating initial feasible solutions, leveraging pheromone-based probabilistic construction; (2) a local search mechanism based on 2-opt further improves individual routes; and (3) a Tabu Search

algorithm is activated in case of stagnation, helping the system escape local optima by performing controlled, memory-guided explorations of the solution neighborhood. Tabu Search is triggered when no better solution is found over a predefined number of iterations, and it operates by modifying the current best-known solution while preventing reversals through a tabu list.

Each ant in the algorithm is implemented as an object-oriented entity encapsulating the coordinates of the depot, vehicle capacity constraints, a list of customers, pheromone information, and parameters controlling the influence of pheromone trails (α), heuristic desirability (β), and time window urgency (*coef_urgency*). During the route construction process, each ant maintains a dynamically updated list of feasible customers to visit next—those that do not violate capacity or time window constraints. Once no feasible customer remains, the ant returns to the depot, completing a route and potentially starting a new one if unvisited customers remain.

The main objective of the algorithm is the minimization of the total transportation cost, which in this context is expressed as the cumulative distance traveled by all vehicles across all routes. By integrating global exploration (ACO), local refinement (2-opt), and intensification-diversification control (Tabu Search), the proposed hybrid approach balances solution quality and convergence speed, showing strong potential for solving real-world instances of the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW).

A. Main algorithm

The proposed hybrid algorithm combines the global search capabilities of Ant Colony Optimization (ACO) with the local refinement of 2-opt and diversification strategies provided by Tabu Search (TS). The objective is to minimize the total distance traveled while satisfying customer demands and time window constraints.

To provide a clear understanding of the solution approach, the pseudocode in Algorithm 1 illustrates the overall procedure of the hybrid algorithm.

B. Choosing next client

Probability of choosing next client during delivery is defined by equation (1):

$$p_{ij}^k = \frac{[f_{ij}(t)]^\alpha \cdot [g_{ij}(t)]^\beta}{\sum_{d \in N_i^k} ([f_{id}(t)]^\alpha \cdot [g_{id}(t)]^\beta)} \quad (1)$$

where p_{ij}^k - probability of choosing unvisited j th client for ant k that is currently located at the position of client i ; α - the influence of pheromone trails; β - the influence of heuristics; N_i^k - the set of client that ant k has not yet visited, and to which there is a path from client i ; $f_{ij}(t)$ - value of the function representing the intensity of pheromone trails on the path between clients i and j ; $g_{ij}(t)$ - value of the function representing the heuristic component of the path between clients i and j .

Using the calculated probability next client is being chosen using the roulette wheel selection.

Algorithm 1 Overall algorithm

```

for Number of iterations do
2:  Ant colony initialization
   for Every ant in the colony do
4:    while Unvisited client exists do
       while Feasible clients exist do
6:         for Every client in feasible
           clients do
            Calculate probability
8:         end for
            Choose next client using Roulette wheel selec-
            tion
10:        Update visited routes
            Remove chosen client from unvisited
12:      end while
            Return to the depot
14:    end while
            Apply two-opt
16:    Check for stagnation
            if No improvement > 10 then
18:      Activate tabu search
            end if
20:    Update best-known solution
            Update pheromone trails
22:  end for
end for

```

C. Heuristic component

During preliminary experiments, the typical calculation of the heuristic component using only the Euclidean metric did not prove effective. Therefore, after conducting thorough analysis and taking into consideration the additional aspect of time windows of the problem, the heuristic determining was extended with a time-based coefficient that defines the urgency of the client and how quickly the vehicle must reach the client for the most effective use of time (2):

$$g_{ij}(t) = \frac{1}{d_{ij}(t)} + \frac{urgency}{\sqrt{\Delta t} + d_{ij}(t) + 1} \quad (2)$$

where $d_{ij}(t)$ - Euclidean distance between clients i and j ; $urgency$ - time-based coefficient of influence of the time window. The bigger coefficient is the bigger constraint of the time windows Δt - difference in time between time remaining until the end of the time window of the client i and current time.

