Contribution to this paper:

- First-author & Corresponding author Prof. Liu proposed the main ideas;
- Second-author Yangguang Wang wrote the code, conducted the experiments, and completed the manuscript.

Minimizing makespan of multi-drone delivery scheduling in the hexagonal network with dynamic parcel arrivals

Chuang Liu[a*], Yangguang Wang[a], Xueping Li[b], Shengchao Zhou[c], and Huaping Chen[a]

[a]School of Management, University of Science and Technology of China, Hefei 230026, PR China; [b]Department of Industrial and Systems Engineering, University of Tennessee, Knoxville, TN 37996, USA; [c]School of Automation, Central South University, Changsha, Hunan 410083, PR China

ARTICLE HISTORY
Compiled August 3, 2021

ABSTRACT

We study the scheduling problem of multiple drones in fulfillment centers (FCs) with dynamic parcel arrivals to minimize the makespan. Drones hold great potentials for speed delivery and improving efficiency. However, due to the limitations of battery technologies, drones' endurance is constricted. Inspired by regular hexagonal cellular networks, we propose an effective delivery network by combining FCs and battery-swapping stations (BSSs) in the hexagonal cellular structure to overcome such restrictions. We propose a mixed-integer programming model to formulate the problem. We develop a memetic algorithm with hill climbing (MAHC) to address the problem. A lower bound of the problem is proposed to measure the performance of MAHC. We propose a strategy to prevent inbreeding in the crossover procedure of MAHC. A local search strategy based on hill climbing is combined with the mutation operator of MAHC. Furthermore, a greedy strategy (GS) and a nearest-priority strategy (NPS) are proposed to select the optimal FC. CPLEX, a genetic algorithm (GA), and the baseline first-come-first-served (FCFS) scheduling algorithm are employed as comparison algorithms to evaluate the performance of MAHC. Extensive experiments show that MAHC outperforms CPLEX, FCFS, GA, and EDA. Besides, comparative experiments between GS and NPS verified the efficiency of GS.

KEYWORDS
Unmanned aerial vehicles (UAVs); Drone; Delivery network; Scheduling; Makespan; Memetic Algorithm

1. Introduction

As the pace of social and economic life accelerates, customers are becoming increasingly sensitive to the delivery time of parcels, and the required delivery time ranges from weekly to daily or even hourly. For some emergency scenarios, medicines and emergency supplies must be delivered even within minutes. Drones, also known as unmanned aerial vehicles (UAVs), have the advantage of meeting this speedy delivery requirement. Compared with traditional trucks, drones are relatively low-cost, environmentally friendly, and non-contact. Based on these advantages, some e-commerce and courier giants, such as Amazon, JD, UPS, and SF Express have tried to use drones to deliver parcels. For example, Amazon launched its Prime Air plan in 2013 and has obtained approval from

CONTACT Chuang Liu. Email: chuang@mail.ustc.edu.cn

the Federal Aviation Administration (FAA) to operate its Prime Air delivery fleet for customer deliveries in the US, Amazon (2019); Palmer (2020). Particularly during the COVID-19 epidemic, drones are being widely used in the delivery of daily and emergency supplies; for example, JD in China has used drones to deliver parcels ordered from JD online grocery, Yang and Reuter (2020); Kidron (2020). CVS and UPS have also used drones to deliver prescription medications in a Florida retirement community Bursztynsky (2020).

Due to the limitation of current battery technologies, however, drones have a limited flight time. To tackle the short endurance problem, some technical solutions have been proposed for drone delivery. One example is the combination of trucks and drones. With this approach, drones and parcels are stored on the truck. When the destination of a parcel enters the delivery range of a drone, the drone can take off from the truck and complete the delivery task. This method reduces the distance between the drones in the truck and the destination of the parcels. Studies on the combination of trucks and drones can be found in the literature, including Carlsson and Song (2018); Poikonen, Golden, and Wasil (2019); Gonzalez-R et al. (2020). However, because of its limited capacity, a truck can only carry a few drones. In addition, the uncertainty of the environment in which the truck is located makes a battery-swapping operation for a drone even more difficult and time-consuming.

To enhance the flight time of drones, we propose a new drone delivery model that combines fulfillment centers (FCs) and battery-swapping stations (BSSs) to form a drone delivery network. A drone takes off from an FC. The battery of each drone can then be replaced at any BSS. Compared with battery charging for a drone, a swapping operation can significantly reduce the time required. The results given in Shen et al. (2020) indicate that a battery-swapping method can obtain a higher throughput capacity in most instances. The endurance of drones can be greatly improved in this way. In each relay station, the battery-swapping operation can be automatically completed without a manual intervention. Amazon has proposed an efficient structural design for an FC as in Curlander et al. (2017). Technical solutions for an automatic battery replacement system have also been made available, such as Suzuki, Kemper Filho, and Morrison (2012).

In this study, two FCs and a certain number of BSSs are considered. All FCs and BSSs together form a logistics delivery network. After receiving an order from a customer, an FC needs to be chosen to handle the order according to the customer's location coordinates and the idle conditions of the two FCs. In general, there will be multiple feasible routes between an FC and a customer, and the number of BSSs for each route will be different. Some routes take longer, and others are shorter. The optimization objective of the problem is to minimize the maximum completion time (makespan).

To illustrate the problem clearly, a simple example is given in Figure 1. Real-life applications of the problem can be easily found in many fields, including e-commerce, shipping and delivery, and rescue operations. Wide adoption of drones in the logistics field can greatly reduce carbon dioxide emissions. Some people are even projecting that in the near future, drones flying in the sky may become a familiar sight, DiNota, Douglas, and Marcontell (2019).

The remainder of this paper is organized as follows. Section 2 provides a review of the problem under research. Section 3 defines the scope of the problem, the notations, the structure of the delivery network, assumptions, and a mathematical formulation. The proposed algorithms are presented in Section 4. Computational experiments are conducted in Section 5. Finally, Section 6 provides some concluding remarks regarding this research, discussions, and areas of future study.

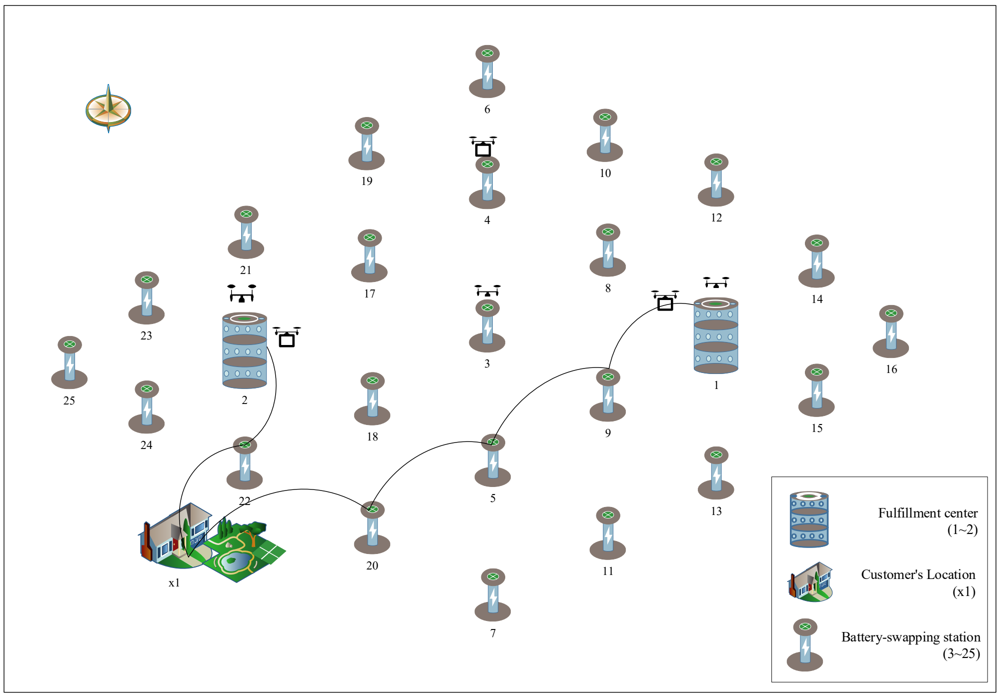


Figure 1. An example of delivering a parcel in a delivery network. The delivery network is composed of 2 FCs (vertices: $1 \sim 2$) and 23 BSSs (vertices: $3 \sim 25$). The destination of the parcel to be delivered is $x1$. Both FCs 1 and 2 can deliver the parcel to location $x1$. The paths for the two FCs to deliver the parcel are, $F_{(1,x1)} = \{(1,9)(9,5)(5,20)(20,x1)\}$ and $F_{(2,x1)} = \{(2,22)(22,x1)\}$. It's obvious that FC 2 is better than FC 1. Therefore, FC 2 is chosen to deliver the parcel to location $x1$.

2. Literature review

From the literature, we can see that studies on delivery using drones focuses on a combination of drones and traditional vehicles, for example, Carlsson and Song (2018); Agatz, Bouman, and Schmidt (2018); Poikonen, Golden, and Wasil (2019); Chung, Sah, and Lee (2020); Gonzalez-R et al. (2020); Dayarian, Savelsbergh, and Clarke (2020). In a drone-assisted truck delivery system, the truck is a mobile mothership that can carry a certain number of drones; drones can take off from the truck and return to it after completing the delivery mission. Based on whether the truck delivers parcels or not, the related studies can be divided into two categories. First, the truck used in the delivery system can also deliver parcels directly to customers; hence, the truck will have two tasks, delivering parcels and carrying drones, which is referred to as DTS1. Second, the truck can only carry drones, and all parcels must be delivered by drones only, which is denoted to DTS2. Compared with DTS2, DTS1 may improve the efficiency of the delivery system. Here, we reviewed the relevant literature based on these two categories. For DTS1, the truck and drones can both provide parcel delivery service for customers. In addition, the loading operation of parcels by drones is generally only allowed at the customers' location. In Agatz, Bouman, and Schmidt (2018), the authors studied a combined drone-assisted truck delivery system under the last-mile delivery scenario, and proposed several fast route-first, cluster-second heuristics for the traveling salesman problem with a drone (TSP-D), i.e., a greedy partitioning heuristic and an exact partitioning algorithm. The authors also showed that the combined system can reduce costs substantially compared to the truck-only delivery system. Based on the partitioning approach of Agatz, Bouman, and Schmidt (2018), in Poikonen, Golden, and Wasil (2019), the authors proposed several new heuristics to address the TSP-D problem, for instance, BAB+L, BAB+Q, and the divide-and-conquer heuristic (DCH). The DCH technique they used can decrease the running time by breaking a larger problem into a sequence of smaller subproblems. In Gonzalez-R et al. (2020), the authors proposed an iterated greedy heuristic based on the iterative process of destruction and reconstruction of the solution to address the drone-truck logistics problem.

For DTS2, only the drones can provide delivery service to customers, and the truck is itself moving. In Carlsson and Song (2018), the coordinated logistics problem is addressed with a truck and a drone, and determined the efficiency of the drone-assisted truck delivery system. The improvement in efficiency is related to the square root of the ratio of the speeds of the truck and drone. In Poikonen and Golden (2020), the authors introduced the mothership and drone routing problem. They applied the BAB approach to solve small instances and proposed several heuristic methods to address the larger instances, for instance, the greedy sequence heuristic, the greedy sequence with local search heuristic, and the partial solve with greedy insert heuristic.

In addition to DTS1 and DTS2, in Dayarian, Savelsbergh, and Clarke (2020), a new drone-assisted truck delivery system was proposed. The truck in this system can deliver parcels to customers, and drones are responsible for delivering the newly arrived parcels from the FC to the truck. In addition, a drone can carry more than one parcel at a time. This type of delivery system is denoted as DTS3 in our study. The more details on drone-assisted truck delivery systems can be found in Chung, Sah, and Lee (2020). The authors provided a detailed overview of the drone and drone-truck combined operations and surveyed the optimization algorithms used in the civil application of drone and combined drone-truck operations.

