symmetry

MDPI

Article

A Hybrid Algorithm for the Heterogeneous Fixed Fleet Open Vehicle Routing Problem with Time Windows

Zakir Hussain Ahmed [1,*] and Majid Yousefikhoshbakht [2]

[1] Department of Mathematics and Statistics, College of Science, Imam Mohammad Ibn Saud Islamic University (IMSIU), Riyadh 11432, Saudi Arabia
[2] Department of Mathematics, Faculty of Sciences, Bu-Ali Sina University, Hamedan 6517838695, Iran
* Correspondence: zaahmed@imamu.edu.sa

Abstract: Nowadays, using a rental fleet to transport goods for delivering to customers at a particular time frame is very important in the services and industry. That is why, in this study, we consider the heterogeneous fixed fleet open vehicle routing problem with time windows, which is one version of the vehicle routing problem with time windows. The problem has not attracted attention so much in the operational research literature than the usual vehicle routing problem. The problem consists of determining the minimum cost routes for a fleet of a fixed number of vehicles with various capacities in order to fulfil the demands of the customer population. Moreover, the vehicles start at the headquarters and terminate at one of the customers. In this study, we introduce a mixed integer programming model and then integrate an exact algorithm to solve this model. Furthermore, a hybrid algorithm (HA) based on modified rank-based ant system is developed and then its efficiency is compared with the exact method and some metaheuristic methods on some standard instances in literature. The results proved the effectiveness of our proposed HA.

Keywords: open vehicle routing problem; heterogeneous fixed fleet; time windows; mixed integer linear programming; rank-based ant system

Citation: Ahmed, Z.H.; Yousefikhoshbakht, M. A Hybrid Algorithm for the Heterogeneous Fixed Fleet Open Vehicle Routing Problem with Time Windows. *Symmetry* **2023**, *15*, 486. https://doi.org/10.3390/sym15020486

Academic Editors: Wenyin Gong and Libao Deng

Received: 25 January 2023
Revised: 2 February 2023
Accepted: 9 February 2023
Published: 12 February 2023

1. Introduction

Growing population and urban expansion have produced some problems which are attached to our daily life and are essential to solve. In this problem, type, logistics and supply chain have considerable places because they deal with the created problems from the starting of the production of a goods till the delivery of it to the customer. In this phase, which exists in all commodities, the cost of production and the final price of the goods are so important. Furthermore, the satisfactory price is an important factor of a goods which has a significant impact on its sale. On the other hand, the lower the production cost of a goods, the cheaper the company is in the production of goods and delivering services to customers. Thus, decreasing the production cost is an important factor for more profit [1].

In the vehicle routing problem (VRP), vehicles in the depot must serve some customers (n) distributed around the depot such that, firstly, the demands of the customers allotted to each vehicle must not surpass the capacity of the vehicle (Q), and secondly, every customer is only met by only one vehicle. In addition, all vehicles are similar having finite capacity and only one type of goods must deliver to the customers [2]. Figure 1, for example, shows a feasible solution toa VRP instance. In this figure, there are 50 customers and 9 vehicles, and the capacities of all customers assigned to each vehicle are less than the Q.

This problem performs an important role in the distribution management as well as in the supply chain such that over the past half century several researchers have addressed it and its expansion. For example, these expansions include the open VRP (OVRP) [1], the green VRP (VRPG) [3], the VRP with drones (VRPD) [4], the VRP with the time window (VRPTW) [5], the VRP with pickup and delivery (VRPPD) [6], the distance constrained VRP (DVRP) [7].

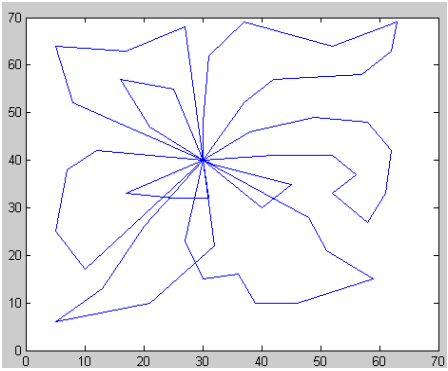

Figure 1. A feasible solution for VRP.

The specialization of jobs and increasing competition for sales and services have caused the time interval of science production and its use in various sciences, especially transportation, to be very low. Alternatively, nowadays, some production companies neither advertise nor transport nor sell their goods but leave it to some dedicated enterprises. Thus, outsourcing to dedicated enterprises has become commonplace and they invest large amounts on transportation deals for distributing their goods. Therefore, the transportation enterprises bring their vehicles to the depot, load the goods, and deliver them to the customers. The vehicles in these matters, known as leased, will no longer come back to the depot after performing their duties and end their path at the last customer. This problem is known as the OVRP where the Hamiltonian route of each vehicle is open, in contrast with VRP, which is closed [8].

The OVRP is extensively used in today's world, for example, the transportation of postal goods, the delivery of newspapers to different houses, etc. More applications can be found in [9]. This problem also does not have a long-sighted record, in contrast with the VRP, and it was presented first in [10]. The authors introduced a first clustering second routing technique. Of course, since this problem is highly regarded by investigators and several procedures were introduced to solve it, such as general variable neighborhood algorithm [11], genetic algorithm [12], and iterated local search [13] and so on.

In everyday transportation issues, due to variations in the production types or the type of delivery to customers across manufacturing firms, these firms close their contracts to distribute products with several service companies that have similar vehicles. Alternatively, production companies utilize numerous fleets of vehicles to disseminate their products so that they can be delivered to customers at lower costs. Therefore, the heterogeneous fleet VRP (HFVRP) has many service and industrial applications that attracted the attention of many investigators such as Gendreau et al. [14], Li et al. [15], Li et al. [16], Tarantilis et al. [17], and Soman and Patil [18]. In this type of problem, several characteristics are considered as follows:

- The fleet is heterogeneous and has various capacities.
- Limited and certain number of vehicles.
- Fixed cost to use each vehicle.
- Non-fixed cost to use any vehicle per distance unit.

Chu [19], for example, examined the heterogeneous fixed fleet VRP (HFFVRP) in 2005. In that study, it was considered that in addition to a fixed heterogeneous fleet of units, a number of customers were serviced by rented companies if needed. Moreover, a mathematical model and an innovative method for selecting customers who should have been assigned to rented trucks are presented. Moreover, in [20], it states to possibly rent a collection. In this algorithm, first, customers who have to be serviced by rented trucks are selected and then the first solution is made and improved by multiple improved heuristic algorithms. Next, the second solution will be made and re-improved. Among the two improved solutions, one of them is chosen as the ultimate solution that has less cost. In

addition, Prins [21] used two proposed algorithms to solve two HFFVRP instances by determining the fleet size and mix VRP (FSMVRP). In their proposed combined genetic algorithm, the obtained response is improved after intersection and mutation by using a local search procedure. Schmida et al. [22] investigated the issue of ready-mixed concrete distribution. In this problem, the purpose of distribution of concrete made by a fixed fleet of vehicles was different. This paper used a detailed method that was guided by the variable large neighborhood search method. The method was tested on 20 instances of different sizes, which were based on the data of a construction materials distribution company. The reported results showed the usefulness of the method. Furthermore, Euchi and Chabchoub [23] addressed the HFFVRP and presented a meta-based algorithm that combined a tabu search algorithm and adaptive memory to solve this problem. This method was tested on the instances presented in [24]. In 2011, Brandao [24] examined the HFFVRP and presented a banned search algorithm for solving the problem. This algorithm was based on the algorithm presented for FSMVRP, which was tested on the instances presented in [25] and a new set of instances with their results are compared with other algorithms.

