Defence Technology 43 (2025) 1–12

Contents lists available at ScienceDirect

Defence Technology

journal homepage: www.keaipublishing.com/en/journals/defence-technology

Autonomous sortie scheduling for carrier aircraft fleet under towing mode

Zhilong Deng [a], Xuanbo Liu [b], Yuqi Dou [b], Xichao Su [c], Haixu Li [d, e], Lei Wang [b], Xinwei Wang [a, *]

[a] *Department of Engineering Mechanics, State Key Laboratory of Structural Analysis, Optimization and CAE Software for Industrial Equipment, Dalian University of Technology, Dalian, 116024, China*
[b] *School of Mathematical Sciences, Dalian University of Technology, Dalian 116024, China*
[c] *Naval Aviation University, Yantai 264001, China*
[d] *China State Shipbuilding Corporation Systems Engineering Research Institute, Beijing 100094, China*
[e] *Department of Automation, School of Information Science and Technology, Tsinghua University, Beijing 100084, China*

ARTICLE INFO

Article history:
Received 8 May 2024
Received in revised form
13 June 2024
Accepted 23 July 2024
Available online 24 July 2024

Keywords:
Carrier aircraft
Autonomous sortie scheduling
Resource allocation
Collision-avoidance
Hybrid flow-shop scheduling problem

ABSTRACT

Safe and efficient sortie scheduling on the confined flight deck is crucial for maintaining high combat effectiveness of the aircraft carrier. The primary difficulty exactly lies in the spatiotemporal coordination, i.e., allocation of limited supporting resources and collision-avoidance between heterogeneous dispatch entities. In this paper, the problem is investigated in the perspective of hybrid flow-shop scheduling problem by synthesizing the precedence, space and resource constraints. Specifically, eight processing procedures are abstracted, where tractors, preparing spots, catapults, and launching are virtualized as machines. By analyzing the constraints in sortie scheduling, a mixed-integer planning model is constructed. In particular, the constraint on preparing spot occupancy is improved to further enhance the sortie efficiency. The basic trajectory library for each dispatch entity is generated and a delayed strategy is integrated to address the collision-avoidance issue. To efficiently solve the formulated HFSP, which is essentially a combinatorial problem with tightly coupled constraints, a chaos-initialized genetic algorithm is developed. The solution framework is validated by the simulation environment referring to the Fort-class carrier, exhibiting higher sortie efficiency when compared to existing strategies. And animation of the simulation results is available at www.bilibili.com/video/BV14t421A7Tt/. The study presents a promising supporting technique for autonomous flight deck operation in the foreseeable future, and can be easily extended to other supporting scenarios, e.g., ammunition delivery and aircraft maintenance.

1. Introduction

1.1. Background

Considering the growing operation intensity and execution complexity in modern shipborne combat, more carrier aircraft are expected to be equipped onboard. During the sortie phase, each aircraft is dispatched to an assigned catapult for launching. The sortie scheduling poses great challenges in two aspects. The one is how to reasonably allocate the limited supporting resource, e.g., preparing spot, catapult, etc. The other is how to realize collision-avoidance between dispatch entities. Since the World War-II, the U.S. Navy have used the Oujia board for assistance. To deal with the growing number of carrier aircraft, many countries have launched the development of aviation data management and control system to further improve the scheduling performance in the last two decades. However, it's worth noting that existing decision support systems mainly focus on the resource allocation problem. Then staffs still have to artificially plan the maneuvering path for dispatch entities, exposing drawbacks in dispatch safety and planning efficiency. Additionally, as more unmanned aircraft represented by MQ-25 and X–47B are deployed onboard, high-level ship-aircraft adaptation capability are expected. To overcome such a dilemma, a promising way is to develop a mathematical

[*] Corresponding author.
E-mail address: wangxinwei@dlut.edu.cn (X. Wang).
Peer review under responsibility of China Ordnance Society

https://doi.org/10.1016/j.dt.2024.07.011

model that can integrate the resource allocation and dispatch path planning, thus realizing the autonomous sortie scheduling on the flight deck.

1.2. Related works

Existing researches in deck scheduling field mainly focus on realize resource allocation functionality mentioned above. For example, some studies focus on the supporting equipment/staff/spot allocation in specific supporting task, e.g., pre-flight deck operation [1–3], aircraft maintenance [4–6], weapon transportation [7]. In contrast to other deck support scheduling scenarios, a distinct feature of sortie scheduling is that it involves frequent dispatch of aircraft. Once neglecting collision-avoidance in the implemented scheduling, it becomes challenging to ensure safety and easily results in prolonged time blocks, even with massive manual adjustments. There are some studies [8–10] dedicated for multiple dispatch entities collision free trajectory/path planning/tracking. These methods are effective for simultaneous dispatch of a few aircraft (i.e., when the number of aircraft is less than the number of catapults). However, as more aircraft are involved in the sortie wave, it is necessary to further consider the timing sequence of aircrafts and catapult allocation. Liu et al. [11] use the trajectory library and the waiting strategy to achieve catapult assignment and collision-avoidance simultaneously. Similarly, the scheduling for mixed fleet (i.e., including both fixed-wing aircraft and helicopters) sortie is studied by Liu et al. [12] by using the path library. It is noticed that only the taxiing dispatch mode rather than the towing dispatch mode is considered in existing research on sortie scheduling. However, as mentioned previously, equipping unmanned carrier aircraft onboard is an overwhelming trend [13]. For these unmanned aircraft, it generally relies on towing tractor to facilitate the dispatch at the current stage. Hence, sortie scheduling involving tractors still remains wide research space.

With the development of warehousing scheduling, similar studies that consider both cargo transportation and robot coordination motion can be found. Liu et al. [14] use a prediction algorithm first to generate sector-level robot path and post tasks for each robot. And then applied the local cooperative A* algorithm within each sector to generate conflict-free path. Lukas et al. [15] focus on the scheduling of automated guided vehicles in very narrow aisles, achieving vehicle allocation and coordination by prohibit simultaneous entry of two vehicles in aisle. Moreover, other researches [16–22] in this field focus much on the conflict resolution on the predetermined spot (e.g., pallet, storage position). However, as for the flight deck, the spots are farther apart and it is not possible to consider only the coordination on spots. The trajectory between the spots is actually the part that is of greater concern.

Considering that aircraft carriers can be viewed as maritime mobile airports, research in the field of airport ground scheduling can enlighten deck scheduling to some extent. Soltani et al. [23] construct a mixed integer linear programming model that incorporates conflict avoidance at taxiways and at each intersection. It realizes a rational allocation of tow-trucks to towing aircraft, pick-up time, drop-off time and the set of taxiways to complete taxiing operations. Oosterom et al. [24] plan the towing task of the electric towing vehicles first and further optimize the battery recharging route based on the scheduled tasks. And conflicts between aircrafts are avoided by imposing a minimum separation distance between any pair of aircraft on taxi road. Furthermore, other researches [25–29] in this field mostly focus on collision-avoidances on directed/undirected graph. Due to the limited space, there is no structured route dedicated for aircraft dispatching. For this reason, it is difficult to directly apply the aforementioned methods to implement aircraft sortie scheduling.

When tractors are involved, the mathematical modeling of the scheduling problem becomes the primary factor that influences the sortie efficiency. The tractors are seen as a kind of supporting resource which occupies certain space, the resource allocation and collision-avoidance must be considered simultaneously, making the resulted problem hard to be solved. If we decompose the sortie process into several independent procedures meanwhile virtualize related supporting resource as processing machines, the sortie process falls into a flow-shop scheduling problem (FSP). The procedures are further linked so that a hybrid FSP (HFSP) can be constructed. Actually, HFSPs constructed for sortie scheduling under the taxiing mode is already inherently complex. For example, during the takeoff phase, only one aircraft can be launched at a time, forming a block FSP [30,31]; there is a buffering process from the preparing spot to the catapult, which is a limited buffer FSP [32]; after an aircraft is launched, the recovery processes of catapult follow immediately, forming a no-wait FSP [33]. When the tractors are involved, it further poses challenges in the following two aspects. On the one hand, it results in more coupled operation constraints on each procedure, which makes it much complicated to determine the earliest time for an entity to enter its next procedure. Notably, previous studies [11,12] treat the dispatch procedure as a full occupation of the processing machine. However, in reality, spot resources (e.g., preparation spots) occupancy only occurs when the entity approaches its vicinity. On the other hand, there is the requirement for coordination among heterogeneous dispatch entities. That is, within the constructed HFSP, we have to guarantee collision-avoidance between trajectories of machines (e.g., tractors) and jobs (e.g., aircrafts).

