A **Multiactor** Conflict Resolution Algorithm for **Route** Sector Flight **Based** on Expandable Deep **Multiagent** Reinforcement Learning

**Abstract**

The continuous growth of air traffic demand will increase the flight flow within the en-route sector, leading to more frequent flight conflicts between aircraft, and continuous conflicts with more complex structures may occur under certain conditions, which will cause air traffic controllers to bear a higher workload. Aiming at resolving conflicts in air route sectors, this paper proposes a thought: clustering all pairwise conflicts in a certain sector according to temporal and spatial correlation, the pairwise conflicts in each category constitute a multi-actor conflict (MAC), and the processes of conflict resolution are independent of each other. Then, the multiagent deep deterministic policy gradient (MADDPG) algorithm, a multiagent reinforcement learning algorithm, is used to map the policy for MACs. To solve the problem that the number of conflicting aircraft changes in MACs, a parameter sharing mechanism, dynamically scalable recurrent neural network (RNN) structure and meta learning framework are used to reform the MADDPG algorithm, such that the conflict resolution model can train and resolve MACs with different numbers of conflicting aircraft. Second, considering the execution time of action, this paper also proposes a method of combining multiple submodels with different and fixed execution times, which greatly improves the success rate of MACs. Finally, the real flight plan and airspace environment elements (such as route, waypoint, and sector) as well as the self-developed simulation system platform, a large number of high-density MAC scenarios are constructed to train the model. The training curve and test results of numerical experiments show that the model has good performance in computing time and success rate.

*Keywords:* Air Route Sector, Multiagent Reinforcement Algorithm, Tactical Conflict Resolution, Dynamic Expansion Mechanism

# Introduction

In the route sector, air traffic controllers (ATCOs) are responsible for ensuring the safety separation between all aircraft and will issue instructions to command the aircraft to resolve the flight conflict appearing in their airspace. However, the continuous growth of air traffic demand will increase the flight flow within the route sector, leading to more frequent flight conflicts between aircraft, and continuous conflicts with more complex structures may occur under certain conditions, which will cause ATCOs to bear a higher workload. Developing a decision support tool (DST) for conflict detection and resolution (CDR) is an effective way to solve this problem [1]. When the route sector is busy, the CDR module in DSTs will automatically detect the potential conflicts in the airspace and display the conflict information and corresponding conflict resolution policy suggestions on the ATCO operation screen. This not only improves the efficiency of air traffic operation but also reduces the workload of ATCOs for conflict resolution [2]. NASA's Center TRACON Automation System (CTAS) and MITRE's User Request Evaluation Tool (URET) have been embedded into a ground air traffic control automation system as DST [3]. The new ATM system under development in Single European Sky ATM Research (SESAR) of Eurocontrol and Next Generation Air Transportation system (NextGen) of FAA also includes more advanced DSTs with different activity levels [4]. To ensure safe and efficient operation and reduce the workload of ATCOs in dense route sector airspace, this paper uses artificial intelligence (AI) methods to study DSTs.

Multiactor contact resolution (MACR) has been widely studied because it plays an important role in the conflict-free flight of aircraft, especially for high-density and complex airspace in the future. AI methods are applied to MACR, which can improve the computing efficiency and cause the resolution model to solve conflicts that are not met or standardized through its generalization ability. In addition, the resolution model based on AI methods can also learn the conflict resolution mode and scheme of ATCOs in the real sector. For example, to eliminate the impact of uncertain factors (such as clear sky turbulence), ATCOs will provide safer separation between aircraft through a resolution policy. This can make DSTs reliable enough to assist ATCOs in the normal operation of the sector and intelligent enough to give a decision like humans, which helps to improve ATCOs' acceptance of the DST proposal. However, the training mechanism and neural network structure limit the portability of the resolution model, and a model can only be used for a type of conflict resolution with a fixed number of conflict aircraft, which results in a higher cost and lower efficiency of training to reduce the performance of the resolution model. Therefore, this paper uses a multiagent deep reinforcement learning algorithm to train the resolution model, and the parameter sharing mechanism, dynamically scalable recurrent neural network (RNN) structure and meta learning framework are introduced to transform the algorithm, which supports the model to be trained by scenarios with different numbers of conflict aircraft.

## Related Work

In the related studies of MACR, various methods have been proposed [5][6], which can be divided into distributed methods and centralized methods according to the control mode. In distributed methods, the function of conflict resolution is decentralized to a single aircraft, and each aircraft calculates and decides its own resolution policy considering only adjacent aircraft [7][8][9][10], such as the Airborne Separation Assurance System (ASAS)[11], Traffic Alert and Collision Avoidance System (TCAS)[12]/Airborne Collision Avoidance System (ACAS)[13]. ASAS is a kind of airborne system that separates aircraft from other aircraft and provides flight information about surrounding traffic. ASAS reassigns the separation task to the aircraft pilot and helps the aircraft maintain self-separation. TCAS/ACAS is the last defence line for safety, and it operates independently of ground equipment and provides advice to the crew in case of possible collision with aircraft equipped with a secondary radar transponder [4]. In contrast, the former has a longer duration of manoeuvre and belongs to tactical conflict resolution, and the latter will be triggered to prevent collision between aircraft after the former fails, make mistakes or be affected by uncertain factors, i.e., loss of separation. However, the lack of global coordination for the surrounding traffic is a key shortcoming for the distributed methods. For manned aviation, the security of its policy has not yet been effectively verified [14], and it can be overcome by deep study on fully automated air traffic control systems. In the centralized methods, the resolution solution is calculated and formulated by a centralized role [15], and each conflict aircraft follows the corresponding path in the solution, which can eliminate the contradiction between the policies of aircraft and reduce the uncertainty, such that provide a global optimal solution for complex multiactor conflict [16].

Previous centralized MACR methods can be classified into precise algorithms and heuristic algorithms according to whether they can find the global optimal solution. Precise algorithms, such as mixed integer programming (MIP) [17][18][19][20], the stochastic optimal control method [21], and the dynamic programming method [22][23], are common to find the global optimal solution. However, due to the high computing time [24], it is usually used as the control group algorithm. Heuristic algorithms, such as the variable neighbourhood search algorithm (VNS) [25], ant colony optimization algorithm (ACO) [26][27], particle swarm optimization algorithm (PSO) [28][29], evolutionary algorithm (EA) [30][31][32], tabu search algorithm (TS) [33], and greedy algorithm [34], cannot guarantee the global optimality of the resolution policy but can obtain a feasible solution on the premise of shortening the solution time. However, due to a variety of possible manoeuvring combinations of multiple aircraft, the calculation amount of these centralized methods is still tremendous, especially as the number of aircraft increases greatly, which may not meet the real-time requirements of MACR [35]. Then, the above algorithms have low flexibility and do not have the ability of self-learning and generalization, which makes them perform well only in standardized multiactor conflict scenarios [36].

In recent years, deep reinforcement learning (DRL) algorithms have been applied to conflict resolution problems [37][38][39], and a multiagent system framework has been introduced, namely, multiagent deep reinforcement learning (MADRL), to expand to the MACR area. MADRL combines game theory and DRL [40], regards each conflict aircraft as an agent, and there is a cooperative relationship between agents. The trained MACR model can select the optimal action for each agent in a joint or partial state to maximize the overall reward. Most of the MACR methods based on MADRL are distributed methods and used for collision avoidance [41][42] or separation assurance [43][44]. Another part belongs to centralized methods to provide real-time suggestions for ATCOs or pilots [16][45][46]. The model of the above methods can give the resolution policy at the millisecond level, and the performance is verified to be feasible to a certain extent.

## Statements

This paper does not consider uncertainty factors such as weather, and further research on constraints such as conflict probability or risk will be put into future work. This paper mainly studies the tactical MACR in the route sector airspace, which is triggered before the Short-Term Conflict Alert (STCA) [47] and TCAS, and the potential conflict will be detected 5 minutes in advance. In this paper, the centralized method is used, that is, the CDR module of DSTs on the ground formulates the global resolution policy for all conflict aircraft, which are sent to conflict aircraft in the form of instruction through Controller Pilot Data Link Communications (CPDLC) or Data Communications (DATACOM) after ATCOs confirm, and the manoeuvres in the policy are discrete, such as climb 600 metres.

## Contributions

* Combined with the air traffic control operation rules of the route sector, three kinds of discrete actions are designed, including heading adjustment (dogleg manoeuvre), altitude adjustment and speed adjustment. Considering the importance of the execution time of actions, the method of assembling multiple fixed time submodels is proposed to improve the performance and practicability of the resolution model (Section 2.2).
* A multiagent deep reinforcement learning algorithm is used to train the resolution model, and the parameter sharing mechanism for the multiagent system, dynamic RNN network structure and meta learning framework are introduced to transform the algorithm. The former two can support the resolution model to be expanded dynamically to resolve multiactor conflicts with different numbers of conflict aircraft. The latter will protect or even improve the performance and stability of the model (Section 3);
* According to the definition of pairwise conflict and the operation of the route sector, the idea of resolving route sector conflict is proposed, and multiactor conflict is defined (Section 2.1). Using the real flight plan and airspace environment elements such as route, waypoint and sector, high-density multiactor conflict scenarios are constructed and chosen as train samples and test samples of the resolution model, and the training curve and performance test results are analysed (Section 4).

# Problem Formulation

## Problem Statement

The intervals between two aircraft include lateral, longitudinal and vertical intervals, and the former two are also called horizontal intervals. When all kinds of intervals do not meet the minimum safety separation criterion, a pairwise flight conflict will occur [48]. In the route sector, the value of the minimum safety separation depends on the density and the scope of the airspace. Therefore, this paper uses a uniform separation criterion, that is, the horizontal interval is 10 km, and the vertical interval is 300 m.

Although not clearly defined, it is no doubt that a multiactor (multiaircraft) conflict consists of several pairwise conflicts. It is worth noting that the term “multiactor” does not mean that there are multiple aircraft in the scenario but that the number of aircraft involved in the conflict is greater than or equal to 3. Therefore, this paper states the definition of multiactor conflict according to the temporal and spatial distribution of pairwise conflicts in the route sector and provides a solution for the overall resolution of these conflicts. In the future, with the increase in flight flow in sector airspace, the pairwise conflicts will increase, and the relationship between the pairwise conflicts will be closer, which results in mutual influence on their resolution process. Solving large-scale conflicts in a sector as a whole is also a complex and computational problem. Hence, some pairwise conflicts with certain connections can be merged into a multiactor conflict, and the process of solving each multiactor conflict is relatively independent. Then, an algorithm is used to output the solution policy for each multiactor conflict.



Figure 1. Schematic diagram of the time space relationship between pairwise conflicts.

The focus of this paper is not to explore the correlation between pairwise conflicts but to provide algorithm support for resolving multiactor conflict under this idea. Therefore, a simple method is selected to establish the correlation, that is, the conflict time is within a certain space-time range, and/or there is overlapping flight (for example, the call sign of conflict aircraft in pairwise conflict 1 is A and B, and the conflict aircraft of pairwise conflict 2 is B and C, then the overlapping flight between conflict 1 and 2 is B). As shown in Figure 1, there are five multiactor conflicts and three pairwise conflicts from  to and in the limited space range (sector airspace), in which  is set to 240 seconds and  is 30 seconds (conflict detection time step, that is, DST detects conflict once every 30 seconds).

