International Journal of Advanced Robotic Systems
January-February 2019: 1–16

DOI: 10.1177/1729881419828917
journals.sagepub.com/home/arx
SAGE

Carrier-borne aircrafts aviation operation automated scheduling using multiplicative weights apprenticeship learning

Mao Zheng[1], Fangqing Yang[2], Zaopeng Dong[3], Shuo Xie[1] and Xiumin Chu[1]

Abstract

Efficiency and safety are vital for aviation operations in order to improve the combat capacity of aircraft carrier. In this article, the theory of apprenticeship learning, as a kind of artificial intelligence technology, is applied to constructing the method of automated scheduling. First, with the use of Markov decision process frame, the simulative model of aircrafts launching and recovery was established. Second, the multiplicative weights apprenticeship learning algorithm was applied to creating the optimized scheduling policy. In the situation with an expert to learn from, the learned policy matches quite well with the expert's demonstration and the total deviations can be limited within 3%. Finally, in the situation without expert's demonstration, the policy generated by multiplicative weights apprenticeship learning algorithm shows an obvious superiority compared to the three human experts. The results of different operation situations show that the method is highly robust and well functional.

Keywords

Carrier-borne aircrafts, launching and recovery scheduling, Markov decision process, apprenticeship learning, multiplicative weights apprenticeship learning

Date received: 29 November 2018; accepted: 12 January 2019

Topic: AI in Robotics; Human Robot/Machine Interaction
Topic Editor: Pengfei Zhang
Associate Editor: Yangquan Chen

Introduction

The capability of carrier-borne aircrafts launching and recovery is significant for an aircraft carrier's combat capacity[1,2] It is also one of the main concerns that the designers have to deal with in the stage of carrier conceptual design. Carrier-borne aircrafts launching and recovery operations present a complex and uncertain environment in which time-critical scheduling should be done to fulfill mission requirements and ensure the safety of aircrafts, deck crews, equipments, and the carrier. The ultimate goal is to ensure the efficiency of aviation operations on flight deck and in the air, in these complex environments.

Most of the existing literatures mainly focus on modeling the process of carrier-borne aircrafts aviation operations and evaluating the sortie generation rates (SGRs). Feng et al.[3] established the sortie generation model with full consideration of faults and maintenance effecting on aviation operation flow, finally, this multi-agent system

[1] National Engineering Research Center for Water Transport Safety, Wuhan University of Technology, Wuhan, China
[2] Jiangnan Shipyard Group Co., Ltd., Shanghai, China
[3] School of Transportation, Wuhan University of Technology, Wuhan, China

Corresponding author:
Zaopeng Dong, School of Transportation, Wuhan University of Technology, Heping Rd., Wuhan, Hubei, 430070, China.
Emails: dongzaopeng@whut.edu.cn; whutusv@126.com

(MAS)-based simulation gained aircrafts SGRs. However, every aircraft in the MAS system is the same priority, which disagree with the facts (e.g. there are always many aircrafts have priorities over others in fact). Taking the priorities of different aircrafts into consideration, Zheng et al.[4] put forward a sortie generation model based on closed queueing network and achieved a new analytic algorithm to obtain the stationary solutions of this multi-class, multi-server, non-preemptive closed queueing network. In essence, the carrier-borne aircrafts launching and recovery process is a closed queueing network in which every aircraft changes its state after an operation is completed. So, early research concentrated in different queueing networks which were used to simulate the aircrafts aviation operations.[5,6] However, these analytical models are so limited that it is impossible to deal with actual combat operations, such as launching and recovery aircrafts in waves which cannot be regarded as Poisson-distribution statistic process. To research complex queueing networks, Monte Carlo simulation methods were widely introduced,[7] and accurate stationary solutions of queueing network were gained through large amount of stochastic calculations. In 1997, the USS Nimitz and Carrier Airwing Nine conducted surge operations and the highest SGR of Nimitz reached 975 in 4 days.[1,2] To analyze the bottleneck of SGR in surge operations, Zheng et al.[8] established a simulation model using Monte Carlo method to simulate the actual surge operations conducted in 1997, achieving relatively accurate results. These theoretical and simulation-based solutions mentioned above could give reliable results almost the same as the data gained in surge operations, however, were lack of robustness to nonstationary stochastic disturbances such as aircraft malfunction and deck equipment malfunction. So, these methods just could be good tools to analyze the SGR of a carrier, because the policy for dealing with random events is predetermined. To some extent, the lack of robustness becomes a major limitation when developing the aviation operation decision-making system. To improve the robustness of these models needs gaining more accurate operation information and unlocking the system's potential, meanwhile, the scheduling and planning calculation should be accomplished as quickly as possible. As the amount of warfare situation information, aircrafts and carrier condition information is so tremendously huge that human operators could not understand well. Therefore, it is significant to deal with random events and accidents in a more automated way. Operators need decision-making support system which can generate a schedule of coordinated aviation operations for all active aircrafts and deck equipments (catapults, arrestors, aviation support equipments, etc.). The policy should be optimized for safety, efficiency, and better robustness to different types of uncertainty inherent on the flight deck and in the air based on large amount of information.

In the fields of robotics, reinforcement learning, such as State Action Reward State Action (SARSA), Q-learning, and deep reinforcement learning (DRL), was widely applied on the basis of reward functions. In automatic driving areas, it is easy to provide reward functions, for example, we can determine the reward functions in the states when our car is taxiing on the middle of the road, keeping proper distances with other cars as a good value. Unfortunately, for aviation scheduling problems, even specifying a reward function is not easy. However, it is often easier to provide examples of a desired behavior than to define the corresponding reward function for human experts. Learning policies from demonstrations, known as apprenticeship learning (AL) via inverse reinforcement learning (IRL),[9] were also proposed by researchers. To improve the efficiency of aviation operation scheduling on flight deck, Li et al.[10] proposed a computer-aided decision-making system, using IRL, obtaining almost the same performance as human operators. Wu et al.[11] developed a dynamic sequencing algorithm to deal with the sequencing problem for landing a team of aircrafts using ant colony method, gaining the optimal sequence, and guaranteeing a higher level of flight safety and an effective response to dynamic circumstance.

In recent years, researchers in MIT[12,13] (2011–2013) developed the Deck operations Course of Action Planner (DCAP) system to improve the overall mission performance, which allows human operators to highly interact with computer system for scheduling both the carrier-borne aircrafts and different kinds of aviation support equipments. Especially taking more factors which influence the capability of carrier-borne aircrafts aviation operation into consideration,[1,2,14] the system robustness to different types of uncertainty inherent on the flight deck and in the air was improved. DCAP system integrates human operators' thinking with optimization algorithm using IRL which is a kind of artificial intelligence algorithm,[15,16] which can generate risky policies and safety policies. This human–computer interaction system could also adapt unmanned aerial vehicles (UAV) operated on carriers better than human operators.[17]

AL is a kind of machine learning theory which was introduced by Abbeel and Ng in 2004,[18] and from then on, a number of different new algorithms were put forward.[19–21] AL now is a focus of machine learning research and widely applied in the field of robot control and navigation.[22,23] For complex systems, such as aircraft aviation operation process on aircraft carrier, it is difficult to gain the optimum scheduling based on theoretical calculation. On the contrary, machine learning methods could extract information from operators' demonstration and make decisions like human operators, needing not knowing the exact mathematical models. Through AL, the expert's scheduling experience will be extracted and kept as mathematical parameters, then, the scheduling and planning will be created automatically based on learned policies.

AL, introduced by Abbeel and Ng, is motivated by two observations in applications.[24] The first is that the reward functions are difficult to gain directly; however, it is easy to

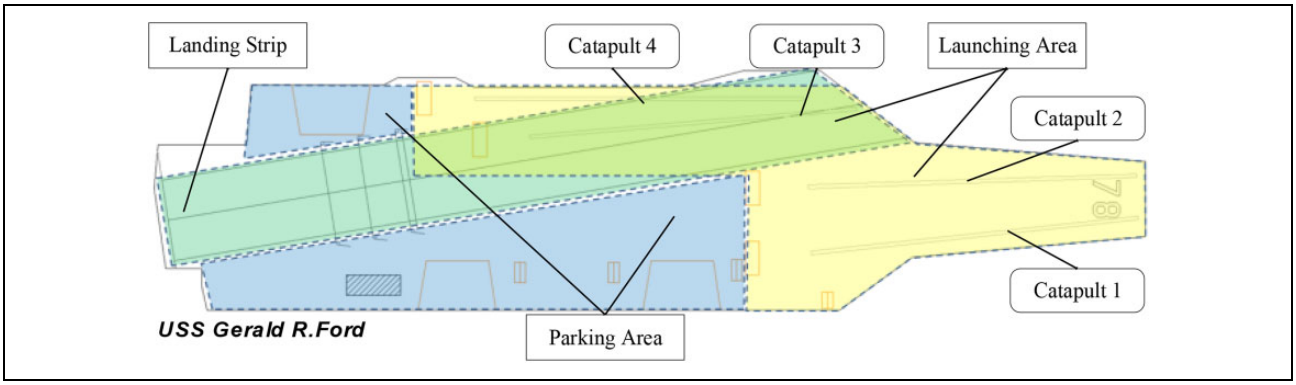

Figure 1. The layout plan of a Nimitz class aircraft carrier's flight deck.

specify what the rewards should depend on. The second is that the human experts could offer enough examples for machine learning. From then on, a number of different new algorithms were put forward. Now, AL is a focus of machine learning research and widely applied in the field of robot control and navigation.[22] For complex systems, such as aircraft aviation operations process on aircraft carrier, it is not only difficult to put forward an optimal scheduling based on theoretical method, but also impossible to describe the reward function. On the contrary, AL could learn the scheduling policies from operators' demonstrations and make decisions like human operators. Through AL, the expert's scheduling experience will be extracted and kept as mathematical parameters; then, the scheduling and planning will be created automatically based on learned policies.