1.3. Contributions

To the best of our knowledge, this is the first paper reports sortie scheduling for carrier aircraft fleet under towing mode. The primary contributions of this paper are as follows:

(1) Addressing the aircraft sortie scheduling planning under the towing mode, a rational HFSP model is established by extracting key processing procedures and analyzing the constraints therein.
(2) A genetic algorithm with specially designed operators and incorporating a chaotic initialization is proposed to realize the robust solution of the sortie scheduling problem.
(3) A spatiotemporal coordination method for heterogeneous dispatch entities was proposed and integrated into the genetic algorithm, ensuring the reliability of the generated scheme in engineering applications.

1.4. Organizations

The rest of the paper is organized as follows. In Section 2, we extracted the key processing procedures of the aircraft sortie scheduling problem and analyzes the constraints to formulate an HFSP. Section 3 introduces the spatiotemporal coordination method applied to heterogeneous dispatch entities. In Section 4, a genetic algorithm based on a chaos-initialized is specially designed. And the proposed method is validated in Section 5 by various scenarios. Finally, conclusions are summarized in Section 6.

2. Modeling of sortie scheduling problem

To covert the aircraft sortie scheduling problem into the HFSP, we extract eight key processing procedures according to the practical sortie process. We then define the related decision variables and analyze the relevant constraints, finally construct a mixed-integer planning model.

2.1. Procedures extraction

To formulate the HFSP, each aircraft is viewed as a processing job, supporting resource such as tractors, preparing spots and catapults are virtualized as processing machines. Based on the type of machines employed and the processing requirements, as shown in Fig. 1, the sortie process is extracted into eight key procedures.

Procedure 1: Dispatch tractor to aircraft parking spot (APS). The tractor is dispatched from the initial tractor parking spot (ITPS) or tractor parking spot attached the catapult (CTPS) to the designated APS. The processing machine is the tractor.
Procedure 2: Connect tractor and aircraft. Following dispatching the tractor to the APS, a connection should be established between the tractor and the aircraft. The processing machine is the tractor.
Procedure 3: Dispatch aircraft to preparing spot. The aircraft is dispatched to the preparing spot by tractor. This procedure utilizes both the tractor and the preparing spot. To facilitate invoking the dispatch trajectory, the preparing spot is designated as the processing machine.
Procedure 4: Separate tractor and aircraft. After dispatch to the preparing spot, aircraft should be disconnected from the tractor. Subsequently, the tractor moves to the corresponding CTPS, preparing for next job. The processing machine is the preparing spot.
Procedure 5: Taxi to catapult. The aircraft taxies from the preparing spot to the corresponding catapult under its own power. The processing machine is the catapult.
Procedure 6: Pre-launch inspection. Before launch, the aircraft undergoes an inspection. Concurrently, preparations for catapult are performed, such as connecting the catapult arresting cable and raising the jet blast deflector (JBD). The processing machine is the catapult.
Procedure 7: Launching. After verification of normal status for the aircraft and the catapult, the aircraft is launching. Although the machine used for this procedure should be the catapult. But, the engine wake of the launched aircraft affects flow field of the flight deck. Therefore, only one is allowed to be launched from the deck at a time. For this reason, a virtual launching machine is designated for launch all aircraft.
Procedure 8: Catapult recovery. After the aircraft is launched, the corresponding JBD undergoes cooling and is retracted to its initial state. Simultaneously, the catapult is recharged, ensuring that subsequent aircraft can be launched. The processing machine is the catapult.

2.2. Constraints description

After extracting the procedures, the constraints between the procedures need to be analyzed to build a mathematical optimization model. We employ the positive integers i, j, and l to represent the job number, the procedure number and the machine number, respectively. For a sortie task involving n aircrafts and m machines, we use $O_{i,j}$ represents the jth procedure for ith job. Then, the decision variables could be defined as illustrated in Table 1.

Additionally, we denote the set composed by all jth procedures as E_j, and the processing time of $O_{i,j}$ on machine l as $p_{i,j,l}$. Then, the following constraints is derived

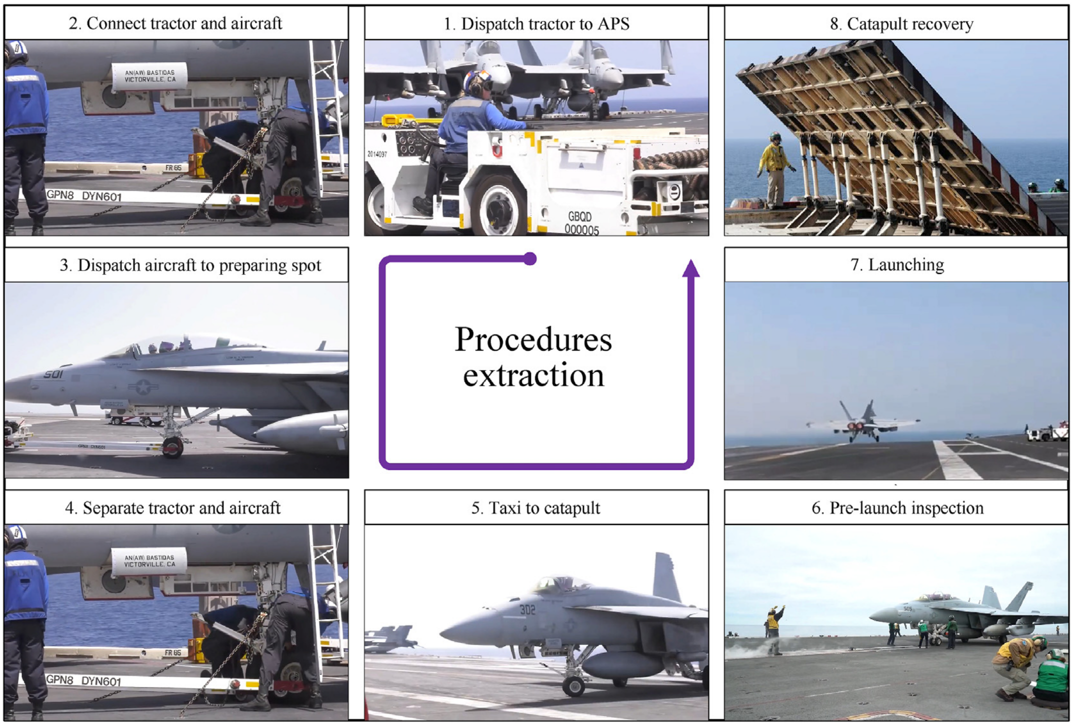

2. Connect tractor and aircraft
1. Dispatch tractor to APS
8. Catapult recovery
3. Dispatch aircraft to preparing spot
Procedures extraction
7. Launching
4. Separate tractor and aircraft
5. Taxi to catapult
6. Pre-launch inspection

Fig. 1. An illustration of extraction of sortie processes.

Table 1
Decision variables of the scheduling problem.

Variables	Description
$S_{i,j,l}$	Start time of $O_{i,j}$ on machine l
$C_{i,j,l}$	Completion time of $O_{i,j}$ on machine l
$Y_{i,j,l}$	Whether the $O_{i,j}$ processing on machine l, if so $Y_{i,j,l} = 1$; otherwise $Y_{i,j,l} = 0$
$X_{i,h,j,l}$	Whether $O_{i,j}$ processing on machine l after $O_{h,j}$ completed, if so $X_{i,j,k,l} = 1$; otherwise $X_{i,j,k,l} = 0$

1) $\forall\ i\in[1,n], j\in[1,m]$, $O_{i,j}$ will be assigned to only one machine

$$\sum_{l=1}^{m_i} Y_{i,j,l} = 1 \tag{1}$$

2) $\forall\ i\in[1,n]$, $O_{i,1}$ and $O_{i,2}$ should processing on same machine

$$Y_{i,1,l} = Y_{i,2,l} \tag{2}$$

3) $\forall\ i\in[1,n]$, $O_{i,3}$, $O_{i,4}$, $O_{i,5}$, $O_{i,6}$ and $O_{i,8}$ should processing on same machine

$$Y_{i,3,l} = Y_{i,4,l} = Y_{i,5,l} = Y_{i,6,l} = Y_{i,8,l} \tag{3}$$

4) $\forall\ i\in[1,n]$, $O_{i,j}$ needs to be performed after $O_{i,j-1}$

$$S_{i,j,l} \geq C_{i,j-1,l} \tag{4}$$

5) $\forall\ i\in[1,n] j\in[1,m]$, The completion time of $O_{i,j}$ should be greater than the sum of its start time and the processing time

$$C_{i,j,l} > S_{i,j,l} + p_{i,j,l} - BM\cdot\left(Y_{i,j,l} - 1\right) \tag{5}$$

where BM is a is large positive number.