## Markov Games Formulation

In this paper, the Markov game (MG) is chosen to model the process of conflict resolution. MG is a product of the combination of the Markov decision process (MDP) and game theory (GT). MDP is applied to the decision process modelling of a single agent with the Markov property [49]. In the multiagent decision process, the environment is complex and dynamic, so an MG is used to address instability [50]. MDP indicates that the state of multi-agent conforms to Markov property, that is, the next state is only related to the current state, but not the previous state; GT represents the relationship between agents: cooperation, competition and the mixture of the two.

Each aircraft in multiactor conflict is regarded as an agent, and the relationship between them is cooperative. The resolution model based on MG can be represented by a tuple , where  is the number of conflict aircraft.  is the joint state space (global observation) composed of partial observations of each aircraft.  is the joint action space of all aircraft, and  is the optional action space in the polcy  of aircraft .  is the state transition function that satisfies the Markov random property, and .  is the sharing (common) reward after all aircraft cooperatively execute their respective actions,  is the sum of the expected reward  from  to , and  represents the discount factor. The resolution model will collect the current state  of aircraft  and output the action  at . After executing , and according to , aircraft  moves to the next state , that is, .

### State Space

State space  is the set of possible states of all agents in the environment, which is used as the input data of the model network:

 (1)

 (2)

 (3)



Figure 2. The content of partial observation (state) of an aircraft agent.

where  refers to the local observation information of conflict aircraft  at , and  refers to the flight status of aircraft  of aircraft  (including aircraft ) within the airspace of 100 km, 100 km and 6,000 m with the position of aircraft  as the centre (as shown in Figure 2), which includes the elements of longitude, latitude, altitude, heading, horizontal speed, vertical speed (or climb and descent rate), aircraft type and the two planned waypoints behind the current flight segment. In this paper, the reason why the planned waypoints of aircraft are added to the state space is that the conflict resolution action based on rules is an integrated command. For example, when the aircraft climbs by 600 metres, it needs some time to reach the target altitude. Therefore, the intention information of the aircraft is to reduce the impact of the time consumption of executing action on the resolution effect so that the mapping relationship between the state and the action in the model is more accurate.

### Action Space

**Action type.** The actions used in the actual conflict resolution work will last for a period of time, and another action will not be executed during this period. To make resolution suggestions given by DSTs more acceptable to ATCOs, the action space  in this paper is set as discrete.

**Action execution object.** In this model, all conflict aircraft in a multiactor conflict are considered, and a global solution is given. Each conflict aircraft will execute action (it is worth noting that each type of action contains zero actions) so that all pairwise conflicts in this multiactor conflict will be resolved at the same time.

**Execution time.** The  in Figure 3 represents the earliest time of loss of separation in the multiactor conflict (that is,  in Figure 1). Because the advanced amount of conflict detection is 300 seconds, the first pairwise conflict in this multiactor conflict is detected at . The execution time of action is within the range of  to , and the duration between two execution times is set to 120 seconds. The resolution effect of different execution times is different. Therefore, this paper proposes a combination method of several submodels with different and fixed execution times. For example, submodel 1 has three groups of actions, and their execution times  are set to . The execution times of submodel 2 are , including three groups of actions (as shown in Figure 3), until the execution time of submodel 12 is  with two groups of actions. All submodels are trained with a large number of identical multiactor conflict scenarios. When testing with the test scenario set, all submodels are traversed, and the actions with higher quality are selected. This combination method can greatly improve the success rate of the resolution model (see Section 4.3 for verification results).

**Resolution effect**. The criterion for judging whether the multiactor conflict has been successfully resolved is that the distance between all conflict aircraft meets regulatory requirements within 10 minutes from  to , and there is no conflict with other aircraft.

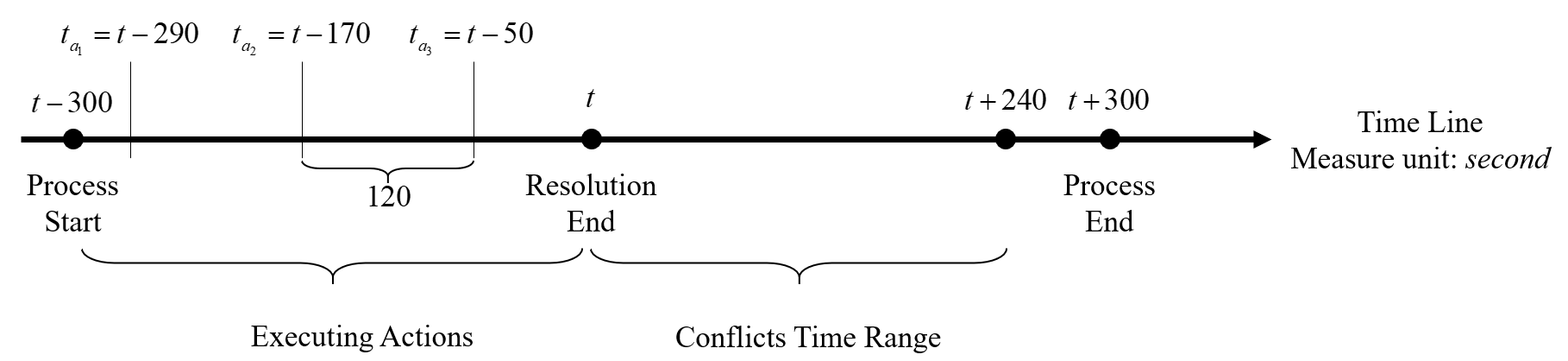


Figure 3. The representation of execution time of action by a time line:  are the execution time of submodel 2.

The actions that each conflict aircraft can choose include heading adjustment , altitude adjustment  and speed adjustment . The detail is as follows:

**Heading adjustment** **.** In the real route sector, ATCOs usually use dogleg manoeuvres to delay the aircraft arriving at a certain point to resolve intersection conflicts, but the deviation angle and distance of manoeuvres are set by ATCOs according to their work experience and conflict situation. To study the conflict resolution policy, this paper proposes a fixed dogleg manoeuvre. As shown in Figure 4, dogleg consists of two stages: “Out” phase and “Return” phase and three parameters: deviation angle , deviation distance  and return angle , in which  is set to 30, 45 and 60;  is 0 km, 11.1 km (6 nm), 14.8 km (8 nm) and 18.5 km (10 nm); To make it easier for the aircraft to cut into the original route,  is 30. In addition, BADA [51] is used to calculate the delay time of dogleg under different parameters, and the delay time is in the range of 20 to 60 seconds according to statistics.

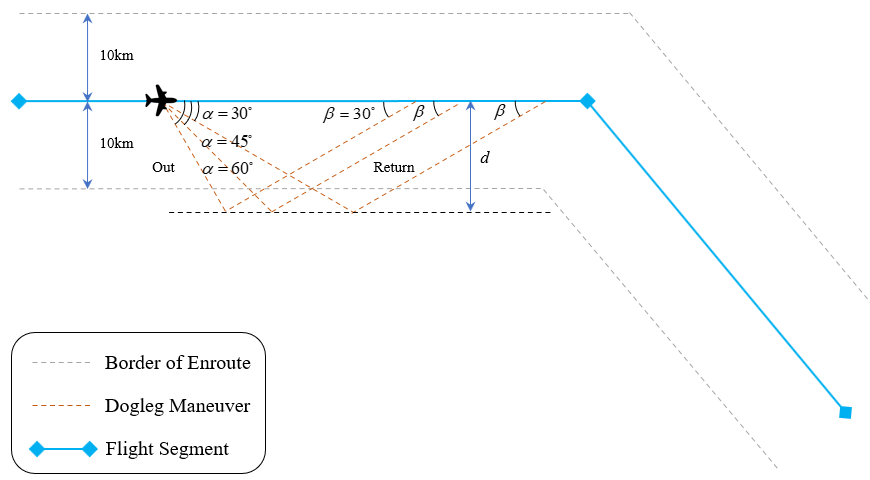


Figure 4. Dogleg manoeuvre diagram.

**Altitude adjustment** 。Combined with RVSM airspace and the allocation principles of flight level, the target altitude of an aircraft is defined as adding or subtracting several levels from its current flight level, and the extent of adjustment  is within . The judgement of the current level of an aircraft in different operation statuses is different: for the climbing aircraft, the current level is the first level that is higher than or equal to the current altitude; for the descending aircraft, its current level is the first one that is lower than or equal to the current altitude; and for cruising aircraft, its current level is the closest one. For example, as shown in Figure 5, the current level of the climbing aircraft is 8,900 m.  means that the target altitude of the aircraft is 4 levels higher than the current altitude.

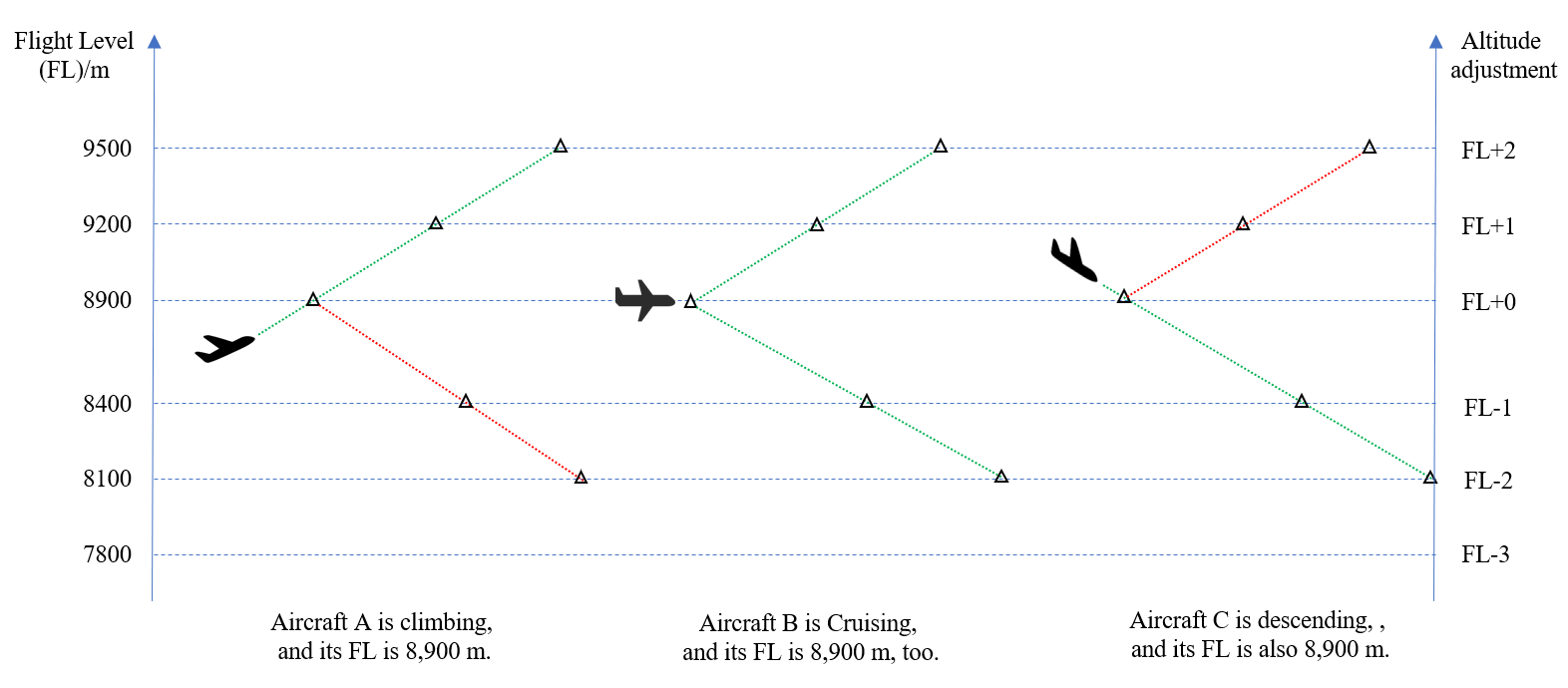


Figure 5. Available range of altitude adjustment: the red dotted line represents that executing this action is not ok when the aircraft is in this state.