If the HFFVRP assumes that the vehicles are rented, then the heterogeneous fixed fleet OVRP (HFFOVRP) arises, which has applications in logistics and transportation problems [26]. The problem is one of the newest versions of the VRP. This is a practical and real issue today because with the expertise of jobs, most of their manufacturing companies no longer market and deliver their goods and leave it to transportation companies. In this case, the stationary trucks move from their location and go to the store of the producer. Next, after loading and delivering the goods to the customers, these devices will no longer return to the warehouse and each of them will finish its movement to the last customer for the delivering the goods. Alternatively, because of the decline and shortage of proper productivity, it is well for these companies to utilize vehicles with variant capacities due to different economic conditions and transportation costs. Therefore, a heterogeneous fleet of vehicles can be used for transporting goods, in that case the transportation cost could be less. Furthermore, for many goods, it needs to deliver to customers within a specific period of time. Therefore, if time windows are applied to problem, the problem becomes HFFOVRP with time windows (HFFOVRPTW) [27]. This problem arises especially in perishable goods which must be delivered to customers at a specific time, otherwise the company cannot make a profit from routing. In this problem, the aim is to find a combination of vehicles to provide service to customers such that the following conditions are satisfied.

- Every vehicle begins its path from the depot and does not return to it after finishing the path.
- All customers' requests are met, and each customer receives the goods within a certain period of time.
- The amount of request for each customer's goods is delivered only by one of the vehicles.
- The amount of customer requests allocated to each vehicle is less than its capacity.
- For each type of vehicle, the number used does not increase the number of predetermined vehicles. Thus, the number of vehicles utilized in a particular problem instance must be less than the predefined number.
- The time for every vehicle is less than the upper limit time for that vehicle.
- Only available vehicles should be used in customer service.

The HFFOVRPTW combines the OVRPTW and the HFFVRPTW which is proposed by decreasing restrictions (Figure 2). For instance, if, no matter what the time window, it is assumed that the vehicles have the same capacity, then it becomes the OVRPTW. Of course, in the simple case of the OVRPTW, there is no fixed and variable cost for the usage of the vehicles, for which the fixed cost is equal to zero and the variable cost is equal to one. On the other hand, if for the HFFOVRPTW, the Hamilton route condition is converted to Hamilton cycle and time windows are not considered, the HFFVRP will be raised. This conversion is possible by assuming no expense between each end node to the depot and

each time window belongs to the real number set. Since the HFFVRP and OVRPTW are NP-hard [28], the HFFOVRPTW is also NP-hard.

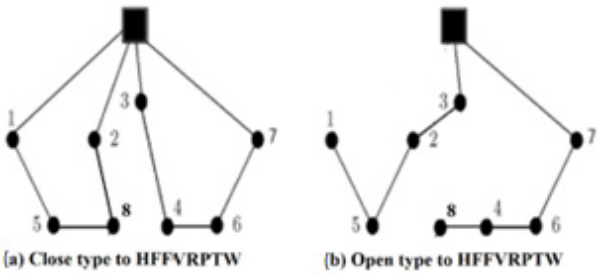

Figure 2. Two feasible solutions for the open and close type of the HFFVRPTW.

In our study, firstly, a mixed integer linear programming model (MILPM) is proposed for our problem, and then a problem instance is solved by an exact approach (AIMMS software version 12.3) for the model verification. Moreover, the Taguchi algorithm is used for tuning parameters and a hybrid algorithm (HA) based on rank-based ant system (RBAS) is developed and applied on some problem instances. Finally, the efficiency of HA is examined by comparing with the exact method and some metaheuristic algorithms. The experimental results demonstrate that our proposed HA has performed very well and produced very high-quality solutions within an acceptable CPU time.

In this study, a MILPM for the HFFOVRP is proposed in Section 2 and in Section 3, the proposed HA is described. In Section 4, several standard instances are considered and solved using exact methods, HA, and some metaheuristic methods. Finally, future conclusions and orientations are presented in Section 5.

2. The Proposed MILPM

One of the most important ways to solve a problem is to use its model, which can express all its data and limitations. Next, using different algorithms, its decision variables can be checked and obtained [29–33]. As a graph, our problem can be stated as G(V, A), where V = {0, 1, . . . , *n*} is the node set and the $A = \{(i, j) \,|i, j \in V \, and \, i \neq j\}$ is the set of edges connecting the nodes. If G is not complete, each unavailable edges are assigned an infinite cost. The 'node 0' denotes the depot, the remaining *n* nodes denote the customers having the same demand q_i. It also corresponds c_{ij} to each arc in an Euclidean distance, where $c_{ii} = 0$ and $c_{ij} = c_{ji}$, for $0 \leq i, j \leq n$. Further, a fleet of K various vehicle types is situated at the depot, such that every kth vehicle has capacity Q_k, a fixed cost f_k (when the vehicle is utilized on request) and the cost varies α_k. Additionally, the number n_k of the device is existing in the fleet. The variable cost is the distance travelled by a vehicle, such that the navigation cost for each arc (*i*,*j*) by the *k*th vehicle type is equal to $c_{ij}^k = c_{ij} \times \alpha_k$. Therefore, in our problem, there is a symmetric cost matrix K and the product amount that the kth vehicle brings from node *i* to *j* node is given by y_{ij}^k when traveling. Every customer has to be provided a service within a prefixed time window that is restricted to the quickest begin time e_i and the end service time l_i. Moreover, if the *k*th vehicle reaches a customer after the end service time or if the vehicle goes to a customer before its begin time, it has to wait an additional waiting time. Similarly, for every *k*th vehicle type, t_i^k is the arrival time at node *i* and f_i^k is a service time for that node. If the vehicles end their routes at the maximum permissible travel time r, then the objective is to find the minimum cost routes for vehicles in which all constraints including travel time, capacity, and time windows are determined for the vehicles.

$$\text{Min} \sum_{k=1}^{K} f_k \sum_{j=1}^{n} x_{0j}^k + \sum_{k=1}^{K} \sum_{i=0}^{n} \sum_{j=0}^{n} c_{ij}^k x_{ij}^k \quad (1)$$

subjest to,

$$\sum_{k=1}^{K}\sum_{i=0}^{n} x_{ij}^{k} = 1 \ \forall j = 1, 2, \ldots, n \tag{2}$$

$$\sum_{k=1}^{K}\sum_{j=1}^{n} x_{ij}^{k} \leq 1 \ \forall i = 1, 2, \ldots, n \tag{3}$$

$$0 \leq \sum_{i=0}^{n} x_{ij}^{k} - \sum_{i=0}^{n} x_{ji}^{k} \leq 1 \ \forall j = 1, 2, \ldots, n, \ \forall k = 1, 2, \ldots, K \tag{4}$$

$$\sum_{j=1}^{n} x_{0j}^{k} \leq n_k, \ \forall k = 1, 2, \ldots, K \tag{5}$$

$$\sum_{k=1}^{K}\sum_{i=0}^{n} y_{ij}^{k} - \sum_{k=1}^{K}\sum_{i=0}^{n} y_{ji}^{k} = q_j, \forall j = 1, 2, \ldots, n \tag{6}$$

$$q_j x_{ij}^{k} \leq y_{ij}^{k} \leq (Q_k - q_i) x_{ij}^{k} \ \forall i, j = 0, 1, \ldots, n, \ i \neq j, \ \forall k = 1, 2, \ldots, K \tag{7}$$

$$\sum_{j=1}^{n} x_{i0}^{k} = 0, \ \forall k = 1, 2, \ldots, K \tag{8}$$

$$\sum_{i=1}^{n}\sum_{j=1, j\neq i}^{n} x_{ij}^{k}(t_{ij}^{k} + f_i^{k} + w_i^{k}) \leq r \quad for\, k \in \{1, 2, \ldots, K\} \tag{9}$$

$$t_0^{k} = w_0^{k} = f_0^{k} = 0 \tag{10}$$

$$\sum_{i=0, i\neq j}^{n} x_{ij}^{k}(t_i^{k} + t_{ij}^{k} + f_i^{k} + w_i^{k}) \leq t_j^{k} \ for\, j \in \{1, 2, \ldots, n\} \, and \, k \in \{1, 2, \ldots, K\} \tag{11}$$

$$e_i \leq (t_i^{k} + w_i^{k}) \leq l_i \ for\, i \in \{1, 2, \ldots, n\} \, and \, k \in \{1, 2, \ldots, K\} \tag{12}$$

$$y_{ij}^{k} \geq 0, x_{ij}^{k} \in \{0, 1\} \ \forall i, j = 0, 1, \ldots, n, \ \forall k = 1, 2, \ldots, K \tag{13}$$