6) $\forall\ i,h\in[1,n]$, once tractor l has been selected for $O_{i,1}$, the tractor cannot be used for the $O_{h,1}$ until the completion of $O_{i,4}$.

$$S_{i,1,l} \geq C_{h,4,l} + BM\cdot(X_{i,1,h,l} - 1) \tag{6}$$

7) Up to one aircraft can be parked in each preparing spot. A safety time T_{safe} is implemented to ensure that, following the departure time T_{safe} of the first-entering aircraft, subsequent aircraft entering the spot will not collide with the preceding one. i.e., $\forall\ i, h\in[1,n]$, $X_{i,h,3,l} = 1$, $O_{h,3}$ should not to be completed on l before $O_{i,5}$ has completed T_{safe} time

$$C_{i,3,l} \geq S_{h,5,l} + T_{\text{safe}} + BM\cdot(X_{i,3,h,l} - 1) \tag{7}$$

8) $\forall\ i,h\in[1,n]$, if a catapult l is selected for $O_{i,5}$, the catapult will be required to serve for $O_{h,5}$ after the completion of the $O_{i,8}$.

$$S_{i,5,l} \geq C_{h,8,l} + BM\cdot(X_{i,5,h,l} - 1) \tag{8}$$

9) Subsequent aircraft must wait for the flow field to stabilize before launching. The flow field stabilization time is denoted as T_{turb}. i.e., $\forall\ i, h\in[1, n]$, $X_{i,h,7,l} = 1$, $O_{i,7}$ should not launch on l before $O_{h,7}$ has completed T_{turb} time. Hence, the following constraint is obtained.

$$S_{i,7,l} \geq C_{h,7,l} + T_{\text{turb}} + \text{BM}\cdot(X_{i,7,h,l} - 1) \tag{9}$$

10) The dispatch trajectory for $O_{i,j}$ is denoted as $Tr_{i,j}$, and for $\forall\ O_{i,j}$, $O_{h,k}\in(E_1\cup E_3)$, $i\neq h$, to achieve collision-avoidance of all dispatch entities, it is necessary that at any moment t satisfies

$$\min\left\|Tr_{i,j}(t) - Tr_{h,k}(t)\right\| \geq D_{\text{allowed}} \tag{10}$$

where $D_{\text{allowed}} = D_{\text{T}} + D_{\text{TS}} + D_{\text{safe}}$, D_{T} and D_{TS} are the external radius of the tractor and traction system respectively; D_{safe} is the margin reserved to ensure the safety of dispatch.

Remark 1. To facilitate the establishment of constraint on preparing spot occupancy in HFSP, Liu et al. [10] assume that Procedure 3 (i.e., dispatch aircraft to preparing spot) totally occupies the corresponding preparing spot. As a result, the constraint on preparing spot occupancy is set as Eq. (11). However, the preparing spot is only occupied when the aircraft is dispatched to its vicinity, which indicates that $S_{i,3,l}$ can be smaller than $C_{h,3,l}$ while ensuring the same spot is occupied by only one dispatch entity. Hence, we introduce constraint (7) to improve the scheduling efficiency.

$$\mathrm{S}_{\mathrm{i,3,l}} \geq \mathrm{C}_{\mathrm{h,3,l}} + BM\cdot\left(X_{\mathrm{i,j,h,l}} - 1\right) \tag{11}$$

Remark 2. Due to multiple types of dispatch entities are considered, trajectory of $O_{i,j}$ cannot represent in a unified form based on the trajectory libraries mentioned later. To facilitate the description of the collision-avoidance constraint, we denote the dispatch trajectory of $O_{i,j}$ by $Tr_{i,j}$. The relationship between $Tr_{i,j}$ and the basic trajectory libraries (BTLs) will be stated in Section 3.

2.3. Formulating the HFSP

The sortie, recovery, and support of aircraft are continuously carried out in rotation on the deck. The sortie process should be completed as quickly as possible to allow sufficient time for other operations, ensuring the sustained combat capability of aircraft carrier. On the other hand, the rapid sortie of aircraft is beneficial for quickly forming aerial formations, reducing the airborne loitering time of other aircraft, and enhancing combat efficiency. In ensuring the satisfaction of all constraints, the key performance metric for scheduling is the makespan. Hence, the objective function is set as the $\min(1/\max(C_{i,8,l}))$. Then, the following mixed-integer programming model of HFSP can be formulated.

$$\begin{array}{l}\min\ (\max(C_{i,8,l})) \\ \text{s.t. Eq. (1)} - \text{Eq.(10)}\end{array} \tag{12}$$

During the sortie process, each aircraft follows the same steps to complete the launch. This evokes associations with the FSP. Consequently, we treat each aircraft as a job awaiting processing, and we divide the entire sortie process into eight processing procedures. Different procedures utilize tow tractors, preparing spots, or catapults as processing machines, and the same procedure for different job can be processed in parallel by machines, thereby forming an HSFP. Constraints such as processing sequence, machine utilization sequence, and collision-avoidance are respectively described by Eqs. (1)–(10). By minimizing the makespan as the optimization objective and integrating process constraints, we successfully construct a combinatorial optimization model. For this complex combinatorial optimization problem, we detail solution method in subsequent sections.

3. Spatiotemporal coordination based on delayed strategy

In this section, the acquisition of the BTLs is presented, and a delayed strategy is implemented for collision-avoidance among heterogeneous dispatch entities.

3.1. Generating the BTLs

In extracted procedures, Procedures 1, 3, 5, and 7 encompass the motion trajectories of dispatch entities. Procedures 5 and 7 involve taxiing trajectories within no-entry zone around the catapult, which are collision-free fixed trajectories. Hence, collision-avoidance for Procedures 5 and 7 is not considered. Once the layout of the deck is determined, APSs, ITPSs, CTPSs, preparing spots are fixed. Therefore, the BTLs for dispatch entities can be pre-generated. This allows direct invocation of trajectories, which improves the efficiency of scheduling planning algorithm.

During generation of BTLs, collision-avoidance with static obstacles is considered. Given the potential presence of stationary dispatch entities, obstacles are considered at relevant spots. Furthermore, no-entry zones are treated as obstacles during trajectory planning process.

For Procedure 1, a BTL is constructed to dispatch the tractor from the ITPS to the APS. This BTL comprises by $m_1 \times n$ trajectories and is denoted as $\{Tra^{A}_{s,t}\}$, where $s = 1, 2, \cdots m_1$, $t = 1, 2, \cdots, n$, m_1 denote the number of tractor. After an aircraft has been dispatched to the preparing spot, if the subsequent aircraft needs to be dispatched using the same tractor, the tractor should depart from CTPS. Hence, the BTL includes the trajectories from CTPSs to the APSs should be constructed for tractor. This library contains $m_3 \times n$ trajectories. It is denoted as $\{Tra^{B}_{s,t}\}$, where $s = 1, 2, \cdots, m_3$, $t = 1, 2, \cdots, n$, m_3 denotes the number of catapult. For Procedure 3, it is required to generate a BTL of traction system comprising $n \times m_3$ trajectories from each APS to all preparing spots. It is denoted as $\{Tra^{C}_{s,t}\}$, where $s = 1, 2, \cdots, n$, $t = 1, 2, \cdots, m_3$.

Remark 3. To facilitate the recognition of trajectories in BTLs, s and t in the subscript are used to denote the start and the terminal position, respectively. In Procedure 3, the trajectory $Tr_{i,3} = Traj^{C}_{i,l}$. When the tractor is processing the job for the first time, i.e., positioned at ITPS, trajectory $Tr_{i,j} = Traj^{A}_{l,i}$. However, after dispatching other aircraft, the tractor needs to depart from CTPS for next aircraft. At this time, the relationship between $Tr_{i,j}$ and trajectory in BTL can expressed as: if $X_{i,1,h,l} = 1$ and $Y_{h,3,k} = 1$, then $Tr_{i,1} = Traj^{B}_{h,k}$.

3.2. Spatiotemporal coordination based on a delayed strategy

In the sortie process, collision-avoidance between heterogeneous entities is essential to ensure the feasibility of the planned scheduling in applications. This paper employs a delayed strategy to achieve spatiotemporal coordination, which can be divided into the following two steps:

Step 1: Time Interception

For trajectory $Tr_{i,j}$ scheduled within the time $[S_{i,j,l}, C_{i,j,l}]$ and any trajectory $Tr_{p,q}$ scheduled within $[S_{p,q,g}, C_{p,q,g}]$, the following six relationships exist between processing time:

a : $S_{i,j,l} \geq C_{p,q,g}$.
b : $S_{p,q,g} \geq C_{i,j,l}$.
c : $C_{p,q,g} \geq S_{i,j,l} \geq S_{p,q,g}$, $C_{i,j,l} \geq C_{p,q,g}$.
d : $C_{p,q,g} \geq S_{i,j,l} \geq S_{p,q,g}$, $C_{i,j,l} \leq C_{p,q,g}$.
e : $S_{i,j,l} \leq S_{p,q,g}$, $S_{p,q,g} \leq C_{i,j,l} \leq C_{p,q,g}$.
f : $S_{i,j,l} \geq S_{p,q,g}$, $S_{p,q,g} \leq C_{i,j,l} \leq C_{p,q,g}$.