All the aforementioned studies used the combination of trucks and drones to expand the delivery range of the FCs; however, this combination cannot make full use of the advantages of drones, for example, speed, environmental friendliness, and low cost. Drones in DTS1, DTS2, and DTS3 only play an auxiliary role, whereas the traditional transportation vehicle plays a major role. In our study, drones are solely responsible for delivering parcels, and battery-swapping stations are introduced to expand the flight endurance of drones.

To the best of our knowledge, there is no research on expanding the delivery distance of drones by introducing battery-swapping stations. In addition, this study aims to fill the gap above. In our study, the drone flight operation can be fully automated without manual interventions, which can further reduce labor costs. Compared with a mobile mothership truck, a fixed-position battery-swapping station is more convenient for battery replacement and charging and maintenance operations. This makes the unified management of the battery easier to achieve. Moreover, considering that there are already mature battery-swapping stations for electric vehicles, as an FC operator, it does not need to build the battery-swapping stations by itse1lf, and can directly outsource the battery-swapping services to the third-party service stations.

3. Problem Statement

In this study, there are m_1 FCs and m_2 BSSs. Each FC maintains m_3 drones and can address all orders. Each BSS can provide a battery-swapping service for all drones. The configurations of all drones are exactly the same. In addition, the battery-swapping operation requires a constant time, t_2. The architectural structure of an FC refers to the patent applied by Amazon Curlander et al. (2017). The multi-level FC in the patent allows several drones to land or take off simultaneously. FCs and BSSs together form a parcel delivery network. All parcels can be delivered through this delivery network using drones.

The number of orders received from customers is n. Each order arrives dynamically, and the ordering time of order o_j is r_j. After packaging the goods for order o_j, the corresponding parcel p_j can be generated. The picking and packing operation requires a constant time, t_1. The time required to complete the delivery of parcel p_j is C_j. Figure 2 shows the procedures used for addressing an order.

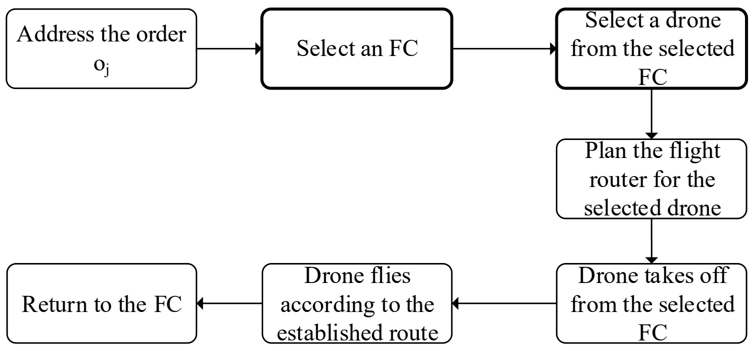

Address the order o_j
Select an FC
Select a drone from the selected FC
Plan the flight router for the selected drone
Drone takes off from the selected FC
Drone flies according to the established route
Return to the FC

Figure 2. Flow chart of the life cycle of an order.

The delivery timeliness of e-commerce platforms is an important factor that affects customer satisfaction. Next-day, same-day, or even 2-h delivery (for example, Amazon provides a free 2-h delivery with Prime) are provided. The earlier the completion of the parcel delivery, the higher the customer satisfaction that is achieved. Therefore, the objective of the problem is set to the maximum completion time of all orders.

3.1. *Notations*

The problem under research takes the scheduling problem into account. It also involves the structural framework of the drone delivery network. Therefore, the notations employed in the problem are complicated. To clarity the problem, indices, sets, parameters, and decision variables used in the problem are summarized in Table 1.

3.2. *Structure of delivery network*

The traditional truck delivery network is generally composed of roads intertwined vertically and horizontally. Compared with trucks, drones have a natural advantage and can fly straight when there are no obstacles. Owing to the limitations of battery technology, the flying distance of drones is limited. To remove this restriction, we introduced the setting of battery-swapping stations in the delivery network. Each BSS can provide battery-swapping services for incoming drones. All BSSs and FCs form the delivery network together.

Considering that the flying range of drones is usually a circular area, we envisage expanding the flying range of drones based on the circle. However, there are unreachable areas between multiple circular areas, as shown by the shaded area in Figure 3.

Table 1. Notations

Notations	Definitions
Indices:	
i	Index of FCs and BSSs, $i = 1, 2, \ldots, m_1, m_1 + 1, \ldots, m_1 + m_2$
j	Index of orders and parcels, $j = 1, 2, \ldots, n$
l	Index of drones in each FC, $l = 1, 2, \ldots, m_3$
Sets:	
V	Set of vertices of FCs and BSSs; $V = \{v_1, v_2, \ldots, v_{m_1}, v_{m_1+1}, v_{m_1+2}, \ldots, v_{m_1+m_2}\}$ [a]
O	Set of orders; $O = \{o_1, o_2, \ldots, o_n\}$
X	Set of vertices of order locations; $X = \{x_1, x_2, \ldots, x_n\}$
A_0	Set of arcs; $A_0 = \{(v_{i_1}, v_{i_2}) : v_{i_1} \in V, v_{i_2} \in V, i_1 \neq i_2\}$
A	Set of arcs between nodes and location of parcels; $A = \{(v_i, x_j) : v_i \in V, x_j \in X\}$
Parameters:	
v_i	A vertex in set V; if $1 \leq i \leq m_1$, v_i is an FC; if $m_1 + 1 \leq i \leq m_1 + m_2$, v_i is a BSS
n	Number of orders (parcels)
m_1	Number of FCs
m_2	Number of BSSs
m_3	Number of drones in each FC
v	Speed of drone flying
$\tau_{i_1 i_2}$	Time spent flying over arc (i_1, i_2), $(i_1, i_2) \in A \cup X$
o_j	The j^{th} order
r_j	The ordering time of order o_j from a customer
p_j	The parcel corresponding to order o_j
x_j	Vertex of location of order o_j
d_{il}	The l^{th} drone in FC v_i
t_0	Max endurance time of a drone with a fully charged battery
t_1	Time spent in picking and packing a parcel
t_2	Time spent in replacing battery for a drone
t_3	Time spent in taking off, accelerating, decelerating, and landing of drone
Decision Variables:	
π_i	An order processing sequence
α_{ilk}	Ready time of the k^{th} parcel to be delivered by drone d_{il}
β_{ilk}	Start time to deliver the k^{th} parcel by drone d_{il}
T_{ilk}	Time spent in delivering the k^{th} parcel using drone d_{il} from FC v_i to the location of the parcel
$\hat{C}_{ilk}$	Completion time of delivering the k^{th} parcel using drone d_{il}
θ_{ilk}	Time to return to FC v_i using drone d_{il} after delivering the k^{th} parcel
Y_{jilk}	1, if parcel p_j is delivered by drone d_{il} at k^{th} order, which is 0, otherwise
$F_{(i,j)}$	Flight route of delivering parcel from FC v_i to the location of p_j [b]
C_j	Completion time of delivering parcel p_j

[a] In vertices, $v_1 \sim v_{m_1}$ indicates the nodes of FCs, and $v_{m_1+1} \sim v_{m_1+m_2}$ denotes the nodes of BSSs.
[b] Arcs in a flight route are all from set $A_0 \cup A$. It should be noted that the arc from the last BSS to the destination of parcel p_j is from A.

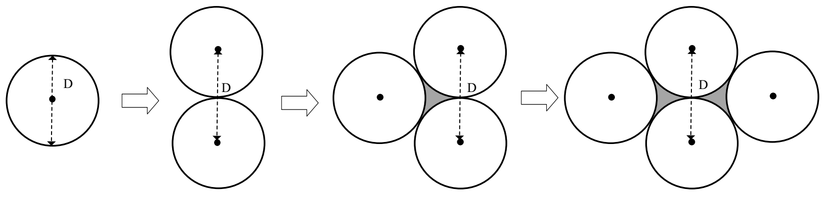


Figure 3. The diameter (D) of all circles equals the maximum distance that drone can fly safely in time t_0. The center of a circle stands for the position of a BSS or an FC.

Inspired by the cellular network structure, we propose a new delivery network using the cellular structure in this study. The cellular network structure is based on the regular hexagon. As shown in Figure 4, there is no unreachable area between multiple regular hexagons. In addition, the scope of the regular hexagons in Figure 4 can be expanded infinitely.

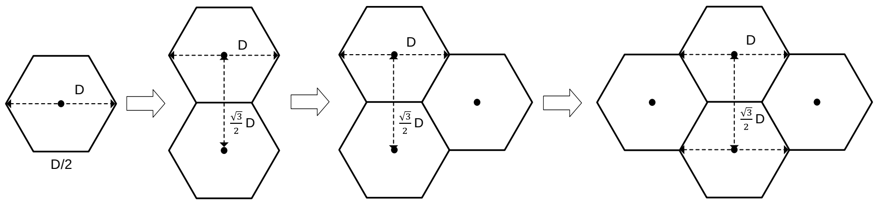


Figure 4. The length of the side of all hexagons is equal to $D/2$. The center of a hexagon stands for the position of a BSS or an FC.

The delivery network discussed in this study consists of FCs and BSSs. The delivery network structure, which is expanded exactly from Figure 4, is given in Figure 5. A regular hexagon in the figure represents a basic cellular structure unit. The center of a regular hexagon represents an FC or a BSS. Each node, whether an FC or a BSS, has the following two functions, delivering all parcels in the hexagonal area and replacing the batteries for the incoming drones. Drones can quickly deliver parcels through this delivery network. The length of the side of all regular hexagons is determined by the range of the drones. There are no bottleneck nodes in this structure. When a node fails, it only affects the delivery of parcels within that hexagonal area. The battery-swapping function of this node can be replaced by its neighboring nodes. In addition, the location problem of BSSs and FCs can be addressed based on the proposed structure.

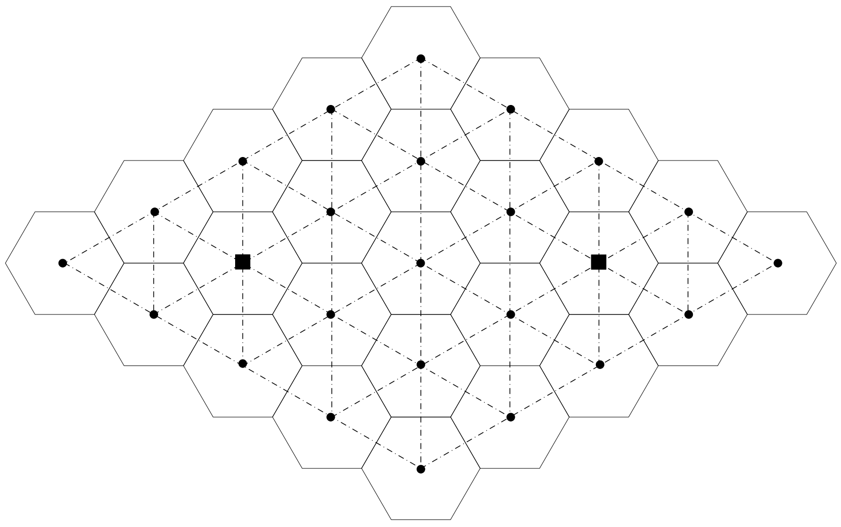


Figure 5. The diameter of the circumscribed circle of each regular hexagon is equal to D. The square point in a regular hexagon denotes the position of an FC, and the circle point stands for the position of a BSS. The dashed line represents the main route for drones. A station (a BSS or an FC) can cover all needs in the regular hexagon where it is located.

