Multi-satellite Online scheduling Method Based on Proximal Policy

Xuefei Li
Wuhan University
Jia Chen
Wuhan University
Xiantao Cai (Caixiantao@whu.edu.cn)
Wuhan University
Ningbo Cui
Wuhan University
Shaohua Wan
Zhongnan University of Economics and Law

Research Article

Keywords: multi-satellite online scheduling, Markov decision process, proximal policy optimization

Posted Date: February 25th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1383799/v1

Multi-satellite Online scheduling Method Based on Proximal Policy

XueFei Li[1], Jia Chen[1], XianTao Cai[1*], NingBo Cui [1], Shaohua Wan[2]
[1] School of Computer Science, Wuhan University, Wuhan/300350, China
[2]School of Information and Safety Engineering, Zhongnan University of Economics and Law, Wuhan 430073, China
Correspondence should be addressed to Xiantao Cai, caixiantao@whu.edu.cn

Abstract

Under the background of the rapid development of network communication and satellite technology, multi-satellite online scheduling has brought many problems such as high labour costs, slow response speed, low task execution efficiency and low solution efficiency. How to effectively develop a real-time satellite scheduling scheme that can maximize the scheduling revenue is an important issue. In this paper, the multi-satellite online scheduling process is regarded as a Markov sequential decision process. When the task arrives, the scheduling decision can be made according to the current scheduling situation and the task information. Since the number and resources of satellites are fixed, but the information of the task itself such as the number of available time windows are unpredictable, this problem can be proved as an NP-Hard problem. There is no specific effective solution yet. As reinforcement learning algorithm has achieved good application results in decision-making optimization problems such as optimization decision, automatic control and production line scheduling, it is feasible to apply reinforcement learning method to multi-satellite online scheduling. Therefore, this paper studies the multi-satellite scheduling method based on reinforcement learning. This paper analyses the Markov property of the basic scheduling process of the multi-satellite online scheduling problem and establishes the corresponding Markov decision pro-cess. Considering the influence of the periodic schedule in the online scheduling on the scheduling strategy of the algorithm-decision network, an improved multi-satellite online scheduling model is proposed. Finally, the model is trained by the proximal policy optimization algorithm (PPO), and by comparing the simulation results with the existing multi-satellite online scheduling model, we verified the validity of our model.

Keywords: multi-satellite online scheduling, Markov decision process, proximal policy optimization.

Introduction

Earth observation is a satellite observation activity that uses various types of satellite remote sensors to run in accordance with the established orbit and obtain ground observation target image data and various information. [1] With the rapid development of satellite observation-related technologies, how to efficiently use the satellite resources for scheduling has become one of the main research directions of satellite practical applications. The satellite scheduling problem is generally expressed as follows: In a specified scheduling cycle, the satellite ground control centre first pre-processes the collected user observation tasks to eliminate those do not meet the observation requirements, and then assigns the processed observation tasks to the satellites with available observation time windows for execution, and formulates an effective task observation scheme to obtain the maximum scheduling rewards.

For solving the satellite scheduling problem, scholars have done many related studies. [5],[6],[9],[10],[11] However, the studies focus on various intelligent optimization algorithms and heuristic algorithms for specific satellite scheduling problems. Among them, intelligent optimization algorithms can achieve better results for small-scale satellite scheduling problems, but in large-scale scheduling problems, intelligent optimization algorithms often achieve slow solving speed and poor results. In addition, the parameter setting of intelligent optimization algorithm has a great influence on the algorithm solution. The heuristic algorithm mainly establishes a model for specific scheduling problems. The model can obtain good scheduling results, but it is not easy to migrate and has strong limitations. In addition, in the dynamic emergency environment, with the timeliness requirements of satellite scheduling, the traditional batch processing mode is no longer able to meet the needs of users. It is increasingly important to explore and improve the rapid processing ability of observation tasks of multi-satellite scheduling method according to the increased number of satellites in satellite scheduling problem.