D. System initialization

1) *Pheromone initialization*: During the analysis stage instead of the typical pheromone initialization function was utilized equation (3) that takes into consideration time window constraint. This equation assigns higher priority, namely higher pheromone intensity to paths leading not only to clients positioned, but also to those that are available earlier. This approach allows to reduce potential delays caused by the

vehicle having to wait for the client's time window to begin before service can be provided:

$$f_{ij}^{(0)} = 1.0 + \frac{1}{d(i,j) + 1} + \frac{1}{|r_j - r_i| + 1}, \forall i \neq j \quad (3)$$

where $f_{ij}^{(0)}$ - initial intensity of pheromones on path between clients i and j ; $d(i,j)$ - Euclidean distance between clients i and j ; r_j - start of the time window of the client j .

2) *Ant colony initialization*: Ant colony is initialized at the depot. All vehicles start and finish their journey at the depot.

E. Pheromone trails

After careful consideration of problem's domain and some preliminary experiments we suggested next algorithm for refreshing pheromone trails (4):

- Pheromone Evaporation - lowering intensity of existing pheromone trails.
- Increasing the pheromone levels based on the analysis of previously traversed path by ant colony.
- Elite ant acknowledgment - reinforcing pheromone level along the best-known path found by the best-performing ant.

$$f_{ij}^{t+1} = f_{ij}^t \cdot (1 - \rho) + \sum_{k \in K} \left(\frac{1}{D_k^t} \cdot \mathbf{h}_{(i,j) \in r_k^t} \right) + \frac{\gamma}{D_{best}} \cdot \mathbf{h}_{(i,j) \in r_{best}} \quad (4)$$

where f_{ij}^t - level of pheromones on path between clients i and j in iteration t ; ρ - evaporation coefficient; K - set of ants representing ant colony; r_k^t - route traversed by ant k in iteration t ; D_k^t - total distance traveled by ant k in iteration t ; $\mathbf{h}_{(i,j) \in r} = 1$ if path ij is in traversed route r (otherwise $\mathbf{h}_{(i,j) \in r} = 0$); γ - elite coefficient for the best ant; D_{best} - total distance traveled by the best ant; r_{best} - best-known found route.

F. Local route optimization using 2-opt

For local improvement of the traversed route implementation of the 2-opt algorithm was used [6]. Main goal of utilizing this algorithm was to find crossing sections of the concrete route and then swap subsections to improve the distance (presented in Algorithm 2).

G. Neighborhood search using Tabu search

After all ants traversed all routes, the neighborhood search is activated that compares clients between routes and tries to swap them if it improves the overall distance. This approach allows for quick exploitation and exploration simultaneously saving the algorithm from stagnating too early — Algorithm 3.

H. Hyperparameters

Efficiency of the algorithm is directly dependent on values of defined hyperparameters. The set of experiments was conducted for every hyperparameter in order to determine the best results for the given value of hyperparameter. Every experiment was performed 3 times in order to eliminate accidental or unstable results. Here is the list of defined hyperparameters:

Algorithm 2 2-opt implementation**Require:** *route* {Route that ant traversed}**Ensure:** A locally optimized route

```

1: improved  $\leftarrow$  true
2: while improved = true do
3:   improved  $\leftarrow$  false
4:   for i  $\leftarrow$  1 to route.length - 2 do
5:     for j  $\leftarrow$  i + 2 to route.length - 1 do
6:       new_route  $\leftarrow$  concatenate(route[1 to i-1],
reverse(route[i to j-1]), route[j to route.length])
7:       old_dist  $\leftarrow$  distance(route[i-1], route[i]) +
distance(route[j-1], route[j])
8:       new_dist  $\leftarrow$  distance(new_route[i-1], new_route[i])
+ distance(new_route[j-1], new_route[j])
9:       if new_dist < old_dist then
10:         route  $\leftarrow$  new_route
11:         improved  $\leftarrow$  true
12:       end if
13:     end for
14:   end for
15: end while
16: return route

