Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(9): 15672–15707.
DOI: 10.3934/mbe.2023699
Received: 30 April 2023
Revised: 26 June 2023
Accepted: 06 July 2023
Published: 28 July 2023

Research article

An improved genetic algorithm for solving the helicopter routing problem with time window in post-disaster rescue

Kaidong Yang[1,2], **Peng Duan**[3,*] **and Huishan Yu**[2]

[1] Shandong Key Laboratory of Optical Communication Science and Technology, Liaocheng University, Liaocheng 252059, China

[2] School of Physics Science and Information Technology, Liaocheng University, Liaocheng 252059, China

[3] School of Computer Science, Liaocheng University, Liaocheng 252059, China

* **Correspondence:** Email: duanpeng@lcu.edu.cn.

Abstract: The vehicle routing problem (VRP) is a highly significant and extensively studied issue in post-disaster rescue. In recent years, there has been widespread utilization of helicopters for post-disaster rescue. However, efficiently dispatching helicopters to reach rescue sites in post-disaster rescue is a challenge. To address this issue, this study models the issue of dispatching helicopters as a specific variant of the VRP with time window (VRPTW). Considering that the VRPTW is an NP-hard problem, the genetic algorithm (GA) as one of the prominent evolutionary algorithms with robust optimization capabilities, is a good candidate to deal with this issue. In this study, an improved GA with a local search strategy and global search strategy is proposed. To begin, a cooperative initialization strategy is proposed to generate an initial population with high quality and diversity. Subsequently, a local search strategy is presented to improve the exploitation ability. Additionally, a global search strategy is embedded to enhance the global search performance. Finally, 56 instances extended from Solomon instances are utilized for conducting simulation tests. The simulation results indicate that the average relative percentage increase (RPI) of the distance travelled by helicopters as obtained by the proposed algorithm is 0.178, 0.027, 0.075 and 0.041 times smaller than the average RPIs obtained by the tabu search algorithm, ant colony optimization algorithm, hybrid GA and simulated annealing algorithm, respectively. Simulation results reveal that the proposed algorithm is more efficient and effective for solving the VRPTW to reduce the driving distance of the helicopters in post-disaster rescue.

Keywords: vehicle routing problem; time window; improved genetic algorithm; post-disaster rescue; helicopter dispatch

1. Introduction

Natural disasters such as earthquakes, tsunamis and hurricanes have devastating consequences for human lives and infrastructure [1]. These catastrophic events not only cause immense damage to society, they also pose significant threats to human health and well-being. In recent years, post-disaster rescue has received extensive attention as a critical research field [2, 3]. The timely and effective response to such emergencies is paramount to saving lives and minimizing the impact on affected communities. Disasters frequently lead to the destruction of roads, which presents substantial challenges for ground transportation and rescue operations. As a result, the utilization of helicopters can be an effective rescue method. Immediately after a large-scale disaster, helicopters are extensively employed for emergency medical services, search and rescue operations and the transportation of supplies and victims [4]. Helicopters possess the unique capability to access remote and inaccessible areas regardless of the terrain [5]. Therefore, the development of efficient and intelligent optimization algorithms for helicopter routing in post-disaster rescue is of paramount importance in improving rescue efforts and minimizing the loss of life and property [6].

The vehicle routing problem (VRP) in post-disaster rescue presents a complex and challenging problem [7]. When dispatching helicopters, considerations should not only encompass transportation costs, they should also prioritize rescue efficiency and effectiveness. First, the capacity constraints of helicopters limit the transport of injured individuals and supplies. In post-disaster rescue, the maximization of transportation capacity for victims and essential relief materials in each flight is of paramount importance. This strategic approach is aimed at enhancing rescue efficiency and effectively increasing the rate of lives saved. Additionally, post-disaster rescue efforts are often characterized by time sensitivity, which directly influences the outcomes of survival. Consequently, rescue sites have imposed heightened requirements on helicopter response times. Finally, a critical objective in helicopter rescue missions is to determine the shortest path. By identifying the most efficient route, response times can be minimized, enabling rescuers and supplies to swiftly reach the affected areas and provide prompt emergency relief.

The dispatch of helicopters in post-disaster rescue operations can be considered as an extension of the VRP problem. The genetic algorithm (GA) is a well-established optimization algorithm that has been successfully applied to various types of optimization problems [8–10]. The GA is known for its competitiveness and adaptability, making it a widely used method in many fields. Notably, the GA has demonstrated its effectiveness in solving VRPs [11–13].

Considering the characteristics of the GA and the aforementioned problems, this paper proposes the improved GA (IGA) to tackle the VRP with a time window (VRPTW) in post-disaster rescue. In the proposed algorithm, each chromosome encodes a two-dimensional vector to enhance the exploration capability of the GA to address such problems. To address the challenge of obtaining feasible solutions for large-scale constraint problems, a repair strategy has been devised to enhance the performance of the proposed algorithm. Furthermore, to enhance the quality and diversity of the population, an efficient cooperative initialization strategy has been developed. Additionally, to improve the exploitation abilities, the proposed algorithm incorporates a local search strategy based on an improved greedy insertion heuristic. Moreover, to reduce the likelihood of the population being trapped in local optima and enhance global search capabilities, a global search strategy based on encoding structure destruction and construction is presented.

The main contributions of this study are as follows:

1) A cooperative initialization strategy is proposed to enhance the quality and diversity of the population at the beginning of the proposed algorithm, which leads to better performance.

2) A local search strategy is embedded to improve the exploitation capabilities, which enables the proposed algorithm to converge to better solutions.

3) A global search strategy is presented to alleviate the issue of getting trapped in local optima and improve the global search capabilities.

The remaining sections of this paper are organized as follows. Section 2 reviews the literature related to the VRP. Section 3 provides a brief introduction to the problem description and formulation. Section 4 presents all of the components of the proposed algorithm. The experimental results and analysis are shown in Section 5, followed by the conclusions of this study in Section 6.

2. Literature review

The VRP is a classical combinatorial optimization problem that was first proposed by Dantzig and Ramser [14], attracting considerable attention from the research community. In recent years, the VRP has found extensive application in various fields, such as home healthcare logistics [15, 16] and municipal solid waste disposal [17]. The research on the VRP is progressing and expanding. Zulvia et al. [18] proposed a green VRP for perishable products, which optimizes operational cost, deterioration cost, carbon emissions and customer satisfaction. Expósito et al. [19] studied a hybrid metaheuristic to solve the VRPTW from the quality of service perspective. Liu et al. [20] proposed an optimized solution for the time-dependent VRPTW by utilizing an improved ant colony algorithm with a congestion-avoiding approach. The approach aims to mitigate traffic congestion during peak hours and temporal traffic congestion. Bianchessi nd Irnich [21] investigated a new and tailored branch-and-cut algorithm in order to avoid traffic congestion and reduce the total costs.

With the increasing complexity of problems, researchers are placing growing emphasis on the development and application of efficient optimization algorithms [22–24]. Continual development and evolution of various optimization algorithms [25–27] are being pursued to solve models of various problems [28, 29]. The optimization algorithms dedicated to solving the VRP have gained significant attention in recent years, making it a prominent research area. Optimization algorithms have achieved significant research accomplishments in various fields [30–32]. Duan et al. [33] developed a robust multiobjective particle swarms optimization approach by incorporating an advanced encoding and decoding scheme, a robustness measurement metric, and a local search strategy to solve the VRPTW under uncertainty. Shen et al. [34] presented a hybrid swarm intelligence algorithm that incorporates both inter-route and intra-route improvement heuristics to solve the VRPTW and minimize the total distance. Brito et al. [35] utilized a variable neighbourhood search algorithm to solve the close–open VRPTW. Keskin and Catay [36] developed an adaptive large neighborhood search algorithm with several removal and insertion mechanisms to solve the VRPTW. Ahmed and Yousefikhoshbakht [37] proposed an improved tabu search (TS) algorithm with the intensification and diversification mechanisms to solve the heterogeneous fixed fleet open VRPTW.

Post-disaster rescue is a critical application field of the VRP, dedicated to ensuring prompt disaster response and facilitating the timely delivery of emergency rescue services [38]. Numerous VRP models and algorithms have been proposed by researchers to address post-disaster rescue scenarios. Vieira et al. [39] proposed hybridizing the ant colony optimization (ACO) metaheuristic with random variable neighborhood descent to solve the VRP problem. The objective of their study was to minimize the operational costs associated with emergency water transport in arid regions. Yi et al. [40] developed an enhanced monarch butterfly optimization algorithm to tackle the emergency VRP in natural disaster relief. Maghfiroh and Hanaoka [41] utilized a modified simulated annealing (SA) algorithm to solve the dynamic truck and trailer routing problem in last mile distribution for disaster response.

Although there have been numerous studies on post-disaster rescue, there are few studies on the problem of helicopter dispatching in situations where roads are blocked. To fill the aforementioned gaps, this paper models the dispatching helicopter problem in post-disaster rescue, taking into account various factors such as time window constraints, helicopter loading capacity, the transport of injured individuals and supplies and travel distance. Moreover, we have designed an IGA for solving the dispatching helicopters problem in post-disaster rescue.

3. Problem description and formulation

This section presents the description and formulation of the VRPTW for helicopter rescue operations. First, a formal description of the VRPTW for helicopter rescue operations is provided, along with key assumptions. Then, the proposed VRPTW is mathematically formulated.

3.1. Problem description

In this study, the dispatching helicopter problem in post-disaster rescue is modeled as an extension of the VRPTW (hereafter called the H-VRPTW). The features of the H-VRPTW are as follows: 1) The system consists of two types of helicopters: rescue helicopters and transport helicopters. 2) Each rescue site has two types of demands, a service duration and a time window constraint. 3) The rescue helicopter is responsible for rescuing all of the victims. 4) The transport helicopter is primarily responsible for delivering supplies to all of the rescue sites, and if the transport helicopter is carrying less than half of its maximum capacity, it can serve as a rescue helicopter and transport injured individuals who are encountered along the way to the rescue center. 5) The rescue sites have perfect communication functionality to ensure smooth information transmission.

The assumptions of the H-VRPTW are as follows:

1) There is only one rescue center and all helicopters must depart from and return to the rescue center.

2) Each rescue site is visited once by each type of helicopter.

3) The demand for rescue sites along the route should not exceed the carrying capacity of the helicopter.

4) Each helicopter must return within the specified maximum route time.

5) The time window of each rescue site cannot be violated.

3.2. Problem formulation

The H-VRPTW is represented by the directed graph $G = (N, A)$, where $N = \{0\} \cup R$. The set R represents nodes of the rescue sites, and the 0 represents the rescue center. The set of arcs connecting vertices of N is denoted by $A = \{(i, j) \mid i, j \in N, i \neq j\}$. Each rescue site $i \in N$ is assigned a service duration t_i and a time window $[a_i, b_i]$, where a_i and b_i define the earliest and latest possible service times for the rescue site. The arrival time of a helicopter at rescue site i is represented by g_{ik}, and the distance traveled for the arc $(i, j) \in A$ is represented by d_{ij}. $TH = \{1, \ldots, n_1\}$ is the set of transport helicopters, and $RH = \{n_1 + 1, \ldots, n_1 + n_2\}$ is the set of rescue helicopters. $H = TH \cup RH$ represents all of the helicopters. Each helicopter $k \in H$ is required to depart from and return to the rescue center.

Taking into account the unique characteristics of the system, the demand for material and casualty care at rescue site i is denoted by dm_i and dn_i, respectively. The maximum capacities of helicopter k for material and casualty care are represented by cm_i and cn_i, respectively. The notations used in the model are as follows:

Indices:

0: Index of the rescue center

i, j: The rescue site index

k: Helicopter index

Sets:

N: Set of rescue sites and rescue center in the system

R: Set of rescue sites

H: Set of helicopters in the system

TH: Set of transport helicopters

RH: Set of rescue helicopters

Parameters:

a_i: The earliest service time for rescue site i

b_i: The latest service time for rescue site i

d_{ij}: Distance between nodes i and j

dm_i: The demand for material at rescue site i

dn_i: The demand for casualty care at rescue site i

cm_i: The maximum capacity for material for helicopter k

cn_i: The maximum capacity for casualty care for helicopter k

t_i: Service duration for rescue site i

Decision variable:

Z_{ijk}: A binary value that is set to 1 if helicopter k travels directly from the rescue site i to rescue site j; otherwise, Z_{ijk} is set to 0, where $i, j \in N$ and $i \neq j$.

Y_{ik}: A binary value that is set to 1 if helicopter k serves the rescue site i; otherwise, Y_{ik} is set to 0, where $i \in R$.

g_{ik}: Arrival time of a helicopter k at rescue site i.

ts_{ik}: The time when helicopter k started service at rescue site i.