Fig. 2 depicts the diagram of these relationship. If the processing time of two trajectories satisfy conditions a or b, collisions are invariably avoided. However, if conditions c, d, e, or f are satisfied, one of the procedure need to be delayed. In these scenarios, we identify the overlapping time segments and introduce a delay to the trajectory scheduled last.

Step 2: Dispatch delay

All scheduled trajectories are re-interpolated at consistent time intervals of ΔT, where ΔT denotes the precision of collision-avoidance detection time. This means for trajectory $O_{i,j}$, it is re-interpolated based on time $[S_{i,j,l}, S_{i,j,l} + \Delta T, S_{i,j,l} + 2\Delta T, \cdots, C_{i,j,l}]$. Following this, all trajectory information within overlapping times is gathered. The trajectory scheduled last undergoes a comparison with others to ascertain if it meets the criteria of Eq. (10). If there is a potential collision interval in the overlapping scheduling time as the orange rectangular area of Fig. 3, we implement a delay ΔT in the start time of that trajectory, i.e., $S_{i,j,l} = S_{i,j,l} + \Delta T$. The constraint

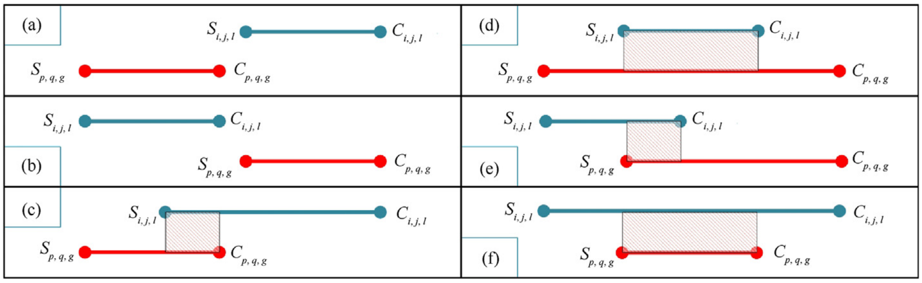


Fig. 2. Relationships between the processing time.

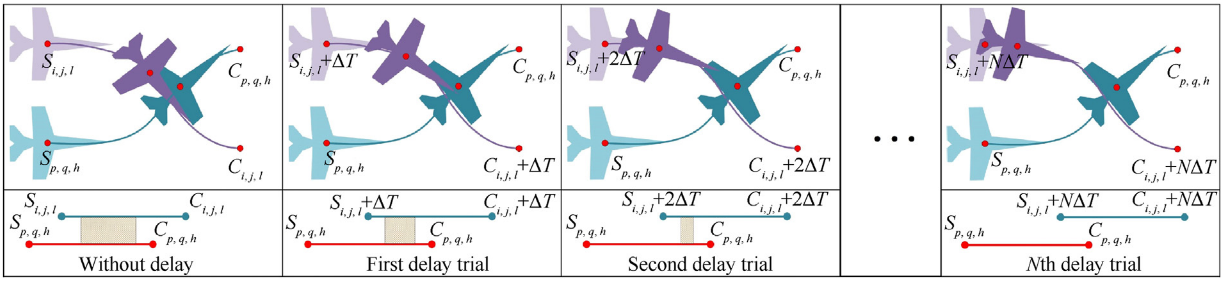
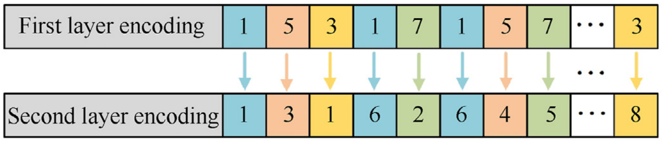


Fig. 3. An illustration of the delayed strategy.



First layer encoding	1	5	3	1	7	1	5	7	···	3
Second layer encoding	1	3	1	6	2	6	4	5	···	8

Fig. 4. An illustration of double-layer encoding method.

verification and time delay are repeated until Eq. (10) is fulfilled.

4. Chaos-initialized genetic algorithm

This section introduces a chaos-initialized genetic algorithm (CiGA) to address the constructed HFSP. It includes the encoding method, the generation of initial population, the decoding method, and the overall workflow.

4.1. Encoding method and initial population generation

A double-layer encoding method [34] as shown in Fig. 4 is applied in CiGA. In that, the first layer represents the job number, and the occurrences number of the job number indicates the procedure number. The second layer identifies the machine number assigned to complete the given procedure for the job. For example, in Fig. 4, the $O_{1,1}$ processing on machine 1; $O_{1,2}$ processing on machine 6, etc.

The constructed HFSP model presents several challenges due to the resultant uneven solution space distribution, particularly when integrating spatiotemporal coordination among heterogeneous dispatch entities. To foster diversity and promote a more even distribution in the solution space, we employ chaotic mapping for population initialization.

For the first layer encoding, consider an example scheduling scenario involving 2 jobs and 4 procedures per job. The initialization method is demonstrated in Fig. 5. The codes are arranged in the order of job numbers at first. And a chaotic sequence $\{y_1, y_2, \cdots, y_8\}$ is generated following the Improved Logistics Mapping [35]. This chaotic sequence is then sorted, and the position of codes are realigned according to the sequence order.

As for second layer encoding, the machine for Procedures 1 and 3 is randomly selected from the available machine set. Due to the presence of constraints (2) and (3), machines for other procedures are determined based on the specified constraints. We use *Pop* to represent the number of individuals in the population. Then, the encoding generation process is conducted *Pop* times to generate an initial population comprising *Pop* chromosomes.

4.2. Chromosome decoding

Constraints (1)–(9) are imposed through logic-based constraints, and the delayed strategy outlined in Section 4 is applied to satisfy constraint (10). Following this, a method, as illustrated in Algorithm 1, is devised for decoding chromosomes.

Algorithm 1. Chromosome Decoding

```
Function $Sche \leftarrow \textbf{Decoding}(Chro, Traj_{1,i,j}, Traj_{3,i,j})$
Initialization $T_{\text{list}}\ M_{\text{list}}\ Sche$
$Order \leftarrow 1$
$P_{\text{list}} \leftarrow \text{ones}(1,8)$
while $Gene \neq \varnothing$ do
    $[J_{ID}, P_{ID}, T] \leftarrow \textbf{AccessInfo}(Chro, Order, P_{\text{list}}, M_{\text{list}}, T_{\text{list}})$
    if $\textbf{CanBeExecuted}(J_{ID}, P_{ID}, M_{ID})$ then
        $[S_{J_{ID},P_{ID},M_{ID}}, C_{J_{ID},P_{ID},M_{ID}}] \leftarrow \textbf{Sequ}(Sche, J_{ID}, P_{ID}, M_{ID})$
        if $\textbf{IsInvoTraj}(P_{ID})$ then
            $T_d \leftarrow \textbf{Calc}(Sche, Tra^A_{s,t}, Tra^B_{s,t}, Tra^C_{s,t}, J_{ID}, P_{ID}, M_{ID})$
            $S_{J_{ID},P_{ID},M_{ID}} \leftarrow S_{J_{ID},P_{ID},M_{ID}} + T_d$
            $C_{J_{ID},P_{ID},M_{ID}} \leftarrow C_{J_{ID},P_{ID},M_{ID}} + T_d$
        end if
        $Sche((J_{ID}-1)+P_{ID}, 1) \leftarrow J_{ID}$
        $Sche((J_{ID}-1)+P_{ID}, 2) \leftarrow P_{ID}$
        $Sche((J_{ID}-1)+P_{ID}, 3) \leftarrow M_{ID}$
        $Sche((J_{ID}-1)+P_{ID}, 4) \leftarrow S_{J_{ID},P_{ID},M_{ID}}$
        $Sche((J_{ID}-1)+P_{ID}, 5) \leftarrow C_{J_{ID},P_{ID},M_{ID}}$
        $P_{\text{list}}(J_{ID}) \leftarrow P_{\text{list}}(J_{ID}) + 1$
        $Gene \leftarrow Gene \backslash Gene(Order)$
        $Order \leftarrow 1$
    else
        $Order \leftarrow Order + 1$
    end if
end while
return $Sche$
```