**Speed adjustment** **.** Speed adjustment means that the aircraft accelerates, decelerates or maintains speed action. The speed adjustment range is , and the unit is knots. The target speed is equal to the current speed plus the speed adjustment.

It is worth noting that due to the limitation of the minimum sector altitude (MSA) and the flight performance envelope of the aircraft, there are upper and lower limits of the target value for altitude adjustment and speed adjustment, so it is necessary to judge whether the output action from the model is correct or not. If the target altitude is more than 12000 metres or less than 6000 metres, the action is invalid, and the corresponding aircraft does not execute any action. The same situation is that the target speed is out of range of flight performance at the current altitude. Second, because the climbing aircraft hardly issues descent instructions (such as the red line part of aircraft An in Figure 5), its altitude adjustment range must be greater than or equal to 0, and the altitude adjustment range of the descending aircraft must be less than or equal to 0 (such as the red line part of aircraft C in Figure 5), and these above actions are also judged to be invalid.

### State Transition through Simulation

The resolution model outputs actions that are assigned to the corresponding conflict aircraft for execution and the current state of aircraft transfer from to the next time step state. The state transfer function  of this paper is embedded in the trajectory prediction module of the Air Traffic Operation Simulation System (ATOSS), developed by our laboratory. According to the flight performance data in BADA, this module calculates and updates the aircraft state in a certain time step, including longitude and latitude, altitude, horizontal speed, heading, turning radius and vertical speed.

### Reward Function

In the multiactor conflict resolution problem, the reward function  determines the joint feedback that can be obtained after the conflict aircraft executes the action instruction output by the resolution model. The model updates its parameter values according to this feedback so that the resolution effect is better in the next cooperative selection action. The design rules of the reward function in this paper are made after consulting ATCOs on the job. The goal is divided into two levels: the resolution effect  and the quality of the action .

The value of  is obtained according to whether the action can successfully resolve the multiactor conflict (the criteria of the resolution effect are mentioned in Section 2.2.2), as shown in Formula 3:

 (3)

where  means that the multiactor conflict has not been resolved and no other conflicts have occurred, but the process of resolution is not over.

 is the further goal of , that is, to be free from conflict first and then to achieve better actions. The formula is as follows:

 (4)

Where  and  represent the difference between the original value and the target value of the aircraft  in the command;  indicates that the command is out of range or belongs to wrong instructions, and the reward is negative; When the type is , the reward is positive; When the type is , the larger the speed adjustment range  is, and the smaller the reward is; When the type is , the higher the altitude adjustment range  is, and the smaller the reward is, too.

The final reward  is the weighted sum of the two targets  and . Because all aircraft cooperate with each other, the Min function is used to extract the smallest value of  as the overall reward. The specific formula is as follows:

 (5)

where  is the weight of  in  and  is the weight of  in . As mentioned above, the premise of obtaining a high-quality solution is that the solution is feasible, so  needs to be greater than . Therefore, we set  as 0.8 and  as 0.2.

It is worth stating that due to the structural limitations of China's airspace and the use rules of airspace resources, it is necessary to coordinate with the owner (military or other) of the airspace when civil aircraft deviate from the route airspace, which will increase the workload of the ATCOs. Therefore, in the actual operation of the route sector, it is also more frequent to use altitude adjustment to resolve conflicts (this is also the reason why the altitude adjustment reward is set higher than the heading adjustment reward in Equation 5). Second, although the reward function is formulated with the help of the current certified and professional controllers, the purpose is to better study the influence of the reward function on the preference of policy output from the resolution model. However, due to the differences in the operation rules between the various route sectors, if it is used for multiactor conflict resolution in other sectors, the reward function needs to be reset and matched to other specific parameter settings according to the operation situation of this sector.

# Solution Approach

As mentioned above, the centralized MACR method is used in this paper. The conflict resolution module of DSTs in the ground collects the flight status information of all aircraft in the air and calculates and formulates the conflict resolution scheme. Considering the large amount of calculation of the centralized method [35], the MADDPG algorithm is selected as the solution algorithm of the resolution model based on MG, and the RNN structure based on the parameter sharing mechanism and meta learning framework are introduced to reform the algorithm. The former can support self-duplication of the model to achieve the dynamic expansion function. The latter uses the generalization ability between multiple tasks to solve the problem of instability and low performance caused by the copying model. Finally, according to the above multi-machine conflict definition, the conflict scene samples and superparameters used in model training are designed. Finally, the method to obtain conflict scenarios and hyperparameters of model training are given.

## **Multiagent** Deep Deterministic Policy Gradient Algorithm

In the cooperative multiagent environment, agents interact with the environment and each other. However, because all agents are constantly learning to improve their policies, it is a major problem for multiple agents. From the perspective of each agent, the environment is dynamic and unstable, which may lead to the model not converging, and a single agent cannot adapt to the unstable environment by changing its own policy to a certain extent.

To solve a similar nonstationary problem in a multiactor conflict environment, this paper uses the MADDPG algorithm [52] to train the conflict resolution model. The MADDPG algorithm is composed of three parts: the multiagent system, deep learning and deterministic policy gradient (DPG) algorithm [53], which is one algorithm of MADRL methods. Each agent trains a critical value network that needs global information and a policy network actor that needs local information, allowing each agent to have its own reward function. Therefore, it can be used for cooperative or competitive tasks, and the action space can be continuous or discrete. The algorithm has the following three characteristics:

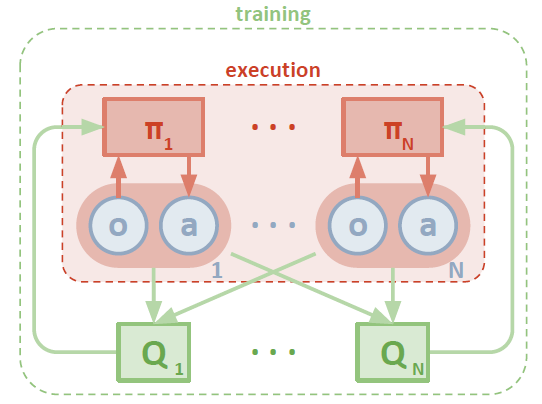


Figure 6. Diagram of centralized training and decentralized execution of the MADDPG algorithm[52].

**Centralized training, decentralized execution.** As shown in Figure 6, during training, the critical network of each agent uses the local observation and action of other agents to estimate the policies of other agents and conduct global learning and guidance. The actor who can only access the local observation information of the agent gradually learns the optimal policy. When applied, only the fully trained actor needs to observe the output action locally.

**Improve experience replay data.** To adapt to the dynamic and unstable environment, each piece of information stored in the experience replay pool is composed of , where  represents the local observation of agent ,  represents the action it performs,  represents the local observation at the next moment after  is executed, and  represents the reward value it obtains.

**Optimization with policy set.** In MADRL, an agent easily overfits a strong policy for other agents, and this strong policy is difficult to adapt to the updating of other agents' policies, which leads to the model performance not being further improved in training**.** To address the above situation, the MADDPG algorithm proposes a policy set idea. Each agent's policy consists of a set with  subpolicies, and only one subpolicy is used in each training episode. For each agent, we maximize the overall reward of its policy set and construct a memory for each subpolicy to improve the stability and robustness of the algorithm. Please refer to the details of the algorithm [52].

## One-to-any Mechanism

The model based on the MADDPG algorithm can quickly give conflict resolution policies according to the local observations of aircraft. However, due to the limitation of the training mechanism of the MADDPG algorithm and the network structure, only scenarios with a fixed number of conflicting aircraft can be used to train the model. If there are multiactor conflict scenarios with different numbers of conflict aircraft, the model cannot be solved, and the corresponding model must be trained additionally. To solve this problem, this section proposes a one-to-any mechanism, that is, using an RNN structure based on a parameter sharing and meta learning framework to reform the MADDPG algorithm, so that it can use different conflict scenarios to train **one** common model, which can resolve multiactor conflicts with **any** number of conflict aircraft.

### Parameter Sharing

In the MADRL algorithms, the MADDPG algorithm is distributed according to the structure of the model, that is, each agent has its own model, the input is its own observation (state), and the output is its own action (as shown in Figure 7.b). Compared with the distributed structure, it is a centralized structure (as shown in Figure 7.a). All agents share a common model. The input is a global observation composed of local observations of all agents, and the actions of all agents are output uniformly. However, with the increase in the number of agents, whether centralized or distributed MADRL algorithms, the weight parameters of the model will increase significantly, which will make the calculation time of model updating very long, and the efficiency of model training is low. The parameter sharing mechanism solves the above problem by reducing the number of parameters as much as possible [54]. It sets the parameters , , and  in Figure 7 to:

 (6)

 (7)

 (8)