3.3. *Problem assumptions*

To describe the problem clearly, the following assumptions are made:

(1) The customer order time obeys a Poisson distribution with rate λ. The time consumed by the picking and packing operations of each order is fixed. In addition, the destination of all orders is within the delivery range. The weight of all parcels is within the safe range of the drones.
(2) The configurations of all FCs are identical, for example, the number of drones, the types of goods, and the ability to handle parcels. The configurations of all BSSs are also identical.
(3) The configurations of all drones are exactly the same. All drones keep flying at a constant speed during the delivery process, except for take-off and landing. A drone can only carry at most one package per flight. Multiple drones can take off from an FC at the same time. There are no route conflicts during the flight of multiple drones in the delivery network.
(4) Each time a drone completes a delivery mission, it backtracks to the FC from which it departed. In addition, a battery-swapping operation must be performed for the drones in all FCs. The time it takes for a drone to replace a battery at any BSS or FC is fixed.
(5) The battery is in a fully charged condition when a drone takes off from an FC. Moreover, the battery replaced from a BSS is also fully charged.

3.4. *Mathematical formulation*

The problem under research is a parallel drone scheduling problem in multiple FCs. If the number of FCs and BSSs in the problem is set to 1 and 0, respectively, this problem can be transformed into the problem described in Liu et al. (2021). The problem addressed in Liu et al. (2021) is a non-deterministic polynomial-time hardness (NP-hard) problem, therefore, the problem under study is also an NP-hard problem. To address the problem, a mixed-integer programming model (MIP) is built as follows:

$$Min\ C_{max} = \max_{1 \le j \le n} C_j \tag{1}$$

Subject to:

$$Y_{jilk} = \begin{cases} 1, \ if\ parcel\ p_j\ is\ addressed\ by\ drone \\ \quad d_{il}\ at\ the\ k^{th}\ order \\ 0, \ otherwise \end{cases} \tag{2}$$

$$\sum_{j=1}^{n} Y_{jilk} \leq 1, \ k = 1, 2, \ldots, n; l = 1, 2, \ldots, m_3; i = 1, 2, \ldots, m_1 \tag{3}$$

$$\sum_{i=1}^{m_1} \sum_{l=1}^{m_3} \sum_{k=1}^{n} Y_{jilk} = 1, \ j = 1, 2, \ldots, n \tag{4}$$

$$\sum_{j=1}^{n} Y_{jilk} \geq \sum_{j=1}^{n} Y_{jil(k+1)}, \ i = 1, 2 \ldots, m_1; \quad l = 1, 2, \ldots, m_3; k = 1, 2, \ldots, n-1 \tag{5}$$

$$if\ Y_{jilk} = 1 \Rightarrow \alpha_{ilk} = r_j + t_1, \ j, k = 1, 2, \ldots, n; \quad l = 1, 2, \ldots, m_3; i = 1, 2, \ldots, m_1 \tag{6}$$

$$if\ Y_{jilk} = 1 \Rightarrow T_{ilk} = f(F_{(i,j)}, t_2, t_3, v), \ j, k = 1, 2, \ldots, n; \quad l = 1, 2, \ldots, m_3; i = 1, 2, \ldots, m_1 \tag{7}$$

$$\beta_{ilk} \geq \alpha_{ilk}, \ k = 1, 2, \ldots, n; l = 1, 2, \ldots, m_3; i = 1, 2, \ldots, m_1 \tag{8}$$

$$\hat{C}_{ilk} = \beta_{ilk} + T_{ilk}, \ k = 1, 2, \ldots, n; l = 1, 2, \ldots, m_3; i = 1, 2, \ldots, m_1 \tag{9}$$

$$\theta_{ilk} = \hat{C}_{ilk} + T_{ilk}, \ k = 1, 2, \ldots, n; l = 1, 2, \ldots, m_3; i = 1, 2, \ldots, m_1 \tag{10}$$

$$\beta_{il1} = \alpha_{il1}, \ l = 1, 2, \ldots, m_3; i = 1, 2 \ldots, m_1 \tag{11}$$

$$\beta_{il(k+1)} = \theta_{ilk} + t_2, k = 1, 2, \ldots, n-1; l = 1, 2, \ldots, m_3; i = 1, 2 \ldots, m_1 \tag{12}$$

$$if\ Y_{jilk} = 1 \Rightarrow C_j = \hat{C}_{ilk}, \ j, k = 1, 2, \ldots, n; \quad l = 1, 2, \ldots, m_3; i = 1, 2, \ldots, m_1 \tag{13}$$

Constraint 2 denotes Y_{jilk} as a binary decision variable. Constraint 3 guarantees that a drone can only deliver one parcel per flight. Constraint 4 ensures that a parcel can only be delivered once. Constraint 5 indicates that drones with the lower labels in an FC are used first. Constraint 6 defines α_{ilk}, the ready time of the k^{th} parcel to be delivered by drone d_{il}. This constraint means that a parcel is ready to be delivered after a picking and packing operation. Constraint 7 defines a variable T_{ilk}. The value of T_{ilk} can be determined by the speed of drone d_{il}, the flight path and the time required for the battery-swapping operation at a BSS. Constraint 8 ensures that the start time to deliver a parcel must be greater than or equal to the ready time of the parcel. It indicates there may exist some parcels that have to wait for a period of time before they can be delivered. Constraint 9 indicates that $\hat{C}_{ilk}$ equals the start time of the delivery plus the time spent in the delivery operation. Constraint 10 indicates that the time of return to the FC equals the time to complete the delivery plus the time spent in delivery. Constraint 11 indicates that the parcel delivered by a drone at the first position does not need to wait. Constraint 12 denotes that the battery must be replaced after a drone delivery mission. Constraint 13 connects two decision variables, C_j and $\hat{C}_{ilk}$.

3.5. *A lower bound of the problem*

This section gives a lower bound by relaxing the constraints of the problem. The number of drones m_3 in each FC can highly affect the delivery completion time of all parcels. The greater the number of drones that are used, the smaller the total delivery completion time for all parcels. When m_3 equals the number of orders ($m_3 = n$), all parcels can be processed immediately without waiting. This indicates that the processing order of parcels is first-in, first-out (FIFO). The constraint 8 can then be transformed into Equation 14. In other words, there are sufficient drones at each FC to deliver all parcels. Therefore, all parcels can be definitely delivered by the nearest FC.

$$\beta_{ilk} = \alpha_{ilk},\ k, l = 1, 2, \ldots, n; i = 1, 2, \ldots, m_1 \tag{14}$$

To clarify the approach in obtaining a lower bound of the problem, a pseudocode is given in Algorithm 1.

Algorithm 1 Pseudocode of the proposed lower bound (LB) algorithm.

```
1:  **Parameters:** $o_j$, $r_j$, $x_j$, $v_i$, $m_1$, and $n$;
2:  Sort all orders in ascending order based on $r_j$;
3:  $C_{max} \leftarrow +\infty$;
4:  for j = 1 to n do
5:      for i = 1 to $m_1$ do
6:          Calculate the distance between $x_j$ and FC $v_i$;
7:      end for
8:      Use the nearest FC to address the order $o_j$;
9:      Calculate $C_j$;
10:     if $C_j < C_{max}$ then
11:         $C_{max} \leftarrow C_j$;
12:     end if
13: end for
14: Obtain $C_{max}$;
```

4. Memetic algorithm with hill climbing strategy for the problem

To solve the NP-hard problem under study, we divide the problem into the two following parts: determining the optimal or suboptimal order processing sequence $\pi_{opt}(o_{j_1}, o_{j_2}, \ldots, o_{j_n})$ and determining the optimal or suboptimal FC and drone allocation scheme for each order. A memetic algorithm with hill climbing strategy (MAHC) used to determine π_{opt} is discussed in detail firstly. Two strategies proposed to determine the optimal or sub-optimal FC are subsequently elaborated.

4.1. *Memetic algorithm with hill climbing strategy*

Memetic Algorithms (MAs) introduced by Moscato et al. (1989) are metaheuristic algorithms based on population evolution. MAs are also commonly known as cultural algorithms, hybrid genetic algorithms, Lamarckian evolutionary algorithms, and Baldwinian evolutionary algorithms in the literature. MAs have been successfully applied to the scheduling problems Xu et al. (2014); Boryczka and Szwarc (2019). MAs combine global search methods and local search strategies to achieve an efficient trade-off between global exploration and local exploitation compared with classical evolutionary algorithms. Besides, the local search strategies generally need problem-specific knowledge.

The pseudocode of MAHC used in the problem is given in Algorithm 2.

Algorithm 2 Pseudocode of MAHC.

```
 1: Parameters: $D_{ld}$, $\rho_c$ and $\rho_m$;
 2: N ← Number of individuals in population;
 3: $I_k$ ← The $k^{th}$ individual in population;
 4: Generate initial population;
 5: $iter \leftarrow 0$;
 6: $POP_{iter}$ denotes the $iter^{th}$ generation of the population;
 7: while Stopping criteria is not satisfied do
 8:     for k = 0 to N do
 9:         Calculate the fitness value of Individual $I_k$;
10:     end for
11:     for k = 1 to N/2 do
12:         if $rand < \rho_c$ then
13:             Select&Crossover($POP_{iter}$, $D_{ld}$);
14:         end if
15:     end for
16:     for k = 1 to N do /* Mutation and local search */
17:         if $rand < \frac{1}{240} + \frac{0.11375}{2^{iter}}$ then
18:             Mutation($I_k$);
19:             HillClimbing($I_k$);
20:         end if
21:     end for
22:     Update the current population;
23:     $iter \leftarrow iter + 1$;
24: end while
```

4.1.1. Representation, initialization and evaluation

A permutation-based encoding method is used to represent the individuals in MAHC. The encoding method used can also be regarded as a sequence of parcel labels (p_j). Besides, the initial population of MAHC is randomly generated, and the number of individuals in each generation remains constant. To evaluating the individuals in the population, a fitness function defined in Equation 15 is used in the problem.

$$f = \max_{1 \le j \le n} C_j \tag{15}$$

4.1.2. Selection and crossover operators

The tournament selection operator based on the fitness function is used to select excellent individuals from the population. Moreover, the position-based crossover (PBX) from Davis (1991) is employed to generate offspring. Inspired by the phenomenon that inbreeding can cause genetic defects, a mechanism to avoid inbreeding is proposed for the PBX operator. Two individuals in the population must be distant relatives to perform the crossover operation. The concept of the inbreeding coefficient in genetics is introduced to measure the similarity between two individuals.

The definition of the coefficient of difference between two individuals used in the problem is given in Definition 4.1. The coefficient of difference between the two individuals used for the PBX operator must be greater than D_{ld}.

Definition 4.1. Let P and Q represent two individuals in the population, and $Diff(P,Q)$ represents the degree of difference between the two individuals. Besides, the memes of the two individuals are $p_1p_2\dots p_n$ and $q_1q_2\dots q_n$ respectively. The coefficient of difference between P and Q can be calculated by the following formulation,

$$Diff(P,Q) = \frac{\Sigma_1^n(x_k)}{n}, \tag{16}$$

where x_k is determined by p_k and q_k $(k = 1, 2, \dots, n)$. If p_k and q_k are different, then x_k is equal to 1; otherwise, x_k equals to 0.

According to the description of inbreeding in Definition 4.1, we give the pseudocode of the crossover operator with inbreeding avoidance mechanism in Algorithm 3.

Algorithm 3 Crossover procedure with inbreeding avoidance mechanism

```
Input: $POP_{iter}$ and $D_{ld}$;
Output: offspring;
1: function Diff($P, Q$)
2:     diff ← 0;
3:     for k = 1 to n do
4:         if $P[k] = Q[k]$ then
5:             x ← 0;
6:         else
7:             x ← 1;
8:         end if
9:         diff ← diff+x;
10:    end for
11:    return $\frac{diff}{n}$;
12: end function
13: Use tournament operator to select two parents $P$ and $Q$ from $POP_{iter}$;
14: $D_{PQ}$ ← Diff($P, Q$);
15: while $D_{PQ} < D_{ld}$ do /* Avoid inbreeding */
16:     $Q$ ← Use tournament selection to select a new parent;
17:     $D_{PQ}$ ← Diff($P, Q$);
18: end while
19: offspring ← PBX($P, Q$);
```

4.1.3. Mutation operator and local search procedure

A shifting mutation operator is used in the problem. In the shifting operator adopted, memetic fragments of an arbitrary random length can be shifted to any position.