In this model, the objective function (1) indicates the cost function which is composed of two terms. Here, the first term indicates the total fixed cost, and the second term indicates the total variable cost of the vehicles. In this model, if decision variables are not taken into consideration, which are continuous and are used in restrictions that prevent unauthorized loading of vehicles, then there are as many variables as the decision to make that after solving the model, n numbers will have a value of one and the remaining of them will get zero. Constraint (2) ensures that exactly one vehicle is arrived at each customer, while relationship (3) points out that at most one vehicle is withdrawn from every customer. The reason that the restrictions (2) and (3) are considered differently is that exactly one vehicle is arrived at each customer while the vehicles' routes can end at any other customer and no vehicles leave them. In addition, relationship (4) causes any vehicle that enters a customer to leave if the node is not terminal. For these customers, the difference in (4) is zero, but if the customers are at the termination of their routes, then the number of incoming vehicles is one unit higher than the number of outgoing vehicles. Constraint (5) suggests that the number of kth type vehicles must be maximum n_k. Here, the number of vehicles utilized in every type can be strictly less than n_k. Relationship (6) ensures that each customer's request is fully met. Alternatively, the needs of each customer must be talked only at one meeting that will be fulfilled by only one vehicle. The relationship (7) indicates that the capacity limit for every vehicle is satisfied. Moreover, the relationship (8) causes no path to the depot. In fact, this limitation, along with (2), (3), and (4) guarantee the openness of the routes. Constraint (9) shows the restriction about the maximum journey time. Finally, limitations (10)–(12) apply time windows. Finally, restriction (13) defines the variable ranges.

3. The Proposed Algorithm

Ant colony optimization (ACO) is an important metaheuristic method [34,35] introduced as a tool for finding solution to the OVRP [8]. This algorithm is a successful example of intelligent group systems, in which each factor performs a simple action, but doing this simple action, in general, solves NP-hard combinatorial optimization problems. In this algorithm, the main task of each artificial ant, such as its natural counterpart, is to obtain the minimum cost path between a pair of vertices in a graph in which this problem is properly moped on it. Therefore, the problem is converted to subproblems in which artificial ants have a duty to choose the next node depending on distance to the next node and the pheromone.

Despite the relatively good solutions obtained by ACO for the TSP, this algorithm did not show much efficiency in the larger sized problem instances compared to other algorithms. For this reason, much research has been conducted to improve this algorithm to solve the problem. The RBAS is a revised version of the ACO. In this algorithm, similar to the ACO algorithm, the decision rule for the *k-ant* located in node i, which wants to select one of the nodes from the set of unsealed nodes, is obtained from Formula (14). Here τ_{ij} represents the pheromone amount on the edge (i, j), while η_{ij} is the inverse distance between the two nodes i and j. Of course, both of them have fixed power α and β which is changeable by the user.

$$P_{ij}^{k} = \begin{cases} \frac{{\tau_{ij}}^{\alpha}{\eta_{ij}}^{\beta}}{\sum_{j \in N_i} {\tau_{ij}}^{\alpha}{\eta_{ij\beta}}} & if\ j \in N_i \\ 0 & if\ j \notin N_i \end{cases} \tag{14}$$

The RBAS improved AS by ranking the solutions obtained by ants. The nature of this method compared to other versions of the ant method is that the pheromone amount released by ants depends on the rank of ants for obtaining the solution, as showed in (15).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{\mu=0}^{\sigma-1} \Delta {\tau_{ij}}^{\mu}(t) \tag{15}$$

ρ : The evaporation rate parameter is in the domain [0, 1]. The evaporation reduces the pheromone on all edges with the same speed $0 \leq 1-\rho \leq 1$. Alternatively, at the end of each repetition of the algorithm, the remaining pheromones on all edges are reduced by coefficient $1-\rho$. This causes the new pheromone footprint to have a moderate weight between the previous pheromone amount left on the edges and the new pheromone amount in the current repetition.

$\tau_{ij}(t)$: The value of the pheromone on the edge (i, j) in repetition t.

$$\Delta {\tau_{ij}}^{\mu}(t) = \begin{cases} (\sigma-\mu)\cdot\frac{Q}{L^{\mu}(t)} & (i,j) \in S^{\mu} \\ 0 & (i,j) \notin S^{k} \end{cases} \tag{16}$$

μ : The variable represents the ranking index, which changes from zero to $\sigma-1$. In this variable, the best ant takes zero and the descending amount remains until the worst ant takes the value $\sigma-1$.

S^{μ} : The edges belong to an ant that has been ranked μ in terms of obtaining the best solution.

σ : The number of ants ranked and cast on the edges belonging to them is the pheromone.

Q: Any constant value given by the user.

$L^{\mu}(t)$: The length of the path obtained by the ant is μ.

Of course, in order to provide a more efficient method for the HFFOVRPTW, some modifications have been made to this algorithm as follows.

3.1. Selection Function

As stated in RBAS algorithm, each ant selects the next j node according to Formula (14), while in the proposed hybrid algorithm (HA) based on RBAS, a new transfer rule is proposed that focuses on both exploitation and exploration. For example, the movement from node i to node j, which is not met $j \in N_i$ and is carried out by ant k according to the passing rule in (17).

$$P_{ij}^{k} = \begin{cases} \frac{{\tau_{ij}}^{\alpha} {\eta_{ij}}^{\beta} (\frac{\pi - \theta_{ij}}{\pi})^{\delta}}{\sum_{j \in N_i} {\tau_{ij}}^{\alpha} {\eta_{ij}}^{\beta} (\frac{\pi - \theta_{ij}}{\pi})^{\delta}} & if\ j \in N_i \\ 0 & if\ j \notin N_i \end{cases} \quad (17)$$

Here, to select the first customer for each ant in the HFFOVRPTW, the value of $(\frac{\pi - \theta_{ij}}{\pi})^{\delta}$ is considered one, while for the remaining clients, the value θ_{ij} is equal to the angular value that the edge $(0, j)$ makes with the previous edge in the mentioned path, which is a number between 0 and 180 degrees. For example, in Figure 3, this angle is shown. In this form, the corresponding ant is located in node 1 and the previous edge in this path is the edge filed between the warehouse node (zero node) with node 1. Therefore, the desired angle is obtained from the intersection of mane (3, 0) with mane (1, 0). It is necessary to pay attention to the fact that in the formula presented $(\frac{\pi - \theta_{ij}}{\pi})^{\delta}$, if θ_{ij} is closer to zero, the more value will obtain. It should be noted that here δ is a similar to parameters α and β are determined by the user.

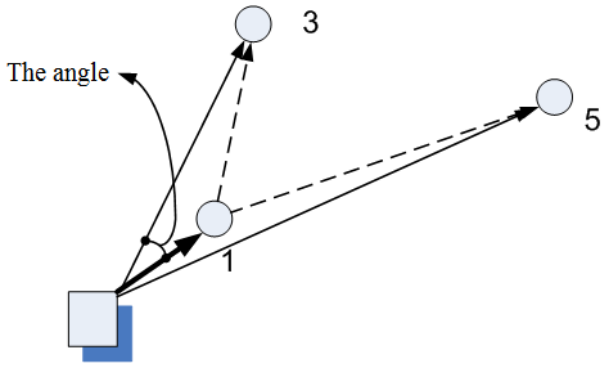

Figure 3. The angle θ_{1j} in Formula (14).

3.2. Updating Pheromone

The drawback of the RBAS algorithm is that in the processing only the best paths are processed in the current repetition of the pheromone. Therefore, the algorithm does not pay attention to the gained experience and does not consider the edges algorithm belonging to the best solutions ever obtained. Therefore, it is better that the algorithm, in addition to searching for new solutions, uses experience to create a good balance for using experience and searching for new solutions. It should be noted that if the algorithm only cares about experience (the best solutions ever obtained), then because of the random structure that the algorithm has, it is possible that the amount of the best solution per repetition will be worse than the previous iterations. Therefore, it is expected that in several limited repetitions, the best solution will not improve, and in spite of knowing that the solution may not be nice (particularly in the beginning of the method). However, in several repetitions it might be a pheromone consecutively. Therefore, in subsequent iterations, the algorithm may improve the solution quality by searching more, and this extra pheromone on an unsuitable solution causes the algorithm to lose time to change the search space without achieving a better solution. Therefore, σ in Formula (15) is an even value is considered and, in each repetition, the number value $\sigma/2$ of the best solutions in the current repetition and $\sigma/2$ from the best values has been obtained, ranked, and pheromone according to this formula.