In Algorithm 1, the Job-Process-Machine list M_{list}, the Job-Process-Time list T_{list} and the scheduling *Sche* are initialized first. *Order* is used to record the current decoding position, and P_{list} is of length n used to record the procedure to be performed for each job. *Chro* is any chromosome, where contain information of both the job and the processing machine. *AcessInfo* function is to invoke the job number J_{ID}, procedure number P_{ID} and processing time T from P_{list}, M_{list}, and T_{list} according to *Chro* and *Order*. Function *CanBeExecuted* is to determine whether the current procedure can be executed, if so, then use the *Sequ* function to arrange the start and completion time of the current procedure based on logical constraint. Then determine whether the current procedure involves trajectory by *IsInvoTraj*, if so, then use *Calc* function to invoke the trajectory and calculate the delay processing time according to the delayed

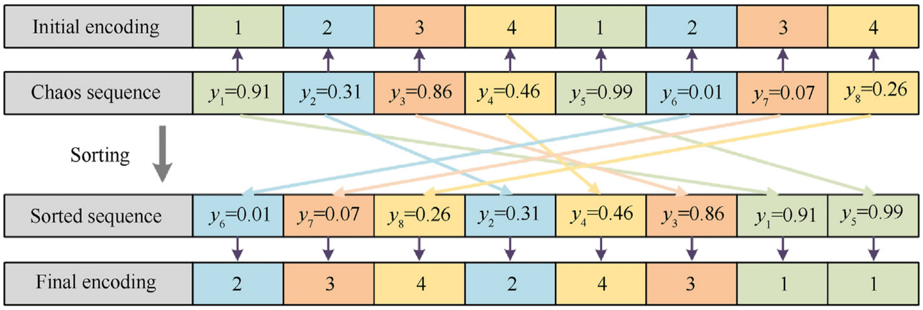

Initial encoding
1 2 3 4 1 2 3 4
Chaos sequence
y_1=0.91 y_2=0.31 y_3=0.86 y_4=0.46 y_5=0.99 y_6=0.01 y_7=0.07 y_8=0.26
Sorting
Sorted sequence
y_6=0.01 y_7=0.07 y_8=0.26 y_2=0.31 y_4=0.46 y_3=0.86 y_1=0.91 y_5=0.99
Final encoding
2 3 4 2 4 3 1 1

Fig. 5. An illustration of generation of the first layer encoding.

strategy presented in Section 4. Finally, the corresponding information is counted into the scheduling, the current code in chromosome is deleted, and the procedure number is updated. If the current code cannot be executed, algorithm update the *Order* and decoding the next code. Finally, return the scheduling *Sche*.

4.3. Population evolution

The population evolution process in this paper is outlined as follows:

Step 1: Selection

Setting the generation gap as P_G, then compute the fitness value for each chromosome. Select the top P_G proportion of chromosomes in the population to form a subpopulation.

Step 2: Crossover

Given the highly uneven distribution of feasible solution space in the constructed HFSP, the entire subpopulation undergoes crossover to enhance population diversity. Specifically, we randomly shuffle the chromosomes in the subpopulation first, and perform crossover between adjacent pairs. For first layer of encoding, we randomly select two crossover points and exchange all codes between these two points. Then, there may be missing or surplus codes in the encoding. To address this, we compare the original encoding with the post-crossover encoding, noting the missing and surplus encodings and their positions in the post-crossover encoding. Then, we replace the surplus encodings with the corresponding missing encodings in the post-crossover encoding. Finally, codes in the second layer are adjusted according to Eqs. (2) and (3).

Step 3: Mutation

To increase population diversity, two mutation operations are applied to the first layer encoding. Additionally, a mutation operation is performed on the second layer encoding.

① Fragment Mutation: The fragment mutation operation firstly randomly selecting two points on the first layer encoding. Then, all the codes between these selected points are mutated, and any missing or surplus code will be adjusted according to the method adopted by the crossover operation. Following the mutation, the second layer encoding is adjusted based on the original machine allocation. The probability of fragment mutation is set as P_F.

② Inversion Mutation: During inversion mutation operation, two points are randomly selected on the first layer encoding. The codes between these points are inverted. Following the mutation, the second layer encoding is adjusted based on the original machine allocation. The probability of inversion mutation is set as P_I.

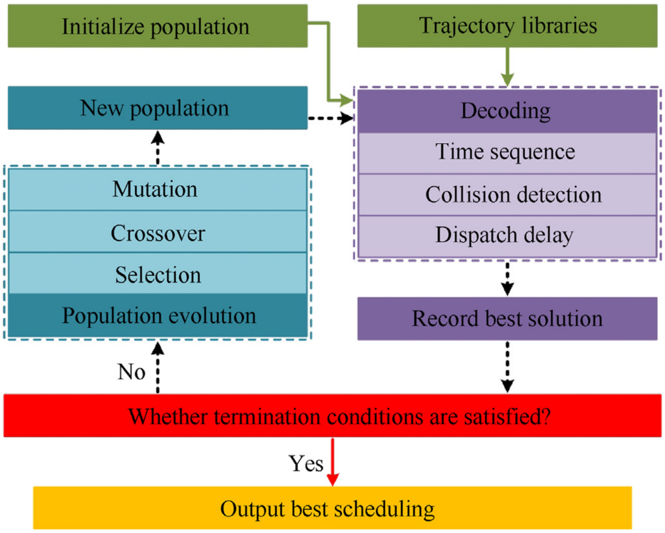


Fig. 6. Overall workflow.

③ Machine Mutation: As a procedure can be performed on different machines, the second layer encoding undergoes mutation. It is important to note that after determining the processing machines for Procedures 1 and 3, the machines assigned to subsequent procedures are determined. Therefore, the mutation is only applied to the codes corresponding to Procedures 1 and 3. After the mutation, the subsequent codes in the second layer are adjusted according to Eqs. (2) and (3). The probability of machine mutation is set as P_M.

Step 4: Population Update

Retain chromosomes in the original population with fitness in the top (1−P_G) proportion and insert them into the subpopulation to generate a new population.

4.4. Generating the scheduling

The overall workflow is shown in Fig. 6. The population iteratively follows the evolution method outlined in subsection 4.3. In each iteration, chromosomes are decoded according to subsection

Table 2
The parameters setting on algorithm.

Parameters	Value	Unit
T_{safe}	5	s
T_{turb}	10	s
ΔT	0.1	s
D_{T}	2	m
D_{TS}	9	m
D_{safe}	2	m
Pop	100	–
P_G	0.9	–
P_F	0.6	–
P_I	0.8	–
P_M	0.6	–
$I_{\max}$	100	–

4.2. The fitness value of each chromosome is recorded, and the optimal chromosome is updated until the iterations exceed $I_{\max}$. Upon completion, the optimal chromosome is re-decoded to generate the optimal scheduling.

5. Numerical simulation

In this section, the numerical simulation of sortie 8 and 20 aircrafts is conducted. The rationality of constructed model and the influence of different number of tractors are analyzed. Related simulation videos can be found at www.bilibili.com/video/BV14t421A7Tt/.

5.1. Parameter setting and BTLs

All simulations are implemented on MATLAB R2021a software on an Intel® Core™ i9-13900HX PC with a 2.20 GHz processor and 32 GB RAM. The default parameter settings of algorithm are detailed in Table 2.

As for BTLs, we have improved the SDC-based trajectory planning method [36] and applied it for both tractor and traction systems. The equipment parameters are referenced to the Ford Class aircraft carrier, F/A-18E/F aircraft and the A/S32A-31A aircraft tow tractor. The initial deck environment and spots setting are illustrated in Fig. 7. In Fig. 7, purple aircrafts J1−J20 are the spots of aircrafts need to be launched (i.e., APSs), red tractors T1−T5 are the ITPSs, orange aircrafts R1−R3 are three preparing spots, blue tractors P1−P3 are the CTPSs, and yellow rectangles C1−C4 are the catapults. The region enclosed by the red polygon is designated as a no-entry zone. Given that only catapults C1−C3 are used in this simulation, C4 is not in the no-entry zone.

Based on the deck layout, we generated BTLs $\{Tra^{A}_{s,t}\}$ as shown in Fig. 8, $\{Tra^{B}_{s,t}\}$ as shown in Fig. 9 and the $\{Tra^{C}_{s,t}\}$ as shown in Fig. 10.

5.2. Small-scale sortie

The planned sortie scheduling of aircrafts J1−J8 is shown in Fig. 11. In Fig. 11, the vertical axis is the aircraft number and the horizontal axis is the processing time. The numbers in bars of Fig. 11 indicate the machine used for corresponding procedure, where machines 1−3 correspond to tractors T1−T3, machines 4−6 correspond to preparing spots P1−P3, machines 7−9 correspond to catapults C1−C3 and machine 10 correspond to the virtual launching machine. Fig. 12 shows a Machine-Time Gantt chart, where the vertical axis represents the processing machine and the horizontal axis represents the processing time. The number in the bars of Fig. 12 indicate the procedure number. The CPU runtime of the simulation is 45.8 s and the sortie duration of generated scheduling is 378.3 s.