To improve the quality of individuals generated by the mutation operator in MAHC, we introduce a local search strategy based on hill climbing algorithm for the mutation operator. For each mutated individual, a local search optimization procedure is performed. For an individual P encoded as $(p_1, p_2, \ldots, p_n)$, the neighborhood structure used in the local search procedure is described as follows.

$$exchange(P,k) : (p_1, p_2, \ldots, \boldsymbol{p_k, p_{k+1}}, \ldots, p_n) \rightarrow (p_1, p_2, \ldots, \boldsymbol{p_{k+1}, p_k}, \ldots, p_n) \quad (17)$$

where k is from the set {1,2,..., n-1}. Lines 4 to 10 of Algorithm 4 give the procedure of the neighborhood structure.

The Lamarckian scheme, which allows the individuals improved by local search procedure to replace the original individuals in the population, is used in the problem. This scheme can pass on excellent memes to future generations.

Algorithm 4 Local search procedure based on hill climbing

```
Input: individual $P(p_1, p_2, \ldots, p_n)$;
Output: The improved $P$;
 1: $f \leftarrow$ The fitness function;
 2: $p[k] \leftarrow$ The $k^{th}$ meme of the individual;
 3: $Rand(i,j) \leftarrow$ Return a random number between $i$ and $j$;
 4: function EXCHANGE($P, k$)
 5:     $neighbor \leftarrow P$;
 6:     temp = neighbor[k];
 7:     neighbor[k] = neighbor[k+1];
 8:     neighbor[k+1] = temp;
 9:     return neighbor;
10: end function
11: while true do
12:     $randPoint \leftarrow Rand(1, n)$;
13:     $neighbor \leftarrow Exchange(P, randPoint)$;
14:     if $f(neighbor) < f(P)$ then
15:         P ← neighbor;
16:     else
17:         return P;
18:     end if
19: end while
```

4.2. *Two strategies to determine the optimal or suboptimal FC*

In the delivery network, all FCs can address orders. However, the distance between the different FCs and the destination of an order is generally different. A greedy strategy and a nearest-priority strategy are proposed to determine the optimal or suboptimal FC for each order. The path between FCs and the destination of parcels is determined by the Floyd-Warshall algorithm.

For a given FC, finding the most suitable drone is also crucial. The process of determining the optimal drone in FC v_i can be described as follows:

(1) Check the idle status of all drones in FC v_i;
(2) If there is an idle drone, Goto (3), otherwise, Goto (4);
(3) The idle drone is selected; End;

(4) Wait for any drone to return to FC v_i;
(5) Swap the battery for the returned drone;
(6) The returned drone is selected; End.

A drone needs to take off from an FC, pass through the delivery network, reach the parcel's destination, deliver the parcel to the customer and then return to the FC. A drone usually has multiple feasible flight paths from an FC to the destination.

The distance between any two nodes in the delivery network is determined by their geographic location. Owing to the limitations of battery technology, we consider the two nodes to be directly connected only if the distance between two nodes is less than the maximum flight distance ($t_0 \cdot v$) of the drones. In this way, the arcs of neighboring points in the delivery network can be constructed. All vertices of the FCs and BSSs (V) and arcs (A_0) together define a directed graph $G = (V, A_0)$. The Floyd-Warshall algorithm from Cormen et al. (2009) is employed in the graph G to determine the optimal path between FCs and the destination of parcels. The number of nodes (including FCs and BSSs) in the delivery network is $m_1 + m_2$, and therefore, the time complexity of the Floyd–Warshall algorithm is $O((m_1 + m_2)^3)$.

4.2.1. A greedy strategy

We propose a greedy strategy (GS) to determine an FC for each order. This strategy requires the delivery of a parcel to be completed as early as possible. It means that even if an FC is farther from the destination of an order, as long as the FC can complete the order delivery earlier, the FC is selected to address the order.

The pseudocode of the GS strategy is given in Algorithm 5. The time complexity of the Floyd–Warshall algorithm is $O((m_1 + m_2)^3)$; therefore, the time complexity of GS is $O(m_1 m_3 (m_1 + m_2)^3)$.

Algorithm 5 Pseudocode of GS.

```
1:  **Parameters:** $r_j, t_1, m_1, m_3, o_j, p_j, v_i, x_j$ and $v_i$;
2:  $t_{release}$ denotes the time to complete packing order $o_j$;
3:  $t_{deliver}$ denotes the time to start delivering parcel $p_j$;
4:  $C_{ij} \leftarrow$ the time to complete the delivery of parcel $p_j$ for FC $v_i$;
5:  for i = 1 to $m_1$ do
6:      Packing order $o_j$ in FC $v_i$;
7:      $t_{release} \leftarrow r_j + t_1$;
8:      for l = 1 to $m_3$ do
9:          Record the next idle time of drone $d_{il}$ in FC $v_i$;
10:     end for
11:     $d_{il_{opt}} \leftarrow$ Select the drone with the smallest next idle time;
12:     if Drone $d_{il_{opt}}$ is busy then
13:         Wait until drone $d_{il_{opt}}$ returns;
14:         Replace the battery for drone $d_{il_{opt}}$;
15:         $t_{deliver} \leftarrow$ The time the selected drone became idle;
16:     end if
17:     $F_{ij} \leftarrow$ The route between FC $v_i$ and $x_j$ determined by Floyd–Warshall algorithm;
18:     $T_{ij} \leftarrow$ Time spent flying along flight route $F_{(i,j)}$;
19:     $C_{ij} \leftarrow t_{deliver} + T_{ij}$;
20: end for
21: The FC with the least $C_{ij}$ is selected;
```

4.2.2. A nearest-priority strategy

An intuitive nearest-priority strategy (NPS) is proposed to determine the optimal FC for order o_j. The FC closest to the destination of the parcel p_j is selected in the NPS strategy. The pseudocode of NPS is given in Algorithm 6. Because the time complexity of the Floyd–Warshall algorithm is $O((m_1 + m_2)^3)$, the NPS's is $O(m_1(m_1 + m_2)^3)$.

Algorithm 6 Pseudocode of NPS.

```
1: Parameters: $x_j$, $m_1$, and $v_i$;
2: for i = 1 to $m_1$ do
3:     $F_{ij} \leftarrow$ Route between FC $v_i$ and $x_j$ determined by Floyd–Warshall algorithm;
4:     $T_{ij} \leftarrow$ Time spent flying along flight route $F_{(i,j)}$;
5: end for
6: The FC with the least $T_{ij}$ is selected;
```

5. Computational experiments

This section discusses the experimental design and setting of parameters in MAHC. The problem-related parameters are also given. In addition, the influence of the order distribution on the problem is also determined.

5.1. *Experimental design*

Extensive comparisons among MAHC, CPLEX, a genetic algorithm (GA), an estimation of distribution algorithm (EDA), and the baseline first-come-first-served (FCFS) scheduling algorithm are conducted. CPLEX (ILOG CPLEX Optimizer 12.10.0) is a commercial solver developed by IBM; The GA is from Liu et al. (2021); the EDA is from Zhou et al. (2016), and FCFS is a classical algorithm used in scheduling problems. The lower bound of the problem obtained by LB in Algorithm 1 is used to measure the difference of the solutions obtained by the above algorithms. We also conducted a comparison experiment between GS and NPS strategies to verify the performance difference between the two strategies.

The number of orders (n) tested in our experiments is from set {20, 25, 30, 35, 40, 45, 100, 200, 400, 800}. The members of a subset {20, 25, 30, 35, 40, 45} are considered small-scale instances, the elements in the subset {100, 200} are regarded as medium-scale instances, and the members in the subset {400, 800} are large-scale instances. In addition, the number of orders arriving within the sixty-second interval obeys the Poisson distribution with the parameter λ, where λ is set to 1. Orders are randomly distributed within the diamond-shaped region in Figure 5, which also known as the shaded region in Figure 7(a). The release time of all orders is generated according to the approach proposed from Knuth (1997). In addition, the geographic locations of all orders are randomly distributed in the reachable area of the delivery network described in Figure 5.

For small-scale problems, three instances are generated; for medium- and large-scale problems, ten instances are generated. For each instance, GA, EDA, and MAHC all run 30 times, whereas CPLEX and LB run just once. In Equation 18, a metric is given to evaluate the quality of the solution obtained, C_{alg} represents the maximum completion time obtained by algorithm alg, C_{lb} denotes the lower bound of the problem obtained by the LB algorithm, and R_{alg} represents the relative distance between the solution obtained by the algorithm alg and the lower bound. The smaller R_{alg} is, the better the quality of the solution.

$$R_{alg} = (\frac{C_{alg}}{C_{lb}} - 1) * 100\% \tag{18}$$

For algorithms that run multiple times, the average running time and the average R_{alg} are calculated to evaluate the algorithm performance. The average R_{alg} of an algorithm is denoted as $\overline{R}$ in the following section. It should be noted that when R_{alg} is equal to zero, it indicates that the algorithm alg has obtained the optimal solution. However, when R is not zero, it is uncertain whether the algorithm alg has obtained the optimal solution.

All algorithms involved in this study are programmed in Julia (Version: 1.6.0) programming language. In addition, the CPLEX solver is called by JuMP embedded in Julia, Dunning, Huchette, and Lubin (2017). Moreover, all experiments run on a PC (Windows 10, 64-bit) equipped with a 3.4-GHz Intel Core CPU (i7-6700) and 16 GB of RAM.

5.2. *Parameter settings*

The main parameters that affect the performance of MAHC are population size (N), the crossover probability (ρ_c), and mutation probability (ρ_m). We will conduct pre-experiments to tune N and ρ_c. The dynamic mutation probability proposed by Fogarty (1989) is adopted in the mutation operator. Line 17 of Algorithm 2 gives the details of calculating the dynamic mutation probability. The stopping condition of MAHC remains the same as that of Liu et al. (2021). The value of D_{ld} is set to 0.0625, which is adopted from Erzurumluoglu et al. (2016). This parameter limits the lower bound of the difference between the two individuals performing the crossover operator.

To show the problem under study more clearly, we describe the parameters related to the problem here. The structure of the delivery network used in all experiments is shown in Figure 5. From the figure, we can see that there are two FCs and twenty-three BSSs. Besides, the number of drones in each FC is set to 30. According to the study in Welch (2015), the speed of a drone can reach up to 50 mph (80.4672 km/h). Besides, the max endurance (t_0) of a drone with a fully charged battery is 30 min. According to Liu et al. (2021), the packing operation of a parcel takes 2 min (t_1). The time spent in replacing the battery for a drone (t_2) is 3 min. In addition, the time (t_3) spent in taking off, accelerating, decelerating, and landing is 2 min.

5.3. *Parameter tuning*

In addition to the parameters that have been determined above, there are two other parameters, the population size (N) and the crossover probability (ρ_c), that affect the performance of MAHC. We tune them through pre-experiment. Considering that the optimal parameter values are not consistent in different scales of the problem, we set up corresponding experiments for each scale (small-scale, medium-scale, and large-scale) separately.

In the pre-experiment, the population size (N) comes from the set {60, 80, 100}. And the crossover probability (ρ_{local}) comes from the set {0.75, 0.85, 0.95}. Five instances are generated for each scale of the problem. MAHC runs ten times on each instance, and the optimal value obtained is recorded.