```

Algorithm 3 Neighborhood search**Require:** *routes*, *vehicle_capacity* {Routes that ants traversed and vehicle capacity}**Ensure:** Optimized routes

```

1: best_routes  $\leftarrow$  routes
2: best_distance  $\leftarrow$  sum(routes)
3: for i  $\leftarrow$  0 to routes.length do
4:   for j  $\leftarrow$  1 to routes[i].length-1 do
5:     for k  $\leftarrow$  0 to routes.length do
6:       if k = j then
7:         continue
8:       end if
9:       for pos  $\leftarrow$  1 to routes[k].length - 1 do
10:        new_routes  $\leftarrow$  routes
11:        customer  $\leftarrow$  new_routes[i].pop(j)
12:        new_routes[k].insert(pos, customer)
13:        if new_routes[k].total_demand  $\leq$ 
vehicle_capacity then
14:          new_distance  $\leftarrow$  sum(new_routes)
15:        end if
16:        if new_distance < best_distance then
17:          best_routes  $\leftarrow$  new_routes
18:          best_distance  $\leftarrow$  new_distance
19:        end if
20:      end for
21:    end for
22:  end for
23: end for
24: return best_routes

```

- **Amount of ants** - small amount of ants (less than 20) caused bad and unstable results, whereas excessive amount (greater than 30) did not introduced improvement. This phenomenon can be explained by greater interference of pheromone trails between ants then required. Hence, as experiments proved, 20 ants as an optimal compromise between quality of solution and time required for computations.
- **Amount of algorithm iterations** - as confirmed in practice, the best found solution stagnated before reaching 30th iteration. So greater amount of iterations unnecessarily increased time required for the algorithm to finish without potentially drastic improvement of the result.
- **Pheromone evaporation** - small level of evaporation (around 0.1) meant, that pheromone trails had too big of influence o later iterations and hindered exploration settling in local minimum. On the other hand too great intensity of evaporation even though sometimes improved results, but they were unstable. On a balance, medium intensity of evaporation of around 0.5-0.6 lead to balance between exploration and exploitation even on big distances between clients.
- **Influence α of pheromone component** - pheromone level coefficient should be big enough for effective exploitation (around 0.5), however excessive value greater then 0.6 destabilizes solution
- **Influence of heuristic component** - heuristic component was implemented considering aspects of the studied problem, therefore this coefficient should be around 1.4-2.0 which produces the best result which was confirmed experimentally
- **Urgency coefficient** - Total neglecting of time window constraint as was expected complicates search of optimal routes, however setting the coefficient greater then 2 does not improve results.
- **Maximum amount of tabu search iterations** - The greater amount of iterations of tabu search the better results are. Nevertheless, on the downside time required for the algorithm to finish also grows drastically. Hence, 30 iterations allows for optimal results without requiring such big amount of time.

IV. EXPERIMENTAL EVALUATION AND RESULTS

All computations were carried out using the Google Colab platform [24], which provided a cloud-based virtual machine environment. The hardware configuration allocated to the runtime included a dual-core Intel(R) Xeon(R) CPU running at 2.20GHz and 12 GiB of available RAM. This environment ensured a consistent and reproducible execution setting for all test instances.

The use of Google Colab allowed for efficient prototyping and rapid experimentation without the need for dedicated local computational infrastructure. Furthermore, the platform's support for Python-based scientific libraries and its seamless integration with Jupyter notebooks greatly facilitated the de-

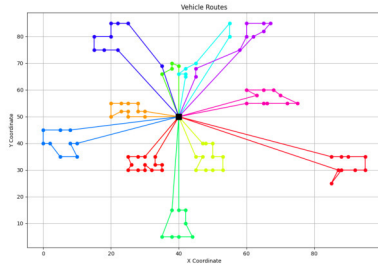


Fig. 2. Visualization of the computed vehicle routes for the VRPTW instance C101

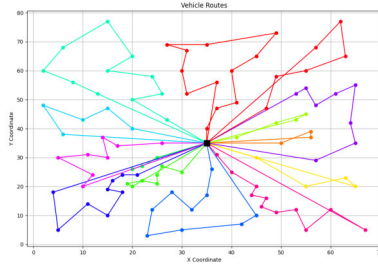


Fig. 3. Visualization of the computed vehicle routes for the VRPTW instance R101

velopment, debugging, and performance analysis of the hybrid algorithm.