Initially inspired by Michini,[15] the work of this article expands the application of the AL method in generating optimized scheduling policy for carrier-borne aircrafts. First, the Markov decision process (MDP) was used to structure the detailed process of carrier-borne aircrafts aviation operation. Second, one of the AL algorithms, multiplicative weights AL (MWAL)[25,26] was applied to generate the learned policy from an expert's demonstration. Third, the MWAL algorithm was also applied to dealing with the situation with no expert available by updating the state features and weight vectors with the iterations. Finally, the computational experimental results in the typical conditions showed that this method is of both reliability and practicability.

The MDP model of carrier-borne aircrafts aviation operation and the MWAL algorithm

The sketch of carrier-borne aircrafts aviation operation

A heavy nuclear-powered aircraft carrier's typical flight deck layout plan is shown in Figure 1. Generally, the flight deck can be divided into three functional areas including the parking area, launching area, and landing strip.[27] In most cases, aircrafts awaiting orders to take off are deposited in their specified parking positions in parking areas on flight deck. The fueling, arming, and other kinds of aviation support operations have to be completed. Once having received the order to take off, they will start jet engines and taxi to an available catapult, checking hydraulic systems, ailerons, rudders, and other flight control surfaces, preparing for launching.

The operating frequency of Catapults 3 and 4 is higher than Catapults 1 and 2. All the four catapults may conflict with each other when operating at the same time, but staggering their working time a little will avoid this problem. Meanwhile, Catapults 3 and 4 overlap with the landing strip and these two catapults' launching and the landing operation could not be conducted at the same time. After launching, these carrier-borne aircrafts climb to the cruising altitude, fly to mission area, and perform conventional missions. When their missions are completed, these aircrafts will fly back and circle around the carrier known as the Marshal stack, waiting for the order to land. If an aircraft attaches the arresting cable and be stopped finally, it will taxi to parking area. However, if it fails to catch any arresting cable, it will take off and climb to the Marshal Stack again and wait for another attempt, until it lands on the flight deck. Figure 2 shows the process of one carrier-borne aircraft aviation operation.

Abstraction of the state features. State features should provide enough information of the MDP, as well as their number should not be so large that the computational expense increases extremely. To approve the quality of the learned policy in the situation with an expert available, just like the model established by Ryan,[12] the following 24 state features are chosen to describe the reward function and they are:

$f^{(1)} \sim f^{(6)}$: numbers of aircrafts with fuel level $1 \sim 6$;
$f^{(7)} \sim f^{(12)}$: numbers of aircrafts in the air with fuel level $7 \sim 12$;
$f^{(13)} \sim f^{(14)}$: numbers of aircrafts crashed and number of aircrafts with no fuel on the flight deck;

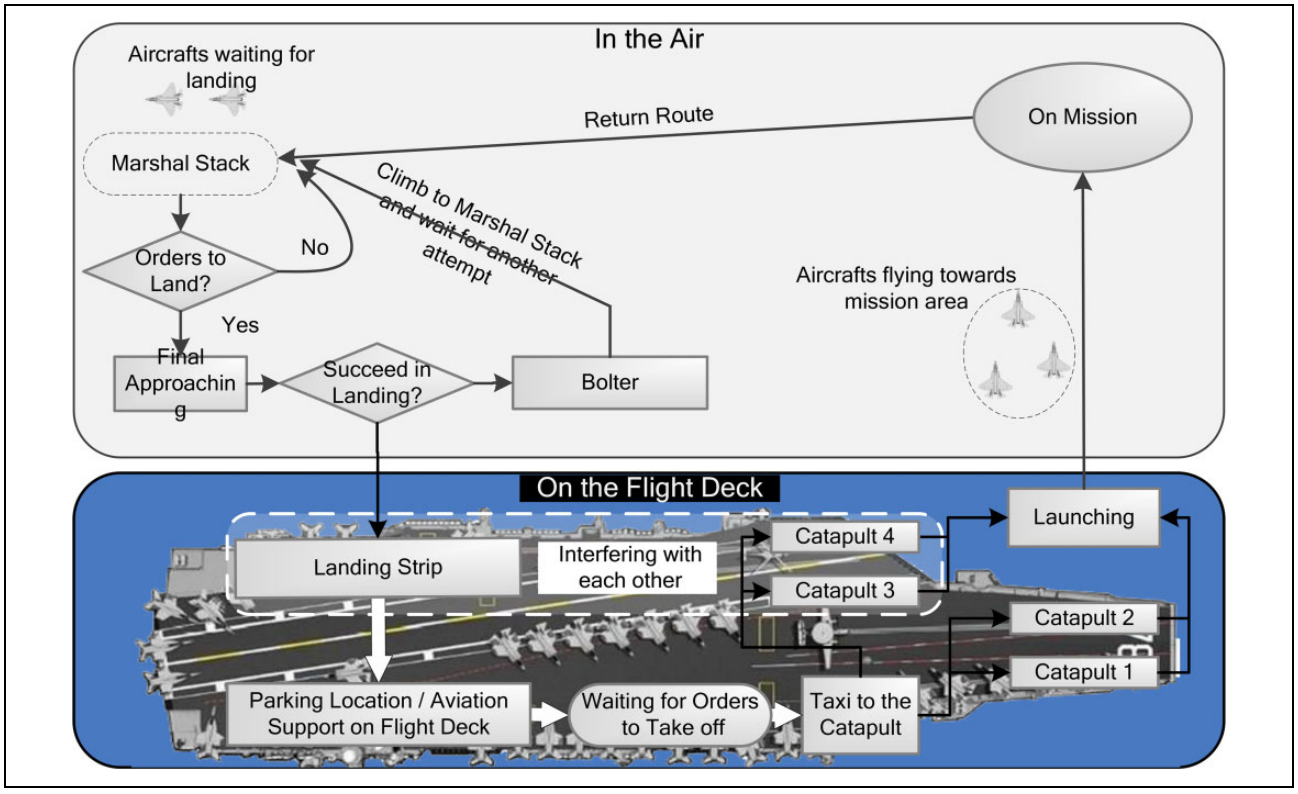

Figure 2. The process of one carrier-borne aircraft aviation operation.

$f^{(15)}$: number of aircrafts in special priority; and
$f^{(16)} \sim f^{(24)}$: numbers of aircrafts in each of the nine locations.

The settings of operation pattern research. There are two kinds of regular combat modes for aircraft carriers, known as concentrated assault and constant assault. In the concentrated assault mode, the carrier should project as much firepower as possible in short time, that means the carrier must launch and recover a large number of aircrafts with high efficiency. In the constant assault mode, the carrier should maintain there will always be enough number of aircrafts in the air waiting for orders to provide allies with fire support, so the launching and landing operations may take place at the same time on the flight deck. Considering the characteristics of these two kinds of combat modes, in this article, two operation patterns, group sortie and alternate sortie are chosen in the computational experiments. Additionally, the case when there is no expert's demonstration available will also be discussed.

Figure 3 shows these two kinds of sortie operations. For group sortie, the initial state is that all the aircrafts (all the aircrafts belong to Squadron A and Squadron B) are deposited in their parking positions on the flight deck, waiting for orders to take off, and the final state is that all the aircrafts return to the flight deck and taxi to their parking positions again. For alternate sortie, the initial state is that half of the aircrafts (Squadron A) circle in the Marshal stack and the rest (Squadron B) are in their parking positions on the flight deck. The final state is the same as group sortie. For each combat mode, scheduling 6 and 12 aircrafts will be the two specific research cases.

Time distributions of aviation operations. To embody the uncertainty of carrier-borne aircrafts aviation operation scheduling, the hypothesis is applied that each operation's time obeys normal distribution and the settings are as follows:

time of refueling $\sim N(8, 0.1)$;
time of taxiing from a parking location to a catapult $\sim N(15, 0.2)$;
time of launching $\sim N(15, 0.2)$;
time of mission $\sim N(60, 0.5)$;
time of landing and arrested to stop $\sim N(10, 0.1)$;
time of taking off again and flying to the Marshal stack $\sim N(12, 0.1)$; and
time of taxiing from the landing strip to a parking location $\sim N(4, 0.05)$.

The simulation program runs in 0.1 unit (we define 1 min as a time unit in simulation) of time as a step size. The failure rate of each aircraft aviation support equipment is set to be 0.05, and the rate of each aircraft leaking fuel in the air is set to be 0.01.