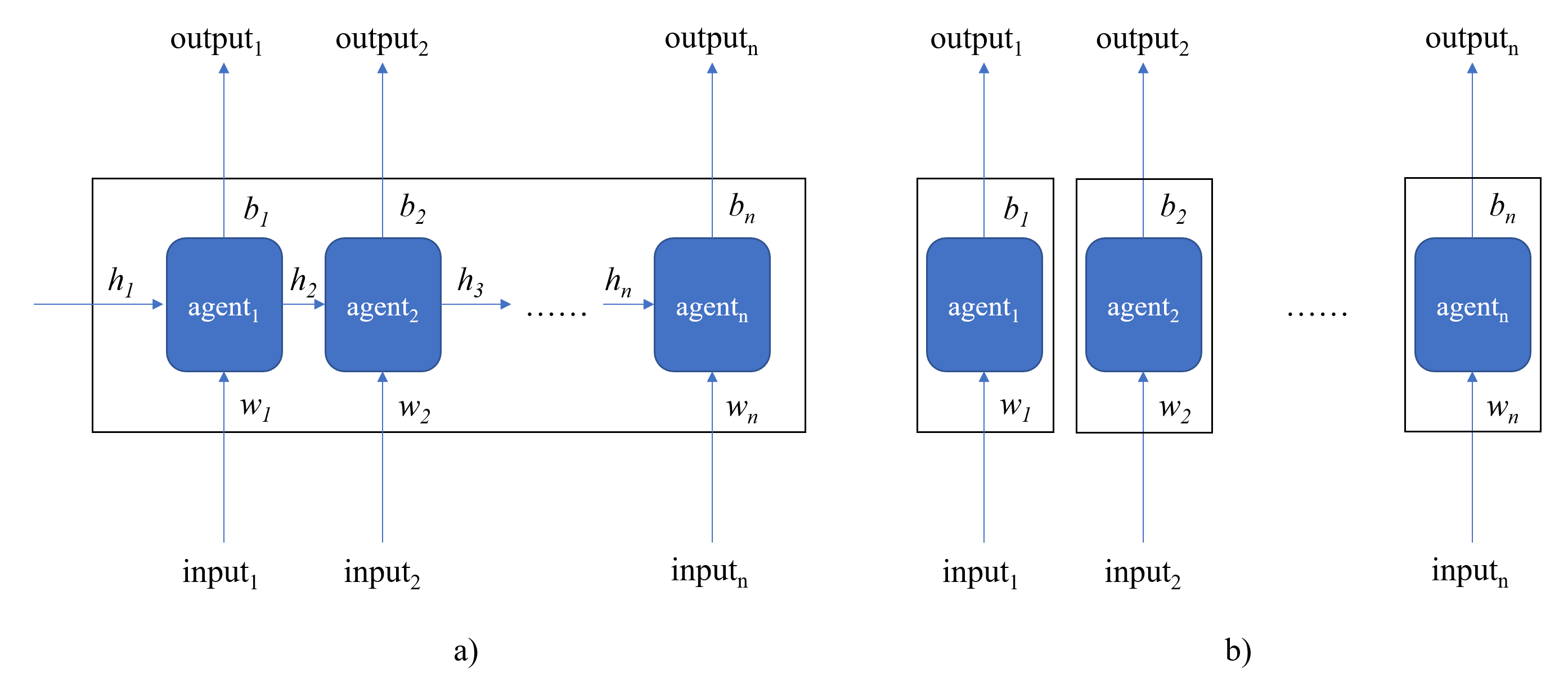


Figure 7: Multiagent reinforcement learning algorithms are classified by their structure: a) centralized structure; b) distributed architecture.

### Model Copying

The one-to-any mechanism in this paper is based on the parameter sharing mechanism. After adding the parameter sharing mechanism to the MADDPG algorithm, all agents share a common model (including a policy network actor and a value network critical). However, different from the centralized structure, the input of the actor network in the model is still the local observation of each agent (as shown in Figure 8). This method is called "model replication". In other words, when the -actor conflict scenario is encountered, the actor network will be copied  times to solve the policy, and the critical network will also be copied  times in the training process. At the same time, the network structure is set as the bidirectional recurrent neural network (Bi-RNN) with a parameter sharing mechanism (as shown in Figure 9) to support the critical network in estimating the value function of the policy with the change in the number of agents. The model replication method is relatively simple, which can not only speed up the training process of the model but also theoretically solve the conflict of any number of aircraft, so the dynamic scalability of MACR is also solved.

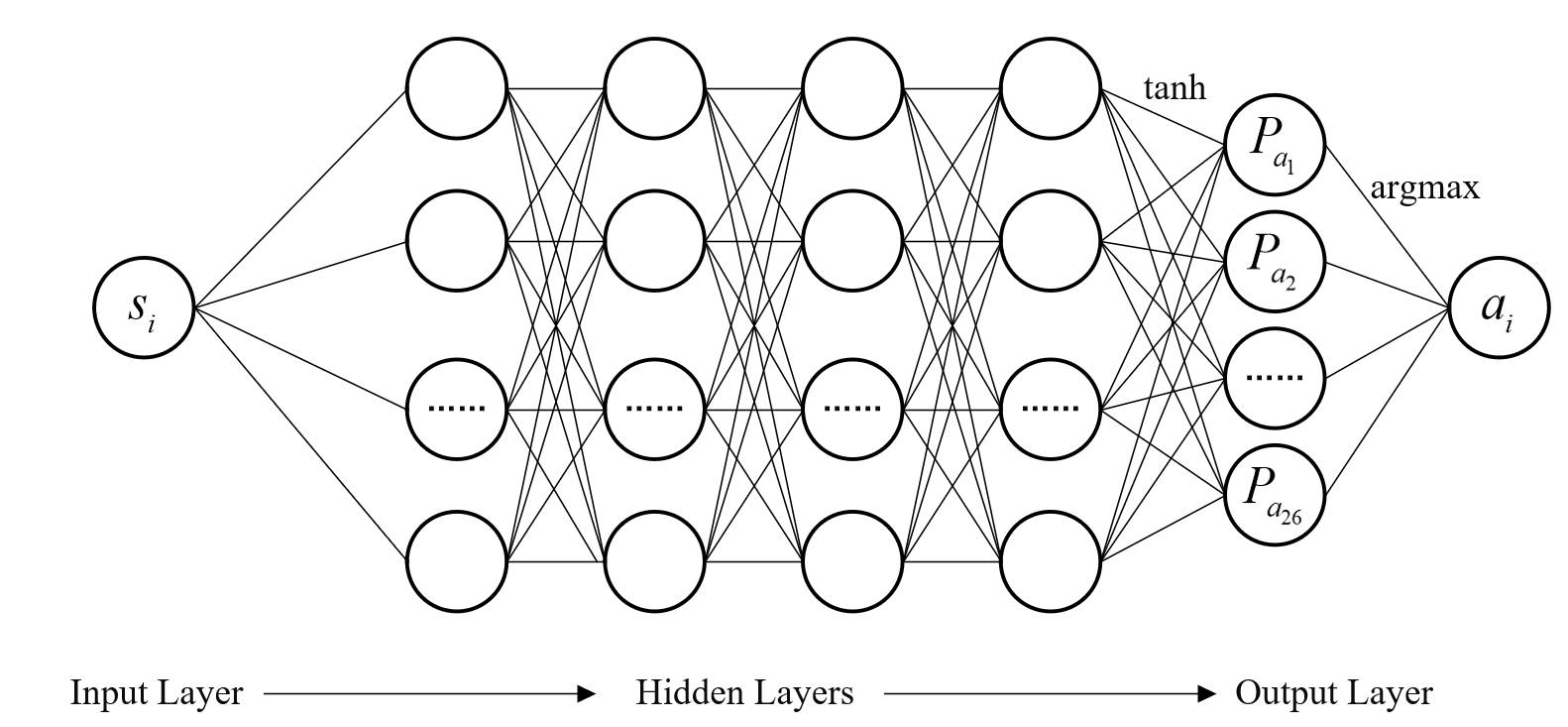
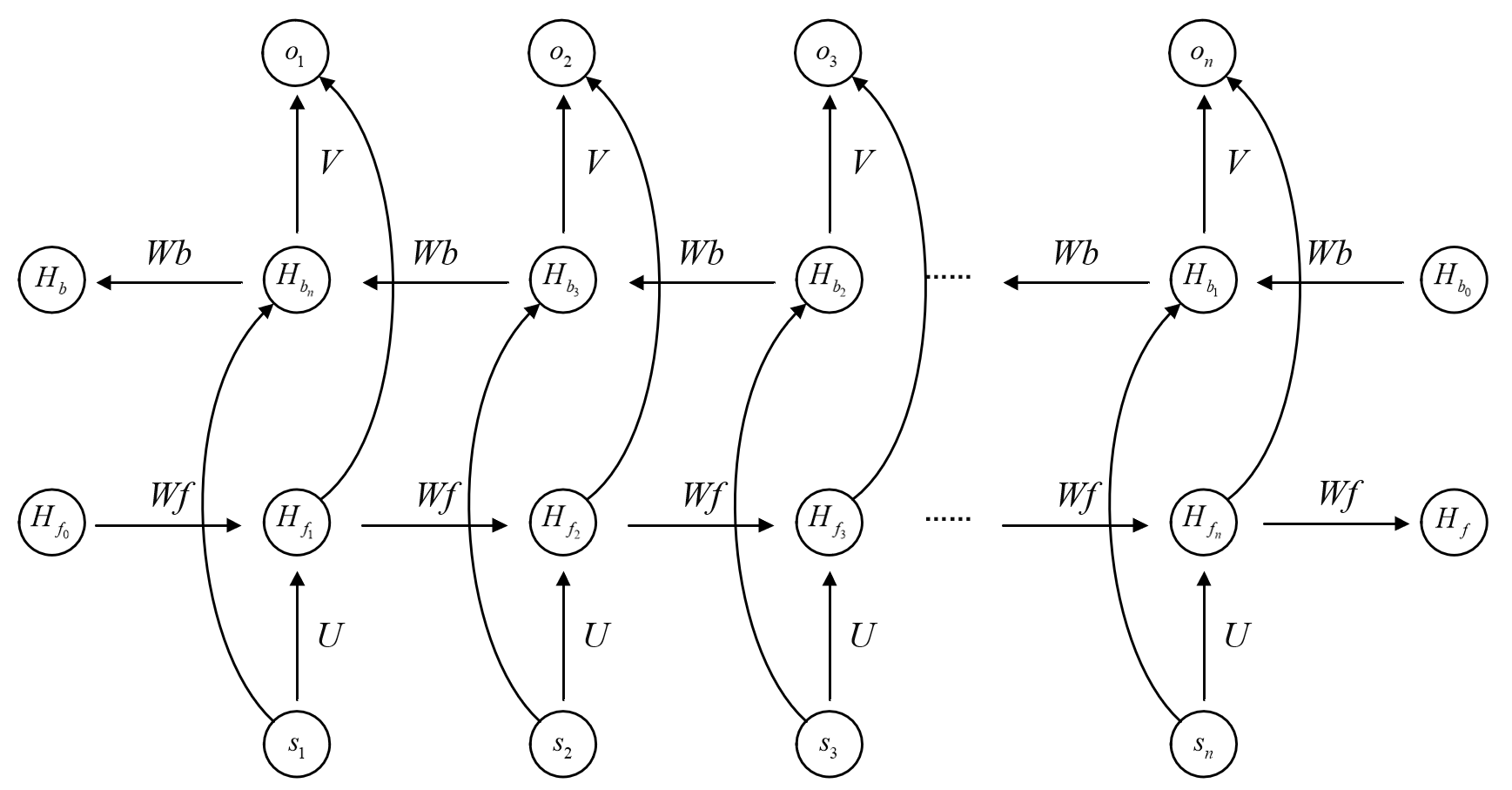


Figure 8. Actor network architecture: Input the state  of aircraft  and output the action .