To demonstrate the advantages of adopted preparing spot usage order constraints in this paper, we construct a new optimization model (model B) by replacing Eq. (7) with Eq. (11) from the proposed model (model A). Then, the optimal chromosome generated by model A is re-decoded by applying the constraints from model B using the CiGA. The generated scheduling is shown in Fig. 13, where the sortie duration for 8 aircrafts is 436.3 s.

To further evaluate the performance of models A and B, a comparative experiment is conducted. In this experiment, we initially determined the number of aircrafts to be dispatched and randomly selected them from the J1−J20 to create a dispatch list. Next, the proposed CiGA is employed to generate scheduling based on two optimization models. For cases with number of aircraft varying from 6 to 12, we conduct experiment ten times for each case. Fig. 14 illustrates the average sortie duration of the generated scheduling, highlighting the efficiency of scheduling obtained by proposed model.

Furthermore, we conducted a comparison among particle swarm optimization (PSO), genetic algorithm (GA) and CiGA. In simulations, the algorithm parameters (including population size and maximum iteration) are kept the same the make the comparison fair. It is often difficult to predict how long it will take for the algorithm to achieve optimal solution. Instead, we are more concerned with which algorithm can produce a better scheduling within a certain timeframe. Therefore, we set the termination condition as a fixed maximum number of iterations to keep the planning time within an acceptable range. The performance of these algorithms are illustrated in Fig. 15. PSO demonstrates a rapid decrease in the average sortie duration of the population as the particles quickly converge towards the current optimal solution. However, this tendency also makes PSO susceptible to getting trapped in a local optimum. By contrast, GA has the ability to escape local optima to some extent due to the inclusion of a mutation operation. Nevertheless, GA exhibits slower convergence efficiency due to the uneven distribution of the initial population. To address this limitation, we introduce a chaotic initialization in GA, resulting in a more even distribution of individuals across the solution space. As a result, CiGA demonstrates higher convergence efficiency and better optimization capability compared to both PSO and GA.

5.3. Large-scale sortie

This simulation considers the sortie of 20 aircraft dispatched by 2−5 tractors, respectively. We have conducted 50 times planning for each scenario, and relevant simulations data are presented in Fig. 16 and Table 3 respectively.

Fig. 16 is a scatter graph that illustrates the relationship between sortie duration (x-axis) and CPU runtime (y-axis). The dots in purple, blue, green, and orange represent planning results using 2, 3, 4, and 5 tractors, respectively. The solid points indicate the average values for each scenario. For 50 experiments of a certain scenario, computational time consumes by spatiotemporal coordination varies much and thus leads to a significant variation in the total CPU runtime. However, the algorithm can restrain such a variation within approximately 60 s. As the number of tractors increases, the average CPU runtime also increases. This can be primarily attributed to the

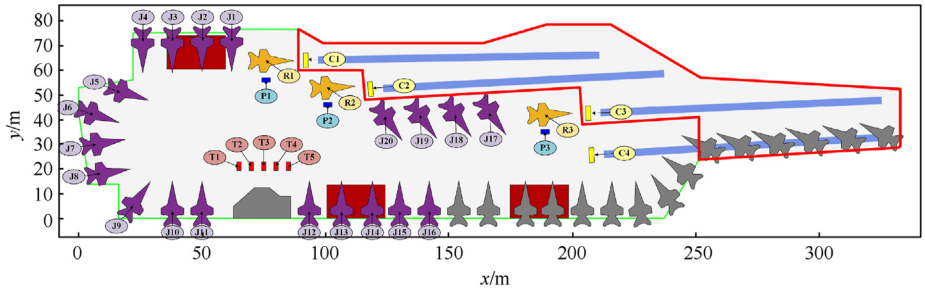


Fig. 7. Initial deck environment and spots setting.

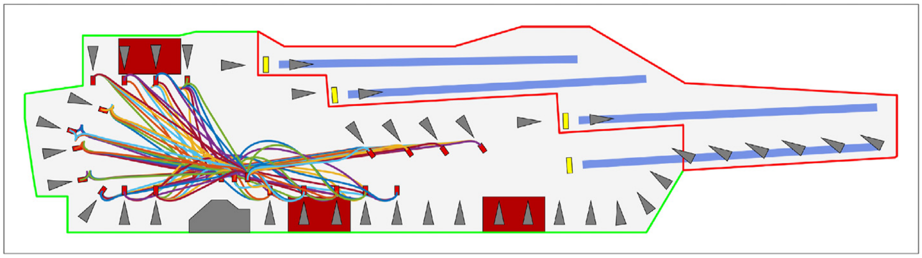


Fig. 8. The trajectory library $\{Tra^{A}_{s,t}\}$.

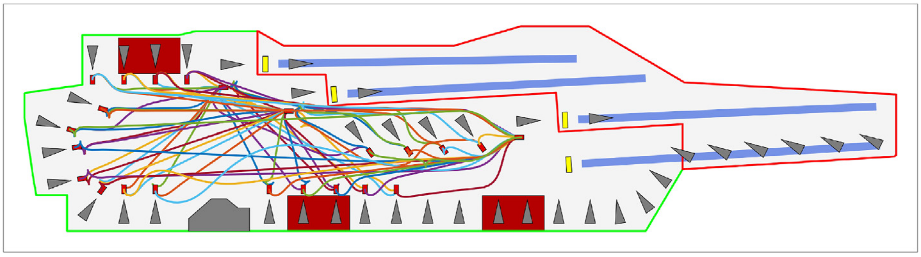


Fig. 9. The trajectory library $\{Tra^{B}_{s,t}\}$.

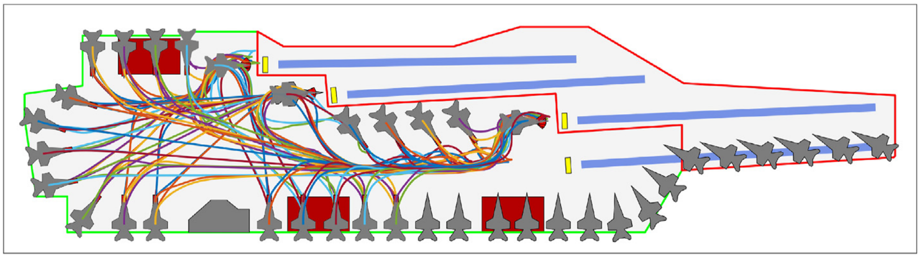


Fig. 10. The trajectory library $\{Tra^{C}_{s,t}\}$.

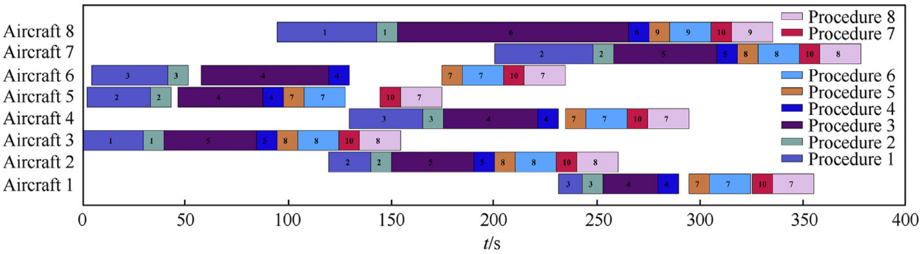


Fig. 11. Timing sequence of each aircraft.

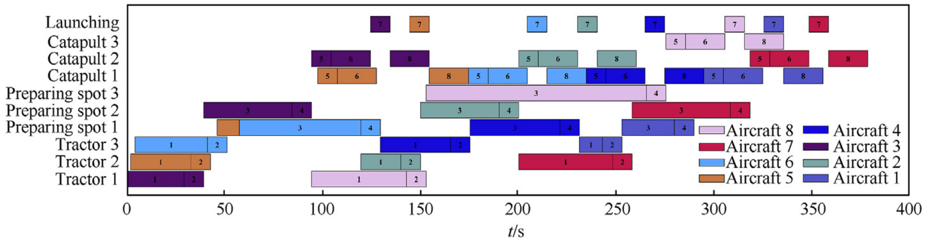


Fig. 12. Timing sequence of 8 aircrafts on each machine.