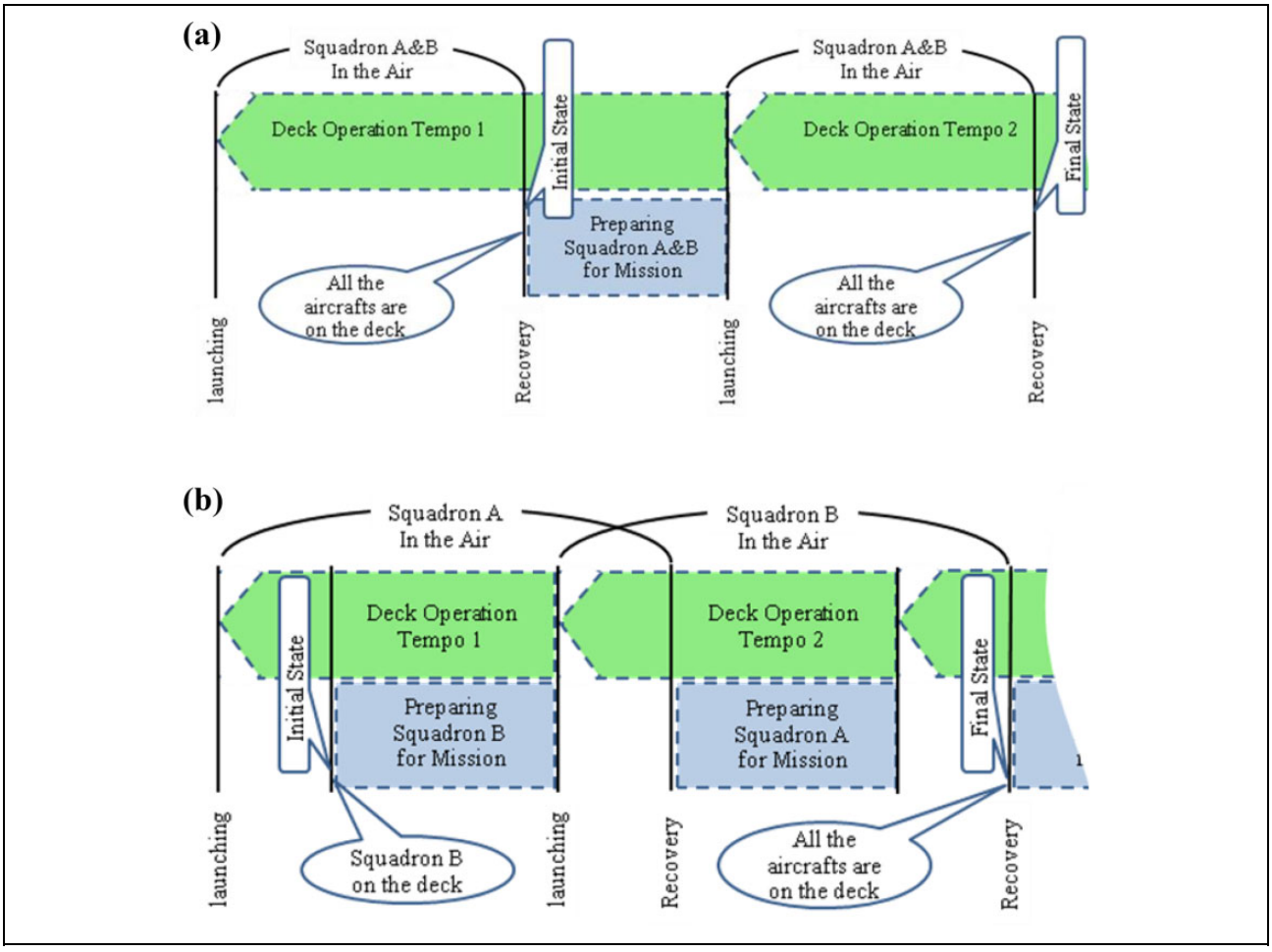

Figure 3. Two kinds of sortie operations on carrier. (a) Group sortie operation period on aircraft carrier and (b) alternate sortie operation period on aircraft carrier.

The MDP model of the scheduling process

The first step in developing a scheduling system is finding a suitable model which captures the relevant system dynamics, actions, constraints, and uncertainties. MDP is a discrete time stochastic control process. It provides a mathematical framework for modeling decision-making in situations where outcomes are partly random and partly under the control of a decision maker. With this in mind, the carrier deck scenario is modeled as a finite-state MDP. The process of carrier-borne aircrafts aviation operation scheduling can be modeled with the use of a finite-state MDP. An MDP contains six elements, including the space of states $\boldsymbol{S}$, the space of actions $\boldsymbol{A}$, the transition probabilities $\boldsymbol{\theta}$, the discount factor $\boldsymbol{\gamma}$, decision time $\boldsymbol{T}$, and the reward function $\boldsymbol{R}$, which can be written as a tuple $\{S, A, \theta, \gamma, T, R\}$.

According to the abovementioned analysis, S includes the aircraft *location*, *fuel level* and the *priority states for landing*, as well as the *condition states of the aviation support equipments* on the flight deck such as catapults and landing strip, and so on. The location states include (1) Parking location/Aviation support operation, (2) Catapult-1, (3) Catapult-2, (4) Catapult-3, (5) Catapult-4, (6) On mission, (7) Marshal Stack, (8) Landing/bolter, and (9) Landing strip. The fuel level of each aircraft can be discretized into seven levels, in which level-6 represents the fuel is full and level-0 represents no fuel in fuel tank. An aircraft will crash if using up all the fuel in the air. The priority states for landing include the normal priority and special priority. The normal priority is determined by the sequence of each aircraft reaching the Marshal Stack, and the special priority concerns whether the aircraft is damaged and the aircraft fuel level in the air. The condition states of the aviation support equipments on the flight deck includes the condition parameters of four catapults and one landing strip. They are *available*, *occupied*, *task conflict*, and *malfunctional* (the task conflict mainly occurs between catapult 3 or 4 and the landing strip). As a result, the size of states space S could be as large as $(9\text{locations} \times 7\text{fuellevels} \times 2\text{priority} \times n)^n \times 4\text{condition}^{5\text{equipments}}$, where n represents the number of operating aircrafts. With the growth of n, the size of S will exponentially increase.

The available actions for each aircraft in actions space A include towing the aircraft to a specified position, launching, returning, landing, and so on, encoding some constraints (e.g. it is impossible to transition to an occupied catapult, or land when the landing strip is inoperative) as well as the key uncertainties (e.g. an aircraft tries to transit from approaching to landing strip, but misses the wires and thus transits to the Marshal Stack instead). Some of these actions could lead the MDP to a determined state and others to a stochastic state using transition probabilities θ.

As described earlier, the size of S is extremely large, so the reward function R can be hardly represented by traditional state-to-action pairs. Instead, a reward function based on a set of state features, which could be represented as $f(s) = \{f^{(1)}(s), f^{(2)}(s), \ldots, f^{(m)}(s)\}^T$, would be an effective way to solve this problem, and the reward function can approximately be a linear combination of these state features

$$R(s) = R\Big(f(s)\Big) = w^T f(s) \tag{1}$$

where w is a weight vector corresponding to $f(s)$ and it is an unknown variable to be computed. So for policy π, its value function can be written as

$$\begin{aligned} V(\pi) &= E\Big[\sum_{l=0}^{\infty} \gamma^l R(s_l)|\pi\Big] = w^T E\Big[\sum_{l=0}^{\infty} \gamma^l f(s_l)|\pi\Big] \\ &= w^T \mu(\pi) \end{aligned} \tag{2}$$

where $\mu(\pi)$ is the discounted expectation vector of the state features, and l indexes over all state features.

MWAL in carrier-borne aircraft scheduling

MWAL is one of the AL algorithms and is easy to be operated and programmed, meanwhile, the iteration converges fast.

As the output of MDP is stochastic, for a given policy π, the mean value of numerous repeated simulations will be calculated to estimate the relatively precise value of the state feature expectations which could be written as follows

$$\mu = E\Big[\sum_{l=0}^{\infty} \gamma^l f(s_l)\Big] \approx \hat{\mu} = \frac{1}{k}\sum_{i=1}^{k}\sum_{l=0}^{\infty} \gamma^l f(s_l) \tag{3}$$

where k represents the number of repeated simulations. Thus, the expectation vectors of the state features and their estimated values can be written as μ_{E} and $\hat{\mu}_{\mathrm{E}}$, respectively, corresponding to the expert's policy π_{E}.

The final goal of AL is to find a learned policy $\pi*$, whose state features can be very close to the expert's policy π_{E}. In this article, the reward function is chosen as a kind of profit map, which means for a given w, the optimal policy is

$$\pi_{\mathrm{opt}} = \mathrm{argmax}_{\pi \in \Pi}\{V(\pi)\} = \mathrm{argmax}_{\pi \in \Pi}\{w^T \mu(\pi)\} \tag{4}$$

where Π is the set of all possible policy, different from traditional gradient optimization algorithms, in every iteration, MWAL updates each component of w by multiplying by a positive parameter β

$$w_{t+1}^{(i)} = w_t^{(i)} \beta^{\hat{\mu}_t^{(i)} - \hat{\mu}_{\mathrm{E}}^{(i)}} \,|\, i = 1, 2, \ldots, m \tag{5}$$

where superscript i for $w_t^{(i)}$ represents ith component of w at iteration t (weight vector w has m components in all corresponding to each state feature). $\hat{\mu}_t$ is the estimated value of state features expectation vector which corresponds to the weight vector w_t and its optimal policy π_t. π_t can be calculated by the use of the approximation method introduced by Bertsekas and Tsitsiklis.[28] Let T is the number of iterations. We can define parameter β as follows

$$\beta = \frac{1}{1 + \sqrt{\frac{2\ln(m)}{T}}} \tag{6}$$

Especially, if there is no expert's demonstration available, let $\mu_{\mathrm{E}} = 0$ to replace the primary expert's demonstration's discounted expectation of the state features. This is an important difference between the MWAL algorithm and the original IRL algorithm. The situation without expert's demonstrations will be discussed in the fourth section.

Finally, when the iteration completes, the objective learned policy of MWAL algorithm will be a mixed policy

$$\pi * = \{(\pi_t, \lambda_t)\}_{t=1}^{T} \tag{7}$$

It means that at the beginning of the MDP, π_t will be chosen due to the probability λ_t and $\lambda_t = \frac{1}{T}$. Thus, the expectation vector of state features of $\pi*$ will be $\mu* = \frac{1}{T}\sum_{t=1}^{T} \mu(\pi_t)$. In all, the iterations of the MWAL algorithm application in carrier-borne aircrafts aviation operation scheduling are presented in Figure 4.

The situation with human expert's demonstrations

The principle of expert's demonstrations

In this section, it is assumed that human expert will try to balance the safety and efficiency, which means when he is conducting the scheduling, before taking off, he would fuel each aircraft to an appropriate level and when arranging the sequences of landing of the aircrafts queuing in the Marshal stack, he would consider both normal priority and special priority. Considering both the precision and time expense, in this article, it is specified that for the expert's policy, the expert would give 10 demonstrations and for the learned policy, 100 iterations would be used to gain the average state features.