Objective:

$$\min \sum_{i \in R} \sum_{j \in R} \sum_{k \in H} d_{ij} Z_{ijk} \tag{1}$$

Constraints:

$$\sum_{j \in N} Z_{0jk} = 1 \quad \forall k \in H \tag{2}$$

$$\sum_{i \in N} Z_{i0k} = 1 \quad \forall k \in H \tag{3}$$

$$\sum_{i \in N} Y_{ik} \leq 1 \quad \forall k \in TH \tag{4}$$

$$\sum_{i \in N} Y_{ik} \leq 1 \quad \forall k \in RH \tag{5}$$

$$\sum_{i \in N} dm_i Y_{ik} \leq cm_k \quad \forall k \in TH \tag{6}$$

$$\sum_{i \in N} dn_i Y_{ik} \leq cn_k \quad \forall k \in H \tag{7}$$

$$a_i \leq ts_{ik} \leq b_i \quad \forall i \in R, \forall k \in H \tag{8}$$

$$ts_{ik} = \max\{a_i, g_{ik}\} \quad \forall i \in R, \forall k \in H \tag{9}$$

$$Z_{ijk} \in \{0,1\} \quad \forall i, j \in N, \forall k \in H \tag{10}$$

$$Y_{ik} \in \{0,1\} \quad \forall i \in R, \forall k \in H \tag{11}$$

The objective function in Eq (1) minimizes the total distance traveled by all helicopters. Constraints given by Eqs (2) and (3) state that each route taken by the helicopter starts and ends at the rescue center. Constraints given by Eqs (4) and (5) ensure that each type of helicopter can only be visited once for each rescue site. Constraints given by Eqs (6) and (7) ensure the capacity constraints for material transport and casualty care for the helicopter. Constraints given by Eqs (8) and (9) require that the time of service for each rescue site falls within its time window. Constraints given by Eqs (10) and (11) imposes a restriction on the decision variable.

4. Proposed algorithm

This section presents the proposed IGA for solving the H-VRPTW problem. The general framework of the IGA is outlined, followed by a detailed description of various components, including encoding and decoding methods, initialization, a crossover operation, a mutation operation, a repair strategy, a local search strategy and a global search strategy. Lastly, the complexity of the proposed IGA is analyzed, providing insights into the computational requirements and efficiency of the proposed algorithm.

4.1. IGA framework

An improved version of the classical GA has been proposed to solve the dispatching problem in post-disaster rescue. The framework of the IGA approach is shown in Algorithm 1. In the population initialization stage, the IGA employs a cooperative initialization strategy (cf. Subsection 4.3) to generate the population of size Ps. In the population evolutionary stage, the crossover operation (cf. Subsection 4.4) and the mutation operation (cf. Subsection 4.5) are performed for the population. Subsequently, the population performs a repair strategy (cf. Subsection 4.6) to ensure that each individual is feasible. After that, the local search strategy (cf. Subsection 4.7) is utilized for the population. Moreover, the better individual is selected to update the population. Then, the global search strategy (cf. Subsection 4.8) is embedded to generate new individuals and replace individuals with long-term evolutionary stagnation. Finally, the evolution stops and the best individual is outputted when the stopping criterion is satisfied.

Algorithm 1: The framework of the IGA

```
Input: the data of rescue sites and helicopters
Output: the best individual
1  Initialize a population of size $Ps$ using initialization strategy (cf. Subsection 4.3);
2  do
3      for each individual $x$ in $Ps$ do
4          Perform the crossover operation (cf. Subsection 4.4) and the mutation operation (cf. Subsection 4.5);
5          Perform the repair strategy (cf. Subsection 4.6) to guarantee individual is a feasible individual;
6          Perform the local search strategy (cf. Subsection 4.7) to generate a new individual $x_{new}$;
7          if $x_{new}$ is better than $x$ then
8              Replace $x$ with $x_{new}$;
9          end
10     end
11     for each individual $x$ do
12         if the number of continuous iterations without improvement exceeds $Lm$ for $x$ then
13             Delete the individual $x$ and insert a new one using global search strategy (cf. Subsection 4.8);
14         end
15     end
16 while stopping criterion is not met;
17 return the best individual;
```

4.2. Encoding and decoding

This study encodes each chromosome as a two-dimensional array, where the first dimension corresponds to all helicopters and the second dimension represents the assigned rescue sites for each helicopter. An example of the encoding is shown in Figure 1, where the sequence {0, 3, 6, 8, 10, 0} indicates that a helicopter starts at the rescue center and then visits rescue sites 3, 6, 8 and 10 before returning to the rescue center. The other two sequences, {0, 5, 1, 4 ,7, 0} and {0, 2, 9, 0}, represent other routes. Notice that the selection of each rescue site in the constructed route must adhere to the constraints outlined in the H-VRPTW model.

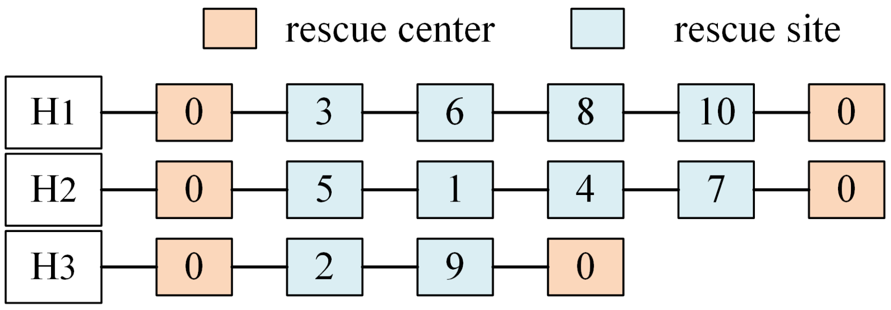


Figure 1. Example of the problems encoding scheme.

In the decoding process, tasks are assigned to helicopters according to the first dimension of the two-dimensional array, and then each rescue site is processed according to the second dimension of the array. The travel distance of the helicopter to the rescue site is calculated in sequence according to the two-dimensional array sequence of the chromosome, and the objective value of the solution can be obtained.

4.3. Initialization

A critical aspect of the algorithm is a population that exhibits a high degree of both solution quality and diversity. To solve the considered problem, three initialization rules are used to jointly generate the initial population. The three initialization rules are as follows: 1) a random distribution method; 2) a sorting assignment method using time windows; 3) a sorting assignment method based on rescue site distance. Assuming the population size is Ps, the initialization is specified as follows:

1) The $Ps-2$ individuals are generated via a random distribution method. First, randomly sequence all rescue sites. Then, for each rescue site, try to insert rescue sites into all of the current helicopters.

2) One individual is generated via the sorting assignment method by using a time window. First, each rescue site is sorted in ascending order according to the starting time window. Then, sequentially insert each rescue site into the current route at the position of minimum distance.

3) One individual is generated by using the sorting assignment method based on rescue site distance. First, calculate the distance between each rescue site and the rescue center. Second, arrange the rescue sites in ascending order of distance. Finally, sequentially insert each rescue site into the current route at the position of minimum distance.

The main steps of the cooperative initialization strategy are described in Algorithm 2.

Algorithm 2: Cooperative initialization strategy

```
Input: system parameters
Output: the initial population
1  for $n = 1$ to $Ps - 2$ do
2      Randomly sequence rescue sites.
3      for each rescue site $i$ do
4          Try to insert rescue site $i$ into all current helicopters.
5          if rescue site i cannot be inserted then
6              Add a new helicopter and service rescue site $i$.
7      Store the generated individual into the current population.
8  Sort all rescue sites in ascending order of start time.
9  for each rescue site $i$ do
10     for each helicopter k in the current individual do
11         for each position p in the current helicopter $k$ do
12             if rescue site i can be inserted into the current position $p$ then
13                 Record the current position $p$ in the helicopter.
14     Select the best position $p$ and insert rescue site $i$ into the position $p$ in the helicopter.
15     if rescue site i cannot be inserted then
16         Add a new helicopter and service rescue site $i$.
17 Store the generated individual into the current population.
18 Calculate the distance between each rescue site and the rescue center.
19 Sort all rescue sites in ascending order of distance.
20 for each rescue site $i$ do
21     for each helicopter k in the current individual do
22         for each position p in the current helicopter $k$ do
23             if rescue site $j$ can be inserted into the current position $p$ then
24                 Store the current position $p$ in the helicopter.
25     Select the best position $p$ and insert rescue site $i$ into the position $p$ in the helicopter
26     if rescue site i cannot be inserted then
27         Add a new helicopter and service rescue site $i$
28 Store the generated individual into the current population.
29 return the initial population;
```

4.4. Crossover operation

Crossover operation is a crucial step in the GA as it enables the transfer of excellent genes from the parent generation to the offspring. A well-designed crossover operation based on chromosome structure is presented as follows:

Step 1: The current chromosome serves as the parent A; randomly select another chromosome as parent B.

Step 2: Randomly select one helicopter from parent B and insert it at the end position of a randomly selected helicopter from parent A. Remove any duplicate rescue sites to generate offspring 1.

Step 3: Randomly select one helicopter from parent A and insert it at the end position of a randomly selected helicopter from parent B. Remove any duplicate rescue sites to generate offspring 2.

Step 4: Compare the fitness of offspring 1 and offspring 2, and retain the superior offspring.

In the crossover operation, the fitness of two offspring is compared, and the one with higher fitness is preserved. This selection process ensures that individuals with superior fitness are retained within the population, allowing favorable individuals to be introduced into the next generation. This gradual evolution of the population, driven by the preservation of higher-fitness individuals, helps the algorithm converge towards the optimal individual as the iterations progress.

Figure 2 shows a schematic of the crossover operation; the rescue sites {6, 4, 7, 9} served by helicopter H_2 are selected from parent B and sequentially inserted behind the path of helicopter H_3 in parent A; then, a new solution offspring 1 is generated. Similarly, the rescue sites {5, 1, 4} served by helicopter H_2 from parent A are inserted sequentially after the path of helicopter H_3 in parent B, generating offspring 2. Finally, offspring 1 and offspring 2 are compared, and the superior offspring 1 is retained.

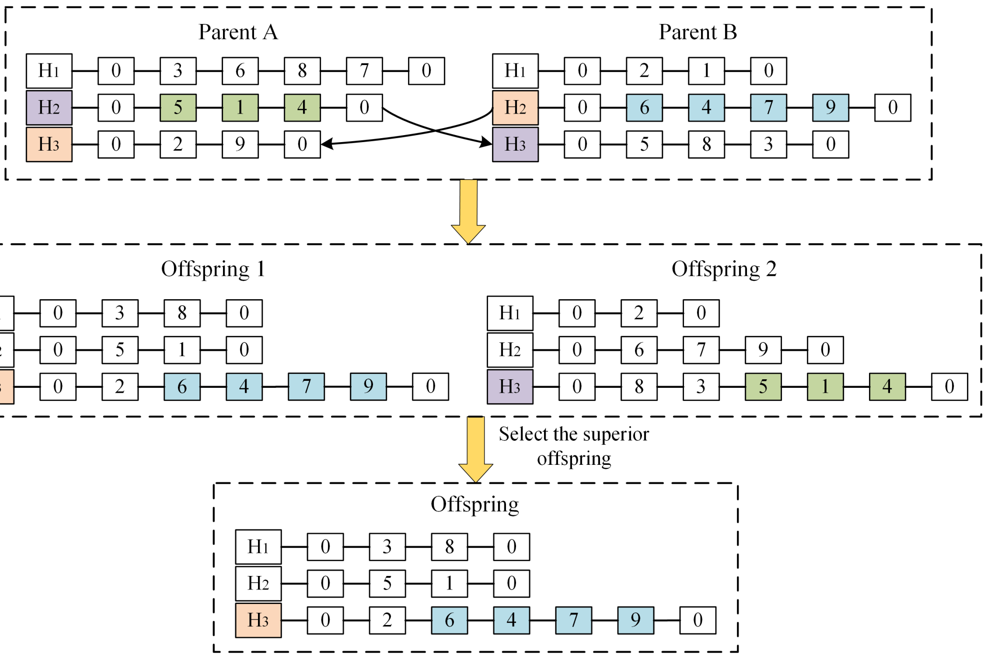


Figure 2. Example of the crossover operation.

4.5. Mutation operation

The mutation operation in the GA serves to imitate the mutation phenomenon of some genes on chromosomes in nature. The mutation operation can improve the situation of the GA falling into the local optimum to some extent, while keeping the diversity of the population. In this study, the swap mutation operator is adopted. First, randomly select two mutation operators from parent A, and then the two selected mutation operators are swapped to generate a new individual. Figure 3 provides a schematic of the mutation operator.

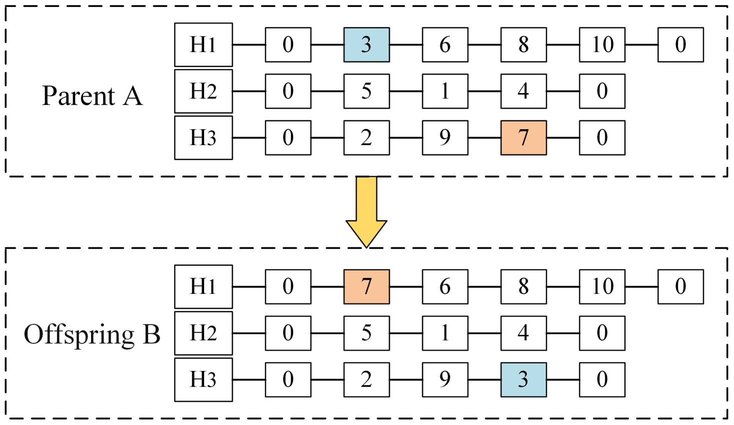


Figure 3. Example of the mutation operation.