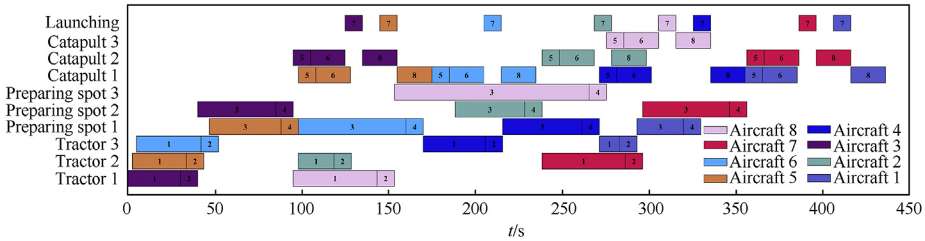


Fig. 13. Timing sequence with modeling method of previous studies.

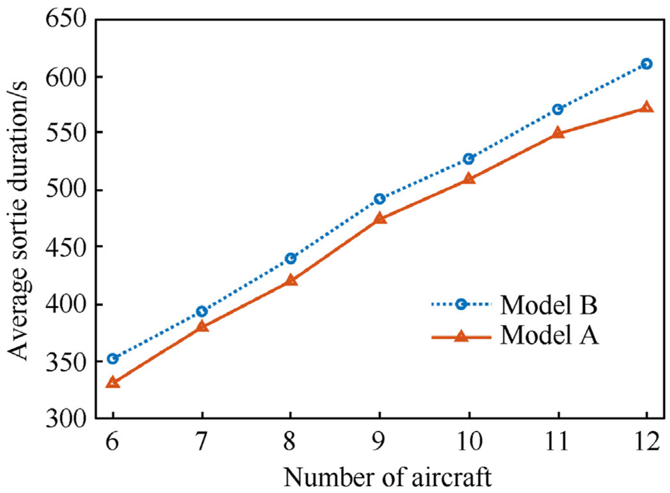


Fig. 14. Average sortie duration obtained by two models.

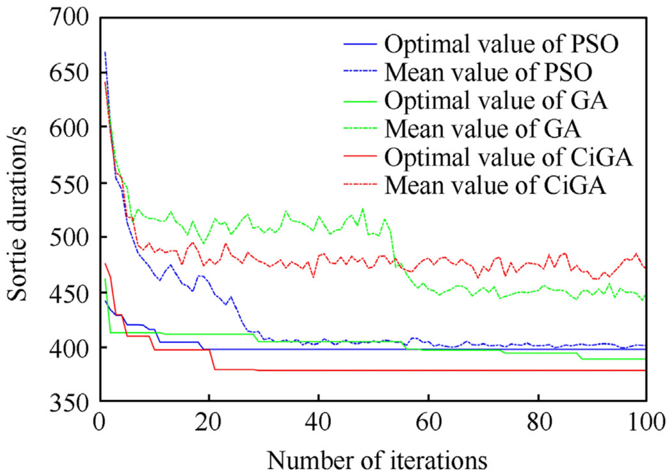


Fig. 15. The performance of three algorithms.

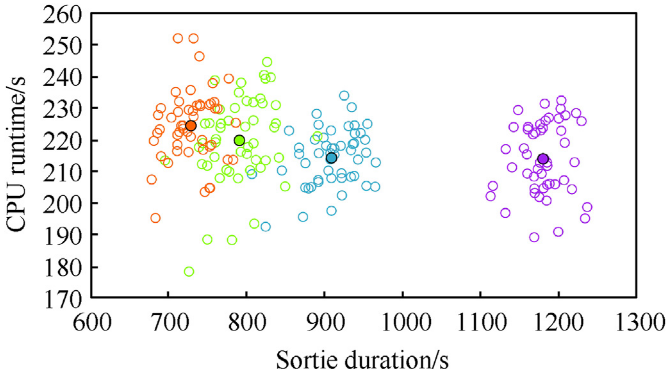


Fig. 16. Scatter of all simulations.

higher number of simultaneously dispatched trajectories, which necessitates handling spatiotemporal coordination for more entities within the limited deck space. Moreover, as the number of tractors increases, the scatter clusters gradually get closer, indicating a diminishing improvement in sortie efficiency. This trend is further confirmed by the average and minimum sortie durations presented in Table 3. This phenomenon is primarily caused by the limitation on the number of virtual launching machines. As the number of tractors increases, the tasks handled by the virtual launching machine approach its capacity limit, leaving less room for efficiency improvement. In addition, benefiting from the use of chaotic mapping for population initialization, the standard deviations of the sortie durations in Table 3 are less than 5% of mean values.

To further examine the scheduling characteristics with different numbers of tractors, we illustrate the minimum sortie duration scheduling for the four scenarios in a machine-time Gantt chart as shown in Fig. 17. As can be seen in Figs. 17 (a) and 17(b), with fewer tractor resources, the aircraft mainly launching on catapults C1 and C2, which are wider surrounding and closer to most involved aircrafts. As the number of tractors increases, tasks are more evenly distributed among catapults as there are sufficient tractors available for dispatching. Meanwhile, the number of simultaneously dispatched entities on the deck increases, which corresponds to the

Table 3
Data of planning with different tractors (unit: s).

Number of tractors	Best sortie duration	Average sortie duration	Standard deviation
2	1113.0	1180.2	28.2
3	806.9	908.9	34.9
4	695.5	790.8	37.1
5	678.3	728.6	28.1

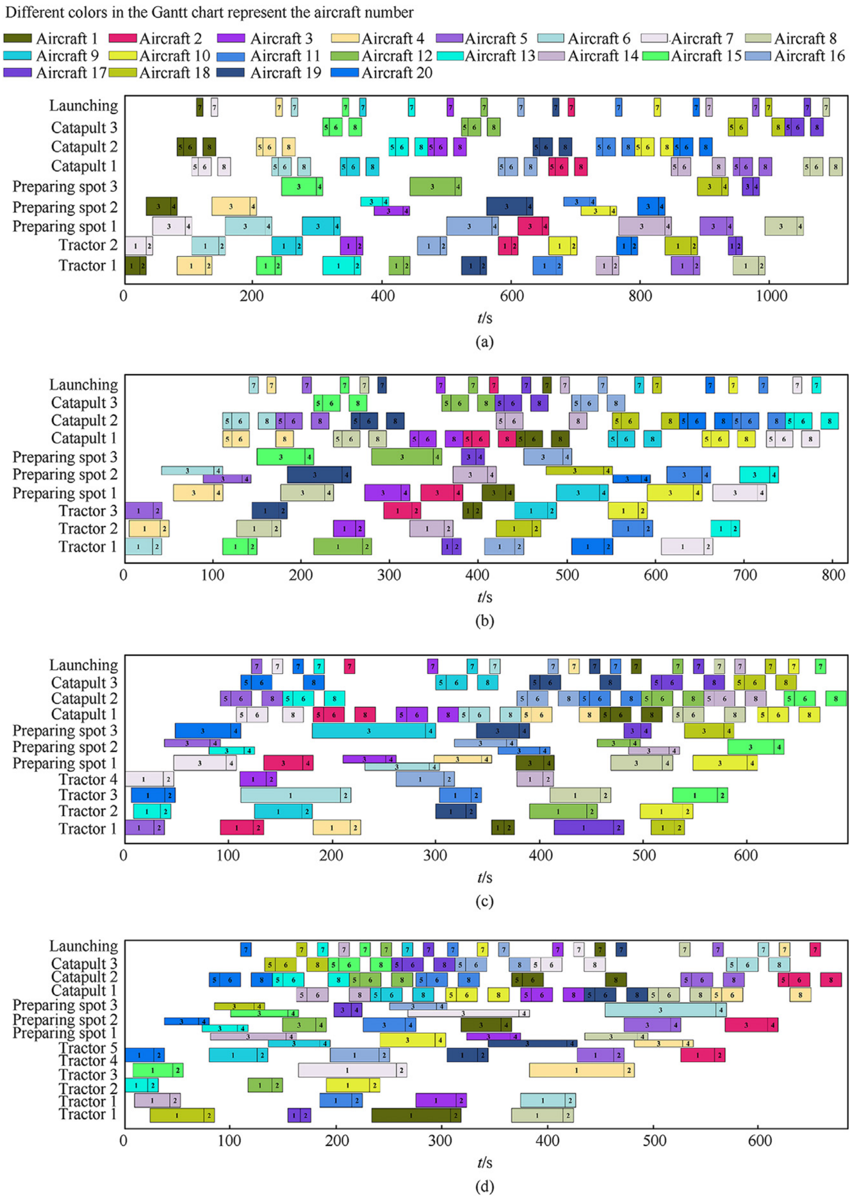


Fig. 17. Timing sequence of 20 aircrafts on each machine.

increase in CPU runtime in Fig. 16. In addition, the launching procedure on the virtual launching machine are lined up more densely, which corresponds to less scheduling efficiency improvement in Fig. 16.