Choosing the mixed objective policy

As has been discussed in "MWAL in carrier-borne aircraft scheduling" section, the typical output of the MWAL

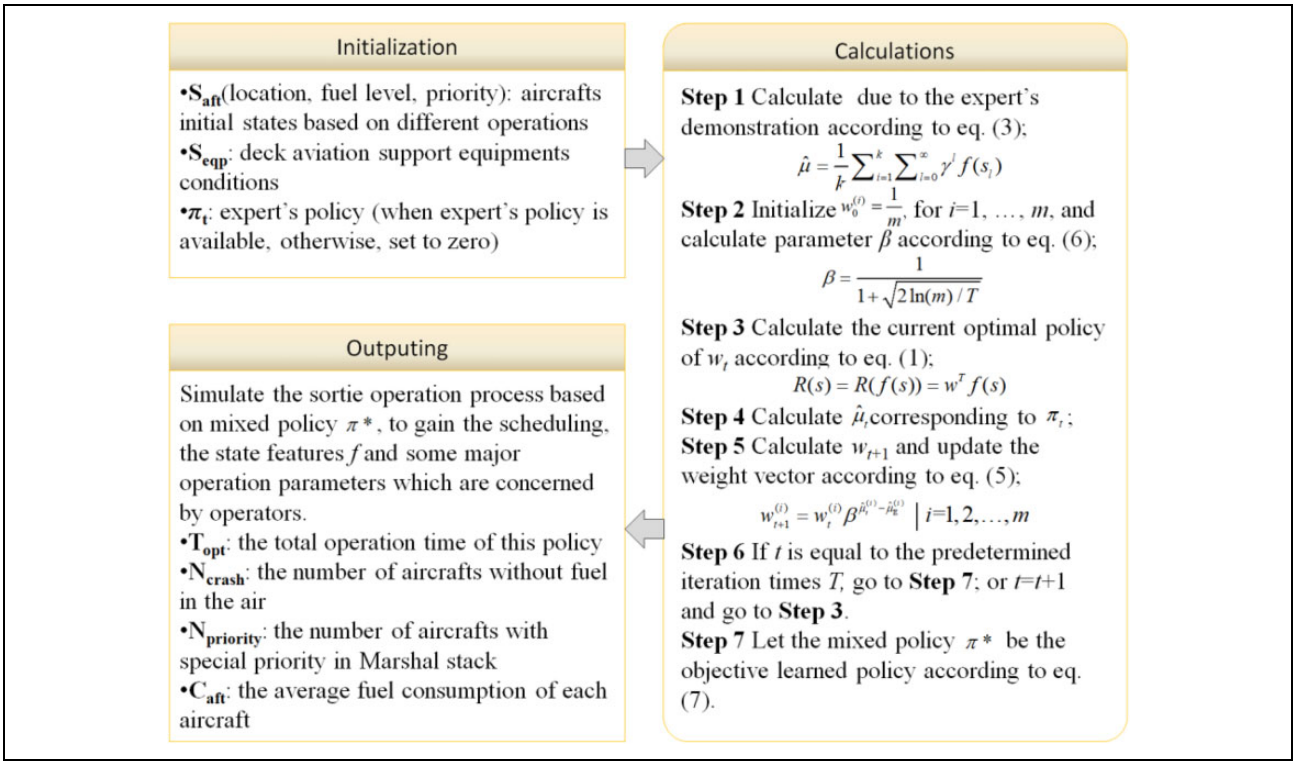

Figure 4. The strategy of the algorithm application in carrier-borne aircrafts aviation operation scheduling.

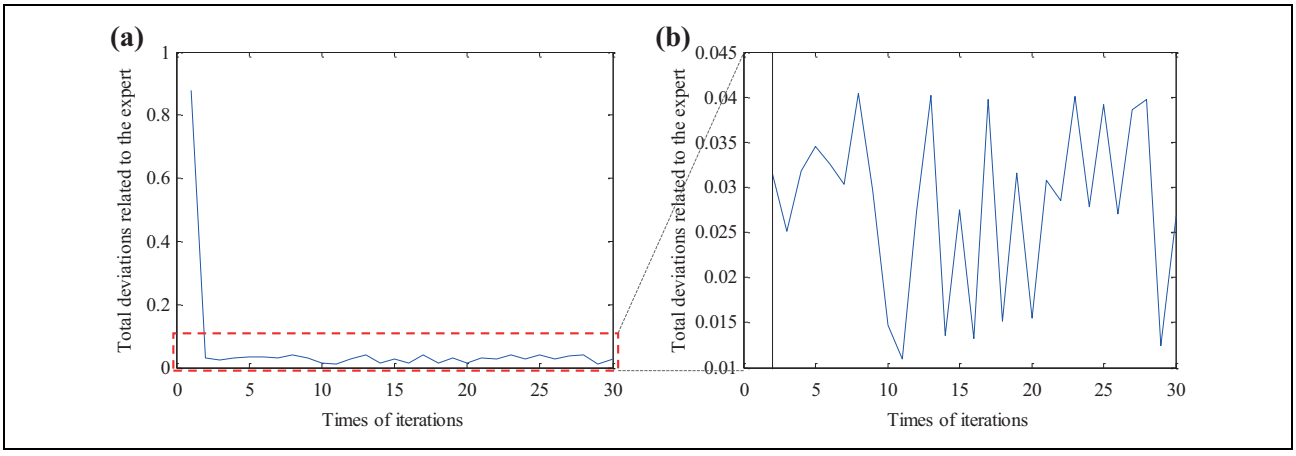

Figure 5. The total deviations of the expectation vectors of each iteration related to the expert' demonstration. (a) When $t \geq 1$ and (b) When $t \geq 2$.

algorithm is a mixed policy like equation (7). To exam the convergent performance of MWAL, taking scheduling 6 aircrafts group sortie as the experimental condition, the total deviations of the expectation vector of each iteration $\hat{\mu}_t$ related to that of the expert's demonstration $\hat{\mu}_E$ are calculated and when the iteration times $T = 30$, the changes of $\hat{\mu}_t$ versus the increase of T are shown in Figure 5(a) and (b)

$$\pi* = \{(\pi_t, \lambda_t)\}_{t=2}^{T} = \frac{1}{T-1}\sum_{t=2}^{T}\mu(\pi_t) \qquad (8)$$

Figure 5 implies that the MWAL algorithm converges very fast. When $t = 1$ (Figure 5(a)), the total deviation of the expectation vector of the initial learned policy π_1 related to the expert's demonstration can be as large as 90%. However, after just one iteration, when $t = 2$ (Figure 5(b)), the total deviation of that of π_2 descends to a rather low level and fluctuates in a small range less than 5%. So, to guarantee the performance of the objective learned policy, equation (7) is modified to be equation (8) and π_1 won't be included.

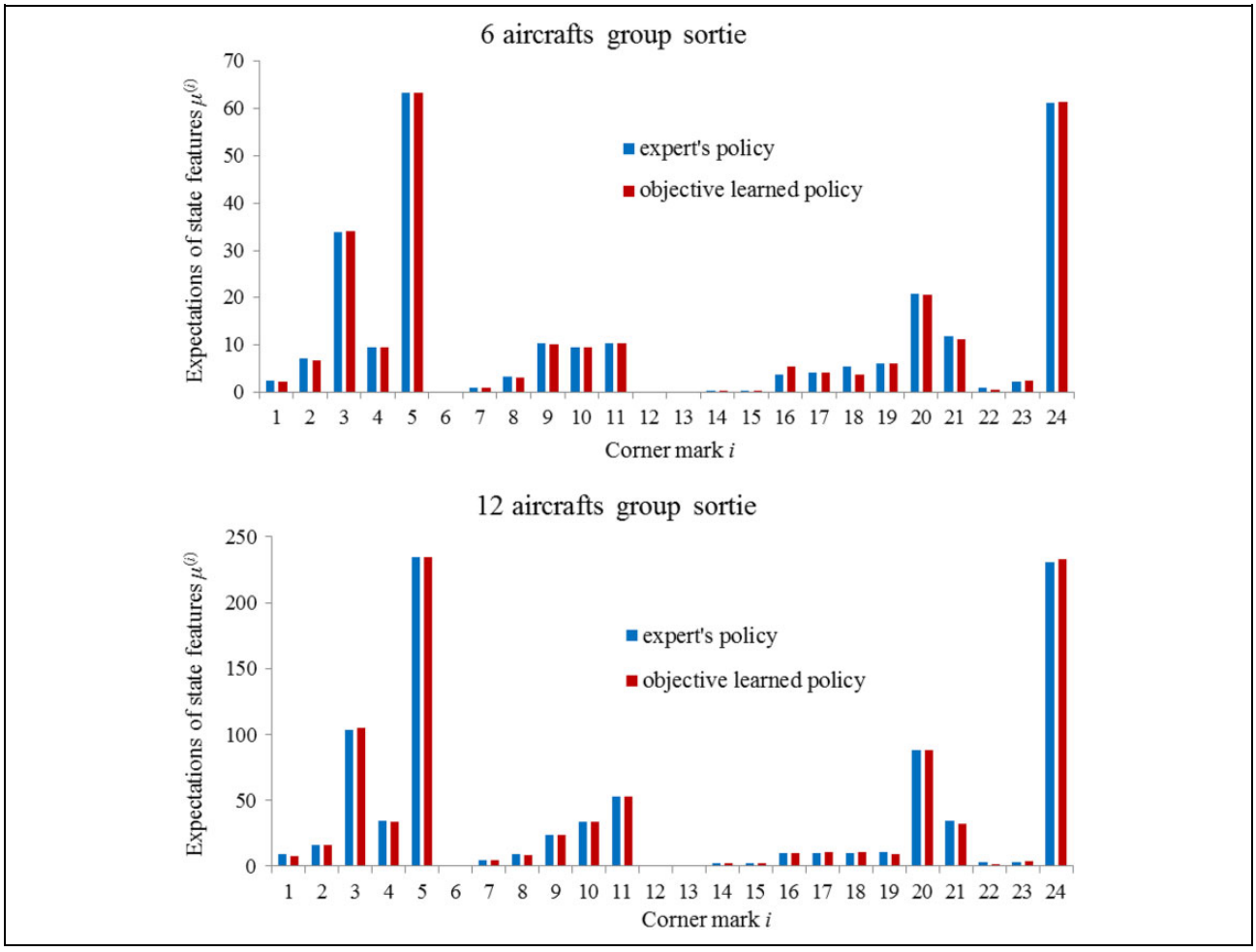

Figure 6. Expectations of the state features of the expert's and the objective learned policies in the two simulation experiments in group sortie condition.