4.6. Repair strategy

After crossover and mutation operations, certain rescue sites in the chromosome may violate the time window constraints. Hence, a repair strategy has been devised to guarantee the feasibility of each individual in the population. First, the rescue sites that violate the time window are removed. Then, for each rescue site that violates the time window, it is inserted into the current route at the position of minimum distance. Algorithm 3 shows the specific steps of the repair strategy.

Figure 4 illustrates the process of chromosome repair. In the figure, helicopter H_3 undergoes crossover and mutation operations, resulting in a modified path from {2, 6} to {2, 6, 4, 7}. However, it is observed that the arrival time of helicopter H_3 at rescue site {4} is T = 40, which falls outside of the specified service time window of [20, 35]. Therefore, the rescue site {4} is excluded from the service path of helicopter H_3 and inserted at a position that satisfies the time window constraint and minimizes the distance. As a result, the revised service path for H_3 becomes {2, 6, 7}, while helicopter H_1 is assigned to service the rescue site {4}.

Algorithm 3: Repair strategy

```
Input: an infeasible solution
Output: a feasible solution
for each helicopter $k$ in the current solution do
    for each rescue site $i$ assigned to helicopter $k$ do
        if $i$ violates its time window then
            Record the rescue site $i$.
            Delete rescue site $i$ from helicopter $k$.
        end
    end
end
for each rescue site $i$ that violates the time window do
    for each helicopter $k$ in the current solution do
        for each position $p$ in the current helicopter $k$ do
            if rescue site $i$ can be inserted into the current position $p$ then
                Record the position $p$ in the helicopter.
            end
        end
    end
    Select the best position $p$ and insert rescue site $i$ into the position $p$ in the helicopter.
    if rescue site $i$ cannot be inserted then
        Add a new helicopter and service rescue site $i$.
    end
end
return a feasible solution;
```

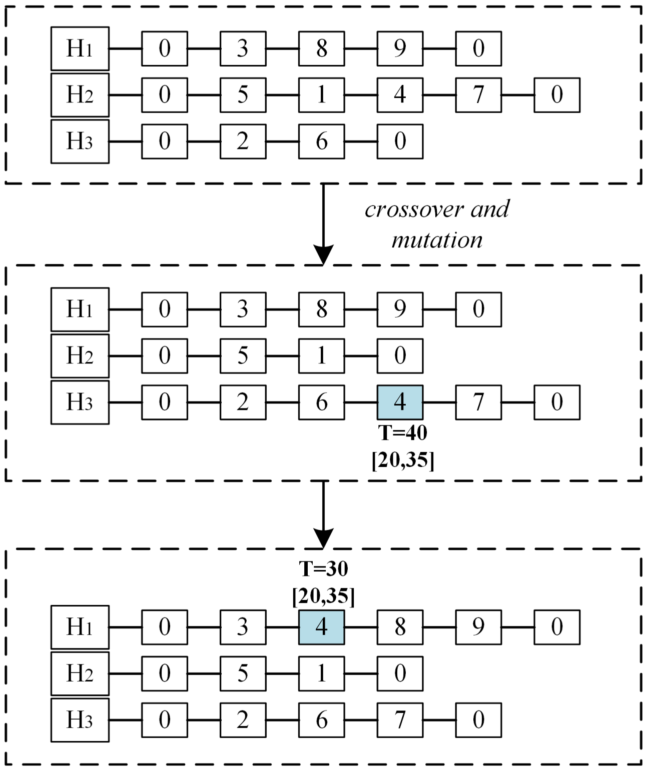


Figure 4. Example of the repair strategy.

4.7. Local search strategy

To enhance the exploitation ability of the IGA, a local search strategy based on an improved greedy insertion heuristic is incorporated. First, all rescue sites are placed into a set R_s and the total number n of rescue sites is calculated. Then, a rescue site is randomly selected from R_s and added to the set R_r, which is then removed from R_s. Third, a rescue site i is randomly selected from R_r, and for each rescue site j in R_s, the distance between i and j is calculated. The set R_s is then sorted based on the distances calculated, and the rescue site with the shortest distance is added to R_r and removed from R_s. This process is repeated $n/10$ times. Fourth, the set R_r is removed from the current solution. Finally, for each rescue site in the set R_r, it is inserted into the current route at the position of minimum distance. Algorithm 4 shows the specific steps of the local search strategy.

Figure 5 illustrates the local search approach. Initially, the rescue site {11} is selected from helicopter H_2 and added to the set R_r for recording. Subsequently, the nearest rescue site {16} from the remaining unselected rescue sites is chosen and included in the set R_r. From the set R_r, one rescue site (in this case, {16}) is randomly selected, and then the nearest unselected rescue site {7} to {16} is identified and added to the set R_r. Finally, the removed rescue sites {11, 16, 17} in set R_r are reinserted into current route at the position of minimum distance.

Algorithm 4: Local search strategy

```
Input: a solution
Output: an improved solution
1  Put all rescue sites into the set $R_s$ and calculate the number $n$ of all rescue sites;
2  Randomly select a rescue site $i$ from $R_s$ and add the rescue site to the set $R_r$;
3  Remove the selected rescue site $i$ from $R_s$;
4  for $m$ = 1 to $n/10$ do
5      Randomly select a rescue site $i$ of set $R_r$;
6      for each rescue site $j$ in $R_s$ do
7          Calculate all distance arcs $a(i, j)$;
8      end
9      Sort the set $R_s$ from smallest to biggest according to the arcs $a(i, j)$;
10     Store the top rescue site of $R_s$ into $R_r$ and remove the rescue site from set $R_s$;
11 end
12 Remove the set $R_r$ from current solution;
13 for each rescue site $i$ in the set $R_r$ do
14     for each helicopter $k$ in the current solution do
15         for each position $p$ in the current helicopter $k$ do
16             if rescue site $i$ can be inserted into the current position $p$ then
17                 Record the current position $p$ in the helicopter;
18             end
19         end
20     end
21     Select the best position $p$ and insert rescue site $i$ into the position $p$ in the helicopter;
22     if rescue site $i$ cannot be inserted then
23         Add a new helicopter and service rescue site $i$;
24     end
25 end
26 return an improved solution;
```

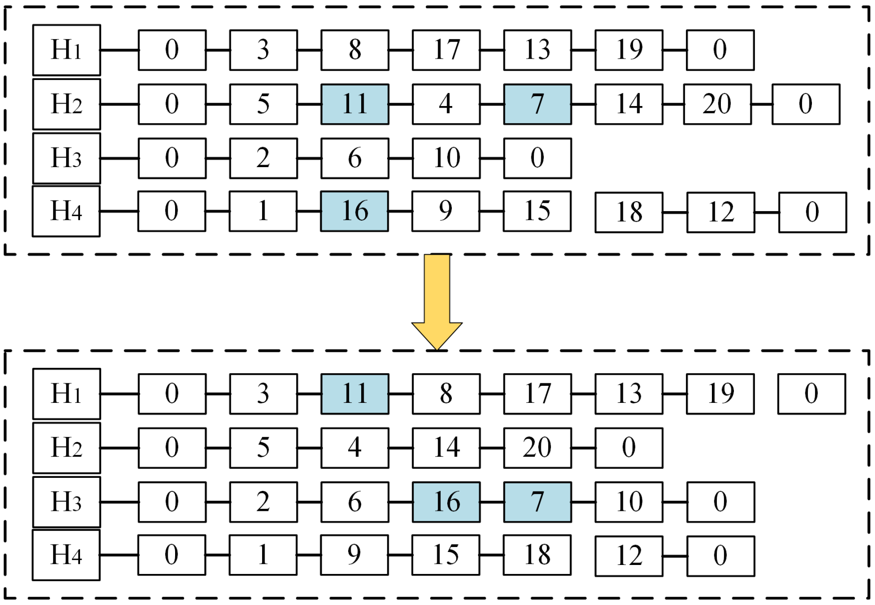


Figure 5. Example of the local search strategy.

4.8. Global search strategy

When an individual within the population fails to improve after multiple iterations, the individual is regarded as being trapped in local optima. In order to enhance the global search performance, and to avoid getting stuck in local optima, a global search strategy based on encoding structure destruction and construction is presented. First, randomly select z helicopters from the current solution and record the set of rescue sites served by the helicopters in set R_d. Then, delete the z helicopters from the current solution. Finally, for each rescue site i in the set R_d, insert this rescue site into current route at the position of minimum distance. The steps of the global search strategy are shown in Algorithm 5.

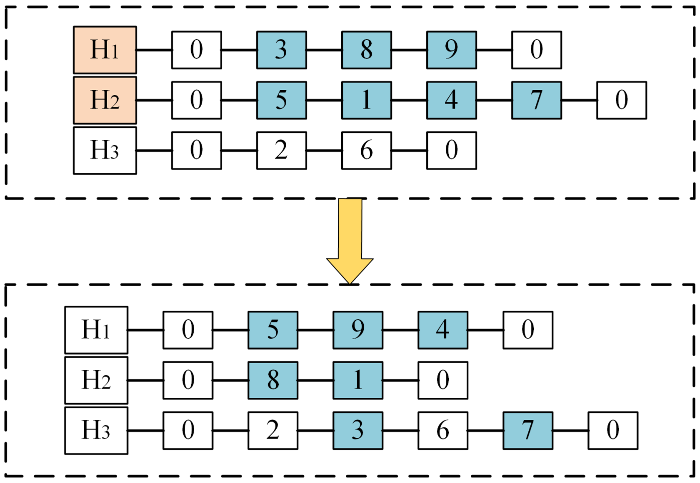


Figure 6. Example of the global search strategy.

Figure 6 illustrates a global search example. In the figure, the rescue sites {3, 8, 9} for helicopter H_1 and {5, 1, 4, 7} for helicopter H_2 are initially excluded from their respective routes. Subsequently, these rescue sites are individually reintroduced at positions that minimize the distance of the current route, generating new chromosomes.

Algorithm 5: Global search strategy

```
Input: a solution
Output: a new solution
1  Randomly select $z$ helicopters from the current solution;
2  Store the set of rescue sites served by the $z$ helicopters in set $R_d$;
3  Delete the $z$ helicopters from the current solution;
4  for each rescue site $i$ in the set $R_d$ do
5      for each helicopter $k$ in the current solution do
6          for each position $p$ in the current helicopter $k$ do
7              if rescue site $i$ can be inserted into the current position $p$ then
8                  Record the current position $p$ in the helicopter;
9              end
10         end
11     end
12     Select the best position $p$ and insert rescue site $i$ into the position $p$ in the helicopter;
13     if rescue site $i$ cannot be inserted then
14         Add a new helicopter and service rescue site $i$;
15     end
16 end
17 return a new solution;
```

4.9. Complexity of the proposed algorithm

The complexity of the proposed algorithm primarily depends on the complexity of each operational step. In the initialization operation, each chromosome in the population is initialized, which has a complexity of $O(n^2)$. In the crossover operation, rescue paths are selected for crossover and duplicate rescue sites are removed. The complexity of the crossover operation is $O(n)$. In the mutation operation, two operators are randomly selected for swapping mutation, with a complexity of $O(1)$. For the repair strategy, each rescue site that violates the time windows constraint is inserted into the current route at the position of minimum distance, resulting in a complexity of $O(n^2)$. Similarly, the local search strategy involves inserting the removed rescue site into the current route at the position of minimum distance, resulting in a complexity of $O(n^2)$. The global search strategy entails inserting the removed rescues site into the current route at the position of minimum distance, with a complexity of $O(n^2)$. Considering all of these factors, the overall complexity of the proposed algorithm is determined to be $O(n^3)$.

5. Experimental results

This section presents the experiments conducted to assess the effectiveness of the proposed algorithms. Firstly, the simulation instances of the H-VRPTW are described. Next, the simulation parameters used in the proposed algorithm were tested and selected. Then, the effectiveness of the local search strategy and the global search strategy in the proposed algorithm was respectively evaluated. Afterward, small instances were utilized to verify the performance of the proposed algorithm. Subsequently, the Solomon benchmark test was conducted to verify the effectiveness of the algorithm. Finally, the proposed algorithm was compared with other algorithms to assess its performance.

5.1. Simulation instances

To better incorporate the system constraints, the simulation test design incorporates an extension of the canonical Solomon instances [42]. The extended instances comprise 56 instances, each of which contains 100 rescue sites. The geographical data used in these instances are similar to those used in the canonical Solomon instances, and include three types of instances: cluster instances, random instances and semi-cluster instances. The extended instances maintain consistency with the Solomon instances in terms of the coordinates of rescue sites, time windows, service times and material demands. The number of victims at each rescue site was randomly generated. All algorithms were evaluated by using a standardized termination criterion, with a maximum elapsed CPU time of 100 seconds. This consistent time duration ensures fairness and comparability in the evaluation process.