The results are summarized in Table 2. It can be seen from Table 2 that for small-scale instances ({20, 25, 30, 35, 40, 45}), the values of N and ρ_c have no effect on the results. Therefore, we might as well take the parameter value combination (N=60 and ρ_c=0.75) as the choice of small-scale instances. In 100-scale instances, there is still no difference between different parameter combinations. However, when the problem scale reaches 200, the parameter value combination (N=80 and ρ_c=0.95) obtains the best solution in 2 of the five instances. Therefore, it is selected for medium-scale instances. In addition, the parameter combination (N=100 and ρ_c=0.95) is selected for the large-scale instances because it obtains the best solution in 8 of the ten instances.

5.4. *Comparison between MAHC, GA, EDA, and CPLEX*

To verify the effectiveness of MAHC, we conduct a comparison experiment between MAHC and the commercial solver CPELX from IBM in this section. Considering that CPLEX has difficulty solving the large-scale NP-hard problems, the small-scale instances ({20, 25, 30, 35, 40, 45}) are employed in the experiment. For each number of orders, three instances are generated. For each instance, MAHC, GA, and EDA run 30 times, and CPLEX runs once. In addition, CPLEX is allowed to run for 3600 s for each instance.

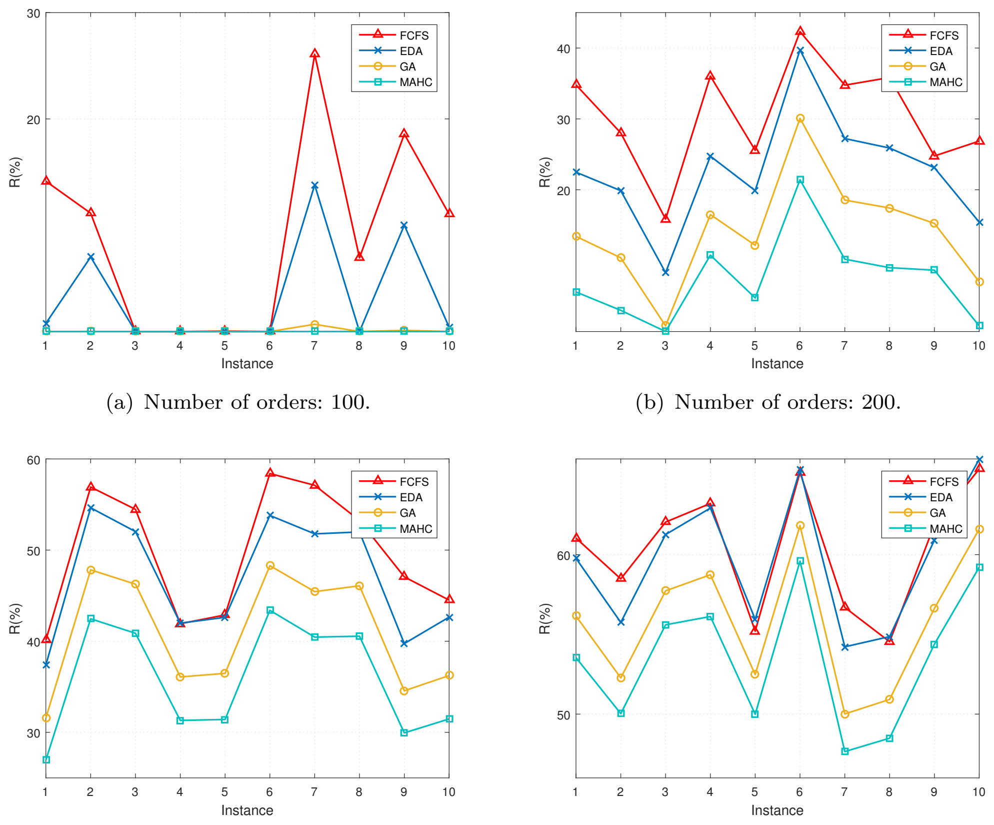
The experimental results are shown in Table 3. The first column indicates the number of orders and the label of the instance. The second column shows the values of the lower bounds obtained by the LB algorithm. The relative distance to the lower bound (R), running time (t), and gap obtained by CPLEX are given in the following three columns. Columns 6 to 14 show the average R ($\overline{R}$), average t ($\overline{t}$), and standard deviation (STD) of GA, EDA, and MAHC. The best R or $\overline{R}$ for each instance is displayed in bold. As shown in Table 3, MAHC and GA obtain the optimal solution within 1 second and EDA in 3 seconds. However, CPLEX can only obtain the optimal solution at instances of an ultra-small-scale ({20, 25, and 30}). When the number of orders increases to 35, CPLEX cannot obtain the optimal solutions within the given 3600 s.

5.5. *Extensive comparison between MAHC, GA, EDA, and FCFS*

The medium- and large-scale instances are used to evaluate the performance of MAHC. Moreover, the classical and efficient scheduling policy, first-come-first-served (FCFS), EDA from Zhou et al. (2016), and GA from Liu et al. (2021) are adopted to compare the difference of MAHC.

The experimental results are shown in Tables 4 and 5. The structure of the two tables is exactly the same. The first column indicates the number of orders and the label of the instance. The second column provides the values of the lower bounds obtained by the LB algorithm. In addition, columns 3–4 give the R and running time (t) of FCFS. Finally, columns 5 to 13 show the average R ($\overline{R}$), average t ($\overline{t}$), and standard deviation (STD) of GA, EDA, and MAHC. The best R or $\overline{R}$ for each instance is shown in bold.

It can be seen from Table 4 that MAHC outperforms GA, EDA, and FCFS in all instances. Moreover, the standard deviation of MAHC is greater than GA and EDA in only one of the 20 instances. When the number of orders equals 100, MAHC can obtain the optimal solution in all instances; however, GA and EDA can only obtain the optimal solution in eight and five instances, respectively. In addition, the running times of MAHC and GA are close. However, EDA takes more time compared with MAHC and GA. When the number of orders increases to 200, the R or $\overline{R}$ of the four algorithms are both greater than zero. Moreover, the $\overline{R}$ of MAHC is the smallest among the four algorithms. Therefore, the solutions obtained by the four algorithms may not be optimal. As shown in Table 5, the performance of MAHC completely surpasses those of the other three algorithms. Meanwhile, MAHC takes less time compared with GA and EDA. Figure 6 illustrates the difference in the quality of the solutions obtained by the four algorithms. It is evident that MAHC outperforms the other four algorithms in all instances.



(a) Number of orders: 100.

(b) Number of orders: 200.

(c) Number of orders: 400.

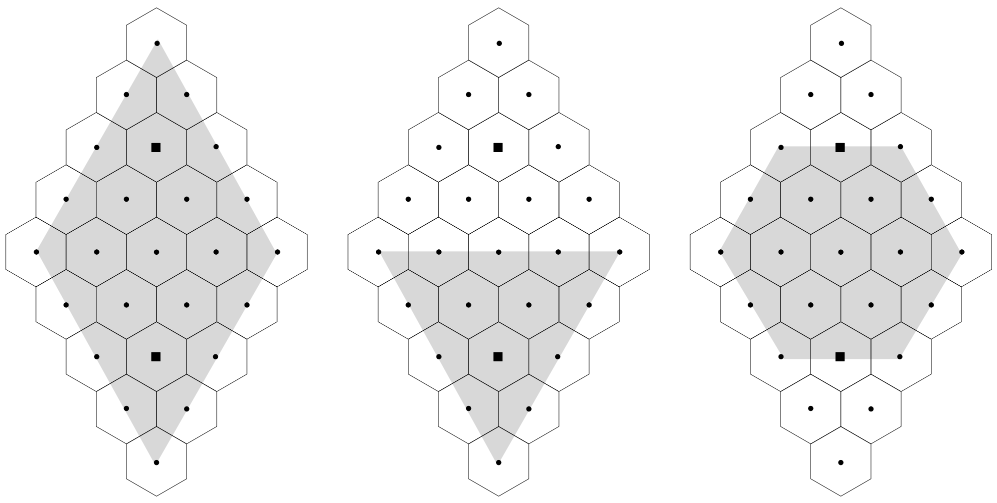
(d) Number of orders: 800.

Figure 6. Comparison of the four algorithms, MAHC, GA, EDA, and FCFS. For MAHC, GA, and EDA, R on the vertical axis stands for $\overline{R}$. The definition of R can be found in Equation 18.

To further verify the difference in the performance of MAHC and GA, we conduct the Wilcoxon rank-sum test, a non-parametric statistical hypothesis test. The null hypothesis (H_0) asserts that two samples containing 30 solutions obtained from MAHC and GA come from the same distribution. As an alternative hypothesis (H_1), the two samples are drawn from a different distribution. The significance level of the hypothesis test is 0.05. The results of Wilcoxon's test are shown in Table 6. The results in Table 6 show that the p-value is more than 0.05 in nine instances with a problem size of 100; therefore, there is no significant difference between MAHC and GA in the corresponding instances. When the number of orders reaches 200, the p-value is much less than 0.05 in all instances; therefore, the null hypothesis is rejected. In other words, there is a significant difference between the solutions obtained by GA and MAHC.

5.6. *Comparison of GS and NPS in different order distribution scenarios*

In all the previous experiments, the geographic locations of the orders are randomly distributed within the diamond-shaped area. To verify the performance of GS, we conducted comparison experiments between GS and baseline policy NPS under different order distribution scenarios. The other two order distributions introduced are a random distribution in the skewed area and a random distribution in the middle area. The three types of order distribution used in this section are illustrated in Figure 7.



(a) Random distribution in the diamond-shaped area. (b) Random distribution in the skewed area. (c) Random distribution in the middle area.

Figure 7. Three types of random distribution of orders. The shaded area in each figure covers the location of orders allowed.

Considering that the performances of GS and NPS are the same in small-scale problems, herein, we only conduct comparative experiments on medium- and large-scale problems. The results can be found in Table 7, where the meanings of $\overline{R}$, $\bar{t}$, and STD in the table are the same as those in Table 5. Table 7 lists the results of three instances for each number of orders. As shown in Table 7, the performance of GS can significantly exceed that of NPS in all distribution scenarios. The advantage of GS lies in its ability to choose a better FC from a global perspective. To show in detail the difference between the two strategies GS and NPS in the selection of FCs, Figure 8 is drawn based on the results from $O_{800}I_1$ (where the number of orders is equal to 800, and the label of the instance is 1). As can be seen from Figure 8, the GS strategy outperforms the intuitive NPS strategy. The difference shown in Figure 8 is that the two-color arcs (corresponding to two FCs) do not overlap in NPS. However, there are overlapping parts in GS. The reason is that NPS selects the closest FC for each parcel, whereas GS allows the more distant FC to be selected for obtaining a better solution.

6. Conclusions and future studies

This study investigated minimizes the makespan of parallel drones with dynamic order arrivals in multiple FCs. It is the first attempt to address the scheduling problems for a drone-based delivery system without the assistance of other vehicles. Inspired by the hexagonal structure of cellular networks, we propose a novel delivery network structure, which can be used to help drone operators design a drone flight network and provide a novel idea for solving the depot location problem. The BSSs are introduced to expand the delivery range of FCs in the delivery network. Drones can deliver parcels through the delivery network. Compared with the use of a truck-assisted method to expand the drone delivery range, the BSS relay method is more flexible and can make full use of the low-cost advantages of drones. The delivery method of drone flying without the assistance of traditional trucks can relieve the pressure of ground traffic congestion and reduce carbon emissions.

An MAHC is developed to address the problem under study. A strategy to prevent inbreeding is proposed in the crossover procedure of MAHC; a local search strategy based on the hill climbing algorithm is used to improve the quality of individuals after mutation. The efficient GS and intuitive NPS strategies are proposed to determine the optimal FC for each order. In addition, a lower bound of the problem is proposed to evaluate the performances of MAHC. Extensive comparative experimental results show that MAHC outperforms GA, EDA, FCFS, and CPLEX. In addition, comparative experiments of two different strategies (GS and NPS) used to select FC show that GS can obtain the best solution in various order distribution scenarios. However, the running time of GS is more than that of NPS in large-scale instances.