Experimental results

In group sortie condition. The program of MWAL algorithm application in carrier-borne aircrafts aviation operation scheduling was developed in MATLAB 2012b and performed on a Lenovo Think Centre PC (3.4-GHz Intel I7 CPU, 16-GB RAM, Windows 7 64-bit operation system). When scheduling 6 and 12 aircrafts in group sortie condition, it takes 10773 and 36928 s for the program to gain the objective learned policy, respectively.

With the state features expectation vectors of the expert's 10 demonstrations and the objective learned policy's 100 simulations (actually, 30 iterations are enough, more simulations could improve the accuracy), the histogram of their components is shown in Figure 6. Ocularly, the objective learned policy is very close to the expert's performance. For 6 and 12 aircrafts scenarios, the total deviations of the objective learned policy state features expectations related to the expert's demonstrations are 2.68% and 1.17%, respectively, corresponding to scheduling 6 and 12 aircrafts. Considering the complexity of the carrier-borne aircrafts aviation operation scheduling, the MWAL algorithm application in this article is of high precision.

When scheduling is being conducted, both combat efficiency and safety should be considered, the mission time of scheduling, number of crashed aircrafts, number of aircrafts in special priority, and the average fuel consumption of each aircraft are chosen as the four indexes to evaluate the performance of the MWAL algorithm applied in carrier-borne aircrafts aviation operation scheduling. With the data of expert's demonstration and the objective learned policy, the box-whisker plot is drawn in Figure 7, presented with 2-σ error bars. The mean values of these performances are shown in Table 1.

According to the data in Figure 7 and Table 1, these four indexes show that when scheduling 6 and 12 aircrafts, both these two objective learned policies are close to the expert's demonstrations, judging from no matter the mean values or the variance. It indicates that the MWAL algorithm applied in carrier-borne aircrafts launching and recovery scheduling could obtain policies which are similar to experts' policies. In Figure 7, considering the time of

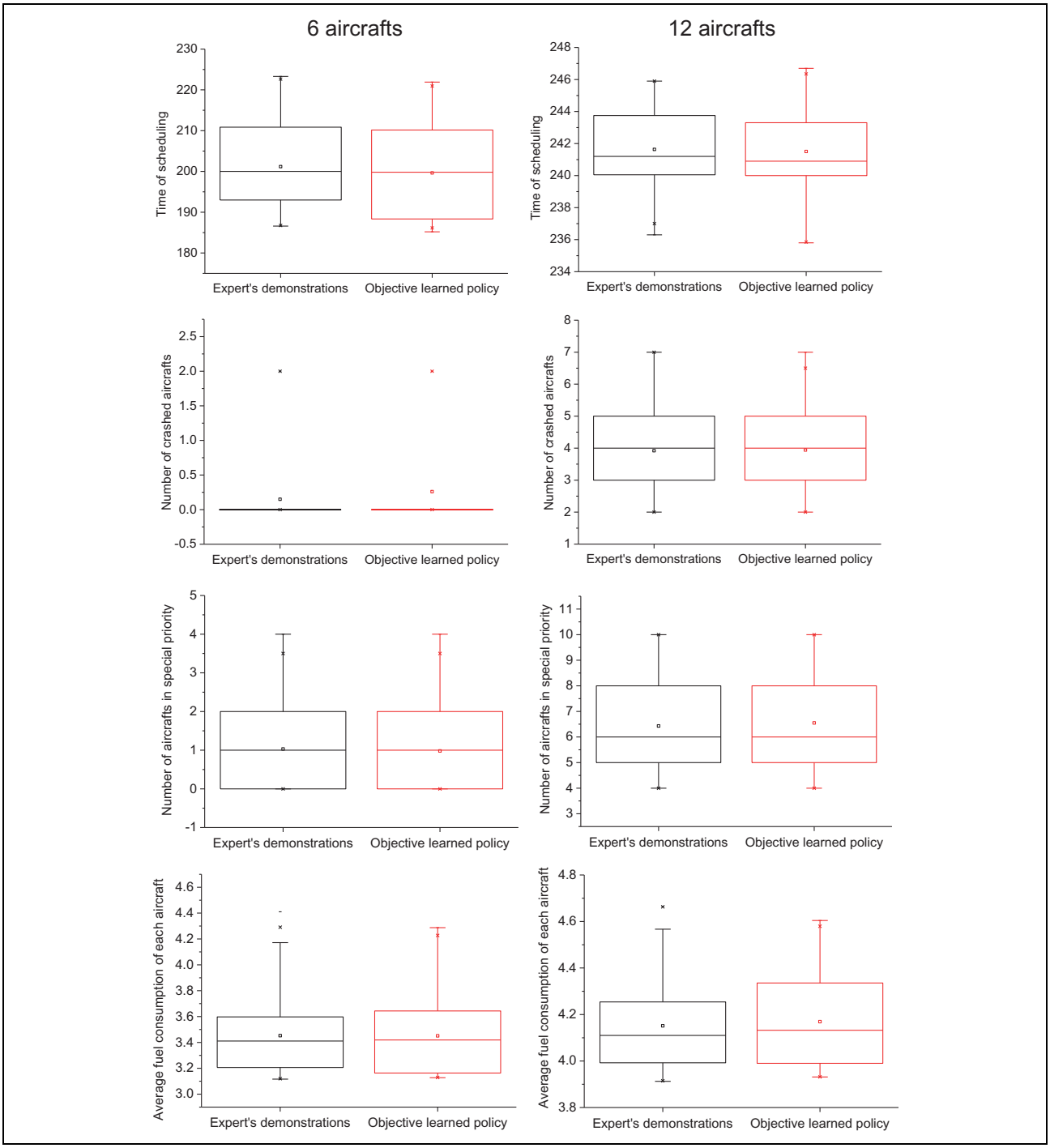

Figure 7. Statistic data of the indexes of the expert's demonstrations and the objective learned policies when scheduling 6 and 12 aircrafts in group sortie condition.

scheduling, the objective learned policy of scheduling 6 aircrafts is closer to the performance of the expert's demonstration than that of scheduling 12 aircrafts. However, the concentration ratio of the 12 aircrafts case is better than that of the 6 aircrafts case. This shows that by the increase in aircrafts number in the sortie operation process, the random factors that affect the time of scheduling will weaken and stabilize. When scheduling 6 aircrafts, the number of aircrafts in special priority or crashed is pretty small. However, when scheduling 12 aircrafts, the crashed aircrafts numbers of both learned policy and expert's demonstration increase a lot (about 1/3 of these aircrafts crashed, meanwhile, more than 1/2 of these aircrafts are in special priority). From a security point of view, the expert's policy is not suitable for the case of scheduling 12 or more aircrafts. These aircrafts should be fueled to a higher level to insure

Table 1. Average values of the indexes of the expert's demonstrations and the objective learned policies when scheduling 6 and 12 aircrafts in group sortie condition.

Number of aircrafts	6		12	
Operator	Expert's demonstrations	Objective learned policy	Expert's demonstrations	Objective learned policy
Time of scheduling (min)	201.185	199.628	241.633	241.504
Number of crashed aircrafts	0.15	0.30	3.92	3.94
Number of aircrafts in special priority	1.03	0.98	6.43	6.55
Average fuel consumption of each aircraft (*t*)	3.4523	3.4512	4.1514	4.1692

Table 2. Average values of the indexes of the expert's demonstrations and the objective learned policies when scheduling 6 and 12 aircrafts in alternate sortie condition

Number of aircrafts	6		12	
Operator	Expert's demonstrations	Objective learned policy	Expert's demonstrations	Objective learned policy
Time of scheduling (min)	152.874	154.284	203.041	201.581
Number of crashed aircrafts	0.14	0.15	1.26	1.01
Number of aircrafts in special priority	0.24	0.40	3.55	3.5
Average fuel consumption of each aircraft (*t*)	1.9757	2.0210	2.6253	2.6165

the security. Considering the fuel consumption, when scheduling 12 aircrafts, each aircraft will consume more fuel in average, in other words, these aircrafts spend more time in queuing in the Marshal stack.

In alternate sortie condition. When scheduling 6 and 12 aircrafts in alternate sortie condition, it takes 7124 and 17599 s for the program to gain the objective learned policy, respectively. The results are shown in Figure 9 and Table 2. It can be implied that when scheduling the same number of aircrafts, the amount of calculation in alternate sortie condition is much less than that in group sortie condition, because only half of these aircrafts (such as Squadron B, whereas Squadron A just lands and parks on deck until the end of sortie operation) experience the whole scheduling process as shown in Figure 4(b). The state features expectation vectors of the expert's demonstrations and the objective learned policy are shown in Figure 8.