5.2. Parameter selection

The experimental parameters consisted of the population size Ps, the crossover probability Pc, the mutation probability Pm and the maximum number of iterations without improvement Lm. Table 1 presents the different values used for the four parameters. An orthogonal array L_{16} was constructed using Taguchi's design of experiments method [43]. For each combination of experimental parameters, the proposed algorithm was executed 30 times independently to obtain the average fitness value, which was recorded as the response variable. The results of the experiments are presented in Table 2. Figure 7 presents the factor level trends of the four parameters. According to the results, the proposed algorithm achieves the best performance when Ps is set to 100, Pc is set to 0.3, Pm is set to 0.3 and Lm is set to 10.

Table 1. The levels of the four parameters.

Parameter	Values			
	1	2	3	4
Ps	50	100	150	200
Pc	0.10	0.30	0.50	0.70
Pm	0.10	0.20	0.30	0.40
Lm	5	10	15	20

Table 2. The response variables for the four parameters.

Ps	Pc	Pm	Lm	Average values
1	1	1	1	1528.04
1	2	2	2	1489.70
1	3	3	3	1479.63
1	4	4	4	1491.98
2	1	2	3	1497.30
2	2	1	4	1482.92
2	3	4	1	1492.38
2	4	3	2	1475.64
3	1	3	4	1526.38
3	2	4	3	1528.69
3	3	1	2	1522.07
3	4	2	1	1528.51
4	1	4	2	1548.95
4	2	3	1	1532.93
4	3	2	4	1558.38
4	4	1	3	1546.37

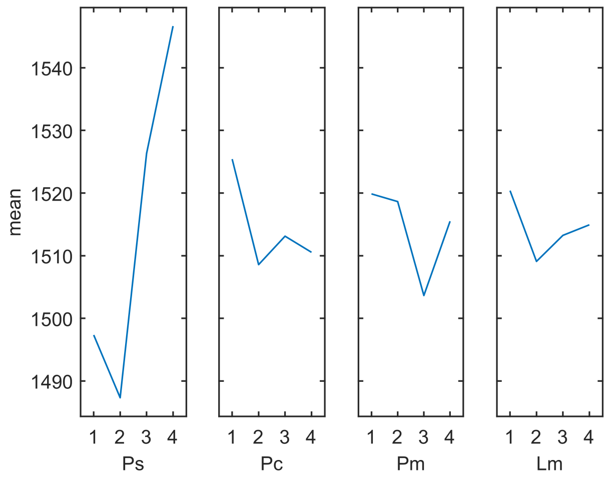


Figure 7. Factor level trends of the four parameters.

5.3. *Efficiency of the local search strategy*

To evaluate the effectiveness of the local search strategy, the IGA without an embedded local search strategy (IGA_NL) was designed to enable comparison with the IGA. The experimental results are presented in Table 3, which includes the name of the instance, the optimal values of the results obtained by the two algorithms, the experimental results for the two algorithms and the relative percentage increase (RPI). The RPI is given by Eq (12).

$$RPI = (f_c - f_b)/f_b \times 100 \tag{12}$$

where f_b represents the best solution found by all compared algorithms, and f_c represents the best solution obtained by a given compared algorithm.

The results in Table 3 show that the IGA achieved 51 optimal solutions out of the 56 instances, while the IGA_NL only obtained five optimal solutions. The IGA obtained an average RPI value of 0.24, which is significantly lower than the average RPI value obtained by the IGA_NL. The results of analysis of variance (ANOVA) for the two different methods are shown in Figure 8. It can be concluded from Table 3 and Figure 8 that incorporating the local search strategy can significantly enhance the search ability of the algorithm.

Table 3. The results of the IGA and IGA_NL.

Instance	Best	Algorithm		RPI	
		IGA	IGA_NL	IGA	IGA_NL
ac101	1465.43	**1465.43**	1608.26	**0**	9.75
ac102	1467.75	**1467.75**	1846.36	**0**	25.80
ac103	1464.08	**1464.08**	2001.75	**0**	36.72
ac104	1482.83	**1482.83**	1919.55	**0**	29.45
ac105	1594.27	**1594.27**	1991.62	**0**	24.92
ac106	1547.95	**1547.95**	1913.76	**0**	23.63
ac107	1753.11	**1753.11**	1953.77	**0**	11.45
ac108	1642.76	**1642.76**	1858.11	**0**	13.11
ac109	1654.43	1658.69	**1654.43**	0.26	**0**
ac201	1580.74	**1580.74**	1920.05	**0**	21.47
ac202	1563.36	**1563.36**	2368.65	**0**	51.51
ac203	1662.92	**1662.92**	2551.24	**0**	53.42
ac204	1593.98	**1593.98**	2487.79	**0**	56.07
ac205	1540.65	**1540.65**	1959.65	**0**	27.20
ac206	1620.99	**1620.99**	2215.69	**0**	36.69
ac207	1521.53	**1521.53**	2166.87	**0**	42.41
ac208	1544.53	**1544.53**	2340.18	**0**	51.51
ar101	1822.29	1854.46	**1822.29**	1.78	**0**
ar102	1636.85	**1636.85**	1805.09	**0**	10.29
ar103	1637.85	**1637.85**	1779.09	**0**	8.62
ar104	1532.78	**1532.78**	2011.33	**0**	31.22
ar105	1680.99	1736.36	**1680.99**	3.29	**0**
ar106	1686.24	**1686.24**	1738.66	**0**	3.11
ar107	1541.48	**1541.48**	1595.80	**0**	3.52
ar108	1419.75	**1419.75**	1701.84	**0**	19.89
ar109	1508.53	**1508.53**	1612.93	**0**	6.92

Continued on next page

Instance	Best	Algorithm		RPI	
		IGA	IGA_NL	IGA	IGA_NL
ar110	1542.03	1604.68	**1542.03**	4.06	**0**
ar111	1427.47	**1427.47**	1669.30	**0**	16.94
ar112	1330.68	**1330.68**	1594.45	**0**	19.82
ar201	1871.04	**1871.04**	2308.66	**0**	23.39
ar202	1731.76	**1731.76**	2143.01	**0**	23.75
ar203	1683.38	**1683.38**	2469.88	**0**	46.72
ar204	1453.17	**1453.17**	2482.92	**0**	70.86
ar205	1681.16	**1681.16**	2080.95	**0**	23.78
ar206	1559.62	**1559.62**	2261.40	**0**	45.00
ar207	1602.26	**1602.26**	2668.41	**0**	66.54
ar208	1460.86	**1460.86**	2585.65	**0**	76.99
ar209	1657.39	**1657.39**	2332.18	**0**	40.71
ar210	1706.34	**1706.34**	2411.02	**0**	41.30
ar211	1482.63	**1482.63**	2486.20	**0**	67.69
arc101	2167.70	**2167.70**	2377.66	**0**	9.69
arc102	1857.56	**1857.56**	2005.42	**0**	7.96
arc103	1942.29	**1942.29**	2119.01	**0**	9.10
arc104	1714.73	**1714.73**	1954.48	**0**	13.98
arc105	2026.26	**2026.26**	2211.13	**0**	9.12
arc106	1946.39	**1946.39**	2061.30	**0**	5.90
arc107	1817.38	**1817.38**	2035.77	**0**	12.02
arc108	1871.91	**1871.91**	2105.14	**0**	12.46
arc201	2191.61	**2191.61**	2650.46	**0**	20.94
arc202	2046.69	**2046.69**	2788.39	**0**	36.24
arc203	2028.24	**2028.24**	3060.31	**0**	50.89
arc204	1931.42	**1931.42**	3601.99	**0**	86.49
arc205	2078.97	**2078.97**	2821.23	**0**	35.70
arc206	1971.11	2048.38	**1971.11**	3.92	**0**
arc207	2027.87	**2027.87**	2769.83	**0**	36.59
arc208	1877.74	**1877.74**	2915.08	**0**	55.24
Mean	1693.85	1697.99	2160.54	0.24	27.94

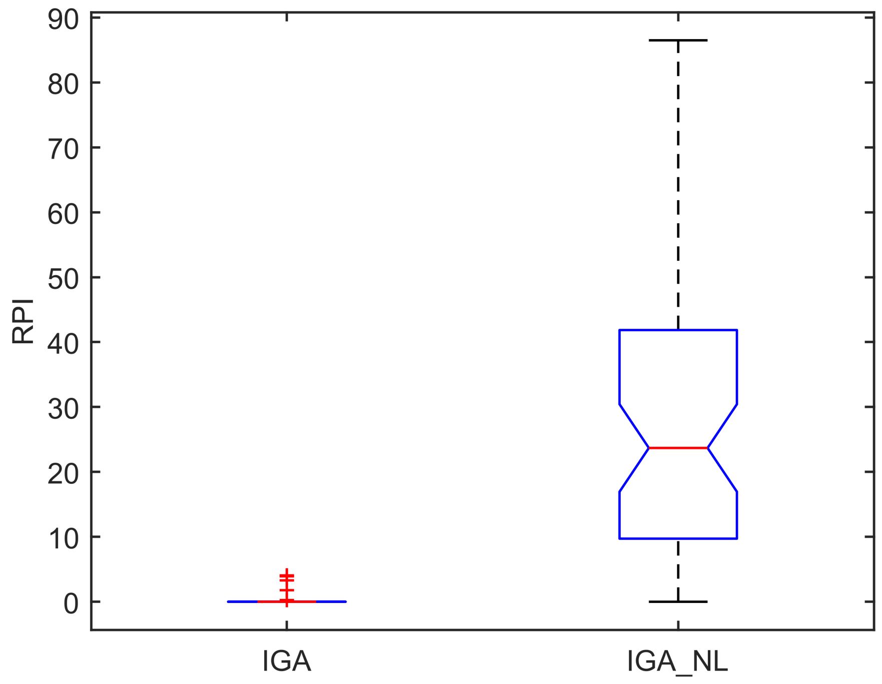


Figure 8. ANOVA results for the IGA and IGA_NL.

5.4. *Efficiency of the global search strategy*

To verify the effectiveness of the proposed global search strategy, the proposed IGA was compared with the IGA without the global search strategy (IGA_NG). The experimental results are shown in Table 4, where a total of 56 instances are considered. The IGA obtained the best solutions for 35 instances, while the IGA_NG obtained the best solutions for 21 instances. The superiority of the proposed global search strategy is further demonstrated by the RPI values in the last two columns. Figure 9 presents the ANOVA comparison between the two methods, confirming that the proposed global search strategy significantly improved the performance.

Table 4. The results of the IGA and IGA_NG.

Instance	Best	Algorithm		RPI	
		IGA	IGA_NG	IGA	IGA_NG
ac101	1465.43	**1465.43**	1503.24	**0**	2.58
ac102	1401.30	1467.75	**1401.30**	4.74	**0**
ac103	1464.08	**1464.08**	1503.36	**0**	2.68
ac104	1482.83	**1482.83**	1639.77	**0**	10.58
ac105	1594.27	**1594.27**	1625.84	**0**	1.98
ac106	1492.18	1547.95	**1492.18**	3.74	**0**

Continued on next page

Instance	Best	Algorithm		RPI	
		IGA	IGA_NG	IGA	IGA_NG
ac107	1753.11	**1753.11**	1767.12	**0**	0.80
ac108	1642.76	**1642.76**	1652.65	**0**	0.60
ac109	1656.73	1658.69	**1656.73**	0.12	**0**
ac201	1580.74	**1580.74**	1607.93	**0**	1.72
ac202	1563.36	**1563.36**	1626.73	**0**	4.05
ac203	1662.92	**1662.92**	1727.69	**0**	3.89
ac204	1593.98	**1593.98**	1602.94	**0**	0.56
ac205	1540.65	**1540.65**	1579.35	**0**	2.51
ac206	1620.99	**1620.99**	1634.47	**0**	0.83
ac207	1521.53	**1521.53**	1528.12	**0**	0.43
ac208	1544.53	**1544.53**	1584.81	**0**	2.61
ar101	1845.67	1854.46	**1845.67**	0.48	**0**
ar102	1617.62	1636.85	**1617.62**	1.19	**0**
ar103	1637.85	**1637.85**	1653.33	**0**	0.95
ar104	1497.79	1532.78	**1497.79**	2.34	**0**
ar105	1623.61	1736.36	**1623.61**	6.94	**0**
ar106	1591.89	1686.24	**1591.89**	5.93	**0**
ar107	1511.86	1541.48	**1511.86**	1.96	**0**
ar108	1419.75	**1419.75**	1464.94	**0**	3.18
ar109	1468.94	1508.53	**1468.94**	2.70	**0**
ar110	1463.71	1604.68	**1463.71**	9.63	**0**
ar111	1427.47	**1427.47**	1453.95	**0**	1.86
ar112	1330.68	**1330.68**	1490.29	**0**	11.99
ar201	1851.65	1871.04	**1851.65**	1.05	**0**
ar202	1731.76	**1731.76**	1737.18	**0**	0.31
ar203	1683.38	**1683.38**	1687.94	**0**	0.27
ar204	1453.17	**1453.17**	1539.67	**0**	5.95
ar205	1681.16	**1681.16**	1716.45	**0**	2.10
ar206	1559.62	**1559.62**	1703.55	**0**	9.23
ar207	1564.44	1602.26	**1564.44**	2.42	**0**
ar208	1379.81	1460.86	**1379.81**	5.87	**0**
ar209	1626.36	1657.39	**1626.36**	1.91	**0**
ar210	1706.34	**1706.34**	1732.59	**0**	1.54
ar211	1482.63	**1482.63**	1532.57	**0**	3.37
arc101	2106.99	2167.70	**2106.99**	2.88	**0**
arc102	1857.56	**1857.56**	2008.94	**0**	8.15
arc103	1851.89	1942.29	**1851.89**	4.88	**0**
arc104	1681.82	1714.73	**1681.82**	1.96	**0**
arc105	2026.26	**2026.26**	2064.04	**0**	1.86
arc106	1946.39	**1946.39**	1957.51	**0**	0.57