6. Conclusions

In this paper, we address sortie scheduling for carrier aircraft fleet under towing mode by transforming it into an HFSP. By integrating a delayed strategy in CiGA, we reliably resolve the HFSP. Numerical simulations validate the robustness of the proposed algorithm and its efficiency under different resource configurations and fleet sizes. This approach is not limited to aircraft sortie scheduling and can easily be adapted for other onboard supporting scenarios, such as aircraft maintenance, ammunition delivery.

During simultaneous sortie and recovery, the landing time of aircraft exhibits an uncertainty. Dynamic planning becomes imperative, placing stringent demands on the efficiency of algorithm. Genetic programming (GP) [37,38], which is capable of pre-generating optimal scheduling rules, demonstrates superior performance in addressing this challenge. In future studies, our focus will be on implementing the proposed spatiotemporal coordination method in the simultaneous sortie and recovery scenario, and developing GP-based solution algorithms tailored for this specific context.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors are thankful to the financial support of the National Key Research and Development Plan (2021YFB3302501); the financial support of the National Natural Science Foundation of China (12102077).

References

[1] Su X, Cui R, Li C, Han W, Liu J. A heuristic solution framework for the resource-constrained multi-aircraft scheduling problem with transfer of resources and aircraft. Expert Syst Appl 2023;228:120430.
[2] Cui R, Han W, Su X, Liang H, Li Z. A dual population multi-operator genetic algorithm for flight deck operations scheduling problem. J Syst Eng Electron 2021;32(2):331–46.
[3] Liu Y, Han W, Su X, Cui R. Optimization of fixed aviation support resource station configuration for aircraft carrier based on aircraft dispatch mission scheduling. Chin J Aeronaut 2023;36(2):127–38.
[4] Li C, Zhang Y, Su X, Wang X. An improved optimization algorithm for aeronautical maintenance and repair task scheduling problem. Mathematics 2022;10(20):3777.
[5] Luan T, Sun M, Hu Z, Fu Q, Wang H. A novel T-S fuzzy robust control for part transportation of aircraft carrier considering transportation time and stochastic demand. Aero Sci Technol 2021;119:107096.
[6] Li C, Su X, Zhang Y, Han W, Guo F, Li X, Wang X. Integrated scheduling method for fleet wave sorties and maintenance of naval distributed platforms. Adv Eng Inf 2024;59:102340.
[7] Guo F, Han W, Su X, Liu Y, Cui R. A bi-population immune algorithm for weapon transportation support scheduling problem with pickup and delivery on aircraft carrier deck. Defence Techn 2023;22:119–34.
[8] Wang X, Liu J, Su X, Peng H, Zhao X, Lu C. A review on carrier aircraft dispatch path planning and control on deck. Chin J Aeronaut 2020;33(12):3039–57.
[9] Liu J, Dong X, Wang X, Cui K, Qie X, Jia J. A homogenization-planning-tracking method to solve cooperative autonomous motion control for heterogeneous carrier dispatch systems. Chin J Aeronaut 2022;35(9):293–305.
[10] Wang X, Peng H, Liu J, Dong X, Zhao X, Lu C. Optimal control based coordinated taxiing path planning and tracking for multiple carrier aircraft on flight deck. Defence Techn 2022;18(2):238–48.
[11] Liu J, Han W, Li J, Zhang Y, Su X. Integration design of sortie scheduling for carrier aircrafts based on hybrid flexible flowshop. IEEE Syst J 2020;14(1): 1503–11.
[12] Liu Z, Han W, Wu Y, Su X, Guo F. Automated sortie scheduling optimization for fixed-wing unmanned carrier aircraft and unmanned carrier helicopter mixed fleet based on offshore platform. Drones 2022;6(12):375.
[13] Wang X, Wang Y, Su X, Wang L, Lu C, Peng H, Liu J. Deep reinforcement learning-based air combat maneuver decision-making: literature review, implementation tutorial and future direction. Artif Intell Rev 2024;57:1.
[14] Liu Z, Wang H, Wei H, Liu M, Liu Y. Prediction, planning, and coordination of thousand-warehousing-robot networks with motion and communication uncertainties. IEEE Trans Autom Sci Eng 2021;18(4):1705–17.
[15] Lukas P, Simon E. Scheduling automated guided vehicles in very narrow aisle warehouses. Omega-Intern J Mana Sci 2021;99:102204.
[16] Zhen L, Wu J, Li H, Tan Z, Yuan Y. Scheduling multiple types of equipment in an automated warehouse. Ann Oper Res 2023;322(2):1119–41.
[17] Liu J, Liew S. Dynamic order-based scheduling algorithms for automated retrieval system in smart warehouses. IEEE Access 2021;9:158340–52.
[18] Mishra N, Kumar V, Kumar N, Kumar M, Tiwari MK. Addressing lot sizing and warehousing scheduling problem in manufacturing environment. Expert Syst Appl 2011;38(9):11751–62.
[19] Oliveira JA. Scheduling the truckload operations in automatic warehouses. Eur J Oper Res 2007;179(3):723–35.
[20] Yan B, Liu Y, Huang Y. Improved discrete imperialist competition algorithm for order scheduling of automated warehouses. Comput Ind Eng 2022;168: 108075.
[21] Li Z, Barenji AV, Jiang J, Zhong RY, Xu G. A mechanism for scheduling multi robot intelligent warehouse system face with dynamic demand. J Intell Manuf 2020;31(2):469–80.
[22] Sun B, Zhang X, Qiao H, Li G, Chen Y. Multi-type resources collaborative scheduling in automated warehouse with fuzzy processing time. J Intell Fuzzy Syst 2020;39:899–910.
[23] Soltani M, Ahmadi S, Akgunduz A, Bhuiyan N. An eco-friendly aircraft taxiing approach with collision and conflict avoidance. Transport Res C Emerg Technol 2020;121:102872.
[24] Oosterom SV, Mitici M, Hoekstra J. Dispatching a fleet of electric towing vehicles for aircraft taxiing with conflict avoidance and efficient battery charging. Transport Res C Emerg Technol 2023;147:103995.
[25] Chen J, Weiszer M, Locatelli G, Ravizza S, Atkin J, Stewart P. Toward a more realistic, cost-effective, and greener ground movement through active routing: a multiobjective shortest path approach. IEEE Trans Intell Transport Syst 2016;17(12):3524–40.
[26] Simaiakis I, Balakrishnan H. A queuing model of the airport departure process. Transport Sci 2015;50(1):94–109.
[27] Clare GL, Richards AG. Optimization of taxiway routing and runway scheduling. IEEE Trans Intell Transport Syst 2011;12(4):1000–13.
[28] Wang F, Bi J, Xie D, Zhao X. Quick taxi route assignment via real-time intersection state prediction with a spatial-temporal graph neural network. Transport Res C Emerg Technol 2024;158:104414.
[29] Brownlee AEI, Weiszer M, Chen J, Ravizza S, Woodward JR, Burke EK. A fuzzy approach to addressing uncertainty in airport ground movement optimization. Transport Res C Emerg Technol 2018;92:150–75.
[30] Meng L, Zhang C, Shao X, Zhang B, Ren Y, Lin W. More MILP models for hybrid flow shop scheduling problem and its extended problems. Int J Prod Res 2020;58(13):3905–30.
[31] Riahi V, Khorramizadeh M, Newton MAH, Sattar A. Scatter search for mixed blocking flowshop scheduling. Expert Syst Appl 2017;79:20–32.
[32] Zhang C, Tan J, Peng K, Gao L, Shen W, Lian K. A discrete whale swarm algorithm for hybrid flow-shop scheduling problem with limited buffers. Robot Comput Integrated Manuf 2021;68:102081.
[33] Zhao F, Qin S, Zhang Y, Ma W, Zhang C, Song H. A hybrid biogeography-based optimization with variable neighborhood search mechanism for no-wait flow shop scheduling problem. Expert Syst Appl 2019;126:321–39.
[34] Long X, Zhang J, Zhou K, Jin T. Dynamic self-learning artificial bee colony optimization algorithm for flexible job-shop scheduling problem with job insertion. Processes 2022;10(3).
[35] Mao Q, Wang Y. Adaptive T-distribution seagull optimization algorithm combining improved logistics chaos and sine-cosine operator. J Chinese Comp Sys 2021;43(11):2271–7.
[36] Wang X, Li B, Su X, Wang L, Lu C, Wang C. Autonomous dispatch trajectory planning on flight deck: a search-resampling-optimization framework. Eng Appl Artif Intell 2023;119:105792.
[37] Braune R, Benda F, Doerner KF, Hartl RF. A genetic programming learning approach to generate dispatching rules for flexible shop scheduling problems. Int J Prod Econ 2022;243:108342.
[38] Xu M, Mei Y, Zhang F, Zhang M. Genetic programming for dynamic flexible job shop scheduling: evolution with single individuals and ensembles. IEEE Trans Evol Comput 2023. Early Access.