Analysis of the experimental results

When the MWAL algorithm is applied to restore an expert's scheduling demonstration, the experimental results show that the learned policy matches the expert's well, and the deviations of the state features expectations are considerably small, which means that the MWAL algorithm could achieve a strong matching between learned policy and expert demonstration. It can be speculated that the MWAL algorithm could deal with some uncertain inherent events like a human expert.

The situation without human expert

In the third section, the MWAL algorithm generating policies based on expert's demonstration was discussed. The results indicate that the MWAL algorithm will achieve a similar performance as the IRL projection algorithm introduced by Ryan[12] when applied to learn from an expert's demonstrations. However, the precondition of this work is that there exists such an expert, and he is supposed to be excellent and experienced enough. In fact, it is very possible that there is no expert available currently or the expert performs not perfect when giving his demonstration so that the performance of learned policy becomes worse. In this section, the MWAL algorithm is adopted to gain policy without an expert's demonstration.

Modification of the original state features

Neu and Szepesvari[20], the original article where the MWAL algorithm was put forward, discussed the case when there is no expert's demonstrations available. As has been described in "MWAL in carrier-borne aircraft scheduling" section, the situation without expert's demonstrations could be treated as that the expectations of the expert's policy state features can be defined as $\hat{\mu}_{\mathrm{E}}^{(i)}=0\,|\,i=1,2,\ldots,m$. Thus, equation (5) changes into $w_{t+1}^{(i)}=w_t^{(i)}\beta^{\hat{\mu}_t^{(i)}}\,|\,i=1,2,\ldots,m$($m$ is the number of state features) in which the weight vector w will evolve with iterating. It should be noted that the types of state features will orient the final output of the MWAL algorithm in some

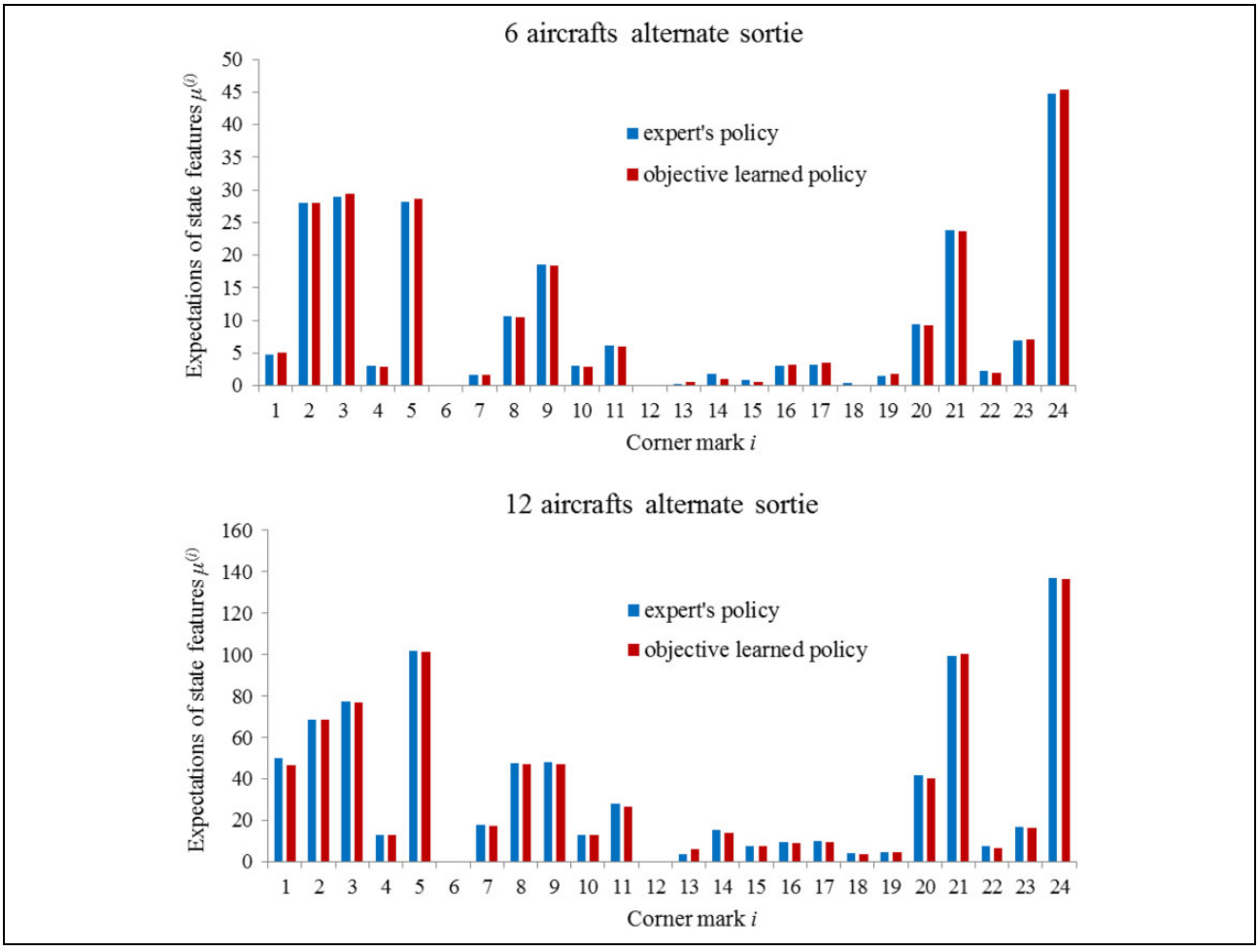

Figure 8. Expectations of the state features of the expert's demonstrations and the objective learned policies in the two simulation experiments in alternate sortie condition.

aspects. In order to gain the policy which could achieve better performance in aspects that we concern, the state features of the MDP can be changed as follows

$$\begin{cases} ff^{(1)} = \dfrac{1000}{Pm_1} \\ ff^{(2)} = n - Pm_2 \\ ff^{(3)} = n - Pm_3 \\ ff^{(4)} = \dfrac{10}{Pm_4} \end{cases} \tag{9}$$

Where n represents the number of aircrafts, while $Pm_1 \sim Pm_4$ represent time of scheduling, number of crashed aircrafts, number of aircrafts in special priority, and average fuel consumption of each aircraft, respectively, as four major indexes used to evaluate scheduling policies. In an operator's view, a good scheduling should take less time, has less crashed aircrafts and less aircrafts in special priority, and the average fuel consumption of each aircraft should be as little as possible. Thus, we define the reward function as a linear combination of features $V(\pi) = E[\sum_{l=1}^{m} \gamma^l R(s_l)|\pi] = w^T \cdot \mu(\pi)$, where, $m = 4$ is the number of new state features. Obviously, the values of these four new state features are expected to become higher but restrict each other, as the robot control problem using Hierarchical AL introduced by Kolter.[22] The searching of objective policy can be treated as an optimization problem which can be solved by cross entropy method efficiently.[29] This optimization problem can be shown as follows

$$\begin{aligned} &\max_{w,\pi} w^T \cdot \mu(\pi) = \max_{t=1}^{\mathrm{T}} \sum_{i=1}^{m} w_i \cdot ff^{(i)}(\pi_t) \\ &\sum_{i=1}^{m} w_i \cdot ff^{(i)}(\pi*) \geq \sum_{i=1}^{m} w_i \cdot ff^{(i)}(\pi_t) \ \forall \pi_t \end{aligned} \tag{10}$$

Through T-time iteration, we can achieve the optimum policy $\pi*$ whose reward function is the maximum among all policies produced by cross entropy method. This iteration is essential to ensuring that the performance of the scheduling policy generated by the MWAL without expert is at least as good as the policy generated by experts' demonstration.