Continued on next page

Instance	Best	Algorithm		RPI	
		IGA	IGA_NG	IGA	IGA_NG
arc107	1817.38	**1817.38**	1949.73	**0**	7.28
arc108	1765.45	1871.91	**1765.45**	6.03	**0**
arc201	2157.37	2191.61	**2157.37**	1.59	**0**
arc202	2046.69	**2046.69**	2185.35	**0**	6.77
arc203	2028.24	**2028.24**	2062.76	**0**	1.70
arc204	1931.42	**1931.42**	2062.55	**0**	6.79
arc205	2078.97	**2078.97**	2284.27	**0**	9.88
arc206	1920.87	2048.38	**1920.87**	6.64	**0**
arc207	2027.87	**2027.87**	2174.11	**0**	7.21
arc208	1877.74	**1877.74**	1983.51	**0**	5.63
Mean	1676.10	1697.99	1716.20	1.34	2.36

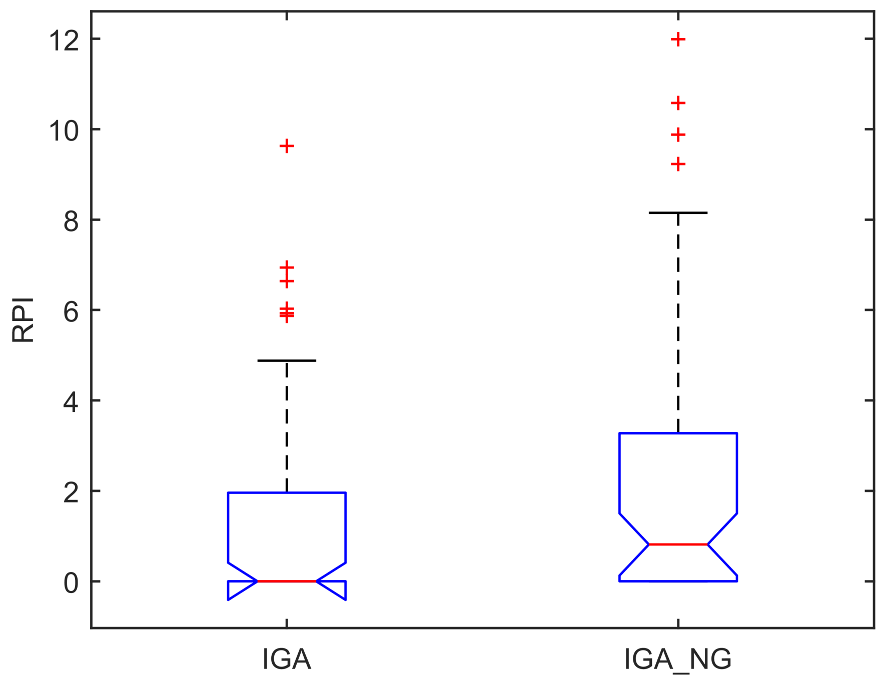


Figure 9. ANOVA results for the IGA and IGA_NG.

5.5. Comparison of small instances

To further assess the feasibility of the IGA, the IBM ILOG CPLEX Optimization Studio solver was utilized to solve small instances; its performance was compared with the IGA. The small instances were generated by selecting a subset of rescue sites from the experimental instances, with the number of rescue sites ranging from four to 20. The name of each instance indicates the number of rescue sites it contains. For example, the instance labeled as “rs4” indicates that it consists of four rescue sites.

The experimental results, as shown in Table 5, demonstrate that CPLEX finds the optimal solution for several small instances. The proposed IGA also achieves the same solution values as CPLEX for instances with 4, 6, 8, 10, 14, 18, and 20 rescue sites. This confirms the feasibility of the proposed IGA approach for solving the VRPTW. Analyzing the results from the perspective of the RPI, it can be observed that the solutions obtained by the IGA exhibit a minor deviation from those obtained via CPLEX. The average RPI was merely 0.28, which demonstrates the effectiveness of the IGA.

Table 5. Comparison of IGA and CPLEX solutions.

Instance	Best	Algorithm		RPI	
		IGA	CPLEX	IGA	CPLEX
rs4	70.10	**70.10**	**70.10**	**0**	**0**
rs6	121.04	**121.04**	**121.04**	**0**	**0**
rs8	133.13	**131.13**	**131.13**	**0**	**0**
rs10	152.09	**152.09**	**152.09**	**0**	**0**
rs12	178.63	180.97	**178.63**	1.31	**0**
rs14	184.15	**184.15**	**184.15**	**0**	**0**
rs16	204.03	206.57	**204.03**	1.24	**0**
rs18	224.92	**224.92**	**224.92**	**0**	**0**
rs20	241.75	**241.75**	**241.75**	**0**	**0**
Mean	167.76	168.30	167.76	0.28	0

5.6. *Solomon benchmark test*

Previous studies often use classical benchmark datasets to test the performance of algorithms [44]. To validate the effectiveness and search capability of the proposed algorithm, it was compared with other well-known algorithms, including the TS [45], ACO [46], hybrid GA (HGA) [47], and SA [48]. The comparison was conducted using the widely recognized Solomon benchmark dataset, which is commonly used to evaluate vehicle routing algorithms. The obtained results were compared with the current international optimal values. In order to optimize the performance of the comparison algorithm, we meticulously fine-tuned its parameters; the outcomes for each parameter are presented in Table 6.

The Solomon benchmark dataset consists of various types of information, including vehicle capacity, customer coordinates, customer demand, time windows and service time. The geographic data in the Solomon instances can be categorized into three types: clustered instances labeled as “cl” and “c2”, random instances labeled as “r1” and “r2” and semi-clustered instances labeled as “rc1” and “rc2”.

Table 6. Tuned parameters for the algorithms.

Algorithm	Parameters
TS	Tabu length $L = 9$
	Tabu list clearing of iteration times $m = 50$
ACO	Pheromone factor $\alpha = 1$
	Expectation factor $\beta = 3$
	Urgency of the service node factor $\gamma = 2$
	Control of the exploitation factor $q0 = 0.5$
	Pheromone volatility coefficient $\rho = 0.5$
HGA	Population size $Ps = 100$
	Crossover probability $Pc = 0.95$
	Mutation probability $Pm = 0.05$
SA	Temperature $T0 = 5$
	Final temperature $Tf = 0.05$
	Coefficient of control $alpha = 0.9$

The experimental results are presented in Table 7. Although a slight gap exists between the results obtained by using the IGA and the current international optimal values, this gap is relatively small in terms of the RPI, with an average RPI value of only 2.99. Moreover, the proposed algorithm exhibited significantly better performance compared to other algorithms, with an average RPI of 0.289, 0.080, 0.137 and 0.109 times that of the TS, ACO, HGA and SA algorithm, respectively. The ANOVA results presented in Figure 10 demonstrate that the proposed algorithm exhibits superior effectiveness and stability. The experimental results clearly demonstrate the competitive performance of the proposed IGA in terms of solving the VRPTW.

Table 7. Comparison of algorithms using the Solomon benchmark dataset.

Instance	International best	Algorithm					RPI				
		IGA	TS	ACO	HGA	SA	IGA	TS	ACO	HGA	SA
c101	828.94	**853.89**	892.25	1180.76	1001.04	1017.96	**3.01**	7.64	42.44	20.76	22.80
c102	828.94	**863.97**	951.73	1221.41	1064.43	1141.58	**4.23**	14.81	47.35	28.41	37.72
c103	828.06	**869.45**	949.88	1194.42	1048.17	1277.15	**5.00**	14.71	44.24	26.58	54.24
c104	824.78	874.09	**856.92**	1071.06	53.92	1284.72	5.98	**3.90**	29.86	15.66	55.77
c105	828.94	**831.87**	900.83	1133.87	1019.24	1401.60	**0.35**	8.67	36.79	22.96	69.08
c106	828.94	**833.31**	875.33	1069.86	994.80	1172.32	**0.53**	5.60	29.06	20.01	41.42
c107	828.94	**892.70**	920.52	1032.48	1041.22	1231.11	**7.69**	11.05	24.55	25.61	48.52
c108	828.94	**842.66**	932.33	1168.82	1029.51	1275.85	**1.66**	12.47	41.00	24.20	53.91
c109	828.94	**878.38**	882.53	1209.93	1085.97	1256.70	**5.96**	6.46	45.96	31.01	51.60
c201	591.56	**611.81**	677.03	891.56	772.77	951.48	**3.42**	14.45	50.71	30.63	60.84
c202	591.56	**603.42**	662.57	880.63	783.11	911.38	**2.00**	12.00	48.87	32.38	54.06
c203	591.17	608.67	**605.75**	897.92	815.97	825.03	2.96	**2.47**	51.89	38.03	39.56

Continued on next page

Instance	International best	Algorithm					RPI				
		IGA	TS	ACO	HGA	SA	IGA	TS	ACO	HGA	SA
c204	590.60	605.25	**601.91**	811.13	792.75	950.34	2.48	**1.91**	37.34	34.23	60.91
c205	588.88	**593.56**	672.36	763.27	670.89	848.62	**0.79**	14.18	29.61	13.93	44.11
c206	588.49	**600.91**	616.92	744.59	746.93	882.59	**2.11**	4.83	26.53	26.92	49.98
c207	588.29	**599.10**	632.17	848.84	822.78	822.46	**1.84**	7.46	44.29	39.86	39.81
c208	588.32	**597.87**	676.15	849.39	811.79	900.56	**1.62**	14.93	44.38	37.98	53.07
r101	1650.80	**1685.99**	1776.00	2100.17	1803.82	1877.78	**2.13**	7.58	27.22	9.27	13.75
r102	1486.12	**1506.06**	1596.94	1921.18	1654.18	1689.17	**1.34**	7.46	29.27	11.31	13.66
r103	1292.68	**1309.08**	1374.75	1679.57	1460.61	1436.05	**1.27**	6.35	29.93	12.99	11.09
r104	1007.31	**1037.79**	1092.32	1230.71	1307.41	1285.36	**3.03**	8.44	22.18	29.79	27.60
r105	1377.11	1452.18	**1382.16**	1832.38	1518.69	1536.34	5.45	**0.37**	33.06	10.28	11.56
r106	1252.03	1303.89	**1278.55**	1527.96	1442.99	1423.96	4.14	**2.12**	22.04	15.25	13.73
r107	1104.66	**1114.82**	1199.45	1373.92	1282.56	1299.39	**0.92**	8.58	24.37	16.10	17.63
r108	960.88	**979.99**	1335.61	1227.23	1214.43	1118.18	**1.99**	39.00	27.72	26.39	16.37
r109	1194.73	**1255.15**	1354.57	1385.94	1416.81	1405.59	**5.06**	13.38	16.00	18.59	17.61
r110	1118.84	**1179.18**	1230.78	1357.30	1326.29	1215.41	**5.39**	10.01	21.31	18.54	8.63
r111	1096.75	**1107.01**	1106.92	1265.13	1254.83	1284.40	**0.94**	0.98	15.35	14.41	17.11
r112	982.14	**1047.18**	1110.62	1322.18	1183.21	1149.07	**6.62**	13.08	34.62	20.47	17.00
r201	1252.37	**1269.09**	1426.01	1960.14	1460.15	1427.60	**1.34**	13.86	56.51	16.59	13.99
r202	1191.70	**1216.73**	1296.98	1773.74	1311.93	1321.12	**2.10**	8.83	48.84	10.09	10.86
r203	939.50	**979.58**	1108.98	1321.34	1193.57	1228.11	**4.27**	18.04	40.64	27.04	30.72
r204	825.52	**859.58**	911.66	1055.90	935.25	1042.18	**4.13**	10.43	27.91	13.29	26.25
r205	994.43	**1028.75**	1117.82	1156.38	1299.73	1191.16	**3.45**	12.41	16.29	30.70	19.78
r206	906.14	**992.74**	1071.62	1240.84	1196.74	1169.12	**4.04**	18.26	36.94	32.07	29.02
r207	890.61	**958.88**	1002.22	1165.64	1101.88	1065.96	**7.67**	12.53	30.88	23.72	19.69
r208	726.82	**758.14**	867.29	1122.97	985.58	931.30	**4.31**	19.33	54.50	35.60	28.13
r209	909.16	**915.36**	1007.71	1433.75	1159.87	1100.13	**0.68**	10.84	57.70	27.58	21.01
r210	939.37	**958.19**	1045.30	1546.78	1166.22	1094.28	**2.00**	11.28	64.66	24.15	16.49
r211	885.71	919.37	**915.24**	1173.67	1062.75	963.66	3.80	**3.33**	32.51	19.99	8.80
rc101	1696.95	**1756.31**	1862.38	2173.13	1865.15	1913.87	**3.50**	9.75	28.06	9.91	12.78
rc102	1554.75	**1571.54**	1727.85	2155.68	1744.86	1734.85	**1.08**	11.13	38.65	12.23	11.58
rc103	1261.67	**1289.22**	1494.44	1820.56	1469.70	1590.02	**2.18**	18.45	44.30	16.49	26.03
rc104	1135.48	**1175.74**	1320.59	1587.93	1383.91	1440.84	**3.55**	16.30	39.85	21.88	26.89
rc105	1629.44	**1633.97**	1666.53	2174.46	1674.26	1733.74	**0.28**	2.28	33.45	2.75	6.40
rc106	1424.73	**1526.33**	1585.73	1916.42	1597.25	1563.19	**7.13**	11.30	34.51	12.11	9.72
rc107	1230.48	**1254.66**	1450.95	1746.27	1496.80	1447.29	**1.97**	17.92	41.92	21.64	17.62
rc108	1139.82	**1206.98**	1327.56	1585.90	1348.84	1319.33	**5.89**	16.47	39.14	18.34	15.75
rc201	1406.94	**1424.86**	1519.46	2246.91	1707.92	1520.85	**1.27**	8.00	59.70	21.39	8.10
rc202	1365.65	**1392.23**	1409.22	1976.80	1597.64	1535.55	**1.95**	3.19	44.75	16.99	12.44
rc203	1049.62	**1068.27**	1107.95	1310.47	1283.23	1241.32	**1.78**	5.56	24.85	22.26	18.26
rc204	798.46	**812.55**	817.68	1071.87	990.79	1178.41	**1.76**	2.41	34.24	24.09	47.59