While the hardware provided by Google Colab may not match the performance of high-end servers or GPU-accelerated environments, it is sufficiently powerful for evaluating medium-scale instances of the vehicle routing problem. The reported computation times in Table I reflect the algorithm's performance under these standard conditions, offering practical insight into its applicability for real-world use cases on accessible computing resources.

A. Dataset

Algorithm was tested using benchmark created by Solomon [25]. Dataset contains 56 sets of clients which are divided in 6 groups. Letter in the group name means density

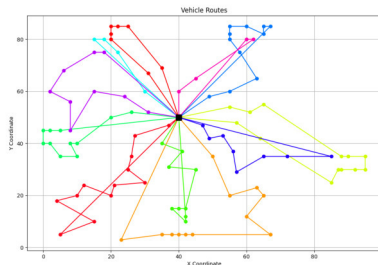


Fig. 4. Visualization of the computed vehicle routes for the VRPTW instance RC101

TABLE I
AVERAGE RESULTS FOR EACH GROUP

Client group	Distance	Vehicles	Time (min)
R1	1062.91	11.72	3.61
R2	1044.70	8.85	1.73
C1	1072.88	11.93	2.52
C2	771.07	5.29	2.96
RC1	1254.60	11.92	3.87
RC2	1004.34	7.25	3.09

of the client set ('R' - sparse density, 'C' - clustered clients and 'RC' is a mix of both of them) and the first number means the travel time type ('1' - short scheduling horizon and '2' - long scheduling horizon). Each client set contains 100 clients. Vehicle has constant capacity which varies between groups of client sets (C1*, R1* and RC1* - 200 vehicles; C2* - 700 vehicles; R2* and RC2* - 1000 vehicles). Furthermore, each client set has a depot from which every route should start and should finish. Each client has their demand which should be fulfilled fully by vehicle. Additionally clients have time constraints, namely each client has the time period during which he can be served by vehicle. Moreover each client has its service time which required to serve that client. Number of vehicles required to serve the client is not set so together with the total traversed distance is the subject for optimization. In our algorithm we focused on optimizing the total distance and the number of required vehicles was secondary.

B. Experiments

For final results each set of 100 clients was tested for 3 times which eliminated accidental results and ensured stability

In order to evaluate the performance of the proposed hybrid algorithm, a series of computational experiments were conducted on benchmark instances with varying numbers of customers, time window constraints, and vehicle capacities. The results of these experiments are summarized in Table I.

Table I presents not only the total minimum distance achieved for each problem instance and the number of vehicles required to serve all customers, but also the total computational time measured in minutes. These metrics allow for a comprehensive assessment of both the solution quality and the efficiency of the algorithm.

The total minimum distance reflects the main optimization objective and serves as a primary indicator of route efficiency. The number of vehicles used corresponds to the operational feasibility and reflects how well the algorithm can consolidate deliveries. Finally, the computation time provides insight into the algorithm's scalability and runtime behavior across different instance sizes.

This experimental setup allows for both qualitative and quantitative comparison of the algorithm's performance. Additionally, the data collected enables further analysis of the trade-offs between solution quality and computation time under various parameter settings.

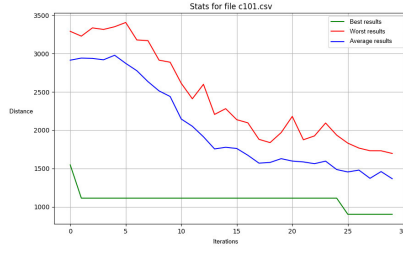


Fig. 5. Best, worst and average results in iteration receipts for instance C101

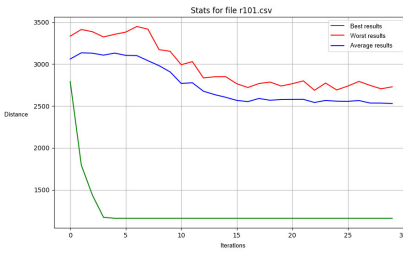


Fig. 6. Best, worst and average results in iteration receipts for instance R101

V. EVALUATION AGAINST STANDARD BENCHMARKS

For each set, drawings were generated showing the routes of individual vehicles and statistics were presented showing the best, worst and average route searches in individual iterations.