According to the different timeliness requirements of satellite observation tasks, the research on satellite task scheduling can be divided into static scheduling problem, dynamic scheduling problem and online scheduling problem. Compared with online scheduling of satellite tasks, both static and dynamic scheduling of satellite tasks re-quire the satellite control center to collect user observation tasks in advance, which can also be collectively called off-line scheduling of satellite tasks. At present, scholars mostly focus on the off-line scheduling problem of satellite tasks, but the research on online scheduling problem of satellite tasks is relatively less, and the research on multi-satellites online scheduling problem is even less.

Some scholars first studied the processing of emergency tasks in the process of satellite scheduling. Cui et al. [13] constructed an emergency task priority calculation model for seven influencing factors of satellite task scheduling in emergency situations, and proposed a scheduling algorithm based on task priority to reduce the complexity of scheduling scheme adjustment. Niu et al. [14] focusing on the dynamic satellite task scheduling problem for large area natural disaster emergency response and constructed a satellite scheduling model based on the maximization of task revenue and scheduling scheme robustness, and a hybrid optimization algorithm (HA-NSGA-II) is proposed to generate scheduling scheme. However, these studies have only partly solved the rapid handling of small-scale emergency tasks, but cannot cope with the arrival of a large-scale emergency tasks.

With the rise of machine learning, scholars have realized the possibility of real-time task allocation from the model of machine learning[16][17],[19], so some scholars have begun to introduce AI algorithms such as neural networks and reinforcement learning to solve the scheduling problem [[21],[22]. Wang [20] first classified the arrival of the above large-scale emergency tasks as the online scheduling problem of satellite tasks, and analysed the shortcomings of the traditional satellite-ground control mode in solving the online scheduling problem of satellite tasks. According to the characteristics of the online scheduling problem of satellite tasks, the centralized and distributed online scheduling models and algorithms of satellite tasks were proposed. The centralized satellite task online scheduling problem is solved by the reinforcement learning A3C algorithm, and the distributed satellite task online scheduling problem is solved based on the reinforcement learning MADDPG algorithm. However, in the study, the design of environmental state in the centralized task online scheduling model is relatively simple, and the actual influence of periodic schedule on scheduling decision-making strategy is not considered. Moreover, the A3C algorithm is

sensitive to the setting of super parameters, so the algorithm performance is unstable and the solution effect is general.

In this paper, the centralized online scheduling model of satellite tasks is improved. Considering the influence of periodic schedule on the decision-making policy of the algorithm in the online scheduling process of satellite tasks, an improved multi-satellite online scheduling model is constructed to optimize the scheduling solution process and provide more specific environmental parameters. A multi-satellite online scheduling model based on proximal policy algorithm is proposed. Finally, the performance comparison between this model and the existing online scheduling model is carried out through the simulation experiments.