Continued on next page

Instance	International best	Algorithm					RPI				
		IGA	TS	ACO	HGA	SA	IGA	TS	ACO	HGA	SA
rc205	1297.65	**1321.77**	1419.42	1537.97	1481.42	1547.04	**1.89**	9.38	18.52	14.16	19.22
rc206	1146.32	**1169.20**	1244.16	1687.14	1497.67	1280.87	**2.00**	8.54	47.18	30.65	11.74
rc207	1061.14	**1091.66**	1263.05	1536.65	1349.97	1310.68	**2.88**	19.03	44.81	27.22	23.52
rc208	828.14	**834.05**	865.97	1266.76	952.80	1007.73	**0.71**	4.57	52.96	15.05	21.69
Mean	1021.19	1051.34	1124.99	1391.78	1225.66	1263.79	2.99	10.33	37.00	21.80	27.45

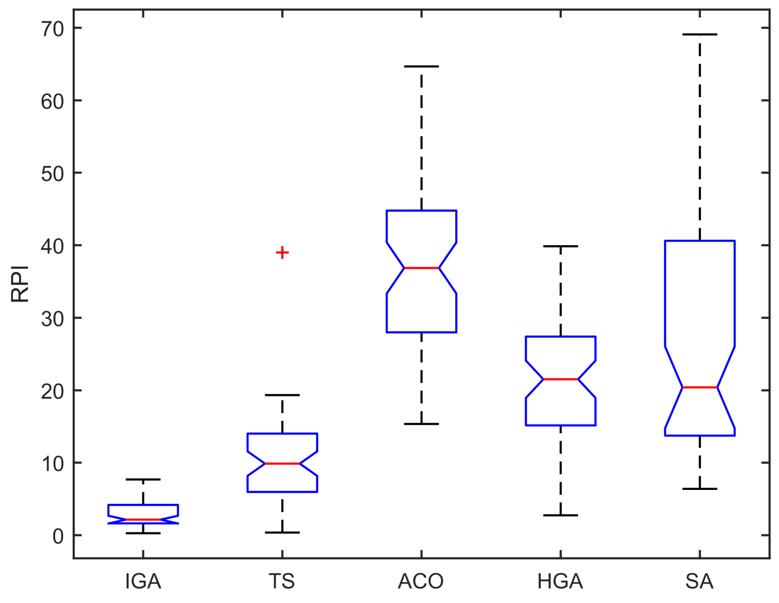


Figure 10. ANOVA results for the algorithms on the Solomon benchmark dataset.

5.7. Comparison of IGA with other algorithms

To evaluate the effectiveness of the IGA, the proposed algorithm was compared with the TS algorithm, ACO algorithm, HGA and SA. In the TS algorithm, the tabu list is an $N \times N$ matrix, where N represents the number of customers. The tabu objects are vertex pairs (j_1, j_2) involved in the neighborhood operation, and their corresponding tabu lengths are stored in the tabu list. When a candidate solution (S^{candi}) corresponding to a vertex pair (j_1, j_2) is selected as the current solution (S^{now}) during the neighborhood operation, the corresponding tabu length is assigned to the matrix element (j_1, j_2). The tabu length is then decremented after each iteration until it reaches 0.

The experimental results are presented in Table 8, demonstrating that the IGA achieved superior solutions for 41 out of 56 instances, accounting for approximately 73.2% of the total instances. Notably, the average RPI of the IGA was only 1.28. Moreover, the proposed algorithm exhibited significantly better performance than the other algorithms, with an average RPI of 0.178, 0.027, 0.075 and 0.041 times those obtained via the TS, ACO, HGA, and SA algorithm, respectively. The effectiveness and stability of the proposed algorithm are found to be considerably higher than those of the other algorithms, as confirmed by the ANOVA results depicted in Figure 11. Furthermore, to illustrate the convergence ability of the algorithm when solving the H-VRPTW, Figure 12 displays the convergence curves for four selected instances: ac102, ac208, ar204 and arc106. These convergence curves clearly demonstrate the remarkable convergence ability of the IGA. In addition, Figure 13 shows the four charts of selected instances: ac106, ac201, ar108 and arc204, further affirming the effectiveness of the proposed algorithm in solving the H-VRPTW.

As the above experimental results and analysis show, the proposed IGA is an effective algorithm for solving the H-VRPTW. The reasons can be concluded as follows. First, a cooperative initialization strategy is employed to generate the population that possesses both high quality and diversity, enabling the IGA to explore a wide range of solutions effectively. Additionally, the incorporation of a local search strategy significantly enhances the exploitation capability of the IGA, leading to improved quality of the solution. Finally, the introduction of a global search strategy helps to prevent the solution from falling into local optima and improves the global search capability of the IGA. Based on the above analyses, the cooperative initialization strategy, the effective local search strategy and the global search strategy are reasons for the outstanding performance of the IGA.

Table 8. The results of comparison with the TS, ACO, HGA and SA.

Instance	Best	Algorithm					RPI				
		IGA	TS	ACO	HGA	SA	IGA	TS	ACO	HGA	SA
ac101	1455.70	1465.43	**1455.70**	2196.51	1821.62	1824.45	0.67	**0**	50.89	25.14	25.33
ac102	1467.75	**1467.75**	1633.37	2400.79	1811.20	1770.30	**0**	11.28	63.57	23.40	20.61
ac103	1464.08	**1464.08**	1916.85	2633.05	1830.23	2077.56	**0**	30.93	79.84	25.01	41.90
ac104	1482.83	**1482.83**	1515.53	2228.27	1841.89	2122.12	**0**	2.21	50.27	25.01	43.11
ac105	1584.88	1594.27	**1584.88**	1994.79	1880.82	1912.08	0.59	**0**	25.86	18.67	20.65
ac106	1547.95	**1547.95**	1599.01	2233.64	1811.51	1959.46	**0**	3.30	44.30	17.03	26.58
ac107	1578.51	1753.11	**1578.51**	2351.18	1991.71	2022.07	11.06	**0**	48.95	26.18	28.10
ac108	1549.68	1642.76	**1549.68**	1873.25	1800.83	1884.54	6.01	**0**	20.88	16.21	21.61
ac109	1490.78	1658.69	**1490.70**	1995.30	1786.18	2030.72	11.26	**0**	33.84	19.82	36.22
ac201	1580.74	**1580.74**	1635.63	2228.17	2034.78	2274.18	**0**	3.47	40.96	28.72	43.87
ac202	1563.36	**1563.36**	1637.29	2432.74	2007.43	2315.95	**0**	4.73	55.61	28.40	48.14
ac203	1510.38	1662.92	**1510.38**	2357.13	2058.05	2349.39	10.10	**0**	56.06	36.26	55.55
ac204	1593.98	**1593.98**	1653.74	2435.95	2004.36	2494.30	**0**	3.75	52.82	25.75	56.48
ac205	1477.92	1540.65	**1477.92**	2358.25	1939.53	2319.41	4.24	**0**	59.57	31.23	56.94
ac206	1620.99	**1620.99**	1626.60	2377.39	1940.02	2435.91	**0**	0.35	46.66	19.68	50.27
ac207	1484.86	1521.53	**1484.86**	2121.41	1828.67	2326.36	2.47	**0**	42.87	23.15	56.67
ac208	1544.53	**1544.53**	1636.52	2233.59	1867.09	2262.77	**0**	5.96	44.61	20.88	46.50
ar101	1809.25	1854.46	**1809.25**	2702.87	1938.82	1828.59	2.50	**0**	49.39	7.16	1.07
ar102	1636.85	**1636.85**	1656.97	2630.13	1763.33	1719.50	**0**	1.239	60.68	7.73	5.05
ar103	1637.85	**1637.85**	2068.84	2230.11	1709.41	1705.97	**0**	26.31	36.16	4.37	4.16
ar104	1447.26	1532.78	**1447.26**	1835.58	1582.75	1658.14	5.91	**0**	26.83	9.36	14.57
ar105	1684.61	1736.36	**1684.61**	2323.50	1715.81	1709.21	3.07	**0**	37.93	1.85	1.46
ar106	1659.15	1686.24	1724.45	2088.68	**1659.15**	1713.98	1.63	3.94	25.89	**0**	3.30

Continued on next page

Instance	Best	Algorithm					RPI				
		IGA	TS	ACO	HGA	SA	IGA	TS	ACO	HGA	SA
ar107	1541.48	**1541.48**	1723.28	1939.89	1551.27	1570.04	**0**	11.79	25.85	0.64	1.85
ar108	1419.75	**1419.75**	1561.72	1719.14	1487.39	1564.56	**0**	10.00	21.09	4.76	10.20
ar109	1508.53	**1508.53**	1668.01	1987.56	1581.34	1670.86	**0**	10.57	31.75	4.83	10.76
ar110	1604.68	**1604.68**	1682.98	1871.64	1669.10	1654.65	**0**	4.88	16.64	4.01	3.11
ar111	1427.47	**1427.47**	1725.69	1968.65	1511.69	1523.06	**0**	20.89	37.91	5.90	6.70
ar112	1330.68	**1330.68**	1609.20	1621.22	1515.37	1573.78	**0**	20.93	21.83	13.88	18.27
ar201	1871.04	**1871.04**	2187.61	3089.48	2315.91	2702.34	**0**	16.92	65.12	23.78	44.43
ar202	1731.76	**1731.76**	2032.40	2907.14	2051.94	2504.20	**0**	17.36	67.87	18.49	44.60
ar203	1683.38	**1683.38**	1685.10	3020.26	2090.83	2419.49	**0**	0.10	79.42	24.20	43.73
ar204	1453.17	**1453.17**	1641.42	2460.35	1966.83	2112.92	**0**	12.95	69.31	35.35	45.40
ar205	1681.16	**1681.16**	1766.16	2697.61	2227.62	2112.92	**0**	5.06	60.46	32.50	56.77
ar206	1559.62	**1559.62**	1692.58	2637.81	2070.37	2335.70	**0**	8.53	69.13	32.75	49.76
ar207	1602.26	**1602.26**	1763.93	2626.03	1998.26	2226.69	**0**	10.09	63.90	24.72	38.97
ar208	1460.86	**1460.86**	1519.17	2409.22	1852.45	2237.09	**0**	3.99	64.92	26.81	53.14
ar209	1657.39	**1657.39**	1833.05	2695.11	2017.70	2312.15	**0**	10.60	62.61	21.74	39.51
ar210	1706.34	**1706.34**	1765.23	2695.93	2099.48	2261.15	**0**	3.45	57.99	23.04	32.51
ar211	1482.63	**1482.63**	1692.78	2395.10	1912.46	2228.67	**0**	14.17	61.54	28.99	50.32
arc101	1992.26	2167.70	**1992.26**	2952.81	2165.05	2102.98	8.81	**0**	48.21	8.67	5.56
arc102	1857.56	**1857.56**	2153.43	2632.86	2043.57	2052.98	**0**	15.94	41.74	10.01	10.52
arc103	1942.29	**1942.29**	2013.90	2407.75	1963.19	2193.77	**0**	3.69	23.96	1.08	12.94
arc104	1714.73	**1714.73**	1835.47	2169.45	1818.20	1873.67	**0**	7.04	26.51	6.03	9.27
arc105	2026.26	**2026.26**	2092.61	2576.05	2119.26	2138.42	**0**	3.27	27.13	4.59	5.54
arc106	1946.39	**1946.39**	2278.06	2465.41	1988.13	2101.47	**0**	17.04	26.67	2.14	7.97
arc107	1817.38	**1817.38**	1892.43	2298.93	1901.54	2034.05	**0**	4.13	26.50	4.63	11.92
arc108	1871.91	**1871.91**	2093.49	2248.72	1941.13	1964.23	**0**	11.84	20.13	3.70	4.92
arc201	2191.61	**2191.61**	2446.45	3924.36	2650.47	3234.19	**0**	11.63	79.06	20.94	47.57
arc202	2046.69	**2046.69**	2347.31	3685.62	2505.02	3162.45	**0**	14.69	80.08	22.39	54.52
arc203	2013.35	2028.24	**2013.35**	3327.14	2325.68	2977.31	0.74	**0**	65.25	15.51	47.88
arc204	1931.42	**1931.42**	1942.74	2705.06	2237.96	2951.20	**0**	0.59	40.06	15.87	52.80
arc205	2078.97	**2078.97**	2417.22	3720.10	2562.86	3227.89	**0**	16.27	78.94	23.28	55.26
arc206	2048.38	**2048.38**	2301.28	3256.33	2437.83	3254.05	**0**	12.35	58.97	19.01	58.86
arc207	2027.87	**2027.87**	2133.07	3155.49	2335.31	2995.21	**0**	5.19	55.61	15.16	47.70
arc208	1834.64	1877.74	**1834.64**	2700.97	2077.34	2603.18	2.35	**0**	47.22	13.23	41.89
Mean	1677.46	1697.99	1798.51	2486.28	1957.46	2179.50	1.28	7.20	47.82	17.02	30.95