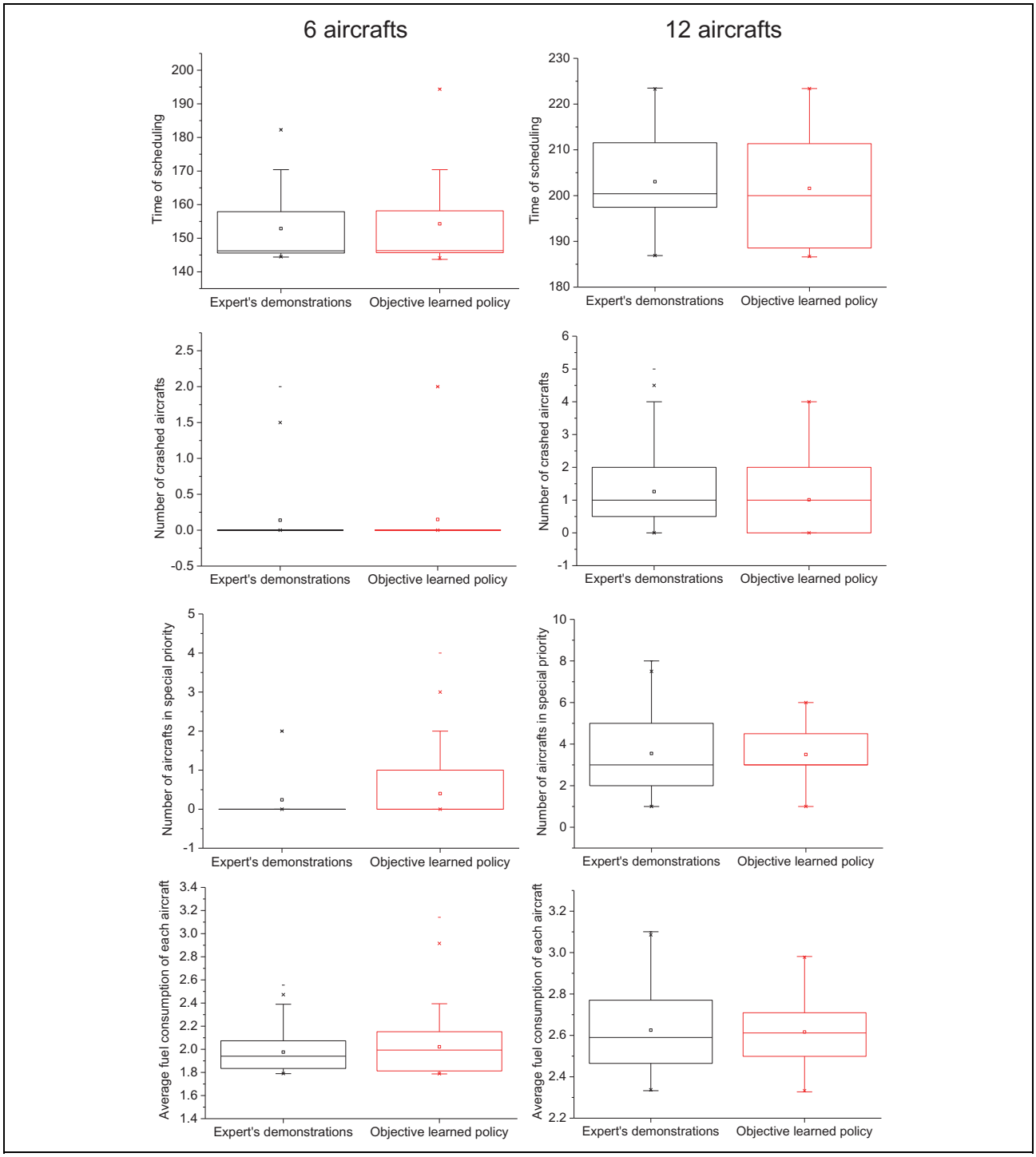

Figure 9. Statistic data of the indexes of the expert's demonstrations and the objective learned policies when scheduling 6 and 12 aircrafts in alternate sortie condition.

Experiments and comparison

The calculation of the mixed policy is the same as equation (8). The experimental conditions are set the same as those in the third section, those are the group sortie condition and the alternate sortie condition. To evaluate the quality of the output scheduling policy, three kinds of experts' operations will be compared. The first kind of expert mainly considers the safety, while the second kind mainly considers the efficiency. They are just like the "safety expert" and "risky expert" in the study by Michini and How.[15]

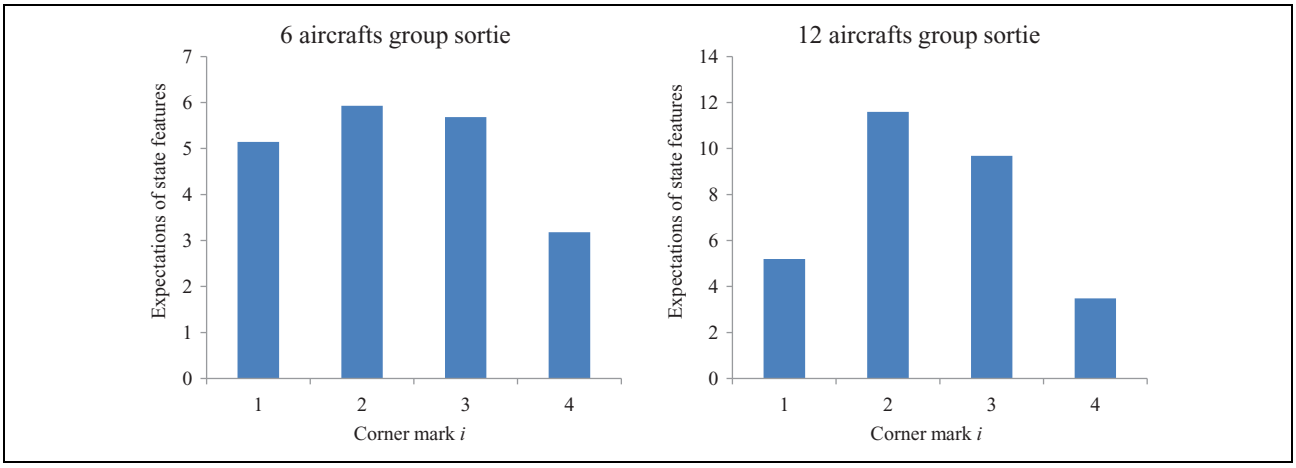

Figure 10. Expectations of state features of the generated policy with no expert in group sortie condition.

Table 3. The comparison between the feature expectations of the experts and the generated scheduling policy.

Number of aircrafts	6				12			
Operator	Expert 1 (safety)	Expert 2 (efficiency)	Expert 3 (balanced)	Generated policy with no expert	Expert 1 (safety)	Expert 2 (efficiency)	Expert 3 (balanced)	Generated policy with no expert
Time of scheduling (min)	214.108	175.799	201.185	194.9626	280.157	203.165	241.633	194.2897
Number of crashed aircrafts	0	1.16	0.15	0.069	2.36	5.51	3.92	0.434
Number of aircrafts in special priority	0.05	2.75	1.03	0.333	3.95	8.57	6.43	2.38
Average fuel consumption of each aircraft (t)	3.481	3.272	3.4523	3.1608	4.537	3.6025	4.1514	2.908

Safety expert: Refueling all aircrafts fully before launching, however, landing aircraft paying more attention to priority aircrafts.

Risky expert: Refueling all aircrafts with minimum fuel level before launching, while, landing aircraft as soon as they arrive in the Marshal stack, paying no attention to priority aircrafts.

The third kind of expert is the same as that in the third section, considering both safety and efficiency. Generally, the last kind of expert is supposed to be the most reasonable one.

In group sortie condition. It takes 12536 and 27424 s for the program to gain the policy for scheduling 6 and 12 aircrafts in group sortie condition, respectively. The expectations of four state features are shown in Figure 10 and it can be seen that the four state features are in the same order of magnitude. For comparison, the average values of the four indexes of 1000 simulations of the generated policy and those of 100 simulations of the experts' demonstrations are listed together in Table 3.

It implies that the MWAL algorithm could generate policy performing quite satisfying in Table 3. When scheduling 6 aircrafts, the generated policy spends slightly less time than the Expert 3, but causes only half number of crashed aircrafts and consumes the least fuel. When scheduling 12 aircrafts, the generated policy performs even more outstanding and exceeds all the three experts in all the four indexes. While the crash rates of these three human experts are relatively high, the generated policy still keeps it at a fairly low level.

This research case shows that in the precondition of the modified state features, the MWAL algorithm is able to generate a policy for group sortie that is both acceptable and feasible, performing even better than some human experts. With the number of the aircrafts increases, the complexity and workload also increase, and superiority of the generated policy is enlarged.

In alternate sortie condition. It takes 11,065 and 32,238 s for the program to generate the policy for scheduling 6 and 12 aircrafts in alternate sortie condition, respectively, and the state features expectation are shown in Figure 11. The comparison between human experts and the generated policy is shown in Table 4.

As discussed in "Experimental results" section, there is more randomness in the alternate sortie condition than that in the group sortie condition. Therefore, the performance of the three experts is a little different than that in the group

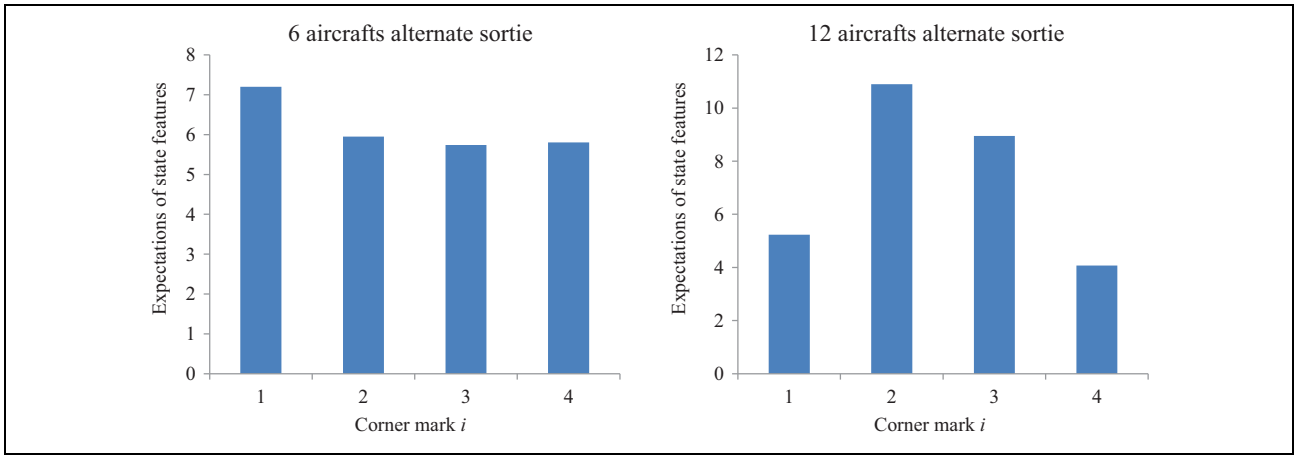

Figure 11. Expectations of state features of the generated policy with no expert in alternate sortie condition.

Table 4. The comparison between the feature expectations of the experts and the generated scheduling policy.

Number of aircrafts	6				12			
Operator	Expert 1 (safety)	Expert 2 (efficiency)	Expert 3 (balanced)	Generated policy with no expert	Expert 1 (safety)	Expert 2 (efficiency)	Expert 3 (balanced)	Generated policy with no expert
Time of scheduling (min)	164.569	142.201	152.874	138.8355	213.939	175.894	203.041	191.986
Number of crashed aircrafts	0.07	0.25	0.14	0.045	1.29	2	1.26	1.128
Number of aircrafts in special priority	0.16	0.79	0.24	0.224	2.41	5.42	3.55	2.979
Average fuel consumption of each aircraft (t)	1.906	2.1105	1.9757	1.7291	2.583	2.5175	2.6253	2.4639

sortie condition. For example, when scheduling 6 aircrafts, the expert 2 spends the least scheduling time but consumes the largest amount of fuel. When scheduling 12 aircrafts, the expert 3 loses nearly as few aircrafts as the expert 2 does but still causes obviously more aircrafts in special priority when arranging the landing operation.