3.3. Local Search Algorithms

For more improving of the solution algorithm, the hill climbing (HC) algorithm is adopted to improve the HA exploitation and applied to the feasible position created. After finding a new feasible solution, in this step, HC is applied to the solution created for improvement. In fact, the solution created as the input solution to this algorithm as a local search is given to increase the exploitation around the solution. This algorithm is a popular local search algorithm starting with an initial solution X. Next, it iteratively creates a neighbor solution X′ using a neighborhood structure. In HA, three neighborhoods, named relocate, exchange, and end-customer interchange, have been used to create a neighboring solution. Next, if the qualification of X′ is obtained well than X, replace X with X′. Otherwise, X′ will be deleted and a new iteration will begin. The search will be repeated until the stopping criterion is met. The hill climbing algorithm is shown in Algorithm 1.

Algorithm 1 Hill Climbing Algorithm

```
1.  //The objective is to obtain a minimum cost
2.  input X
3.  // This initial solution is obtained by another algorithm
4.  repeat
5.  Generate a neighborhood for X
6.     Select the best solution in the neighborhood and named X*
7.     if f(X) < f(X*)
8.        Break
9.     else
10.       X = X*
11.    end if
12. until (the stopping criterion is satisfied)
13. return X
```

HC generates a new position by using the three neighborhood structures, respectively, applied to the new position:

- Relocate the operator, which transfers a customer from one route to another.
- Exchange operator, which swaps two customers in different routes.
- End customer interchange operator, which swaps two end customers in different routes.

3.4. Stopping Condition

Since the proposed algorithm is metaheuristic, similar to other metaheuristic algorithms, the conditions for stopping the algorithm must be specified, otherwise the algorithm will run indefinitely. For HA, the final condition is to perform the main loop in the algorithm according to the number of customers. Alternatively, the algorithm will be completed until the number of iterations of the main loop reaches the number of customers.

3.5. The Algorithm

The proposed algorithm runs on the HFFOVRPTW is stated as: Let there are n set of ants where each set has m ants which are placed at the depot initially. Every ant creates a Hamiltonian path using the state transition rule repeatedly. While creating its path, it also modifies the pheromone amount of the traversed edges using the global updating rule. In other words, the moment every ant finished their paths, the amount $\sigma/2$ of each the best known and the current solutions are selected and their pheromones on the edges are updated using the global updating rule. In our proposed algorithm, ants are managed to create their open tours using Formula (17), by exploitation and exploration. The rules for updating pheromones are constructed to provide extra pheromone to edges that are supposed to be traversed by ants.

Most literature on the ACO suggests that a probable method to obtain good quality solutions is to incorporate a local search method for creating initial solutions. In our proposed algorithm, once a set obtains a solution, the best set's solution is further rectified using three local search algorithms. In this, the main point is that a relatively good quality solution might have a good possibility to lead to a global optimal solution. We first implement the displacement method to an ant, and then implement the exchange method to two ants. Finally, the end customer interchange operator is applied for more improvement. The resultant solution would be accepted if the leading tour would be better than the former solutions. The algorithm will be stopped when the iteration number is equal to number of nodes. The pseudocode of our proposed hybrid algorithm for the HFFOVRPTW is presented in Algorithm 2.

Algorithm 2 Our Proposed Hybrid Algorithm

```
Input all data
Initialize all parameters
while number of iteration is not equal to nodes
    for each from n colony
        for each ant in a colony
            Set all nodes as unvisited
            while number of unvisited nodes > 0 do
                if there is no feasible solution then
                    Go to 3
                end if
                Select a depot
                Compute ant's probability of going to unvisited nodes satisfied time window
                if there is no node to be selected then
                    Go to 8
                end if
                Select a node according to the probability
                if ant (load)+node (load) > ant (capacity) then
                    return to the depot and go to 7
                else
                    Visit the selected node
                end if
            end while
        end for
    end for
    Save the best solution if found
    Select σ/2 of the best known solutions and σ/2 of the best current solutions
    Evaporate pheromone trails
    Update pheromone trails
    Apply three local search algorithms
end while
return the best solution
```

4. Results and Discussion

Our proposed algorithm, HA, was encoded in MATLAB 2016 and all experiments were implemented on a laptop with core 3 2.6 GHZ and 4 GB RAM running Windows 7 Home Basic Operating system. In this section, the parameters' tuning is presented, and the results of the HA are shown.

4.1. Parameters Tuning

Metaheuristic algorithms can obtain very good solutions in a limited time because they use the concept of random to create solutions. In addition, in order to create this concept, the algorithm must use many parameters so that there are different ways to find

optimal solutions. Even to adjust these parameters, algorithms have been proposed that because in this paper, the solutions are presented at the acceptable time of the target, so it is tried to use an easier strategy to adjust the parameters in this section. For this goal, based on the results reported in other articles, the important parameters, including α, β, δ, σ, *Map* and ρ, of our proposed algorithm are considered and in Table 1, candidate values are shown for them. Now, at this stage, the a5 problem is considered as a candidate and has been tested with different parameters of this table. It should be noted that for each parameter setting, this problem is tested 10 times and the best results are reported in this table.

Table 1. Range of parameters of the proposed algorithm.

Parameters	Range	The Best Value
Alpha (α)	0.1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5	1.5
Beta (β)	1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5	1.5
Delta (δ)	1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5	4
Sigma (σ)	2, 4, 6, 8, 10, 12, 14, 16, 18	6
Map	0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9	0.5
Rho (ρ)	0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45	0.1

Since the algorithm quality is often lower than best known solution (BKS), then the gaps are positive. Conversely, value zero shows very good functioning of the algorithm for this parameter and is able to reach BKS.

By comparing the results in Table 1 for alpha, one can conclude that the gap change diagram is almost ascending based on its values. Alternatively, the higher the amount of alpha, the lower the solution quality. Thus, 1.5 for this parameter causes the importance of pheromone information not to have much effect on choosing a new path, especially at the beginning of the algorithm, which has not yet obtained good solutions to the problem. In addition, obtaining value 1.5 of beta has led the algorithm to seek more global searches in the initial iterations and prevent early convergence at a slow rate to increase the pheromone on the edges. The value of δ is obtained which shows the importance of this parameter compared to other parameters. Moreover, for the best solutions obtained in the current iteration and until now, the number of six has been obtained. This, while the neighboring algorithm searches for the best solutions, prevents early convergence to one of the once due to the updating of more of the solutions. The minimum amount of pheromone is also considered here and the value 0.5 is obtained through nine candidates. Finally, the amount of pheromone evaporation for this problem is 0.1, which reduces the amount of pheromone on all edges belonging to the problem by 0.1 per replication. It should be noted that high evaporation values cause the difference between the edges to increase rapidly, and the algorithm does not have enough time to search globally and to search in the problem space.

4.2. Computational Results

The numerical results are presented using three sets of instances existing in the literature. In the first set shown in Table 2, seven instances are considered as a1, a2, a3, a4, a5, a6, and a7. This category of instances includes customer numbers between 10 and 50 with specific demands that are scattered randomly nearby the source. Additionally, the fleet contains 4 to 18 various vehicles along with capacities where the route length for each vehicle is unrestricted. It should be added that these seven tested instances are made up of Taillard instances. Taillard built these instances depending on Golden's biggest instances related to the FSMVRP, along with variable and fixed vehicle costs. It should be added that all of these instances may be taken from the following URL: http://mistic.heig-vd.ch/taillard/problemes.dir/vrp.dir/vrp.html (accessed on 8 February 2023).

Table 2. Characteristics of the first test set.