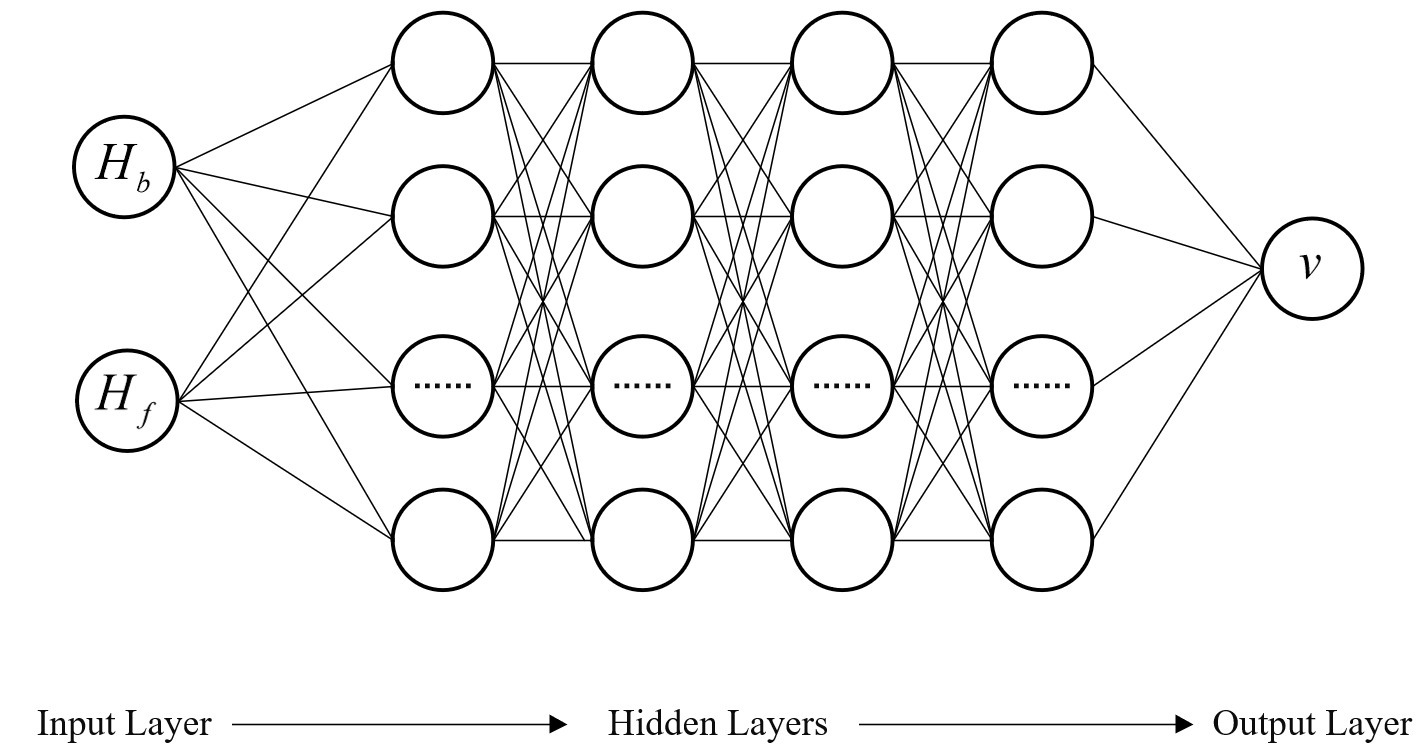


Figure 9. Critic Network Architecture: Bidirectional Recurrent Neural Network, Full Connected Layers and the output is the estimated value  to state-action map of actor network.

Although the "model replication" method can better solve the problem of aircraft quantity change, the performance and stability of the extrication model may be reduced. First, from the perspective of multiple agents, the essence of model replication is to integrate the policies originally belonging to multiple agents into a common policy. Because different policies may be used by conflicting aircraft during multi-aircraft conflict resolution, the critical network of each agent will guide its own actor network to update its policies according to the overall observation and the estimation policies of other agents. When all agents share a critical network, the updating direction of each agent may be inconsistent, which leads to model updating in a worse direction. Second, from the perspective of the number of conflicting aircraft, the policies of multiple aircraft conflicts, such as three aircraft, four aircraft and five aircraft, are also different and not universal. For example, the policies of resolving three aircraft conflicts may not be fully applicable to conflicts with more aircraft. In short, the "model replication" method reduces the training parameters of the model and makes the model more "versatile", but the cost is more tortuous, with a bumpy training process and unstable performance.

In view of the above limitations, this paper uses the "meta learning" framework to transform the model. Meta learning has been proposed to solve the problem of multitask generalization [55]. It regards the conflict resolution of different numbers of conflict aircraft as different tasks to reduce the loss of policy integration for different tasks. It also reduces the complexity of training, reduces the "detour" of the model, and restores or even improves the overall performance reduced by the model replication mechanism. Figure 10 is a pseudocode of the MADDPG algorithm based on meta learning.

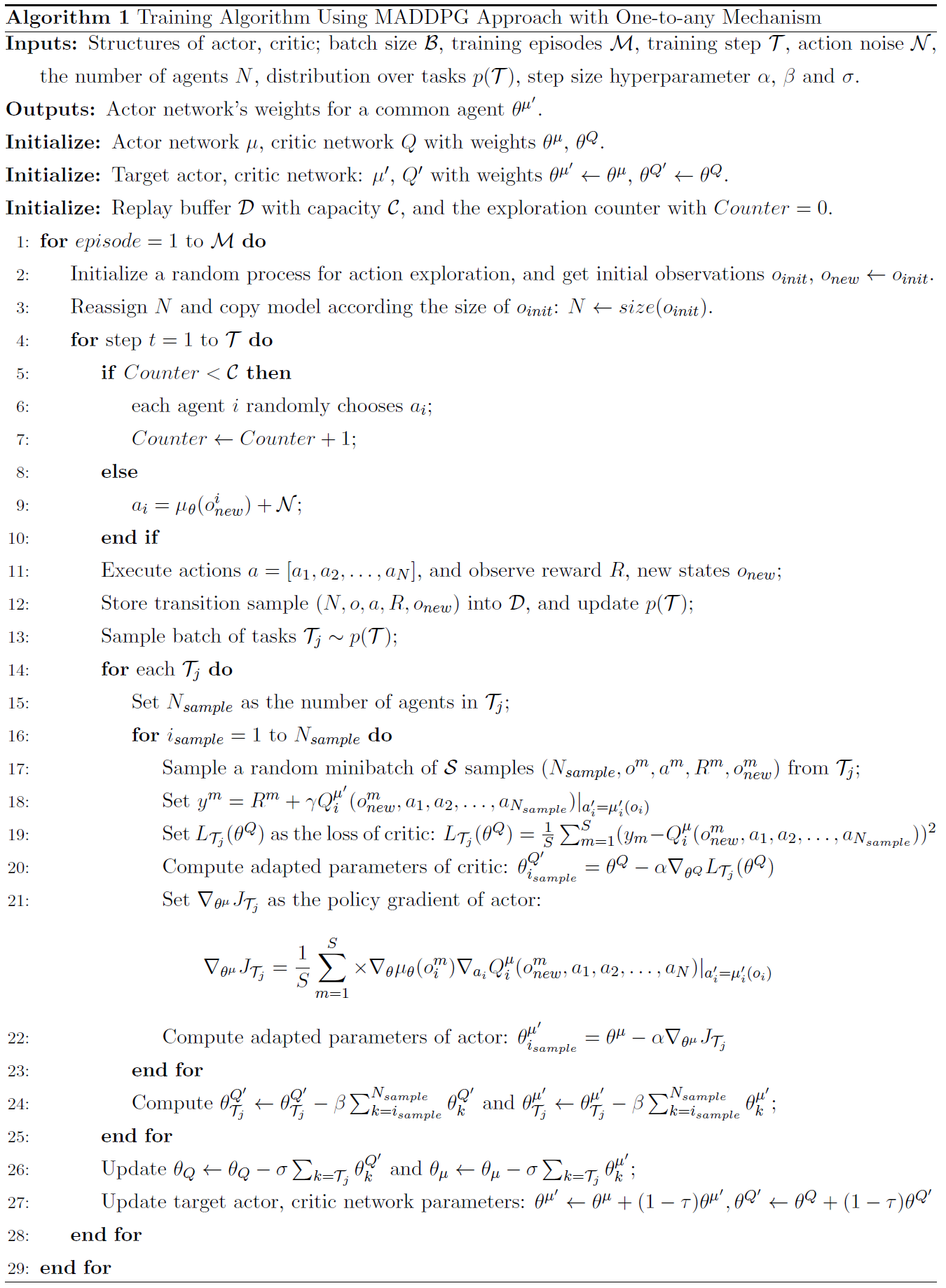


Figure 10. Pseudocode of the dynamically extensible MADDPG algorithm based on meta learning.

## Model Training

### Training Environment Design

**Basic Database:** A basic databaseis used to store the data needed for simulation operation, including navigation databases and aircraft performance databases. Through the processing of the National Aeronautical Information Publication (NAIP) data in 2018 applied from the airspace centre of Civil Aviation Administration of China, the longitude and latitude are converted into decimal form in the unit of degree, the probe code is used as the unique name of the navigation station and waypoint, and the sector boundary is fitted to form a closed polygon. By analysing and processing the Operation Performance File (OPF) of BADA applied from Eurocontrol, a database containing more than 200 types of aircraft performance data is obtained.

**Hardware:** HP Z840 Workstation, Intel Xeon(R) CPU E5-2630 v3 @ 2.4 GHz, RAM 32 GB.

**Software:** Windows 10 OS, Python, IntelliJ Pycharm, Pytorch, and ATOSS (simulation platform).

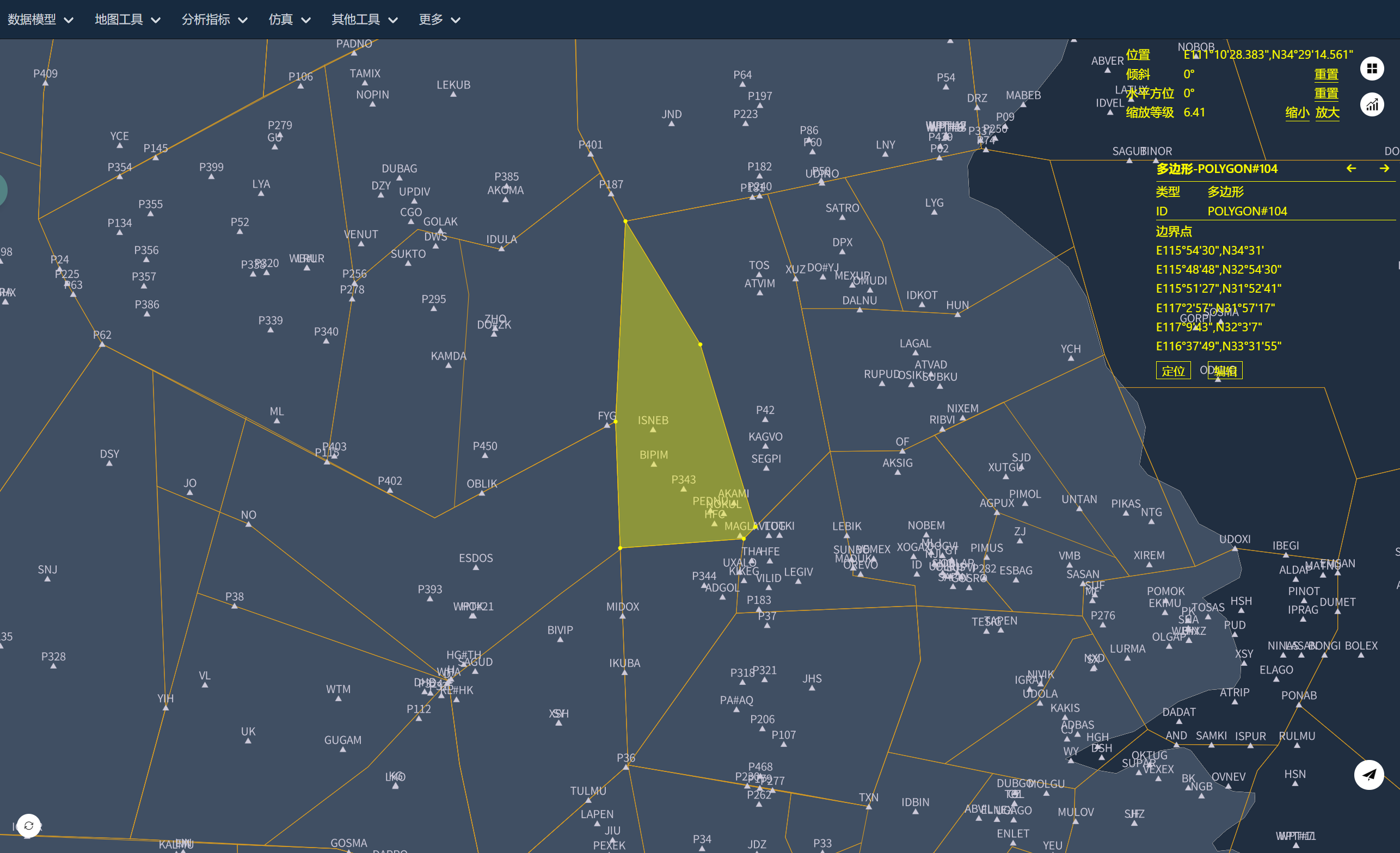


Figure 11: Building the airspace scope of the conflict scenario on the ATOSS simulation platform, and the display language of the interface is Chinese.