Future research can be extended to more complex practical environments. For example, the geographic location data of the merchant's historical orders can be used to optimize the location of FCs based on the delivery network structure as proposed in this study. Considering that the proposed lower bound of the problem is not sufficiently tight in large-scale instances, it is of interest to propose a tighter lower bound based on the nature of the problem and the intrinsic structure of the delivery network.

Funding

This research received funding from the National Natural Science Foundation of China [Grant number 71671168]. This research was also supported by the University of Tennessee Health Innovation Technology & Simulation (HITS) Laboratory.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Agatz, Niels, Paul Bouman, and Marie Schmidt. 2018. "Optimization approaches for the traveling salesman problem with drone." *Transportation Science* 52 (4): 965–981.

Amazon. 2019. "Amazon.com: Prime Air." `https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011`. [Online; accessed 10-Oct-2020].

Boryczka, Urszula, and Krzysztof Szwarc. 2019. "Selected variants of a Memetic Algorithm for JSP–a comparative study." *International Journal of Production Research* 57 (22): 7142–7157.

Bursztynsky, Jessica. 2020. "CVS and UPS will use drones to deliver prescriptions in a retirement community amid coronavirus outbreak." `https://www.cnbc.com/2020/`

04/27/coronavirus-cvs-ups-delivering-prescriptions-with-drones.html. [Online; accessed 10-Oct-2020].

Carlsson, John Gunnar, and Siyuan Song. 2018. "Coordinated Logistics with a Truck and a Drone." *Management Science* 64 (9): 4052–4069.

Chung, Sung Hoon, Bhawesh Sah, and Jinkun Lee. 2020. "Optimization for Drone and Drone-truck Combined Operations: A Review of the State of the Art and Future Directions." *Computers & Operations Research* 105004.

Cormen, Thomas H, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *The Floyd-Warshall algorithm*, 3rd ed., Chap. 25.2. MIT press.

Curlander, James Christopher, Asaf Gilboa-Amir, Lauren Marie Kisser, Robert Arthur Koch, and Ricky Dean Welsh. 2017. "Multi-level fulfillment center for unmanned aerial vehicles." June. US Patent 9777502.

Davis, Lawrence. 1991. *Schedule optimization using genetic algorithms.* New York: Van Nostrand Reinhold.

Dayarian, Iman, Martin Savelsbergh, and John-Paul Clarke. 2020. "Same-day delivery with drone resupply." *Transportation Science* 54 (1): 229–249.

DiNota, Anthony, Steve Douglas, and Dave Marcontell. 2019. "Why The Skies Aren't Filled With Delivery Drones ... Yet." https://www.forbes.com/sites/oliverwyman/2019/10/07/why-the-skies-arent-filled-with-delivery-drones-yet/?sh=7af2a82c3bc9. [Online; accessed 10-Oct-2020].

Dunning, Iain, Joey Huchette, and Miles Lubin. 2017. "JuMP: A Modeling Language for Mathematical Optimization." *SIAM Review* 59 (2): 295–320.

Erzurumluoglu, A Mesut, Hashem A Shihab, Santiago Rodriguez, Tom R Gaunt, and Ian NM Day. 2016. "Importance of genetic studies in consanguineous populations for the characterization of novel human gene functions." *Annals of human genetics* 80 (3): 187–196.

Fogarty, Terence C. 1989. "Varying the probability of mutation in the genetic algorithm." In *Proceedings of the 3rd International Conference on Genetic Algorithms*, 104–109.

Gonzalez-R, Pedro L, David Canca, Jose L Andrade-Pineda, Marcos Calle, and Jose M Leon-Blanco. 2020. "Truck-drone team logistics: A heuristic approach to multi-drop route planning." *Transportation Research Part C: Emerging Technologies* 114: 657–680.

Kidron, Ella. 2020. "WORLD ECONOMIC FORUM HIGHLIGHTS JD'S USE OF DRONES IN COVID-19 FIGHT." https://jdcorporateblog.com/world-economic-forum-highlights-jds-use-of-drones-in-covid-19-fight/. [Online; accessed 10-Oct-2020].

Knuth, Donald Ervin. 1997. *Art of computer programming, volume 2: Seminumerical algorithms.* 3rd ed. Addison-Wesley Professional.

Liu, Chuang, Huaping Chen, Xueping Li, and Zeyu Liu. 2021. "A scheduling decision support model for minimizing the number of drones with dynamic package arrivals and personalized deadlines." *Expert Systems with Applications* 167: 114157.

Moscato, Pablo, et al. 1989. "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms." *Caltech concurrent computation program, C3P Report* 826: 1989.

Palmer, Annie. 2020. "Amazon wins FAA approval for Prime Air drone delivery fleet." https://www.cnbc.com/2020/08/31/amazon-prime-now-drone-delivery-fleet-gets-faa-approval.html. [Online; accessed 10-Oct-2020].

Poikonen, Stefan, and Bruce Golden. 2020. "The mothership and drone routing problem." *INFORMS Journal on Computing* 32 (2): 249–262.

Poikonen, Stefan, Bruce Golden, and Edward A Wasil. 2019. "A branch-and-bound approach to the traveling salesman problem with a drone." *INFORMS Journal on Computing* 31 (2): 335–346.

Shen, Yaohan, Xianhao Xu, Bipan Zou, and Hongwei Wang. 2020. "Operating policies in multi-warehouse drone delivery systems." *International Journal of Production Research* 0 (0): 1–17.

Suzuki, Koji AO, Paulo Kemper Filho, and James R Morrison. 2012. "Automatic battery replacement system for UAVs: Analysis and design." *Journal of Intelligent & Robotic Systems* 65 (1-4): 563–586.

Welch, Adrienne. 2015. "A cost-benefit analysis of Amazon Prime Air." Master's thesis, University of Tennessee at Chattanooga, Chattanooga, TN, US. Honors Theses.

Xu, Jianyou, Yunqiang Yin, TCE Cheng, Chin-Chia Wu, and Shusheng Gu. 2014. "An improved memetic algorithm based on a dynamic neighbourhood for the permutation flowshop scheduling problem." *International Journal of Production Research* 52 (4): 1188–1199.

Yang, Junwei, and Timothy Reuter. 2020. "3 ways China is using drones to fight coronavirus." `https://www.weforum.org/agenda/2020/03/three-ways-china-is-using-drones-to-fight-coronavirus`. [Online; accessed 10-Oct-2020].

Zhou, Shengchao, Xueping Li, Huaping Chen, and Cong Guo. 2016. "Minimizing makespan in a no-wait flowshop with two batch processing machines using estimation of distribution algorithm." *International Journal of Production Research* 54 (16): 4919–4937.

Table 2. Pre-experiment

Instance	$N = 60$			$N = 80$			$N = 100$		
	$\rho_c = 0.75$	$\rho_c = 0.85$	$\rho_c = 0.95$	$\rho_c = 0.75$	$\rho_c = 0.85$	$\rho_c = 0.95$	$\rho_c = 0.75$	$\rho_c = 0.85$	$\rho_c = 0.95$
$O_{20}I_1$[a]	6812.6	6812.6	6812.6	6812.6	6812.6	6812.6	6812.6	6812.6	6812.6
$O_{20}I_2$	6918.7	6918.7	6918.7	6918.7	6918.7	6918.7	6918.7	6918.7	6918.7
$O_{20}I_3$	7292.73	7292.73	7292.73	7292.73	7292.73	7292.73	7292.73	7292.73	7292.73
$O_{20}I_4$	5096.19	5096.19	5096.19	5096.19	5096.19	5096.19	5096.19	5096.19	5096.19
$O_{20}I_5$	5382.58	5382.58	5382.58	5382.58	5382.58	5382.58	5382.58	5382.58	5382.58
$O_{25}I_1$	5793.41	5793.41	5793.41	5793.41	5793.41	5793.41	5793.41	5793.41	5793.41
$O_{25}I_2$	5537.88	5537.88	5537.88	5537.88	5537.88	5537.88	5537.88	5537.88	5537.88
$O_{25}I_3$	7144.88	7144.88	7144.88	7144.88	7144.88	7144.88	7144.88	7144.88	7144.88
$O_{25}I_4$	6863.76	6863.76	6863.76	6863.76	6863.76	6863.76	6863.76	6863.76	6863.76
$O_{25}I_5$	7489.48	7489.48	7489.48	7489.48	7489.48	7489.48	7489.48	7489.48	7489.48
$O_{30}I_1$	7272.69	7272.69	7272.69	7272.69	7272.69	7272.69	7272.69	7272.69	7272.69
$O_{30}I_2$	6359.94	6359.94	6359.94	6359.94	6359.94	6359.94	6359.94	6359.94	6359.94
$O_{30}I_3$	6380.02	6380.02	6380.02	6380.02	6380.02	6380.02	6380.02	6380.02	6380.02
$O_{30}I_4$	7678.43	7678.43	7678.43	7678.43	7678.43	7678.43	7678.43	7678.43	7678.43
$O_{30}I_5$	7744.63	7744.63	7744.63	7744.63	7744.63	7744.63	7744.63	7744.63	7744.63
$O_{35}I_1$	7026.4	7026.4	7026.4	7026.4	7026.4	7026.4	7026.4	7026.4	7026.4
$O_{35}I_2$	5847.75	5847.75	5847.75	5847.75	5847.75	5847.75	5847.75	5847.75	5847.75
$O_{35}I_3$	8015.35	8015.35	8015.35	8015.35	8015.35	8015.35	8015.35	8015.35	8015.35
$O_{35}I_4$	7546.99	7546.99	7546.99	7546.99	7546.99	7546.99	7546.99	7546.99	7546.99
$O_{35}I_5$	7858.63	7858.63	7858.63	7858.63	7858.63	7858.63	7858.63	7858.63	7858.63
$O_{40}I_1$	6601.92	6601.92	6601.92	6601.92	6601.92	6601.92	6601.92	6601.92	6601.92
$O_{40}I_2$	7590.32	7590.32	7590.32	7590.32	7590.32	7590.32	7590.32	7590.32	7590.32
$O_{40}I_3$	8661.73	8661.73	8661.73	8661.73	8661.73	8661.73	8661.73	8661.73	8661.73
$O_{40}I_4$	7748.64	7748.64	7748.64	7748.64	7748.64	7748.64	7748.64	7748.64	7748.64
$O_{40}I_5$	7709.65	7709.65	7709.65	7709.65	7709.65	7709.65	7709.65	7709.65	7709.65
$O_{45}I_1$	7863.15	7863.15	7863.15	7863.15	7863.15	7863.15	7863.15	7863.15	7863.15
$O_{45}I_2$	8207.94	8207.94	8207.94	8207.94	8207.94	8207.94	8207.94	8207.94	8207.94
$O_{45}I_3$	7844.33	7844.33	7844.33	7844.33	7844.33	7844.33	7844.33	7844.33	7844.33
$O_{45}I_4$	8004.78	8004.78	8004.78	8004.78	8004.78	8004.78	8004.78	8004.78	8004.78
$O_{45}I_5$	9133.8	9133.8	9133.8	9133.8	9133.8	9133.8	9133.8	9133.8	9133.8
$O_{100}I_1$	10373.53	10373.53	10373.53	10373.53	10373.53	10373.53	10373.53	10373.53	10373.53
$O_{100}I_2$	9932.61	9932.61	9932.61	9932.61	9932.61	9932.61	9932.61	9932.61	9932.61
$O_{100}I_3$	10985.01	10985.01	10985.01	10985.01	10985.01	10985.01	10985.01	10985.01	10985.01
$O_{100}I_4$	12277.76	12277.76	12277.76	12277.76	12277.76	12277.76	12277.76	12277.76	12277.76
$O_{100}I_5$	12186.47	12186.47	12186.47	12186.47	12186.47	12186.47	12186.47	12186.47	12186.47
$O_{200}I_1$	18705.93	18339.03	18031.14	18646.37	18402.69	**17957.02**	18694.24	18414.4	18066.45
$O_{200}I_2$	17955.82	17437.13	**17035.44**	17998.3	17780.38	17206	17589.75	17532.87	17125.99
$O_{200}I_3$	18346.28	18346.28	18346.28	18346.28	18346.28	18346.28	18346.28	18346.28	18346.28
$O_{200}I_4$	18463.09	18292.38	18098.14	18607.35	18240.11	**17966.03**	18134.19	18175.26	18095.59
$O_{200}I_5$	18306.26	17990.7	17957.28	18335.47	18147.56	17782.65	18447.05	18155.65	**17731.18**
$O_{400}I_1$	38219.68	37819.37	37651.85	38305.49	37919.91	37535.4	38193.71	37898.68	**37209.7**
$O_{400}I_2$	41203.96	40760.82	40629.28	41008.02	40840.28	40361.69	41032.54	40819.15	**40242.96**
$O_{400}I_3$	40662.97	40313.24	40300.72	40722.21	40254.61	40165.55	40673.5	40288.66	**39994.92**
$O_{400}I_4$	38621.47	38812.42	38298.28	38875.45	38547.14	38234.72	38736.07	38374.02	**38053.57**
$O_{400}I_5$	41118.91	40414.6	40411.4	40855.52	40526.78	40339.98	40820.28	40422.06	**40076.58**
$O_{800}I_1$	80572.05	80425.28	80661.08	80837.42	80367.66	80177.58	80467.98	80137.7	**79959.96**
$O_{800}I_2$	80178.81	79921.16	79604.53	80060.44	79791.39	79698	80035.79	79625.38	**79487.42**
$O_{800}I_3$	80236.88	80042.58	80163.87	80235.12	80075.3	**79521.37**	80451.61	79750.75	79689.02
$O_{800}I_4$	83498.54	83546.53	83076.76	83496.89	83336.31	**82733.14**	83384.12	83347.77	82843.26
$O_{800}I_5$	81551.47	81130.94	81004.63	81524.9	81155.48	80793.53	81345.25	81320.89	**80781.21**