Instance	Number of Customers	Kind of Vehicles	Capacity of Vehicles	Fixed Cost	Variable Cost	Number of Each Kind of Vehicles
a1	10	1	20	20	1	1
		2	30	40	1.1	1
		3	40	70	1.3	2
		4	70	200	1.7	1
a2	15	1	30	60	1	1
		2	60	100	1.1	1
		3	80	250	1.5	1
		4	150	300	2	1
a3	20	1	20	70	1	1
		2	35	120	1.1	2
		3	50	200	1.2	2
		4	120	250	2	3
a4	25	1	25	50	1	2
		2	35	80	1.1	2
		3	50	200	1.2	3
		4	120	250	1.7	3
a5	30	1	25	35	1	3
		2	35	50	1.1	2
		3	50	75	1.2	4
		4	120	150	1.7	4
		2	120	75	1.1	2
		3	160	100	1.2	3
a6	35	1	50	60	0.7	1
		2	120	75	1	2
		3	160	200	1.1	3
		2	140	75	1.7	2
		3	100	225	2.5	2
		4	200	400	2	2
a7	40	1	25	20	1	1
		2	35	35	1.1	1
		3	40	50	1.2	3
		4	70	120	1.7	4
		5	100	225	2.5	2
		6	200	400	3.2	1

In Figure 4, a 25-node instance for this problem is shown, solved by the exact algorithm (AIMMS). Looking at the shape, one can see that 4 vehicle types with 7 vehicles are used. The first kind of vehicles is brought with the black line, while the solution to the issue is used for the second type vehicles, which are shown with the green line. Moreover, the red line shows the solutions obtained by type 3 vehicles. In addition, given that there are only two fourth vehicles, that only one of them has been used in this form. It is noted that in these instances, similar to as in all available literature, the fixed vehicle cost is not considered, in the computations, only the variable vehicle cost is used.

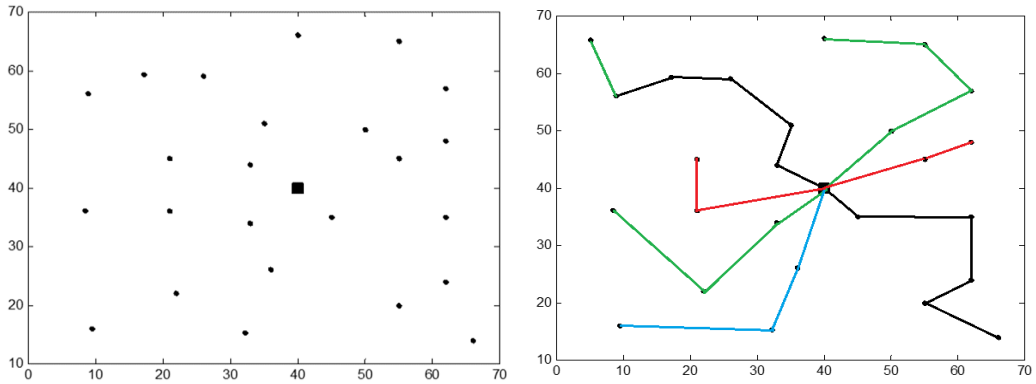

Figure 4. The results obtained by AIMMS using different colors for different type of vehicles.

In Table 3, we compared our results by HA with the results by the exact algorithm. In this table, for each instance, n shows the number of customers while the third and fourth columns show the results by the exact algorithm (AIMMS) and its computational time, respectively. Moreover, the worst solution (WS), mean solution (MS), and best solution (BS) obtained by HA in the ten iterations are shown in the fifth, sixth, and seventh columns. Finally, the computational time for obtaining the best solution by HA is presented in the very last column. From this table, one can see HA has performed better than the AIMMS and is able to acquire good quality solutions. In more detail, our proposed HA has obtained the same solutions at a much lesser time compared to AIMMS for a1, a2 and a3 instances. Alternatively, our algorithm found better solutions for instance a4. Finally, for the instances a5, a6, and a7, the AIMMS could reach a solution, but our HA could achieve a suboptimal solution in just about 30 s. Generally, it can be said that our algorithm has performed well that it can obtain solutions of the instances very quickly. For these instances, it is seen that as the number of customers grows, the computation time differences between these two algorithms grows rapidly. Therefore, in everyday applications of this problem in industry and services, a good quality solution can be achieved by our proposed algorithm in an acceptable time.

Table 3. Comparing the solutions by our HA with exact algorithm (AIMMS).

Instance	n	AIMMS		HA			
		Solution	Time (s)	WS	MS	BS	Time (s)
a1	10	205.32	2.23	205.32	205.32	205.32	0.61
a2	15	299.00	19.72	299.00	299.00	299.00	0.99
a3	20	412.65	94.73	412.65	412.65	412.65	2.12
a4	25	512.11	212.72	489.02	482.38	480.12	4.82
a5	30	587.76	43,353.53	573.92	550.82	524.02	11.82
a6	35	Na	-	399.02	386.29	383.12	25.82
a7	40	Na	-	650.26	630.19	620.15	33.20

In Table 4, a comparison has been made between the proposed and exact algorithm from another perspective. The difference between this table and Table 2 is that here is the run time for the two identical algorithms as shown in the fourth column of the table. It is noted that in spite of the fact that AIMMS software is one of the best optimization software, but for the four instances a4, a5, a6, and a7, it failed to achieve a solution at the specified time. Instead, for the three instances a1, a2, and a3, the proposed HA is cable to obtain much better solutions than the implemented exact algorithm within short computational time. Thus, one can conclude that the proposed HA is clearly better than the exact method.

Table 4. Comparing exact algorithm (AIMMS) and our HA within the same time.

Instance	*n*	AIMMS	Time (s)	HA
a1	10	235.18	0.61	215.12
a2	15	476.33	0.99	315.35
a3	20	498.45	2.12	456.82
a4	25	Na	4.82	591.12
a5	30	Na	11.82	763.82
a6	35	Na	25.82	998.125
a7	40	Na	33.20	673.82

The second dataset is provided by Taillard and consists of 8 problem instances from a13 to a20 with sizes ranging from 50 to 100 customers. In this medium size instance, time windows are added to create a new set for testing the proposed algorithm. In order to assess the usefulness of the HA compared to the classical version for these instances, results by the proposed HA and the ACO [36] are reported in Table 5. Furthermore, this is a new version of VRP, and no algorithm has been proposed until now, some classical meta-heuristics are considered here, and their results are compared with our algorithm. For any more details about the considered instances, one can click on the link: http://mistic.heig-vd.ch/taillard/problemes.dir/vrp.dir/vrp.html (accessed on 8 February 2023).

Table 5. Comparing algorithms for standard HFFOVRPTW problem instances.

Instance	*n*	GA		ACO		HA		BKS
		Solution	Time	Solution	Time	Solution	Time	
a13	50	998.13	101.26	992.76	98.15	990.11	94.32	990.11
a14	50	480.73	102.81	448.25	101.42	448.25	104.11	448.25
a15	50	789.27	99.02	714.67	88.73	709.31	92.22	705.12
a16	50	878.02	102.81	802.95	99.52	791.01	106.31	791.01
a17	75	898.26	152.72	823.56	123.52	815.05	129.11	815.05
a18	75	1698.92	147.92	1635.67	135.78	1601.55	145.04	1601.55
a19	100	1004.82	178.92	938.62	172.77	900.11	189.78	900.11
a20	100	1193.17	182.92	1088.62	178.02	1038.58	179.81	1035.12
Instance	***n***	**TS**		**ICA**		**SA**		**BKS**
		Solution	**Time**	**Solution**	**Time**	**Solution**	**Time**	
a13	50	990.11	99.12	992.62	100.23	993.12	105.22	990.11
a14	50	448.25	102.11	478.27	103.82	478.12	99.23	448.25
a15	50	705.12	88.92	780.32	105.92	780.02	105.22	705.12
a16	50	795.12	110.23	870.23	187.23	869.24	106.92	791.01
a17	75	825.72	145.82	900.12	155.82	889.12	144.23	815.05
a18	75	1601.55	155.82	1655.72	179.23	1614.27	155.92	1601.55
a19	100	905.11	199.12	999.12	199.82	950.12	175.92	900.11
a20	100	1035.12	204.11	1180.27	200.82	1056.82	198.02	1035.12