Figure 12. The number of flights in the scenarios: the abscissa is arranged in ascending order, but the interval is different.

**Conflict Scenarios Acquisition:** First, a real route sector is selected as the range of the scenario in Chinese airspace (as shown in Figure 11). From the flight plan data in China's airspace on June 1, 2018, 20 to 100 flight plans were selected (the distribution of the flight number in the scenario is shown in Figure 12). To obtain more samples, this paper also adds random changes to the start time, flight type and cruise level in the flight plan. After the simulation runs, the conflict state information detected is collected, recorded, and filtered according to the above multiactor conflict definition (the conflict scenario sample information is shown in Table 1 and Table 2). Finally, a total of 37,232 multimachine conflict scenes were obtained, including 27,232 training samples, 5,000 test samples, and 5,000 test samples. The ratio of three-, four- and five-actor conflict scenario samples is 7:2:1.

Table 1. Information on the three-actor conflict scenario (part).

|  |  |  |  |
| --- | --- | --- | --- |
| Scenario ID | Conflict Aircraft | Conflict Start Time  (UTC) | Flight Plans |
| 1 | CQH8696-CSN6251-CXA8697 | 1:41:51 | FPL\_1 |
| 2 | CSN6894-DKH1256-GCR7577 | 1:56:26 | FPL\_2 |
| 3 | CDG4730-CQH8865-CSN6974 | 0:42:27 | FPL\_3 |
| 4 | CSC8763-DKH1679-KNA8291 | 1:18:06 | FPL\_4 |
| 5 | CGZ7136-CSN3657-CSN6690 | 1:09:30 | FPL\_5 |
| 6 | CES5129-CES5134-CES5137 | 0:49:48 | FPL\_6 |
| 7 | CCA1806-CCA1843-CHH7523 | 1:01:52 | FPL\_7 |
| 8 | CCA1787-CSN6569-FZA6556 | 1:15:30 | FPL\_8 |

Table 2. The format of flight plan data in scenario 1 (part).

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Flight  Plans | Callsign | Registration | Aircraft  Type | Start Time  (UCT) | Route | Current  Level/m | Target  Level/m |
| FPL\_1 | CQH8696 | B1807 | AT45 | 0:00:01 | ZSNB-ZYTX | 8,100 | 8,100 |
| FPL\_1 | CSN6251 | B8995 | D728 | 0:11:38 | ZYHB-ZSPD | 8,100 | 8,100 |
| FPL\_1 | CXA8697 | B1971 | A380 | 1:36:51 | ZYTL-ZSHC | 9,800 | 8,100 |
| FPL\_1 | CSH9354 | B7862 | B763 | 1:36:51 | ZYCC-ZSWH | 9,800 | 7,500 |
| FPL\_1 | CDG4678 | B5785 | A320 | 1:36:51 | ZBAA-ZSRZ | 8,900 | 8,400 |
| FPL\_1 | EPA6241 | B1533 | A330 | 1:36:51 | ZGSZ-ZSLG | 8,900 | 9,800 |
| FPL\_1 | CHH7596 | B5338 | B744 | 1:36:51 | ZSJN-ZYTL | 8,100 | 7,800 |
| FPL\_1 | CES2060 | B6005 | B744 | 1:36:51 | ZSQD-ZLXY | 7,500 | 7,500 |
| … | … | … | … | … | … | … | … |
| FPL\_1 | CES2954 | B2411 | B732 | 1:36:51 | ZYCC-ZSNJ | 10,100 | 10,100 |

### Training Parameters

For the simulation experiment, this paper sets the experimental group as the MADDPG algorithm with dynamic expansion and the control group as the MADDPG algorithm with nondynamic expansion. The following are the hyperparameter settings used in the training of each algorithm:

Table 3. Training hyperparameters of the model based on the dynamically extensible MADDPG algorithm.

|  |  |  |  |
| --- | --- | --- | --- |
| Training parameters | Parameter values | Training parameters | Parameter values |
| Number of agents | 1 (shared) | Buffer size | 100000 |
| Critic network | Bidirectional RNN | Batch size | 32 |
| Actor network | Full Connected Layer | Gamma | 0.99 |
| Number of hidden networks | 64 | Tau | 0.001 |
| Learning rate of Critic | 1e-4 | Print frequency | 100 |
| Learning rate of Actor | 1e-4 | Update interval | 1 |
| Total iteration steps | 100000 | Learning start | 5000 |

Table 4. Training hyperparameters of the model based on the traditional MADDPG algorithm.

|  |  |  |  |
| --- | --- | --- | --- |
| Training parameters | Parameter values | Training parameters | Parameter values |
| Number of agents | 3/4/5 | Buffer size | 100000 |
| Critic network | Full Connected Layer | Batch size | 32 |
| Actor network | Full Connected Layer | Gamma | 0.99 |
| Number of hidden networks | 64 | Tau | 0.001 |
| Learning rate of Critic | 1e-4 | Print frequency | 100 |
| Learning rate of Actor | 1e-4 | Update interval | 1 |
| Total iteration steps | 100000 | Learning start | 5000 |

# Numerical Results and Analysis

This section can be divided into three parts: First, in the training process of using conflict scenario samples, the training effect of the model is judged by collecting and analysing the change of average reward value every 100 episodes; The second is to test the generalization performance of the model with the test set samples which are mutually exclusive with the training set samples; The three is to validate the method of combining sub models into a resolution model using validation set samples. The performance indicators are reflected in the efficiency and safety, respectively, in the average computing time and the success rate. The experimental results of [46] show that compared with the genetic algorithm, the average computing time of the reinforcement learning algorithm is much shorter.

## Model Training Curve

The purpose of this experiment is to show the training curve comparison of the resolution model with and without the dynamic expansion mechanism, so the experimental group algorithm is the MADDPG algorithm with dynamic expansion, and the control group algorithm is the MADDPG algorithm with nondynamic expansion. This experiment uses 27,232 conflict scenario samples of the training set, 5,000 of the test set, and 5,000 of the validation set, including three-, four- and five-actor conflict scenario samples with a ratio of 7:2:1. See Table 3 and Table 4 in Section 3.3.2 for training hyperparameters. The experimental design is as follows:

Table 5. Design of the Model Training Curve experiment.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Experiment  number | Experimental  Group | Control  Group | Type of  conflict scenario | Scenario  number | Training  steps |
| 1 |  | √ | 3-actor | 22,500 | 100,000 |
| 2 |  | √ | 4-actor | 6,400 | 100,000 |
| 3 |  | √ | 5-actor | 3,332 | 100,000 |
| 4 | √ |  | 3, 4,5-actor | 27,232 | 300,000 |

Figure 13 is the curve of the average reward value with the increase in training episodes. It can be seen from Figure 13.a that the average reward values of the three models are rising and gradually converging. Moreover, the greater the number of conflicting aircraft is, the lower the average reward value is. This is because the cooperation of more agents is more difficult. It can be seen from Figure 13.b that the average reward value of the experimental group algorithm is larger than the average reward of the three models of the control group algorithm, which means that the training effect of the model with the dynamic expansion mechanism is better.

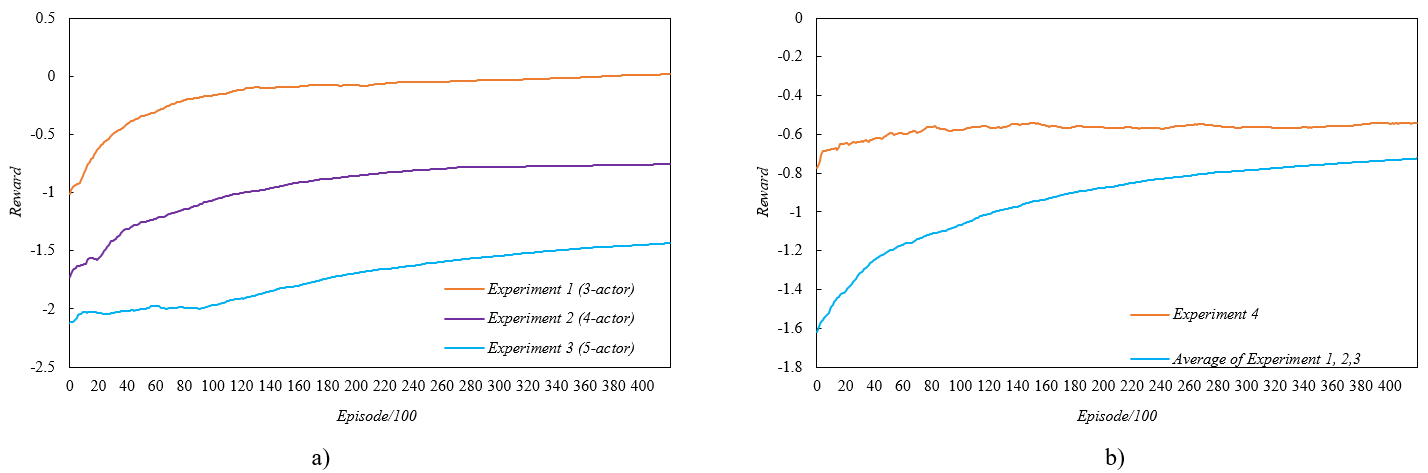


Figure 13. The average reward value with the increase of training episodes: a) experiment 1, 2, 3; b) The experiment 4 and the average reward of experiment 1, 2 and 3.

## Model Performance Testing

The model performance test uses test set samples that are unprecedented in the previous training process of the model to evaluate the generalization ability of the model. The number of test set samples is 5,000, including 3,500 three-actor conflict scenarios, 1,000 four-actor conflict scenarios and 500 five-actor conflict scenarios.

The success rate (SR) refers to the percentage of conflicts that can be successfully resolved by the model in all test sets:

 (10)

where  is the number of scenarios successfully resolved and  is the number of total scenarios of the test set.