[a] $O_{20}I_1$ means that the number of orders is 20, and the label of the instance is 1.

Table 3. Comparison results of MAHC, CPLEX, GA, and EDA in small-scale instances

Instance	LB	CPLEX			GA			EDA			MAHC		
		R	t	GAP	$\overline{R}$	$\bar{t}$	STD	$\overline{R}$	$\bar{t}$	STD	$\overline{R}$	$\bar{t}$	STD
$O_{20}I_1$[a]	6812.60	**0.00**	3600.06	0.12	**0**	0.21	0	**0**	1.04	0	**0**	0.17	0
$O_{20}I_2$	6918.70	**0.00**	3600.05	0.09	**0**	0.21	0	**0**	1.04	0	**0**	0.17	0
$O_{20}I_3$	7292.73	**0.00**	3600.06	0.13	**0**	0.21	0	**0**	1.04	0	**0**	0.17	0
$O_{25}I_1$	5793.41	**0.00**	3600.06	0.22	**0**	0.25	0	**0**	1.31	0	**0**	0.20	0
$O_{25}I_2$	5537.88	**0.00**	3600.08	0.16	**0**	0.25	0	**0**	1.32	0	**0**	0.21	0
$O_{25}I_3$	7144.88	**0.00**	3600.11	0.1	**0**	0.25	0	**0**	1.32	0	**0**	0.21	0
$O_{30}I_1$	7272.69	**0.00**	3600.13	0.08	**0**	0.28	0	**0**	1.69	0	**0**	0.22	0
$O_{30}I_2$	6359.94	**0.00**	3600.11	1.57E+33	**0**	0.27	0	**0**	1.69	0	**0**	0.23	0
$O_{30}I_3$	6380.02	**0.00**	3600.11	0.03	**0**	0.27	0	**0**	1.87	0	**0**	0.22	0
$O_{35}I_1$	7026.40	81.69	3600.10	7.83E+32	**0**	0.29	0	**0**	2.25	0	**0**	0.24	0
$O_{35}I_2$	5847.75	52.08	3600.10	1.12E+33	**0**	0.30	0	**0**	2.2	0	**0**	0.24	0
$O_{35}I_3$	8015.35	42.74	3600.12	8.74E+32	**0**	0.29	0	**0**	2.03	0	**0**	0.23	0
$O_{40}I_1$	6601.92	81.73	3600.21	8.33E+32	**0**	0.31	0	**0**	2.38	0	**0**	0.25	0
$O_{40}I_2$	7590.32	45.63	3600.19	9.05E+32	**0**	0.31	0	**0**	2.38	0	**0**	0.26	0
$O_{40}I_3$	8661.73	19.70	3600.14	9.65E+32	**0**	0.32	0	**0**	2.49	0	**0**	0.25	0
$O_{45}I_1$	7863.15	59.30	3600.15	7.98E+32	**0**	0.34	0	**0**	2.86	0	**0**	0.27	0
$O_{45}I_2$	8207.94	49.49	3600.14	8.15E+32	**0**	0.34	0	**0**	2.84	0	**0**	0.28	0
$O_{45}I_3$	7844.33	35.04	3600.15	9.44E+32	**0**	0.34	0	**0**	2.84	0	**0**	0.27	0

[a] $O_{20}I_1$ means that the number of orders is 20, and the label of the instance is 1.

Table 4. Comparison results of MAHC, FCFS, GA, and EDA in medium-scale instances.

Instance	LB	FCFS		GA			EDA			MAHC		
		R	t	$\overline{R}$	$\bar{t}$	STD	$\overline{R}$	$\bar{t}$	STD	$\overline{R}$	$\bar{t}$	STD
$O_{100}I_1$[a]	10373.53	14.14	0.0012	**0**	1.3486	0	0.77	8.0294	174.11	**0**	1.1404	0
$O_{100}I_2$	9932.61	11.13	0.0013	**0**	1.7526	0.32	7.01	8.0361	307.95	**0**	1.4588	0
$O_{100}I_3$	10985.01	**0**	0.0017	**0**	1.3405	0	**0**	8.0424	0	**0**	1.1464	0
$O_{100}I_4$	12277.76	**0**	0.0012	**0**	1.3385	0	**0**	9.0857	0	**0**	1.1038	0
$O_{100}I_5$	12186.47	0.04	0.0011	**0**	1.3432	0	**0**	9.0933	0	**0**	1.169	0
$O_{100}I_6$	11173.19	**0**	0.0013	**0**	1.3505	0	**0**	8.1839	0	**0**	1.1752	0
$O_{100}I_7$	9921.26	26.08	0.0011	0.68	2.4097	85.36	13.79	8.6538	184.64	**0**	2.0535	0
$O_{100}I_8$	11954.83	6.93	0.0011	**0**	1.3449	0	**0**	9.0911	0	**0**	1.1243	0
$O_{100}I_9$	10247.17	18.57	0.0011	0.11	1.8663	54.05	10	8.7511	258.02	**0**	1.4644	0
$O_{100}I_{10}$	10423.53	11.04	0.0011	**0**	1.3444	0	0.41	9.1192	165.51	**0**	1.1897	0
$O_{200}I_1$	17261.6	34.8	0.002	13.43	9.7145	276.38	22.48	25.8883	261.44	**5.58**	10.6687	190.48
$O_{200}I_2$	17021.91	28.02	0.0017	10.4	10.8452	162.13	19.87	24.1637	171.46	**2.99**	10.9091	266.9
$O_{200}I_3$	18346.28	15.86	0.0017	0.8	16.5844	144.24	8.31	25.3317	245.09	**0**	4.1298	0
$O_{200}I_4$	16458.92	36.04	0.0017	16.5	9.4047	202.7	24.74	25.0761	206.43	**10.78**	9.8535	132.89
$O_{200}I_5$	17281.03	25.57	0.0016	12.17	15.12	261.8	19.89	25.7736	193.76	**4.74**	11.1448	190.3
$O_{200}I_6$	15271.6	42.32	0.0037	30.08	10.1208	264.36	39.74	24.6589	266.17	**21.41**	10.0944	190.84
$O_{200}I_7$	16198.77	34.76	0.0016	18.55	9.8608	287.79	27.23	24.6359	208.45	**10.21**	11.4245	188.55
$O_{200}I_8$	16657.89	35.84	0.0017	17.43	22.0102	208.52	25.89	24.6262	239.4	**9.01**	12.0545	190.58
$O_{200}I_9$	16600.75	24.77	0.0017	15.26	8.8687	210.12	23.11	25.4791	187.72	**8.66**	11.3993	180.77
$O_{200}I_{10}$	17807.52	26.87	0.0017	7.02	9.8691	177.35	15.43	25.2238	224.97	**0.79**	11.8232	129.02

[a] $O_{100}I_1$ means that the number of orders is 100, and the label of the instance is 1.

Table 5. Comparison results of MAHC, FCFS, GA, and EDA in large-scale instances.

Instance	LB	FCFS		GA			EDA			MAHC		
		R	t	$\overline{R}$	$\bar{t}$	STD	$\overline{R}$	$\bar{t}$	STD	$\overline{R}$	$\bar{t}$	STD
$O_{400}I_1$[a]	29668.09	40.11	0.0027	31.57	39.9181	261.88	37.39	71.7263	276.89	**26.95**	26.2556	258.69
$O_{400}I_2$	28461.04	56.92	0.0028	47.83	92.7629	263.69	54.64	72.8755	243.13	**42.46**	24.6603	270.26
$O_{400}I_3$	28528.58	54.47	0.0027	46.26	68.3723	310.41	52.03	71.2837	269.48	**40.87**	24.0668	199.24
$O_{400}I_4$	29229.87	41.92	0.0027	36.08	42.6188	276.29	41.98	66.8748	250	**31.31**	23.1175	198.79
$O_{400}I_5$	30769.95	42.88	0.0035	36.47	37.7104	332.19	42.62	67.3031	293.02	**31.41**	24.3822	302.06
$O_{400}I_6$	27905.54	58.4	0.0028	48.29	50.6639	249.45	53.8	66.1274	208.07	**43.39**	23.3494	340.57
$O_{400}I_7$	28572.29	57.11	0.0028	45.44	56.0898	283.53	51.78	68.3187	284.3	**40.45**	23.0122	252.17
$O_{400}I_8$	27704.48	53.25	0.0027	46.08	37.1283	260.86	51.99	65.3406	243.94	**40.55**	24.5624	312.53
$O_{400}I_9$	30424.64	47.06	0.0028	34.56	29.5665	299.86	39.74	67.2553	269.31	**29.94**	23.148	247.74
$O_{400}I_{10}$	30185.5	44.51	0.0038	36.23	56.8105	291.65	42.64	66.5041	248.7	**31.47**	24.0814	192.66
$O_{800}I_1$	52279.69	61.04	0.0049	56.15	131.5085	226.72	59.77	225.2226	280.24	**53.54**	39.4342	352.84
$O_{800}I_2$	53236.3	58.48	0.005	52.27	116.8084	223.03	55.75	230.4748	262.29	**50.03**	37.7644	240.43
$O_{800}I_3$	51466.95	62.04	0.005	57.73	104.7553	255.41	61.26	231.8442	308.96	**55.6**	34.1234	304.73
$O_{800}I_4$	53298.56	63.21	0.0051	58.74	86.7668	276.53	62.92	231.6295	185.18	**56.12**	37.1374	260.87
$O_{800}I_5$	54110.19	55.2	0.005	52.48	97.2125	268.65	55.94	232.907	387.39	**49.99**	37.5615	286.73
$O_{800}I_6$	53479.38	65.16	0.0049	61.82	86.0399	316.51	65.33	238.0623	280.76	**59.59**	35.5152	278.11
$O_{800}I_7$	55077.29	56.69	0.0052	50.01	65.2828	266	54.22	239.5442	307.66	**47.65**	35.3209	317.28
$O_{800}I_8$	55369.63	54.54	0.0048	50.92	76.7655	295.56	54.86	204.5204	375.5	**48.49**	34.2891	231.5
$O_{800}I_9$	53482.86	61.87	0.005	56.63	88.4526	263.54	60.91	204.7502	262.1	**54.39**	33.0494	266.21
$O_{800}I_{10}$	51393.85	65.37	0.0052	61.61	83.377	298.81	65.98	204.5644	310.61	**59.21**	34.2381	285.43

[a] $O_{400}I_1$ means that the number of orders is 400, and the label of the instance is 1.