In Table 5, the first column shows the names of instances while the second column presents the number of clients of each instance. It is noted that in this table, along with the solution obtained by each algorithm, its execution time is also shown (in seconds). Consequently, in the third and fourth columns, the results by genetic algorithm (GA) [37], the fifth and sixth columns present the results by classical ACO [36], and the seventh and eighth columns present the solutions obtained by the proposed HA. Alternatively, in the second row of the table, the amounts of data lost by three other algorithms are shown. In this row, the third and fourth columns show the results by tabu search (TS) [38], the fifth and sixth columns show the results by the imperialist competitive algorithm (ICA) [39] and finally the seventh and eighth columns present the results by simulated annealing

(SA) [40]. Moreover, in the end column of each row, the best solutions (BKS) found by the above-mentioned algorithms are presented. This column leads to compare the efficiency of each algorithm compared to the best result obtained. In order to make a good comparison between these algorithms, the gap obtained for each instance is shown by each algorithm in Figure 5 (Formula (18)). In this formula, $c(s^*)$ is the best obtained solution by the considered algorithm and BKS is the best obtained solution by all algorithms.

$$Gap = \frac{c(s^*) - BKS}{BKS} \times 100 \tag{18}$$

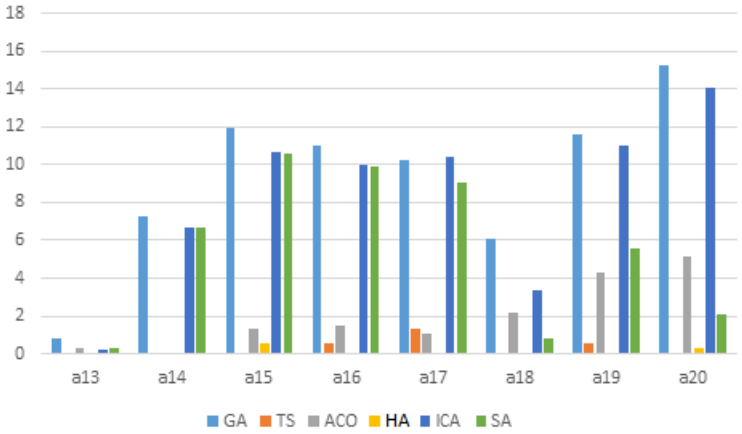

Figure 5. Comparing the gaps of the six algorithms.

In addition, in order to acquire the accurate results for comparing these algorithms, the average gap for all instances is shown in Figure 6. It is seen from this figure that our proposed HA is cable to find very good mean solutions compared to other algorithms. Moreover, TS algorithm could obtain better solutions compared to the other four algorithms and with an average of 0.3 in the second rank of algorithms. The ACO algorithm could not find better solution than the two algorithms, but it has obtained better solutions with an average of 1.97 than the GA, ICA, and SA algorithms. Finally, these three algorithms have almost equal efficiency to solve these problem instances and have an average of 5.6 for SA, 8.3 for ICA, and 9.27 for the GA algorithm. By looking at the results obtained by all algorithms, it is found that the closest solutions are by TS and HA algorithms. Comparing these two algorithms, based on the obtained results, it is found that for two instances a15 and a20, the TS algorithm could find better solutions, while for three instances a16, a17, and a19, our proposed HA has provided better quality solutions. Moreover, for the remaining two instances, these two algorithms have obtained the same solutions. From both points of view, our proposed HA could find the best solutions among these algorithms.

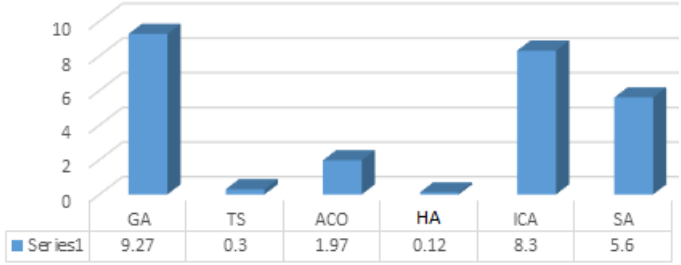

Figure 6. Comparing mean gaps of the six algorithms.

In Table 6, large size instances for the HFFOVRPTW have been created randomly. In these instances, which have a number of nodes from 110 to 400 nodes, the depot is placed at the coordinate center, and the customers are located inside a square with the center of

the depot and 200 sides. In Table 6, the third column shows the kind of vehicles (KV), the fourth column shows capacity of the vehicles (CV), the fifth column presents the fixed cost (FC), and the sixth column shows the variable cost (VC). In addition, the same algorithms presented in the previous table are listed here and the reported computational times are for the best solutions. On the other hand, in the end column, the best solution obtained by which algorithm is presented, thereby best solutions (BKS) of this column have been used for calculating the gap. Therefore, the numbers obtained in this table for all algorithms are a gap compared to the BKS.

Table 6. Comparing algorithms for large sized HFFOVRPTW problem instances.

Instance	*n*	KV	CV	FC	VC	GA		ACO		TS		HA		ICA		SA		BKS by
						Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time	
a21	110	4	20–70	20–70	1–1.8	4.32	170	3.62	172	2.43	201	0.00	187	4.12	172	5.74	175	HA
a22	120	4	20–70	20–70	1–1.8	3.54	168	3.99	171	1.34	200	0.00	176	3.12	173	4.73	176	HA
a23	130	4	20–70	20–70	1–1.8	5.23	175	5.12	176	3.73	212	0.00	189	5.65	174	2.38	180	HA
a24	140	4	20–70	20–70	1–1.8	2.99	164	2.12	165	2.83	199	0.00	192	2.83	168	5.93	170	HA
a25	150	4	20–70	20–70	1–1.8	5.23	184	1.27	167	0.00	212	1.82	195	5.93	169	5.29	173	TS
a26	160	4	20–70	20–70	1–1.8	3.93	187	4.83	178	3.27	231	0.00	192	4.12	166	3.28	175	HA
a27	170	4	20–70	20–70	1–1.8	5.90	192	4.83	179	1.74	212	0.00	189	5.82	165	5.90	180	HA
a28	180	4	20–70	20–70	1–1.8	4.93	188	3.99	180	1.82	211	0.00	189	4.83	187	4.83	181	HA
a29	190	4	20–70	20–70	1–1.8	6.03	186	5.82	181	0.95	204	0.00	192	5.59	183	5.15	189	HA
a30	200	4	20–70	20–70	1–1.8	5.82	178	5.12	187	1.12	223	0.00	194	4.83	189	4.12	201	HA
a31	210	4	20–100	20–70	1–1.8	7.12	179	6.03	189	0.00	245	2.81	198	6.83	183	5.30	204	TS
a32	220	4	20–100	20–70	1–1.8	7.29	197	6.84	200	0.00	253	0.00	212	6.12	179	5.83	199	TS
a33	230	4	20–100	20–70	1–1.8	6.82	186	6.12	201	1.72	248	0.00	232	5.23	198	5.89	215	HA
a34	240	4	20–100	20–70	1–1.8	6.85	200	4.72	190	1.82	234	0.00	223	5.73	200	5.34	225	HA
a35	250	4	20–100	20–70	1–1.8	7.93	198	5.83	199	1.39	212	0.00	243	5.84	198	5.84	256	HA
a36	260	4	20–100	20–70	1–1.8	5.19	234	3.27	212	1.22	245	0.00	236	4.75	197	4.12	267	HA
a37	270	4	20–100	20–70	1–1.8	4.82	212	3.93	202	0.94	265	0.00	238	3.96	169	3.49	256	HA
a38	280	4	20–100	20–70	1–1.8	9.82	189	4.83	200	0.00	243	0.48	251	7.84	198	6.82	218	TS
a39	290	4	20–100	20–70	1–1.8	3.72	206	2.82	213	2.48	213	0.00	242	3.67	189	3.12	267	HA
a40	300	4	20–100	30–80	1–2	5.84	213	4.83	243	1.22	256	0.00	265	5.34	168	4.90	273	HA
a41	310	5	30–90	30–80	1–2	7.60	243	5.93	212	3.82	256	0.00	276	5.73	201	5.12	289	HA
a42	320	5	30–90	30–80	1–2	8.93	265	5.04	254	1.92	255	0.00	254	6.83	222	5.73	256	HA
a43	330	5	30–90	30–80	1–2	7.69	234	5.94	254	0.37	267	0.00	286	6.73	213	5.93	235	HA
a44	340	5	30–90	30–80	1–2	7.99	265	5.39	245	0.00	289	1.82	265	6.39	186	6.65	287	TS
a45	350	5	30–90	30–80	1–2	8.93	213	4.39	223	0.27	267	0.00	287	7.39	198	6.93	273	HA
a46	360	5	30–90	30–80	1–2	8.84	254	4.80	243	0.00	287	0.28	296	8.12	231	7.03	277	TS
a47	370	5	30–90	30–80	1–2	8.12	243	4.83	241	1.92	298	0.00	281	6.72	254	5.06	298	HA
a48	380	5	30–90	30–80	1–2	9.12	245	4.93	246	0.00	278	1.72	283	7.84	245	7.83	273	TS
a49	390	5	30–90	30–80	1–2	8.26	249	5.94	256	1.92	298	0.00	285	6.38	256	5.93	299	HA
a50	400	5	30–90	30–80	1–2	9.63	278	5.93	267	0.12	289	0.00	287	7.93	243	7.95	263	HA