The diagram presents the resulting vehicle routes generated by the proposed hybrid algorithm for the benchmark instance C101 (2). Each colored line represents the route of a single vehicle, starting and ending at the central depot (marked with a black square). Individual customer locations are depicted as dots, and the connected paths illustrate the order of service while respecting both vehicle capacity and time window constraints. Similar diagrams were created for other benchmark instances to validate the robustness and adaptability of the proposed solution across various configurations of customer distribution and time constraints (f.e. Fig. 3 and Fig. 4).

In Fig. 5, the green line represents the best result obtained (minimum route length), the red line represents the worst, and the blue line represents the average value obtained in a given iteration.

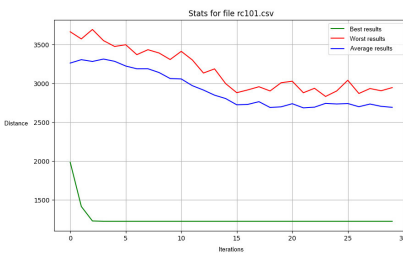


Fig. 7. Best, worst and average results in iteration receipts for instance RC101

A clear trend of improving average results and gradual stabilization of the values of the best solutions can be observed, which indicates effective exploration of the solution space by the algorithm. Significant quality improvement occurs already in the initial iterations, and further calculations allow for successive refinement and local optimization of solutions. A sudden drop in the best result after about 25 iterations suggests effective operation of the tabu search component or local optimization (2-opt), leading to significant improvement of previously found routes.

Analogous graphs were generated for the remaining test instances, which allows for analysis of the stability and repeatability of the proposed approach (f.e. Fig. 6 and Fig. 7).

TABLE II
RELATIVE EFFICIENCY IMPROVEMENT COMPARING TO PREVIOUSLY KNOWN RESULTS

File	Best Known Distance	Distance obtained	Relative efficiency(%)
r101	1650.80	1011.41	38.73
r102	1486.12	1054.03	29.08
r103	1292.68	1029.14	20.39
r104	1007.31	1006.33	0.10
r105	1377.11	975.83	29.14
r106	1252.03	993.79	20.63
r107	1104.66	1060.89	4.00
r109	1194.73	1075.12	10.01
r110	1118.84	992.36	11.30
r111	1096.72	1062.86	3.09
r201	1252.37	1029.12	17.83
r202	1192.70	969.44	18.65
rc101	1696.95	1189.84	29.88
rc102	1554.75	1295.17	16.70
rc103	1261.67	1205.30	4.47
rc105	1629.44	1240.40	23.88
rc106	1424.73	1265.33	11.19
rc107	1230.48	1218.62	0.96
rc201	1406.94	964.95	31.42
rc202	1365.65	990.67	27.46
rc203	1049.62	918.56	12.49
rc205	1297.65	1089.39	16.05
rc206	1146.32	937.28	18.24
rc207	1061.14	885.42	16.56

In Table II, the relative efficiency improvement achieved by the proposed hybrid algorithm is presented in comparison with the best-known solutions reported in [26]. The evaluation was conducted using the efficiency formula defined in equation (5), which quantifies the relative gain by measuring how much shorter the total distance of the obtained solution is compared to the reference result.

$$E_i = \frac{x_i - x_0}{x_0} \cdot 100\% \quad (5)$$

where E_i - relative efficiency of our solution; x_i - distance retrieved as a result of our solution; x_0 - distance for best known solution.

The suggested approach not only outperformed previously known results in the majority of benchmark instances, but in some cases achieved drastic improvements-up to approximately 30% reduction in total distance. Such a substantial gain is particularly impressive given that it was achieved

without significantly increasing computation time. In fact, the algorithm maintained practical execution times, making it suitable for applications where responsiveness is critical.