Table 6. Wilcoxon rank–sum test results of MAHC and GA.

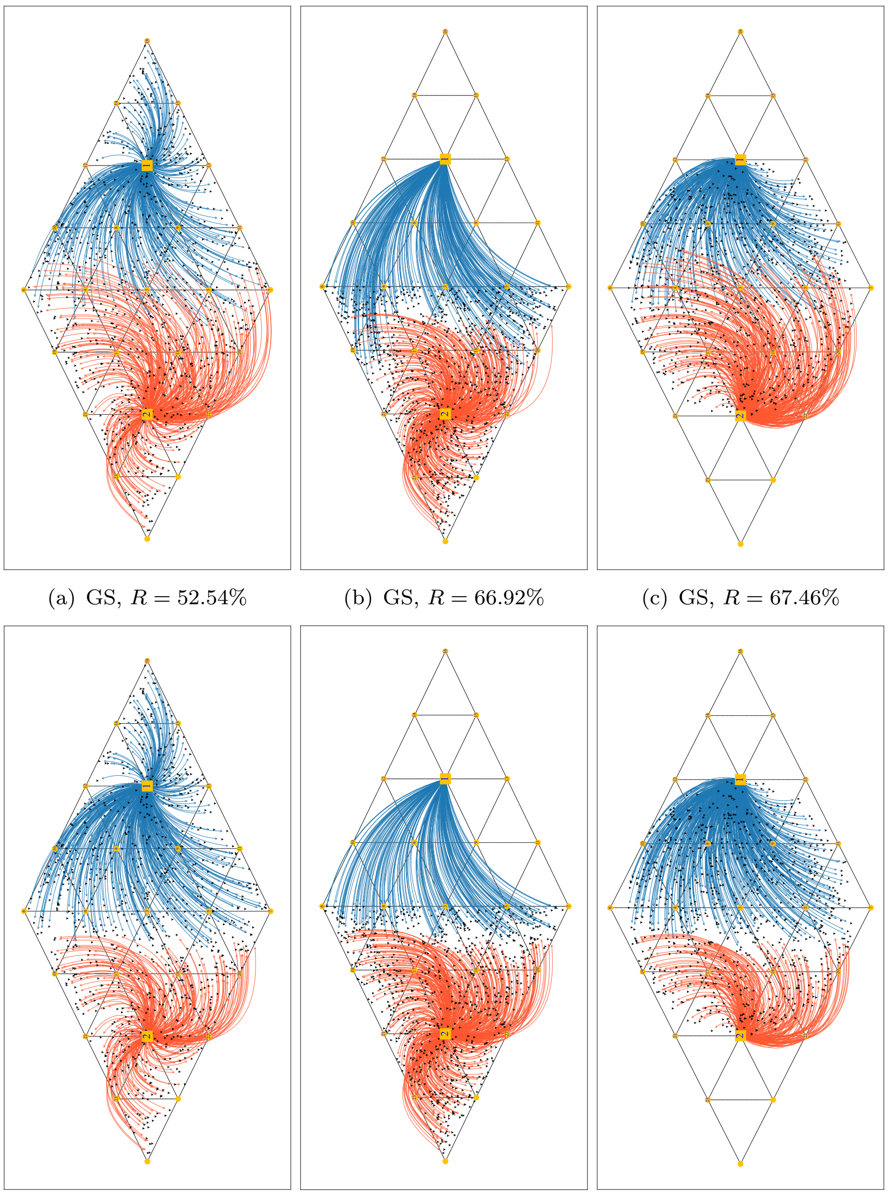
Medium-scale instances	MAHC VS GA		Large-scale instances	MAHC VS GA	
	statistic	p-value		statistic	p-value
$O_{100}I_1$[a]	450	1	$O_{400}I_1$	0	3.02E-11
$O_{100}I_2$	435	0.333711	$O_{400}I_2$	0	3.02E-11
$O_{100}I_3$	450	1	$O_{400}I_3$	0	3.02E-11
$O_{100}I_4$	450	1	$O_{400}I_4$	0	3.02E-11
$O_{100}I_5$	450	1	$O_{400}I_5$	0	3.02E-11
$O_{100}I_6$	450	1	$O_{400}I_6$	3	4.08E-11
$O_{100}I_7$	105	5.84E-09	$O_{400}I_7$	0	3.02E-11
$O_{100}I_8$	450	1	$O_{400}I_8$	0	3.02E-11
$O_{100}I_9$	420	0.160802	$O_{400}I_9$	0	3.02E-11
$O_{100}I_{10}$	450	1	$O_{400}I_{10}$	0	3.02E-11
$O_{200}I_1$	0	3.02E-11	$O_{800}I_1$	0	3.02E-11
$O_{200}I_2$	0	3.02E-11	$O_{800}I_2$	0	3.02E-11
$O_{200}I_3$	90	1.95E-09	$O_{800}I_3$	1	3.34E-11
$O_{200}I_4$	0	3.02E-11	$O_{800}I_4$	0	3.02E-11
$O_{200}I_5$	0	3.02E-11	$O_{800}I_5$	1	3.34E-11
$O_{200}I_6$	0	3.02E-11	$O_{800}I_6$	2	3.69E-11
$O_{200}I_7$	0	3.02E-11	$O_{800}I_7$	1	3.34E-11
$O_{200}I_8$	0	3.02E-11	$O_{800}I_8$	0	3.02E-11
$O_{200}I_9$	0	3.02E-11	$O_{800}I_9$	0	3.02E-11
$O_{200}I_{10}$	0	2.86E-11	$O_{800}I_{10}$	4	4.50E-11

[a] $O_{100}I_1$ means that the number of orders is 100, and the label of the instance is 1.

Table 7. Comparison of GS and NPS strategies in different order distribution scenarios.

Category	LB	GS			NPS		
		$\overline{R}$	$\overline{t}$	STD	$\overline{R}$	$\overline{t}$	STD
$O_{100}I_1D_1$[a]	10373.53	**0**	1.1081	0	7.05	2.2231	170.59
$O_{100}I_1D_2$	12186.47	**0**	1.4921	0	25.1	2.0067	89.8
$O_{100}I_1D_3$	10423.53	**0**	1.0909	0	0.08	1.7334	32.33
$O_{100}I_2D_1$	11160.11	**0**	1.3292	0	**0**	1.2687	0
$O_{100}I_2D_2$	11675.41	**0.13**	3.5397	27.81	43.24	3.1655	140.45
$O_{100}I_2D_3$	10454.57	0	2.0539	0	12.43	2.4443	60.67
$O_{100}I_3D_1$	10537.19	**0**	1.4261	0	19.5	2.0501	71.26
$O_{100}I_3D_2$	10392.98	**0**	1.1686	0	11.06	2.6058	165.13
$O_{100}I_3D_3$	11260.36	**0.01**	1.6622	7.96	19	1.9274	66.16
$O_{200}I_1D_1$	17261.6	**5.48**	10.6716	278.59	40.88	7.4233	149.51
$O_{200}I_1D_2$	17281.03	**2.84**	10.0912	227.05	39.63	6.2996	212.76
$O_{200}I_1D_3$	17807.52	**0**	4.068	0	17.63	7.0233	218.64
$O_{200}I_2D_1$	17326.7	**21.52**	8.8496	174.5	52.15	6.751	194.4
$O_{200}I_2D_2$	17741.93	**13.07**	10.6665	394.47	41.15	6.994	145.22
$O_{200}I_2D_3$	17931.15	**11.99**	7.9284	153.53	30.64	6.8199	155.64
$O_{200}I_3D_1$	16168.52	**19.22**	11.6386	229.39	67.41	7.2851	160.83
$O_{200}I_3D_2$	16344.35	**16.16**	10.5957	254.32	68.96	6.8281	224.37
$O_{200}I_3D_3$	16747.88	**9.95**	11.4758	216.46	57.88	7.3667	146.49
$O_{400}I_1D_1$	29668.09	**26.93**	24.0789	214.45	65.31	13.8977	190.96
$O_{400}I_1D_2$	30769.95	**42.67**	21.2624	199.66	108.63	14.1755	206.93
$O_{400}I_1D_3$	30185.5	**40.94**	21.4702	276.02	89.45	13.8843	220.81
$O_{400}I_2D_1$	29789.82	**44.83**	22.4308	313.12	82.61	13.5184	243.36
$O_{400}I_2D_2$	30377.43	**47.94**	24.5953	354.84	82.99	12.6516	156.38
$O_{400}I_2D_3$	30582.17	**50.76**	25.6226	290.02	91.47	13.3732	223.34
$O_{400}I_3D_1$	30738.38	**42.5**	21.6016	203.46	103.92	12.9724	215.84
$O_{400}I_3D_2$	30345.93	**45.07**	21.1566	210.2	106.42	12.9756	245.48
$O_{400}I_3D_3$	30582.26	**53.31**	22.4914	189.76	115.33	13.3266	193.34
$O_{800}I_1D_1$	52279.69	**53.75**	33.199	279.8	99.63	22.6012	193.46
$O_{800}I_1D_2$	54110.19	**50.06**	36.8987	327.73	97.69	26.0545	249.32
$O_{800}I_1D_3$	51393.85	**55.51**	39.3641	257.2	97.3	27.7586	269.17
$O_{800}I_2D_1$	52217.59	**68.3**	36.929	351.32	105.55	24.6372	219.5
$O_{800}I_2D_2$	55414.83	**65.61**	42.9041	405.9	112.9	22.6629	302.64
$O_{800}I_2D_3$	52499.82	**70.36**	38.0519	400.88	99.02	25.4836	271.36
$O_{800}I_3D_1$	52200.44	**68.55**	40.3121	363.6	144.07	23.0906	340.35
$O_{800}I_3D_2$	55235.17	**65.65**	36.1877	253.28	130.6	24.4665	279.94
$O_{800}I_3D_3$	51508.15	**64.29**	41.3265	319.73	134.04	24.7032	319.72

[a] $O_{100}I_1D_1$ means that the number of orders is 100, and the label of the instance is 1. Moreover, D_1, D_2 and D_3 in the first column represent the order distribution described by the three sub-graphs (a, b and c) in Figure 7 respectively.



(a) GS, $R = 52.54\%$ (b) GS, $R = 66.92\%$ (c) GS, $R = 67.46\%$

(d) NPS, $R = 98.92\%$ (e) NPS, $R = 104.55\%$ (f) NPS, $R = 142.96\%$

Figure 8. Comparison of GS and NPS in selecting the best FC. The distribution of orders in subgraphs (a) and (d) is shown in Figure 7(a). The distribution of orders in subgraphs (b) and (e) is shown in Figure 7(b). The distribution of orders in subgraphs (c) and (f) is shown in Figure 7(c). Each blue arc indicates that the corresponding order is addressed by FC 1, each red edge means that the corresponding order is processed by FC 2, and each black dot represents the destination of an order.