By observing the results reported in Table 6, and especially the last column, one can say that the best solutions are obtained by only TS and our proposed HA among all six algorithms. More precisely, the TS algorithm could obtain the best solutions for six out of 30 instances, while our proposed HA could obtain the best solutions for the remaining 24 instances. On the other hand, although the time of implementation of each algorithm for each instance is different, these differences are not so large and can be ignored compared to the total run time. These results, together with the results presented in previous tables, show that our HA is efficient to solve the instances and could find the best solutions for them.

In order to statistically comparison the quality of these six algorithms, one-way analysis of variance (ANOVA) is used in which the hypothesis H0 is that the average solutions of these algorithms are equal with $\alpha = 0.05$. The results obtained in Tables 7 and 8 and Figure 7 show that the average solutions obtained for these algorithms are not equal and the hypothesis H0 is rejected. In other words, there is a significant difference in the quality of these methods. In other words, our proposed HA has a high-quality solutions for solving this group of problems and can obtain the best solutions compared to other five algorithms according to Figure 6.

Table 7. Analysis of variance (ANOVA).

Source	DF	Adj SS	Adj MS	*F*-Value	*p*-Value
Algorithms	5	995.4	199.071	109.17	0.000
Error	174	317.3	1.823		
Total	179	1312.6			

Table 8. Means and standard deviations for the compared algorithms.

Algorithms	*n*	Mean	StDev	95% CI
ACO	30	4.768	1.243	(4.282, 5.255)
GA	30	6.614	1.950	(6.128, 7.101)
HA	30	0.298	0.722	(−0.189, 0.784)
ICA	30	5.740	1.420	(5.254, 6.227)
SA	30	5.405	1.313	(4.919, 5.892)
TS	30	1.345	1.154	(0.859, 1.832)

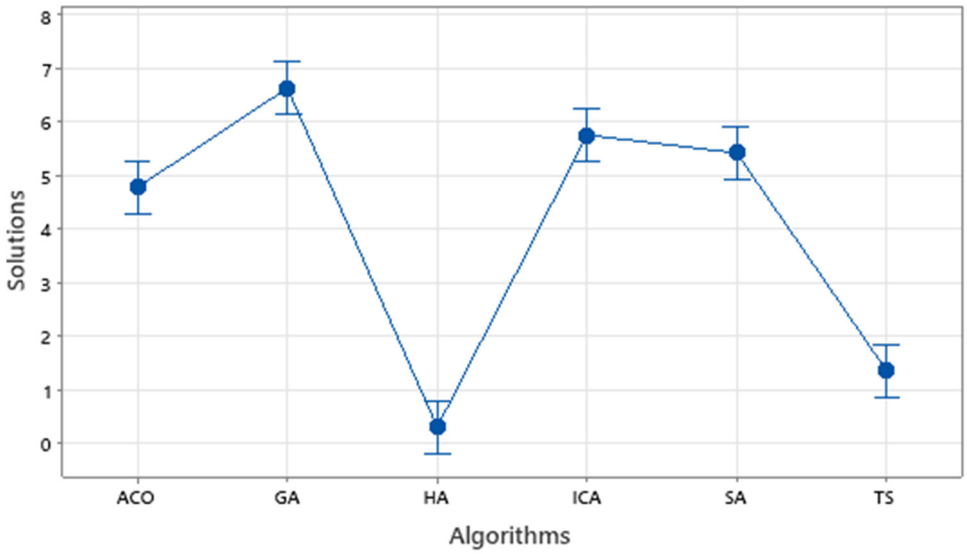

Figure 7. Interval plot of solutions vs algorithms with 95% CI for the mean.

5. Conclusions and Future Works

This paper presents MILP model for the HFFOVRPTW as one of new problems in transportation. Next, several problem instances were solved by the exact algorithm. In addition, four small, medium, and large sized standard instances are solved, and then the results by our proposed hybrid algorithm (HA) are compared against the exact method and five other algorithms. The results showed that our proposed HA could achieve better solutions than the exact method within a very short CPU time. Therefore, it is recommended to apply the proposed HA instead of exact algorithms for small-sized instances. Furthermore, the solutions by our proposed HA were compared with other metaheuristic algorithms for medium-sized and large-sized instances and showed that most of the best solutions were obtained by our algorithm. The use of tabu search and simulated annealing are suggested to improve the intensification of the HA to solve the problem. Other real limitations can also be considered in this problem, such as the pickup and delivery of items or the use of balance for the problem. The new problems could be modeled as well as could be solved that would be considered in upcoming research. In addition, this problem can be studied by other meta-heuristic algorithms, for example, tabu search, genetic algorithms, etc.

Author Contributions: Conceptualization, Z.H.A.; methodology, Z.H.A. and M.Y.; software, M.Y.; validation, Z.H.A. and M.Y.; formal analysis, M.Y.; investigation, M.Y.; resources, Z.H.A.; data curation, M.Y.; writing—original draft preparation, M.Y.; writing—review and editing, Z.H.A.; visualization, Z.H.A. and M.Y.; supervision, Z.H.A.; project administration, Z.H.A.; funding acquisition, Z.H.A. All authors have read and agreed to the published version of the manuscript.

Funding: The authors extend their appreciation to the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University for funding this work through Research Group no. RG-21-09-17.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Acknowledgments: The authors are very much thankful to the anonymous honorable reviewers for their careful reading of the paper and their many insightful comments and constructive suggestions which helped the authors to improve the paper.