Through the statistics and analysis of the success rate of three-, four- and five-actor conflicts and the overall success rate, this experimental result reflects the impact of the dynamic expansion mechanism on the performance of the model. The experimental group algorithm is the MADDPG algorithm with dynamic expansion, and the control group algorithm is the MADDPG algorithm with nondynamic expansion. The experimental design is as follows:

Table 7. Design of the Model Performance Test experiment.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment  number | Experimental  Group | Control  Group | Type of  conflict scenario | Scenario  number |
| 1 | √ | √ | 3-actor | 3,500 |
| 2 | √ | √ | 4-actor | 1,000 |
| 3 | √ | √ | 5-actor | 500 |
| 4 | √ |  | 3, 4,5-actor | 5,000 |

Because the control group algorithm cannot test all the scenarios in an experiment, the overall success rate in Experiment 4 is obtained by averaging the success rates of each scenario type in Experiments 1, 2 and 3. It can be seen from Figure 15 that during the training process, the success rate of the test set gradually increases and then converges. Under the same test conditions, the success rate of the experimental group was higher than that of the control group. Both the training curve and the result of the performance test show that the dynamic expansion mechanism can not only solve different numbers of multiactor conflicts but also improve the generalization performance of the resolution model.

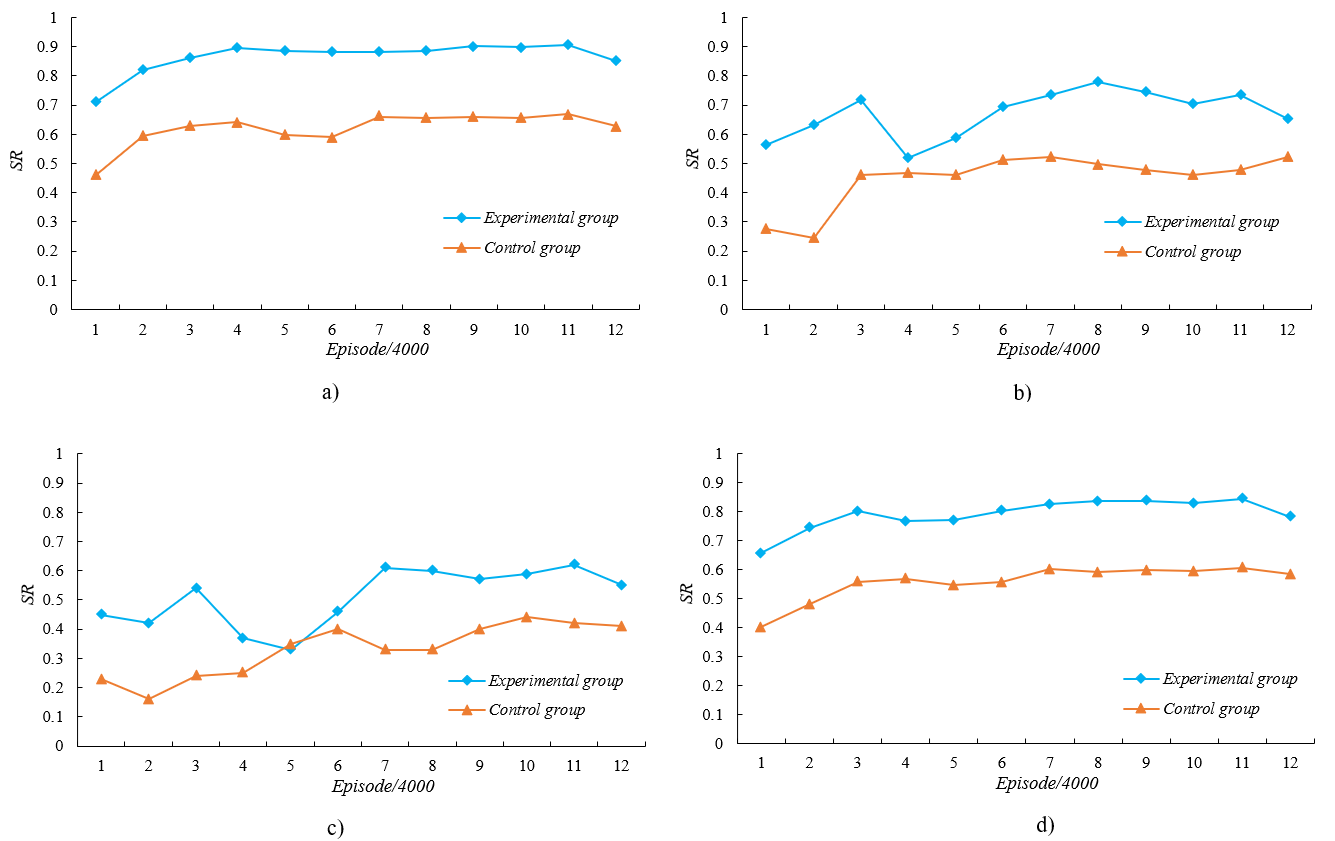


Figure 14. The success rate with the increase in training episodes: a) 3-actor conflict; b) 4-actor conflict; c) 5-actor conflict; d) The overall average success rate.

## Multi-models Validating

The purpose of this experiment is to verify the effectiveness of combining multiple submodels with different and fixed execution times into a single model. The experimental algorithm is the MADDPG algorithm, which can be dynamically extended. The experimental design is as follows:

Table 7：Design of multimodel validation experiment (,  and  see Figure 3)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Experiment  number |  |  |  | Scenario  number |
| 1 | *t*-300 | *t*-180 | *t*-60 | 27,232 |
| 2 | *t*-290 | *t*-170 | *t*-50 | 27,232 |
| 3 | *t*-280 | *t*-160 | *t*-40 | 27,232 |
| 4 | *t*-270 | *t*-150 | *t*-30 | 27,232 |
| 5 | *t*-260 | *t*-140 | *t*-20 | 27,232 |
| 6 | *t*-250 | *t*-130 | *t*-10 | 27,232 |

Each experiment represents a submodel in which three groups of actions are set in the process of training, and the execution time is fixed (as shown in Table 7). The same conflict scenario samples and model parameters are used, and then six submodels are combined after all training is complete. 5,000 samples of validation set are used to this model. The resolution effect, the reward value of the action and the calculation time of each submodel in each scenario are stored and counted.

Figure 15 shows the relationship among the number of submodels, success rate and reward value. The success rate increases from 87.24% to 99.38% when using the combination of six submodels, and the quality of policy is also the highest. In Figure 16, most of the computing time is concentrated in the range of 1 to 3 seconds.

Figure 15. The curve of the relationship between the number of submodels, success rate and reward value in the combination model method.

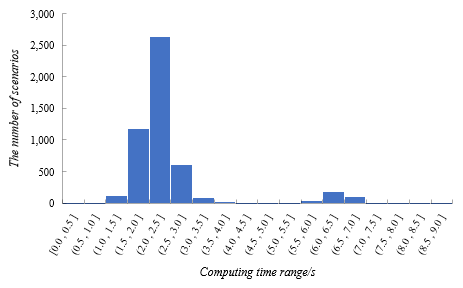


Figure 16. The statistics chart of the computing time of 6 submodels for the combination model solution.

# Conclusions and Future Work

Aiming at the problem of high-density flight conflict resolution under the background of future air traffic in the air route sector, this paper proposes an idea: cluster the pairwise conflicts in the sector are clustered according to the correlation and combined into a multiactor conflict. To support the resolution model to provide the policy for multiactor conflicts with different numbers of conflict aircraft, this paper uses a multiagent deep deterministic policy gradient (MADDPG) algorithm. In addition, the parameter sharing mechanism and the recurrent neural network (RNN) structure are applied to make the model replicable. Moreover, the meta learning framework will protect the model performance from the impact of the dynamic extension mechanism. In the aspect of action design, this paper uses discrete action, including heading adjustment (dogleg manoeuvre), altitude adjustment and speed adjustment, and proposes a combined model composed of several submodels with different and fixed execution times to solve multiactor conflicts. Finally, a large number of high-density multiactor conflict scenarios are built by using the air traffic operation simulation system (ATOSS). The results of the training curve and performance test show that in the training phase, the model can learn and converge to a stable policy. In the test phase, compared with the nondynamic extended MADDPG algorithm, the dynamic extended mechanism can improve the overall success rate of the model. The combination model can further improve the success rate to nearly 100% and provide a higher-quality resolution policy in a short time.

However, with the increase in the number of conflicting aircraft in the scenario, the success rate shows a downward trend (as shown in Figure 14): The more conflicting aircraft there are, the closer their cooperative relationship needs to be, and the more complex the solution process is, the more difficult it is to find a feasible solution. Although increasing the number of training scenarios and properly adjusting the reward function or learning rate can narrow the gap and improve the overall success rate to a certain extent, this problem is not solved completely. Therefore, future research will focus on multiactor conflict resolution with more conflicting aircraft and a higher success rate. In addition, in view of the above ideas to solve the flight conflict in the route sector, we use clustering and other related methods to further study the correlation between the pairwise conflicts. Finally, the reward function of this paper is formulated with the help of air traffic controllers (ATCOs) on the job, but because the conflict resolution policy of the ATCOs in the sector has subjective preference, to improve their acceptable level of the output action of the resolution model, the next step will be how to extract the experience from the historical data.

# Acknowledge

This paper is jointly completed by the project team of the Intelligent Air Traffic Control Laboratory of Nanjing University of Aeronautics and Astronautics (NUAA) and cannot be separated from the efforts of every member. This work was supported by the Safety Ability Project of the Civil Aviation Administration of China (No. TM 2018-5-1/2) and the Open Foundation project of The Graduate Student Innovation Base (Laboratory) of NUAA, China (No. kfjj20190720).

# Data Availability

The conflict scenarios are randomly generated in Python and stored in MongoDB database, and the relevant codes of this method can be found in this link: <https://github.com/Lydia-Yahuhe/cdr_maddpg.git>. However, the detailed data (such as flight plans, ATS route, waypoints, sectors and so on) used to support the findings of this study are available from the corresponding author upon request. For the same reason, the login account and password of Air Traffic Operation Simulation System (ATOSS) autonomously developed by our laboratory need to be applied from the author, and the login GUI interface can be accessed from the following links: <http://106.14.36.245:8444/manager/index.html#/login>.

# Reference

[1] E. Hernández-Romero, A. Valenzuela, D. Rivas, Probabilistic multi-aircraft conflict detection and resolution considering wind forecast uncertainty, Aerosp. Sci. Technol. 105 (2020) 105973. https://doi.org/https://doi.org/10.1016/j.ast.2020.105973.

[2] International Civil Aviation Organization, Doc 9750. Global Air Navigation Plan, third edition, in: 2007: pp. 1–23.

[3] H. Erzberger, Automated conflict resolution for air traffic control, (2006).

[4] J. Tang, Review: Analysis and Improvement of Traffic Alert and Collision Avoidance System, IEEE Access. 5 (2017) 21419–21429. https://doi.org/10.1109/ACCESS.2017.2757598.

[5] J.K. Kuchar, L.C. Yang, A review of conflict detection and resolution modelling methods, IEEE Trans. Intell. Transp. Syst. 1 (2000) 179–189. https://doi.org/10.1109/6979.898217.