These results underline the robustness and effectiveness of the hybrid approach, especially in scenarios with tight time windows and large numbers of customers. The combined use of Ant Colony Optimization for initial solution generation, 2-opt local search for intra-route optimization, and Tabu Search for escaping local minima, proved particularly effective in navigating complex solution landscapes.

Importantly, the efficiency gains translate directly into more resource-effective planning-fewer vehicles and less travel distance, while respecting time constraints. This makes the proposed method highly valuable in real-time logistics and transportation scenarios, where computational speed and solution quality must go hand-in-hand. The consistent improvements across problem instances confirm that the developed algorithm is not only scalable and adaptive, but also practical for deployment in real-world operations.

VI. CONCLUSION

In this study, a novel hybrid algorithm combining Ant Colony Optimization, 2-opt local search, and Tabu Search has been proposed for solving the Vehicle Routing Problem with Time Windows. The algorithm effectively integrates global exploration with local refinement and strategic escape from local optima, resulting in a robust and adaptive solution method.

The experimental results confirmed that the proposed method significantly improves upon previously known solutions in terms of total route distance, with some instances showing improvements of up to 30%. Importantly, these improvements were achieved without incurring substantial computational cost, demonstrating the algorithm's suitability for real-time and large-scale applications.

The results also indicate that the hybrid approach ensures a better balance between intensification and diversification in the search process. This leads to high-quality solutions that respect all problem constraints, including capacity and time windows, while optimizing the number of vehicles and total travel distance.

Future work may involve extending the algorithm to dynamic or stochastic versions of the VRP, testing its performance on real-world datasets, and incorporating learning-based mechanisms to further enhance its adaptability and efficiency.