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1. Brandão, J. A memory-based iterated local search algorithm for the multi-depot open vehicle routing problem. *Eur. J. Oper. Res.* **2020**, *284*, 559–571. [CrossRef]
2. Ahmed, Z.H.; Al-Otaibi, N.; Al-Tameem, A.; Saudagar, A.K.J. Genetic Crossover Operators for the Capacitated Vehicle Routing Problem. *Comput. Mater. Contin.* **2023**, *74*, 1575–1605. [CrossRef]
3. Abdullahi, H.; Reyes-Rubiano, L.; Ouelhadj, D.; Faulin, J.; Juan, A.A. Modelling and multi-criteria analysis of the sustainability dimensions for the green vehicle routing problem. *Eur. J. Oper. Res.* **2021**, *292*, 143–154. [CrossRef]
4. Euchi, J.; Sadok, A. Hybrid genetic-sweep algorithm to solve the vehicle routing problem with drones. *Phys. Commun.* **2021**, *44*, 101236. [CrossRef]
5. Zhang, K.; He, F.; Zhang, Z.; Lin, X.; Li, M. Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach. *Transp. Res. Part C Emerg. Technol.* **2020**, *121*, 102861. [CrossRef]
6. Park, H.; Son, D.; Koo, B.; Jeong, B. Waiting strategy for the vehicle routing problem with simultaneous pickup and delivery using genetic algorithm. *Expert Syst. Appl.* **2022**, *165*, 113959. [CrossRef]
7. Ahmed, Z.H.; Hameed, A.S.; Mutar, M.L. Hybrid Genetic Algorithms for the Asymmetric Distance-Constrained Vehicle Routing Problem. *Math. Probl. Eng.* **2022**, *2022*, 2435002. [CrossRef]
8. Sedighpour, M.; Ahmadi, V.; Yousefikhoshbakht, M.; Didehvar, F.; Rahmati, F. Solving the open vehicle routing problem by a hybrid ant colony optimization. *Kuwait J. Sci.* **2014**, *41*, 139.
9. Li, F.; Golden, B.; Wasil, E. The open vehicle routing problem: Algorithms, large-scale test problems, and computational results. *Comput. Oper. Res.* **2007**, *34*, 2918–2930. [CrossRef]
10. Sariklis, D.; Powell, S. A Heuristic Method for the Open Vehicle Routing Problem. *J. Oper. Res. Soc.* **2000**, *51*, 564. [CrossRef]
11. Sánchez-Oro, J.; López-Sánchez, A.D.; Colmenar, J.M. A general variable neighborhood search for solving the multi-objective open vehicle routing problem. *J. Heuristics* **2020**, *26*, 423–452. [CrossRef]
12. Ruiz, E.; Soto-Mendoza, V.; Barbosa, A.E.R.; Reyes, R. Solving the open vehicle routing problem with capacity and distance constraints with a biased random key genetic algorithm. *Comput. Ind. Eng.* **2019**, *133*, 207–219. [CrossRef]
13. Brandão, J. Iterated local search algorithm with ejection chains for the open vehicle routing problem with time windows. *Comput. Ind. Eng.* **2018**, *120*, 146–159. [CrossRef]
14. Gendreau, M.; Laporte, G.; Musaraganyi, C.; Taillard, D. A tabu search heuristic for the heterogeneous fleet vehicle routing problem. *Comput. Oper. Res.* **1999**, *26*, 1153–1173. [CrossRef]
15. Li, F.; Golden, B.; Wasil, E. A record-to-record travel algorithm for solving the heterogeneous fleet vehicle routing problem. *Comput. Oper. Res.* **2007**, *34*, 2734–2742. [CrossRef]
16. Li, X.; Tian, P.; Aneja, Y. An adaptive memory programming metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *Transp. Res. Part E Logist. Transp. Rev.* **2010**, *46*, 1111–1127. [CrossRef]
17. Tarantilis, C.; Kiranoudis, C.; Vassiliadis, V. A threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem. *Eur. J. Oper. Res.* **2004**, *152*, 148–158. [CrossRef]
18. Soman, J.T.; Patil, R.J. A scatter search method for heterogeneous fleet vehicle routing problem with release dates under lateness dependent tardiness costs. *Expert Syst. Appl.* **2020**, *150*, 113302. [CrossRef]
19. Chu, C.-W. A heuristic algorithm for the truckload and less-than-truckload problem. *Eur. J. Oper. Res.* **2005**, *165*, 657–667. [CrossRef]
20. Bolduc, M.-C.; Renaud, J.; Boctor, F. A heuristic for the routing and carrier selection problem. *Eur. J. Oper. Res.* **2007**, *183*, 926–932. [CrossRef]
21. Prins, C. Two memetic algorithms for heterogeneous fleet vehicle routing problems. *Eng. Appl. Artif. Intell.* **2009**, *22*, 916–928. [CrossRef]

22. Schmid, V.; Doerner, K.F.; Hartl, R.F.; Salazar-González, J.-J. Hybridization of very large neighborhood search for ready-mixed concrete delivery problems. *Comput. Oper. Res.* **2010**, *37*, 559–574. [CrossRef]
23. Euchi, J.; Chabchoub, H. A hybrid tabu search to solve the heterogeneous fixed fleet vehicle routing problem. *Logist. Res.* **2010**, *2*, 3–11. [CrossRef]
24. Brandão, J. A deterministic tabu search algorithm for the fleet size and mix vehicle routing problem. *Eur. J. Oper. Res.* **2009**, *195*, 716–728. [CrossRef]
25. Brandão, J. A tabu search algorithm for the heterogeneous fixed fleet vehicle routing problem. *Comput. Oper. Res.* **2011**, *38*, 140–151. [CrossRef]
26. Li, X.; Leung, S.C.; Tian, P. A multistart adaptive memory-based tabu search algorithm for the heterogeneous fixed fleet open vehicle routing problem. *Expert Syst. Appl.* **2012**, *39*, 365–374. [CrossRef]
27. Ahmed, Z.H.; Yousefikhoshbakht, M. An improved tabu search algorithm for solving heterogeneous fixed fleet open vehicle routing problem with time windows. *Alex. Eng. J.* **2023**, *64*, 349–363. [CrossRef]
28. Repoussis, P.; Tarantilis, C.D.; Ioannou, G. The open vehicle routing problem with time windows. *J. Oper. Res. Soc.* **2007**, *58*, 355–367. [CrossRef]
29. Gharaei, A.; Amjadian, A.; Amjadian, A.; Shavandi, A.; Hashemi, A.; Taher, M.; Mohamadi, N. An integrated lot-sizing policy for the inventory management of constrained multi-level supply chains: Null-space method. *Int. J. Syst. Sci. Oper. Logist.* **2022**, 1–14. [CrossRef]
30. Amjadian, A.; Gharaei, A. An integrated reliable five-level closed-loop supply chain with multi-stage products under quality control and green policies: Generalised outer approximation with exact penalty. *Int. J. Syst. Sci. Oper. Logist.* **2022**, *9*, 429–449. [CrossRef]
31. Rezaei, H.B.; Amjadian, A.; Sebt, M.V.; Askari, R.; Gharaei, A. An ensemble method of the machine learning to prognosticate the gastric cancer. *Ann. Oper. Res.* **2022**, 1–42. [CrossRef]
32. Gharaei, A.; Amjadian, A.; Shavandi, A. An integrated reliable four-level supply chain with multi-stage products under shortage and stochastic constraints. *Int. J. Syst. Sci. Oper. Logist.* **2021**, 1–22. [CrossRef]
33. Gharaei, A.; Shekarabi, S.A.H.; Karimi, M. Optimal lot-sizing of an integrated EPQ model with partial backorders and re-workable products: An outer approximation. *Int. J. Syst. Sci. Oper. Logist.* **2021**, 1–17. [CrossRef]
34. Sangeetha, V.; Krishankumar, R.; Ravichandran, K.S.; Kar, S. Energy-efficient green ant colony optimization for path planning in dynamic 3D environments. *Soft Comput.* **2021**, *25*, 4749–4769. [CrossRef]
35. Sangeetha, V.; Krishankumar, R.; Ravichandran, K.S.; Cavallaro, F.; Kar, S.; Pamucar, D.; Mardani, A. A Fuzzy Gain-Based Dynamic Ant Colony Optimization for Path Planning in Dynamic Environments. *Symmetry* **2021**, *13*, 280. [CrossRef]
36. Yousefikhoshbakht, M.; Didehvar, F.; Rahmati, F. A mixed integer programming formulation for the heterogeneous fixed fleet open vehicle routing problem. *J. Optim. Ind. Eng.* **2015**, *8*, 37–46.
37. Zafari, A.; Hashemi, S.M.T.; Yousefikhoshbakht, M. A hybrid effective genetic algorithm for solving the vehicle routing problem. *Int. J. Ind. Eng. Prod. Res.* **2010**, *21*, 63–76.
38. Yousefikhoshbakht, M.; Didehvar, F.; Rahmati, F. A combination of modified tabu search and elite ant system to solve the vehicle routing problem with simultaneous pickup and delivery. *J. Ind. Prod. Eng.* **2014**, *31*, 65–75. [CrossRef]
39. Yousefikhoshbakht, M.; Sedighpour, M. New Imperialist Competitive Algorithm to solve the travelling salesman problem. *Int. J. Comput. Math.* **2013**, *90*, 1495–1505. [CrossRef]
40. Geng, X.; Chen, Z.; Yang, W.; Shi, D.; Zhao, K. Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search. *Appl. Soft Comput.* **2011**, *11*, 3680–3689. [CrossRef]