The overall performance of the generated policy with no expert is as good as that in group sortie condition, which has been discussed in "In group sortie condition" section. It keeps the lowest crash rate compared to the three experts and consumes slightly less fuel than any expert. It shows that the generated policy with no expert can still conduct the scheduling well in the alternate sortie condition and the four indexes are even better balanced than human experts.

Analysis of the experimental results

The comparison between three different human experts and the generated policy shows that with the application of the MWAL algorithm, through iterations, can generate a scheduling policy that performs at least as good as these experts' demonstrations. With more aircrafts being scheduled, the generated policy performs even better. Compared with the work of Michini and How[15] and the third section in this article which try to restore a human expert's action, a further step is made that a feasible and robust policy for carrier-borne aircrafts aviation operation scheduling can be generated without learning from an expert. The possible reason can be analyzed as follows:

1. When a decision is making, the habits and preferences of operator are not so explicit enough that the decision is of randomness. In other words, there is no "pure" safety expert or efficiency expert, on the contrary, every human operator is a mixture expert.
2. When random events occur, human experts need more time to understand the situation and make a suitable decision, however, computers could gain policy and conduct scheduling faster than human under the condition of predefining the state features and reward functions.

Conclusion and discussion

1. With the application of the MWAL algorithm, the method for generating optimized policies for carrier-borne aircrafts scheduling was put forward. When there is an expert available, the method can learn from the expert's demonstration to produce

the scheduling which could perform quite close to the expert's demonstrators according to the statistic values of the four indexes. When there is no expert available or the expert's capability is not convincing enough, MWAL algorithm can also generate scheduling policy which performs as good as (or even better than) some experts by modifying the initial state features. In both situations, the output policy shows to be highly acceptable and reliable. This result is a little like what the IRL algorithm achieves in the study by Michini and How.[15]

2. According to the experimental results, it can be implied that if the task of scheduling is relatively easy, human experts can handle the situation well and their demonstrations can be restored by the MWAL algorithm. However, with the increase in the scheduling task complexity, it becomes difficult for human experts to deal with all the random influencing factors. Thus, the generated policy without experts' demonstrations performs even more excellent than humans. It should be noticed that, this algorithm could achieve wonderful policies without any human expert, improving the practicability of this algorithm. Especially in complex scenarios, this algorithm could also generate proper policies, as an obvious difference with the projection method used in the study by Michini and How.[15]
3. The method in this article can be used to construct a data base of optimized scheduling policies. Such a data base can not only guide the training of carrier deck operators, but also improve the working-efficiency of deck operators in complex operation situations. Based on MWAL algorithm, an automated deck operation scheduling tool could be developed, providing an efficient decision-making system for deck operators, reducing failure rate.
4. The computational expense of the MWAL method is relatively less than IRL; however, the iteration time is not less than 3 h. It is difficult to shorten the iteration time of Monte Carlo simulation. If other analytical and numerical methods can be introduced to obtain state features, the MWAL algorithm will run more efficiently. On the other hand, the Compute Unified Device Architecture (CUDA)-enabled NVIDIA GPUs could also accelerate the iteration to some extent. However, the original MATLAB codes should be modified, which is left as an area of future work.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the National Natural Science Foundation of China (NSFC; Grant Nos. 51479155, 51309040 and 51709214), China Postdoctoral Science Foundation (Grant No. 2018M642939), Fundamental Research Funds for the Central Universities (Grant Nos. WUT:2017IVA006 and WUT:2018IVB069), Stable Supporting Fund of Science and Technology on Underwater Vehicle Laboratory (Grant No. SXJQR2018WDKT001), and Open Fund of Key Laboratory of High Performance Ship Technology (Wuhan University of Technology), Ministry of Education (Grant No. gxnc18041404)

ORCID iD

Mao Zheng https://orcid.org/0000-0002-5533-2024
Zaopeng Dong https://orcid.org/0000-0002-8909-8395

References

1. Angelyn J. Sortie generation capacity of embarked airwings. *ADA359178* 1998: 9–95.
2. Angelyn J. USS Nimitz and carrier airwing nine surge demonstration. *Defense Report of Center for Naval Analyses* 1998: 32–34.
3. Feng Q, Zeng SK, and Kang R. A MAS based model for dynamic scheduling of carrier aircraft. *ACTA Aeronaut et Astronaut Sin* 2009; 30(11): 2119–2125.
4. Zheng M, Huang S, Wang C, et al. Research on aircraft sortie generation rate using multi-class closed queueing network. In: *International conference on vehicle & mechanical engineering and information technology*, Beijing, China, 7–9 September 2013, pp. 1864–1867. Trans tech publications.
5. Dietz DC and Jerkins RC. Analysis of aircraft sortie generation with the use of a fork-join queueing network model. *Nav Logist Res* 1997; 2(44): 153–164.
6. Reiser M. A queueing network analysis of computer communication networks with window flow control. *IEEE Trans Commun* 1979; 27(8): 1199–1209.
7. James W and Harris J. The sortie generation rate model. In: *Proceedings of the 2002 winter simulation conference*, San Diego, CA, USA, 8–11 December 2002, pp. 864–868. ACM.
8. Zheng M, Huang S, Zhao YZ, et al. Simulation analysis of carrier-borne aircraft surge operation based on Monte Carlo method. *Comput Simul* 2013; 30(2): 62–65.
9. Abbeel P and Ng A. Apprenticeship learning via inverse reinforcement learning. In: *21st International conference on machine learning (ICML'04)*, Banff, Alberta, Canada, 4–8 July 2004, pp. 1–8. ACM.
10. Li Y, Zhu Y, Yang F, et al. Inverse reinforcement learning based optimal schedule generation approach for carrier aircraft on flight deck. *J Natl Univ Def Technol* 2013; 35(4): 171–175.
11. Wu Y, Sun L, and Qu X. A sequencing model for a team of aircraft landing on the carrier. *Aerosp Sci Technol* 2016; 54: 72–87.
12. Ryan JC, Cummings ML, Roy N, et al. Designing an interactive local and global decision support system for aircraft carrier deck scheduling. In: *AIAA Infotech 2011*, St. Louis, USA, 29–31 March 2011, pp. 27–39. AIAA.

13. Ryan JC. *Assessing the performance of human-automation collaborative planning systems*. Master's Thesis, Massachusetts Institute of Technology, USA, 2011, pp. 45–51.
14. Rosenthal RE and Walsh WJ. Optimizing flight operations for an aircraft carrier in transit. *Oper Res* 1996; 44(2): 305–312.
15. Michini B and How JP. A human-interactive course of action planner for aircraft carrier deck operations. In: *AIAA Infotech 2011*, St. Louis, USA, 29–31 March 2011, pp. 1–11. AIAA.
16. Ryan JC and Cummings ML. Comparing the performance of expert user heuristics and an integer linear program in aircraft carrier deck operations. *IEEE Trans Cybern* 2013; 44(6): 761–773.
17. Ryan JC. Investigation possible effects of UAVs on aircraft carrier deck operations. In: *Human systems integration symposium*, Vienna, Virginia, USA, 25–27 October 2011, pp. 1–12. ASNE.
18. Abbeel P. *Apprenticeship learning and reinforcement learning with application to robotic control*. PhD Thesis, Stanford University, USA, 2008, pp. 11–17.
19. Ratliff ND, Bagnell JA, and Zinkevich MA. Maximum margin planning. In: *23rd International conference on machine learning*, Pittsburgh, Pennsylvania, USA, 25–29 June 2006, pp. 729–739. ACM.
20. Neu G and Szepesvari C. Apprenticeship learning using inverse reinforcement learning and gradient methods. In: *23rd Conference on uncertainty in artificial intelligence*, Vancouver, BC, Canada, 19–22 July 2007, pp. 295–302. DBLP.
21. Ziebart BD, Maas A, Bagnell JA, et al. Maximum entropy inverse reinforcement learning. In: *23rd AAAI Conference on artificial intelligence and the 20th innovative applications of artificial intelligence conference*, Chicago, Illinois, July 13–17 2008, Vol. 3, pp. 1433–1438. AAAI.
22. Kolter JZ, Abbeel P, and Ng AY. Hierarchical apprenticeship learning, with application to quadruped locomotion. In: *21st Annual conference on neural information processing systems*, Vancouver, British Columbia, Canada, 08–10 December 2008, pp. 1–8. Curran Associates Inc.
23. Jin ZJ, Qian H, Chen SY, et al. Survey of apprenticeship learning based on reward function approximating. *J Huazhong Univ of Sci & Tech (Nat Sci Ed)* 2008; 36(1): 288–294.
24. Abdeslam B and Brahim C. Apprenticeship learning with few examples. *Neurocomputing* 2013; 104(6): 83–96.
25. Syed U and Schapire RE. A game-theoretic approach to apprenticeship learning. *Sci China* 2013; 46(5): 381–389.
26. Syed U, Michael B, and Schapire RE. Apprenticeship learning using linear programming. In: *25th International conference on machine learning*, Helsinki, Finland, 5–9 July 2008, pp. 1032–1039. ACM.
27. Norman F. *US Aircraft carriers*. Annapolis, Maryland, USA: Naval Institute Press, 1983, pp. 318–321.
28. Bertsekas DP and Tsitsiklis J. *Neuro-dynamic programming. Athena Scientific*. Athena Scientific Press, 1996, pp. 102–105.
29. Deping Z, Changhai N, and Baowen X. Cross-entropy method based on Markov decision process for optimal software testing. *J Soft* 2008; 19(10): 2770–2779.