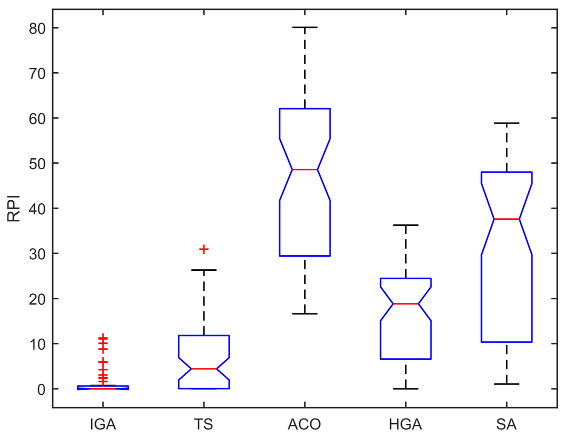
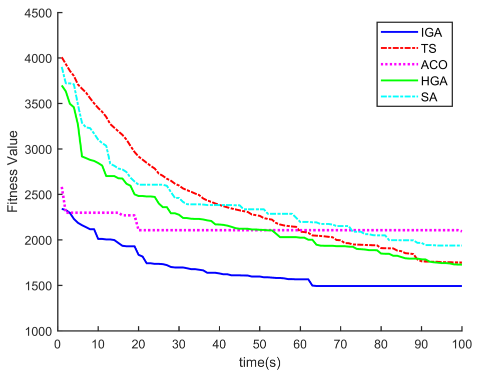
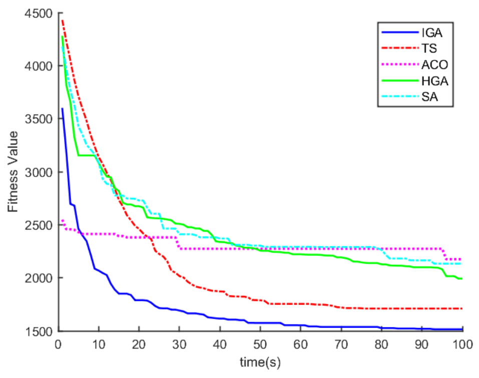


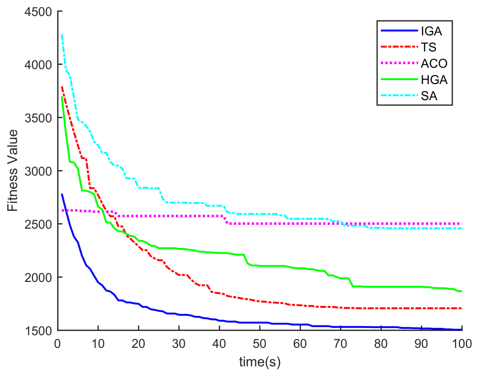
Figure 11. ANOVA results for the IGA, TS, ACO, HGA and SA algorithm.



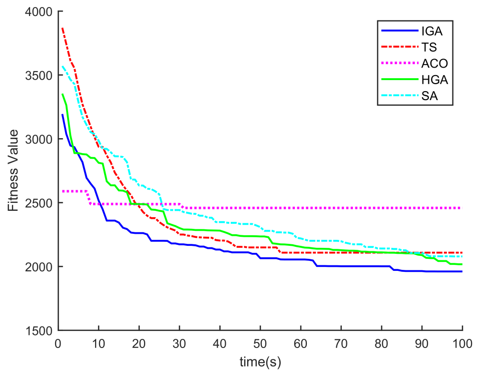
(a) Curve for instance ac102



(b) Curve for instance ac208

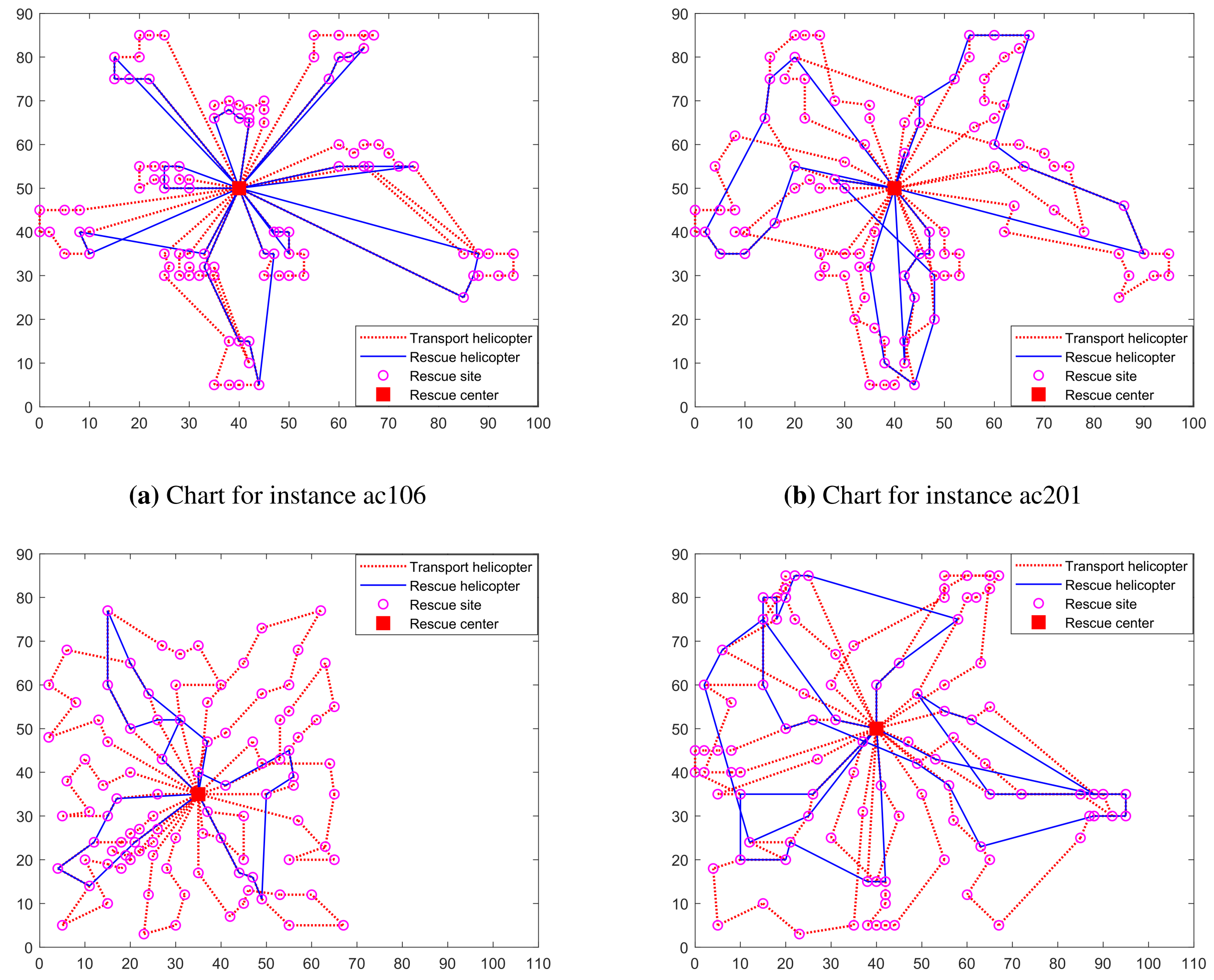


(c) Curve for instance ar204



(d) Curve for instance arc106

Figure 12. Comparisons of convergence curves for the IGA, TS, ACO, HGA and SA algorithm.



(a) Chart for instance ac106

(b) Chart for instance ac201

(c) Chart for instance ar108

(d) Chart for instance arc204

Figure 13. Charts of the helicopter routing schemes of the four instances.

5.8. Theoretical implications and managerial insights

The effective optimization of the H-VRPTW considering the minimum flight distance of the helicopter improves the efficiency of post-disaster rescue and is of great significance for human safety. Therefore, this paper establishes a mathematical model of the H-VRPTW, considering load constraints, material demand, transportation of the wounded, time window constraints and travel distance. To solve the H-VRPTW, an effective IGA has been proposed; the IGA shows better convergence and optimization performance when solving the H-VRPTW. This study provides a new perspective from which disaster management officials can formulate more refined dispatching schemes, which has practical significance.

One managerial insight is that the research results of this paper have significant reference value for solving the helicopter dispatching problem in post-disaster rescue. Another managerial insight is that the helicopter dispatching scheme can effectively arrange rescue tasks for decision-makers and improve rescue efficiency. Finally, the proposed IGA is effective and applicable for solving dispatching schemes for helicopters in post-disaster rescue.

6. Conclusions

In this study, the dispatching helicopters problem in post-disaster rescue is addressed. To solve this problem, a reasonable mathematical model of the H-VRPTW that considers the minimum flight distance of the helicopters is proposed, and an effective IGA has been designed to solve the H-VRPTW. In the IGA, a cooperative initialization strategy is included to generate the initial population. Subsequently, a local search strategy is presented to improve the exploitation ability. Furthermore, a global search strategy is embedded to enhance the global search performance. In the simulation experiments, the extended Solomon instances were used to verify the effectiveness of the proposed algorithm. The proposed algorithm was compared with four existing algorithms to assess its competitiveness. The experimental results show that the IGA yielded an average RPI value that was 0.178, 0.027, 0.075 and 0.041 times those of the TS algorithm, ACO algorithm, HGA and SA algorithm, respectively. The experimental results confirm that the proposed algorithm has higher competitive performance.

One limitation of this study is that the optimization objective was limited to consider the minimum flight distance of the helicopters, without considering other objectives, such as economic cost and energy consumption. To improve the research, a multi-objective dispatching model can be explored to optimize multiple objectives simultaneously in the post-disaster rescue. Another limitation is the lack of consideration for external environmental factors and emergencies, such as situations in which helicopters are unable to reach certain areas. To enhance the research, future studies can employ a dynamic programming approach to simulate the dispatching problem for the post-disaster rescue. Finally, the performance of the proposed algorithm needs further improvement. In future research, the proposed algorithm can be enhanced by combining it with other algorithms, such as the SA algorithm.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (52205529, 61803192), the Natural Science Foundation of Shandong Province (ZR2021QE195, ZR2021MD090, ZR2018BD008, ZR2016FL13), the Special Fund Plan for Local Science and Technology Development Lead by Central Authority and the Guangyue Youth Scholar Innovation Talent Program support received from Liaocheng University (LCUGYTD2022-03).