[6] M. Ribeiro, J. Ellerbroek, J. Hoekstra, Review of Conflict Resolution Methods for Manned and Unmanned Aviation, Aerosp. 7 (2020). https://doi.org/10.3390/aerospace7060079.

[7] C. Tomlin, G.J. Pappas, S. Sastry, Conflict resolution for air traffic management: a study in multiagent hybrid systems, IEEE Trans. Automat. Contr. 43 (1998) 509–521. https://doi.org/10.1109/9.664154.

[8] S. Devasia, D. Iamratanakul, G. Chatterji, G. Meyer, Decoupled Conflict-Resolution Procedures for Decentralized Air Traffic Control, 2009. https://doi.org/10.1109/CCA.2009.5281138.

[9] A. Dudoit, J. Skorupski, A Simulation-Based Approach for the Conflict Resolution Method Optimization in a Distributed Air Traffic Control System, in: 2019: pp. 104–114. https://doi.org/10.1007/978-3-030-27687-4\_11.

[10] H.-X. Chen, Y. Nan, Y. Yang, Real-time Conflict Resolution Algorithm for Multi-UAV Based on Model Predict Control, Algorithms. 12 (2019). https://doi.org/10.3390/a12020047.

[11] P. Brooker, Airborne Separation Assurance Systems: towards a work programme to prove safety, Saf. Sci. 42 (2004) 723–754.

[12] C. Munoz, A. Narkawicz, J. Chamberlain, A TCAS-II resolution advisory detection algorithm, in: AIAA Guid. Navig. Control Conf., 2013: p. 4622.

[13] M.J. Kochenderfer, J.E. Holland, J.P. Chryssanthacopoulos, Next-generation airborne collision avoidance system, 2012.

[14] Y.I. Jenie, E.-J. Van Kampen, J. Ellerbroek, J.M. Hoekstra, Taxonomy of conflict detection and resolution approaches for unmanned aerial vehicle in an integrated airspace, IEEE Trans. Intell. Transp. Syst. 18 (2016) 558–567.

[15] X. Tang, P. Chen, B. Li, Optimal air route flight conflict resolution based on receding horizon control, Aerosp. Sci. Technol. 50 (2015). https://doi.org/10.1016/j.ast.2015.12.024.

[16] P. Zhao, Y. Liu, Physics Informed Deep Reinforcement Learning for Aircraft Conflict Resolution, IEEE Trans. Intell. Transp. Syst. (2021).

[17] L. Pallottino, E. Feron, A. Bicchi, Conflict Resolution Problems for Air Traffic Management Systems Solved with Mixed Integer Programming, Intell. Transp. Syst. IEEE Trans. 3 (2002) 3–11. https://doi.org/10.1109/6979.994791.

[18] J. Omer, A space-discretized mixed-integer linear model for air-conflict resolution with speed and heading manoeuvres, Comput. Oper. Res. 58 (2015). https://doi.org/10.1016/j.cor.2014.12.012.

[19] A. Alonso-Ayuso, L.F. Escudero, F.J. Mart\’\in-Campo, Exact and Approximate Solving of the Aircraft Collision Resolution Problem via Turn Changes, Transp. Sci. 50 (2016) 263–274. https://doi.org/10.1287/trsc.2014.0557.

[20] S. Cafieri, R. Omheni, Mixed-Integer Nonlinear Programming for Aircraft Conflict Avoidance by Sequentially Applying Velocity and Heading Angle Changes, Eur. J. Oper. Res. 260 (2016). https://doi.org/10.1016/j.ejor.2016.12.010.

[21] Y. Matsuno, T. Tsuchiya, J. Wei, I. Hwang, N. Matayoshi, Stochastic optimal control for aircraft conflict resolution under wind uncertainty, Aerosp. Sci. Technol. 43 (2015). https://doi.org/10.1016/j.ast.2015.02.018.

[22] S. Ayhan, P. Costas, H. Samet, Prescriptive Analytics System for Long-Range Aircraft Conflict Detection and Resolution, in: Proc. 26th ACM SIGSPATIAL Int. Conf. Adv. Geogr. Inf. Syst., Association for Computing Machinery, New York, NY, USA, 2018: pp. 239–248. https://doi.org/10.1145/3274895.3274947.

[23] Y. Yi, M. Tong, LiuXi, Flight Conflict Detection and Resolution Based on Digital Grid, in: 2020: pp. 467–479. https://doi.org/10.1007/978-981-15-0187-6\_56.

[24] A. Letchford, The Linear Ordering Problem: Exact and Heuristic Methods in Combinatorial Optimization by Rafael Martí; Gerhard Reinelt, Interfaces (Providence). 42 (2012) 324–325. https://doi.org/10.2307/23254881.

[25] A. Alonso-Ayuso, L. Escudero, F.J. Martín-Campo, N. Mladenovic, A VNS metaheuristic for solving the aircraft conflict detection and resolution problem by performing turn changes, J. Glob. Optim. 63 (2015) 583–596. https://doi.org/10.1007/s10898-014-0144-8.

[26] H. Liu, F. Liu, X. Zhang, X. Guan, J. Chen, P. Savinaud, Aircraft conflict resolution method based on hybrid ant colony optimization and artificial potential field, Sci. China Inf. Sci. 61 (2018) 129103. https://doi.org/10.1007/s11432-017-9310-5.

[27] S. Hao, S. Cheng, Y. Zhang, A multi-aircraft conflict detection and resolution method for 4-Dimensional trajectory-based operation, Chinese J. Aeronaut. 31 (2018). https://doi.org/10.1016/j.cja.2018.04.017.

[28] Y. Hong, B. Choi, G. Oh, K. Lee, Y. Kim, Nonlinear Conflict Resolution and Flow Management Using Particle Swarm Optimization, IEEE Trans. Intell. Transp. Syst. PP (2017) 1–10. https://doi.org/10.1109/TITS.2017.2684824.

[29] J. Xurui, W. Minggong, W. Xiangxi, T. Congliang, W. Zibolin, A multi-aircraft conflict resolution method based on cooperative game, 2017. https://doi.org/10.1109/ICCIS.2017.8274877.

[30] G. Granger, N. Dur, J.-M. Alliot, Optimal Resolution of En Route Conflicts, (2001).

[31] D. Rathbun, S. Kragelund, A. Pongpunwattana, B. Capozzi, An evolution based path planning algorithm for autonomous motion of a UAV through uncertain environments, 2002. https://doi.org/10.1109/DASC.2002.1052946.

[32] A. Sathyan, N. Ernest, L. Lavigne, F. Cazaurang, M. Kumar, K. Cohen, A Genetic Fuzzy Logic Based Approach to Solving the Aircraft Conflict Resolution Problem, 2017. https://doi.org/10.2514/6.2017-1751.

[33] R.K. Cecen, C. Cetek, Conflict-free en-route operations with horizontal resolution manoeuvers using a heuristic algorithm, Aeronaut. J. 124 (2020) 767–785.

[34] S. Cafieri, D. Rey, Maximizing the number of conflict-free aircraft using mixed-integer nonlinear programming, Comput. Oper. Res. 80 (2016) 147–158.

[35] K. Bilimoria, H. Lee, Z.-H. Mao, E. Feron, Comparison of centralized and decentralized conflict resolution policies for multiple-aircraft problems, in: 18th Appl. Aerodyn. Conf., 2000: p. 4268.

[36] K. Kim, R. Deshmukh, I. Hwang, Development of data-driven conflict resolution generator for en-route airspace, Aerosp. Sci. Technol. 114 (2021) 106744.

[37] Z. Wang, H. Li, J. Wang, F. Shen, Deep reinforcement learning based conflict detection and resolution in air traffic control, IET Intell. Transp. Syst. 13 (2019) 1041–1047.

[38] N.P. Tran, D.-T. Pham, S.K. Goh, S. Alam, V. Duong, An Intelligent Interactive Conflict Solver Incorporating Air Traffic Controllers’ Preferences Using Reinforcement Learning, in: 2019 Integr. Commun. Navig. Surveill. Conf., 2019: pp. 1–8.

[39] D.-T. Pham, N.P. Tran, S. Alam, V. Duong, D. Delahaye, A Machine Learning Approach for Conflict Resolution in Dense Traffic Scenarios with Uncertainties, in: ATM 2019, 13th USA/Europe Air Traffic Manag. Res. Dev. Semin., Vienne, Austria, 2019. https://hal-enac.archives-ouvertes.fr/hal-02138135.

[40] T.T. Nguyen, N.D. Nguyen, S. Nahavandi, Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications, IEEE Trans. Cybern. 50 (2020) 3826–3839.

[41] F. Bin, F. XiaoFeng, X. Shuo, Research on cooperative collision avoidance problem of multiple UAV based on reinforcement learning, in: 2017 10th Int. Conf. Intell. Comput. Technol. Autom., 2017: pp. 103–109.

[42] S. Li, M. Egorov, M. Kochenderfer, Optimizing Collision Avoidance in Dense Airspace using Deep Reinforcement Learning, 2019.

[43] M. Brittain, P. Wei, Autonomous separation assurance in an high-density en route sector: A deep multi-agent reinforcement learning approach, in: 2019 IEEE Intell. Transp. Syst. Conf., 2019: pp. 3256–3262.

[44] M. Ribeiro, J. Ellerbroek, J. Hoekstra, Improvement of Conflict Detection and Resolution at High Densities Through Reinforcement Learning, ICRAT 2020. (2020).

[45] M.W. Brittain, P. Wei, One to Any: Distributed Conflict Resolution with Deep Multi-Agent Reinforcement Learning and Long Short-Term Memory, in: AIAA Scitech 2021 Forum, 2021: p. 1952.

[46] S.U.I. Dong, X.U. Weiping, K. ZHANG, Study on the resolution of multi-aircraft flight conflicts based on an IDQN, Chinese J. Aeronaut. (2021).

[47] R.M. Everson, J.E. Fieldsend, Multiobjective optimization of safety related systems: An application to short-term conflict alert, IEEE Trans. Evol. Comput. 10 (2006) 187–198.

[48] I. Doc, 4444--procedures for air navigation services--air traffic management, Montr. QC, Canada Int. Civ. Aviat. Organ. (2016).

[49] S. Temizer, M. Kochenderfer, L. Kaelbling, T. Lozano-Pérez, J. Kuchar, Collision avoidance for unmanned aircraft using Markov decision processes, in: AIAA Guid. Navig. Control Conf., 2010: p. 8040.

[50] M.L. Littman, Markov games as a framework for multi-agent reinforcement learning, in: Mach. Learn. Proc. 1994, Elsevier, 1994: pp. 157–163.

[51] A. Nuic, User manual for the Base of Aircraft Data (BADA) revision 3.10, Atmosphere (Basel). 2010 (2010) 1.

[52] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, I. Mordatch, Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments, Neural Inf. Process. Syst. (2017).

[53] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, M. Riedmiller, Deterministic policy gradient algorithms, in: Int. Conf. Mach. Learn., 2014: pp. 387–395.

[54] J.K. Terry, L.S.B.B. Nathaniel Grammel Ananth Hari, Revisiting Parameter Sharing In Multi-Agent Deep Reinforcement Learning, ArXiv. (2021).

[55] A. Nichol, J. Achiam, J. Schulman, On First-Order Meta-Learning Algorithms, CoRR. abs/1803.0 (2018). http://arxiv.org/abs/1803.02999.