REFERENCES

- [1] A. Schirjver, *On the history of combinatorial optimization (till 1960)*. In: K. Aardal, G.L. Nemhauser, R. Weismantel (Eds.), *Handbook of Discrete Optimization*, 2005, Amsterdam.
- [2] C. Rego, D. Gamboa, F. Glover, C. Osterman, *Traveling salesman problem heuristics: leading methods, implementations and latest advances*, European Journal of Operational Research, 2014, vol. 211 (3).
- [3] G.B. Dantzig, J.H. Ramser, *The Truck Dispatching Problem*, 1959, Management Science, vol. 6 (1).
- [4] G. Ghiani, G. Laporte, R. Musmanno, *Introduction to Logistics Systems Management (2nd ed.)*, 2013, Wiley.
- [5] P. Toth, D. Vigo, *The Vehicle Routing Problem. Monographs on Discrete Mathematics and Applications*, 2001, SIAM, Philadelphia.
- [6] G. A. Croes, *A method for solving traveling-salesman problems*, 1958, Operations Research, vol. 6(6), pp. 791–812, <https://doi.org/10.1287/opre.6.6.791>
- [7] S. Lin, *Computer solutions of the traveling salesman problem*, 1965, Bell System Technical Journal, vol. 44(10), pp. 2245–2269, [urlhttps://doi.org/10.1002/j.1538-7305.1965.tb04146.x](https://doi.org/10.1002/j.1538-7305.1965.tb04146.x).
- [8] N. Christofides, A. Mingozzi, P. Toth, *The vehicle routing problem*, In *Combinatorial Optimization*, 1979, Wiley, pp. 315–338.
- [9] F. Uddin, N. Riaz, A. Manan, I. Mahmood, O.-Y. Song, A.J. Malik, A.A. Abbasi, *An Improvement to the 2-Opt Heuristic Algorithm for Approximation of Optimal TSP Tour*, 2023, Applied Sciences, 13(12):7339, [urlhttps://doi.org/10.3390/app13127339](https://doi.org/10.3390/app13127339)
- [10] M. Dorigo, V. Maniezzo, A. Colnari, *Ant system: Optimization by a colony of cooperating agents*, (1996, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 26(1), pp. 29–41, <https://doi.org/10.1109/3477.484436>
- [11] K. F. Doerner, R.F. Hartl, M. Reimann, *Metaheuristics for the vehicle routing problem with loading constraints*, 2006, Networks, vol. 49(4), pp. 294–307.
- [12] J. Ochelska-Mierzejewska, *Ant Colony Optimization Algorithm for Split Delivery Vehicle Routing Problem*, In International Conference on Advanced Information Networking and Applications (was ICOIN), 2020, https://link.springer.com/chapter/10.1007/978-3-030-44041-1_67.
- [13] F. Glover, *Future paths for integer programming and links to artificial intelligence*, 1986, Computers and Operations Research, vol. 13(5), pp. 533–549, [https://doi.org/10.1016/0305-0548\(86\)90048-1](https://doi.org/10.1016/0305-0548(86)90048-1)
- [14] J.-F. Cordeau, M. Gendreau, G. Laporte, J.-Y. Potvin, F. Semet, *A guide to vehicle routing heuristics*, 2001, Journal of the Operational Research Society, vol. 53(5), pp. 512–522, [urlhttps://doi.org/10.1057/palgrave.jors.2601319](https://doi.org/10.1057/palgrave.jors.2601319)
- [15] M. Gendreau, A. Hertz, G. Laporte, *A tabu search heuristic for the vehicle routing problem*, 1994, Management Science, vol. 40(10), pp. 1276–1290, [urlhttps://doi.org/10.1287/mnsc.40.10.1276](https://doi.org/10.1287/mnsc.40.10.1276)
- [16] J. Ochelska-Mierzejewska, *Tabu Search Algorithm for Vehicle Routing Problem with Time Windows*, 2020, https://link.springer.com/chapter/10.1007/978-3-030-34706-2_7. DOI: 10.1007/978-3-030-34706-2_7.
- [17] N. Paisarnvirosrak, P. Rungrueang, *Firefly Algorithm with Tabu Search to Solve the Vehicle Routing Problem with Minimized Fuel Emissions: Case Study of Canned Fruits Transport*, 2023, LOGI – Scientific Journal on Transport and Logistics, vol. 14(1), pp. 263–274, [urlhttps://doi.org/10.2478/logi-2023-0024](https://doi.org/10.2478/logi-2023-0024).
- [18] X. Ma, C. Liu, *Improved Ant Colony Algorithm for the Split Delivery Vehicle Routing Problem*, 2024, Applied Science, vol. 14(5090), <https://doi.org/10.3390/app14125090>
- [19] L.M. Gambardella, É.D. Taillard, G. Agazzi, *MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows*, In D. Corne, M. Dorigo, F. Glover (Eds.), *New ideas in optimization*, 1999, McGraw-Hill, pp. 63–76.
- [20] M. Tadaros, N.A. Kyriakakis, *A Hybrid Clustered Ant Colony Optimization Approach for the Hierarchical Multi-Switch Multi-Echelon Vehicle Routing Problem with Service Times*, 2024, Computers & Industrial Engineering, <https://diva-portal.org/smash/get/diva2:1802273/FULLTEXT01.pdf>
- [21] J.B. Holliday, E. Osaba, K. Luu, *An Advanced Hybrid Quantum Tabu Search Approach to Vehicle Routing Problem*, 2025, <https://arxiv.org/pdf/2501.12652v1>
- [22] Z. Zheng, B. Ji, S.S. Yu, *An Adaptive Tabu Search Algorithm for Solving the Two-Dimensional Loading Constrained Vehicle Routing Problem with Stochastic Customers*, 2023, Sustainability, vol. 15(2), 1741, <https://www.mdpi.com/2071-1050/15/2/1741>
- [23] Y. Liu, Z. Wang, J. Liu, *A Quick Pheromone Matrix Adaptation Ant Colony Optimization for Dynamic Customers in the Vehicle Routing Problem*, 2024, vol. 12(7), 1167, <https://doi.org/10.3390/jmse12071167>
- [24] Google Colaboratory, 2024, <https://colab.research.google.com/>
- [25] M. Solomon, *Solomon VRPTW Benchmark*, 1987, <http://w.cba.neu.edu/~msolomon/problems.htm>
- [26] Top, VRPTW for 100 customers, 2008, <https://www.sintef.no/projectweb/top/vrptw/100-customers/>