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. F. Wex, G. Schryen, S. Feuerriegel, D. Neumann, Emergency response in natural disaster management: allocation and scheduling of rescue units, *Eur. J. Oper. Res.*, **235** (2014), 697–708. https://doi.org/10.1016/j.ejor.2013.10.029
2. G. Tian, A. M. Fathollahi-Fard, Y. Ren, Z. Li, X. Jiang, Multi-objective scheduling of priority-based rescue vehicles to extinguish forest fires using a multi-objective discrete gravitational search algorithm, *Inf. Sci.*, **608** (2022), 578–596. https://doi.org/10.1016/j.ins.2022.06.052
3. Z. Mahtab, A. Azeem, S. M. Ali, S. K. Paul, A. M. Fathollahi-Fard, Multi-objective robust-stochastic optimisation of relief goods distribution under uncertainty: a real-life case study, *Int. J. Syst. Sci. Oper. Logist.*, **9** (2022), 241–262. https://doi.org/10.1080/23302674.2021.1879305
4. Y. Okuno, K. Kobayashi, H. Ishii, Development of a helicopter operations management system for disaster relief missions, *J. Am. Helicopter Soc.*, **61** (2016), 1–9. https://doi.org/10.4050/JAHS.61.012006
5. A. Andreeva-Mori, K. Kobayashi, M. Shindo, Particle swarm optimization/greedy-search algorithm for helicopter mission assignment in disaster relief, *J. Aerosp. Inf. Syst.*, **12** (2015), 646–660. https://doi.org/10.2514/1.I010362
6. Q. Shao, C. Xu, Y. Zhu, Multi-helicopter search and rescue route planning based on strategy optimization algorithm, *Int. J. Pattern Recognit Artif Intell.*, **33** (2019), 1950002. https://doi.org/10.1142/S0218001419500022
7. J. Zhang, Y. Zhu, X. Li, M. Ming, W. Wang, T. Wang, Multi-trip time-dependent vehicle routing problem with split delivery, *Mathematics*, **10** (2022), 3527. https://doi.org/10.3390/math10193527
8. W. VanDeventer, E. Jamei, G. S. Thirunavukkarasu, M. Seyedmahmoudian, T. K. Soon, B. Horan, et al., Short-term PV power forecasting using hybrid GASVM technique, *Renewable Energy*, **140** (2019), 367–379. https://doi.org/10.1016/j.renene.2019.02.087
9. S. B. Jouida, S. Krichen, A genetic algorithm for supplier selection problem under collaboration opportunities, *J. Exp. Theor. Artif. Intell.*, **34** (2022), 53–79. https://doi.org/10.1080/0952813X.2020.1836031
10. L. Meng, W. Cheng, B. Zhang, W. Zou, W. Feng, P. Duan, An improved genetic algorithm for solving the multi-AGV flexible job shop scheduling problem, *Sensors*, **23** (2023), 3815. https://doi.org/10.3390/s23083815
11. J. Ochelska-Mierzejewska, A. Poniszewska-Marańda, W. Marańda, Selected genetic algorithms for vehicle routing problem solving, *Electronics*, **10** (2021), 3147. https://doi.org/10.3390/electronics10243147
12. H. Park, D. Son, B. Koo, B. Jeong, Waiting strategy for the vehicle routing problem with simultaneous pickup and delivery using genetic algorithm, *Expert Syst. Appl.*, **165** (2021), 113959. https://doi.org/10.1016/j.eswa.2020.113959
13. M. A. Mohammed, M. K. A. Ghani, R. I. Hamed, S. Mostafa, M. S. Ahmad, D. A. Ibrahim, Solving vehicle routing problem by using improved genetic algorithm for optimal solution, *J. Comput. Sci.*, **21** (2017), 255–262. https://doi.org/10.1016/j.jocs.2017.04.003

14. G. B. Dantzig, J. H. Ramser, The truck dispatching problem, *Manage. Sci.*, **6** (1959), 80–91. https://doi.org/ 10.1287/mnsc.6.1.80

15. A. M. Fathollahi-Fard, A. Ahmadi, B. Karimi, Sustainable and robust home healthcare logistics: a response to the COVID-19 pandemic, *Symmetry*, **14** (2022), 193. https://doi.org/10.3390/sym14020193

16. A. M. Fathollahi-Fard, M. Hajiaghaei-Keshteli, R. Tavakkoli-Moghaddam, N. R. Smith, Bi-level programming for home health care supply chain considering outsourcing, *J. Ind. Inf. Integr.*, **25** (2022), 100246. https://doi.org/10.1016/j.jii.2021.100246

17. M. Mojtahedi, A. M. Fathollahi-Fard, R. Tavakkoli-Moghaddam, S. Newton, Sustainable vehicle routing problem for coordinated solid waste management, *J. Ind. Inf. Integr.*, **23** (2021), 100220. https://doi.org/10.1016/j.jii.2021.100220

18. F. E. Zulvia, R. J. Kuo, D. Y. Nugroho, A many-objective gradient evolution algorithm for solving a green vehicle routing problem with time windows and time dependency for perishable products, *J. Cleaner Prod.*, **242** (2020), 118428. https://doi.org/10.1016/j.jclepro.2019.118428

19. A. Expósito, J. Brito, J. A. Moreno, C. Exposito-Izquierdo, Quality of service objectives for vehicle routing problem with time windows, *Appl. Soft Comput.*, **84** (2019), 105707. https://doi.org/10.1016/j.asoc.2019.105707

20. C. Liu, G. Kou, X. Zhou, Y. Peng, H. Sheng, P. E. Alsaadi, Time-dependent vehicle routing problem with time windows of city logistics with a congestion avoidance approach, *Knowledge-Based Syst.*, **188** (2020), 104813. https://doi.org/10.1016/j.knosys.2019.06.021

21. N. Bianchessi, S. Irnich, Branch-and-cut for the split delivery vehicle routing problem with time windows, *Transp. Sci.*, **53** (2019), 442–462. https://doi.org/10.1287/trsc.2018.0825

22. A. Agga, A. Abbou, M. Labbadi, Y. E. Houm, I. H. Ou Ali, CNN-LSTM: An efficient hybrid deep learning architecture for predicting short-term photovoltaic power production, *Electr. Power Syst. Res.*, **208** (2022), 107908. https://doi.org/10.1016/j.epsr.2022.107908

23. S. Han, Y. Qiao, J. Yan, Y. Liu, L. Li, Z. Wang, Mid-to-long term wind and photovoltaic power generation prediction based on copula function and long short term memory network, *Appl. Energy*, **239** (2019), 181–191. https://doi.org/10.1016/j.apenergy.2019.01.193

24. Z. Yu, P. Duan, L. Meng, Y. Han, F. Ye, Multi-objective path planning for mobile robot with an improved artificial bee colony algorithm, *Math. Biosci. Eng.*, **20** (2023), 2501–2529. https://doi.org/10.3934/mbe.2023117

25. L. Li, Z. Liu, M. Tseng, S. Zheng, M. K. Lim, Improved tunicate swarm algorithm: Solving the dynamic economic emission dispatch problems, *Appl. Soft Comput.*, **108** (2021), 107504. https://doi.org/10.1016/j.asoc.2021.107504

26. Z. Liu, L. Li, Y. Liu, J. Liu, H. Li, Q. Shen, Dynamic economic emission dispatch considering renewable energy generation: a novel multi-objective optimization approach, *Energy*, **235** (2021), 121407. https://doi.org/10.1016/j.energy.2021.121407

27. A. T. Eseye, J. Zhang, D. Zheng, Short-term photovoltaic solar power forecasting using a hybrid wavelet-PSO-SVM model based on SCADA and meteorological information optimization approach, *Renewable Energy*, **118** (2018), 357–367. https://doi.org/10.1016/j.renene.2017.11.011

28. L. Meng, K. Gao, Y. Ren, B. Zhang, H. Sang, C. Zhang, Novel MILP and CP models for distributed hybrid flowshop scheduling problem with sequence-dependent setup times, *Swarm Evol. Comput.*, **71** (2022), 101058. https://doi.org/10.1016/j.swevo.2022.101058

29. L. Meng, C. Zhang, Y. Ren, B. Zhang, C. Lv, Mixed-integer linear programming and constraint programming formulations for solving distributed flexible job shop scheduling problem, *IEEE Trans. Cybern.*, **142** (2020), 106347. https://doi.org/10.1016/j.cie.2020.106347

30. H. Guo, H. Sang, B. Zhang, L. Meng, L. Liu, An effective metaheuristic with a differential flight strategy for the distributed permutation flowshop scheduling problem with sequence-dependent setup times, *Knowledge-Based Syst.*, **242** (2022), 108328. https://doi.org/10.1016/j.knosys.2022.108328

31. Z. Li, H. Sang, J. Li, Y. Han, K. Gao, Z. Zheng, et al., Invasive weed optimization for multi-AGVs dispatching problem in a matrix manufacturing workshop, *Swarm Evol. Comput.*, **77** (2023), 101227. https://doi.org/10.1016/j.swevo.2023.101227

32. H. Guo, H. Sang, X. Zhang, P. Duan, J. Li, Y. Han, An effective fruit fly optimization algorithm for the distributed permutation flowshop scheduling problem with total flowtime, *Eng. Appl. Artif. Intell.*, **123** (2023), 106347. https://doi.org/10.1016/j.engappai.2023.106347

33. J. Duan, Z. He, G. G. Yen, Robust multiobjective optimization for vehicle routing problem with time windows, *IEEE Trans. Cybern.*, **52** (2022), 8300–8314. https://doi.org/10.1109/TCYB.2021.3049635

34. Y. Shen, M. Liu, J. Yang, Y. Shi, M. Middendorf, A hybrid swarm intelligence algorithm for vehicle routing problem with time windows, *IEEE Access*, **8** (2020), 93882–3893. https://doi.org/10.1109/ACCESS.2020.2984660

35. J. Brito, A. Expósito, J. A. Moreno, Variable neighbourhood search for close–open vehicle routing problem with time windows, *IMA J. Manage. Math.*, **27** (2016), 25–38. https://doi.org/10.1093/imaman/dpt024

36. M. Keskin, B. Catay, Partial recharge strategies for the electric vehicle routing problem with time windows, *Transp. Res. Part C Emerging Technol.*, **65** (2016), 111–127. https://doi.org/10.1016/j.trc.2016.01.013

37. Z. H. Ahmed, M. Yousefikhoshbakht, An improved tabu search algorithm for solving heterogeneous fixed fleet open vehicle routing problem with time windows, *Alexandria Eng. J.*, **64** (2023), 349–363. https://doi.org/10.1016/j.aej.2022.09.008

38. N. L. Giedelmann, W. J. Guerrero, E. L. Solano-Charris, On the emergency water distribution problem: optimizing vehicle routing decisions with deprivation costs considerations, *IFAC-PapersOnLine*, **55** (2022), 3166–3171. https://doi.org/10.1016/j.ifacol.2022.10.216

39. Y. E. M. Vieira, R. A. de Mello Bandeira, O. S. da Silva Júnior, Multi-depot vehicle routing problem for large scale disaster relief in drought scenarios: the case of the Brazilian northeast region, *Int. J. Disaster Risk Reduct.*, **58** (2021), 102193. https://doi.org/10.1016/j.ijdrr.2021.102193

40. J. Yi, J. Wang, G. Wang, Using monarch butterfly optimization to solve the emergency vehicle routing problem with relief materials in sudden disasters, *Open Geosci.*, **11** (2019), 391–413. https://doi.org/10.1515/geo-2019-0031

41. M. F. N. Maghfiroh, S. Hanaoka, Dynamic truck and trailer routing problem for last mile distribution in disaster response, *J. Humanitarian Logist. Supply Chain Manage.*, **8** (2018), 252–278. https://doi.org/10.1108/JHLSCM-10-2017-0050

42. M. M. Solomon, Algorithms for the vehicle routing and scheduling problems with time window constraints, *Oper. Res.*, **35** (1987), 254–265. https://doi.org/doi:10.1287/opre.35.2.254

43. C. Lu, Q. Liu, B. Zhang, L. Yin, A pareto-based hybrid iterated greedy algorithm for energy-efficient scheduling of distributed hybrid flowshop, *Expert Syst. Appl.*, **204** (2022), 117555. https://doi.org/10.1016/j.eswa.2022.117555

44. X. Ren, S. Chen, L. Ren, Optimization of regional emergency supplies distribution vehicle route with dynamic real-time demand, *Math. Biosci. Eng.*, **20** (2023), 7487–7518. https://doi.org/10.3934/mbe.2023324

45. Y. Xia, Z. Fu, Improved tabu search algorithm for the open vehicle routing problem with soft time windows and satisfaction rate, *Cluster Comput.*, **22** (2019), 8725–8733. https://doi.org/10.1007/s10586-018-1957-x

46. W. Ran, L. Liu, G. Yang, A hybrid ant colony algorithm for vehicle routing problem with time windows, *Inf. Technol. J.*, **12** (2013), 5701–5706. https://doi.org/10.3923/itj.2013.5701.5706

47. Q. Liu, P. Xu, Y. Wu, T. Shen, A hybrid genetic algorithm for the electric vehicle routing problem with time windows, *Control Theory Technol.*, **20** (2022), 279–286. https://doi.org/10.1007/s11768-022-00091-1

48. A. Redi, P. Jewpanya, A. C. Kurniawan, S. F. Persada, R. Nadlifatin, O. Dewi, A simulated annealing algorithm for solving two-echelon vehicle routing problem with locker facilities, *Algorithms*, **13** (2020), 218. https://doi.org/10.3390/a13090218