Problem Formulation

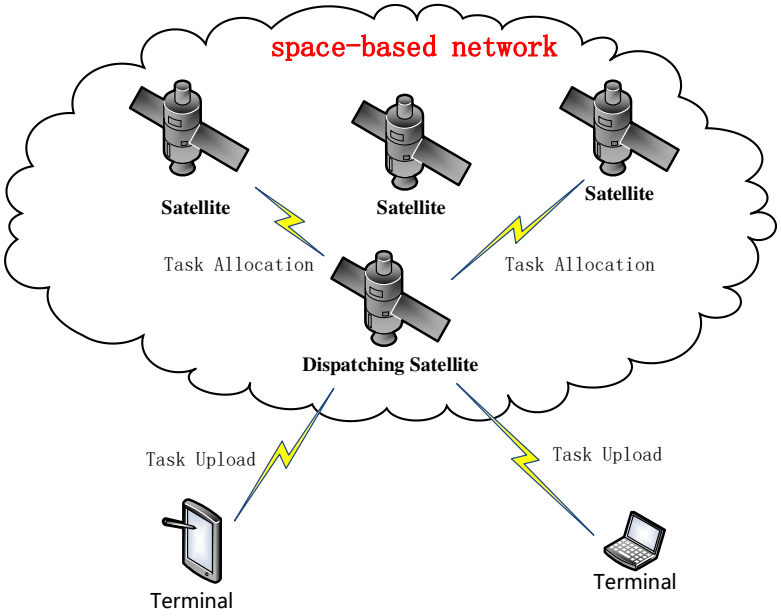

Figure 1: Basic rocket ship design. The rocket ship is propelled with three thrusters and features a single viewing window. The nose cone is detachable upon impact.

The realization of multi-satellite online scheduling is based on the space-based networking technology-an on-satellite space-based network formed by multiple satellites. Therefore, users can use space-based network links to send observation tasks through portable devices such as small-station terminals, without requiring the ground station to wait for the satellite to transit and then make the note of observation instructions like the satellite-ground control mode.

Different from the common satellite receiving information mode (the common satellite cannot receive the user's observation task directly, requires the ground station to carry out measurement and control and instructions on the note), through the space-based earth observation mode, users can directly send observation tasks to the satellite.

The space-based information network receives the request sent by the user and enters the online immediate scheduling process of the satellite. Different from the satellite-ground control process, the time bands and resources of all satellites in the space-based information network are shared. Therefore, it is not necessary to predict the resources. It is only necessary to input the observation task into the multi-satellite online scheduling mode, and allocate resources and make decisions through the online immediate scheduling model of the satellite, so as to realize the immediate response of the task.

The multi-satellite online scheduling problem can be generally defined as: the time of submission of observation tasks is random and independent. The satellite control center processes the observation tasks dynamically arriving over time and gives the corresponding scheduling scheme. It can be seen that the online scheduling problem of satellite tasks is essentially a kind of sequential decision-making problem. If the control center accepts the current arrival of the observation task, it needs to consume the corresponding satellite resources to obtain the corresponding income of the task, which may make the subsequent observation task with greater income cannot be arranged. If the current arrival of the observation task is refused, the decision income of the observation task is set to zero, and the task cannot be scheduled again in this scheduling cycle. Therefore, the scheduling center needs to consider the current observation requirements and satellite resource status to ensure maximum scheduling rewards throughout the scheduling cycle.

According to the dynamic random knapsack theory of Kleywegt et al. [23], multi-satellite online scheduling problem can be described as follows: a series of tasks arrive randomly, each task needs to consume the corresponding resources, and the completion of the task will obtain the corresponding rewards. Before the task arrives, the task resource demand and the corresponding rewards cannot be predicted. Decision makers have fixed resources and can receive or reject tasks. Receiving tasks consume resources to obtain benefits, and refusing tasks may be punished to some extent. The goal of the problem is to find a strategy to maximize the rewards of the task. The corresponding dynamic random knapsack problem is expressed as follows:

$$V_{DKSP}^{\pi} = E\left[\begin{matrix}\sum_{i:A_i<T_{end}} e^{-\alpha A_i}[D_i^{\pi}R_i - (1-D_i^{\pi})p] - \int_0^{T_{end}} e^{-\alpha\tau} c\left(N^{\pi}(\tau)\right)d\tau \\ +e^{-\alpha T_{end}} v\left(N^{\pi}(T)\right)|N^{\pi}(T) = N_0\end{matrix}\right] \quad (1)$$

A_i is the arrival time of the task i, D_i^{π} is the decision for task i, R_i is the reward of task i,$N^{\pi}(t^{+})$ is the remaining resource after time t, $N^{\pi}(t^{-})$is the remaining resource before time t, p is the penalty when the task is refused.

Multi-satellite online scheduling requires immediate decision-making after each task arrives, including three steps: current state acquisition, task decision-making and state update. The decision flow chart is shown in Fig 2.

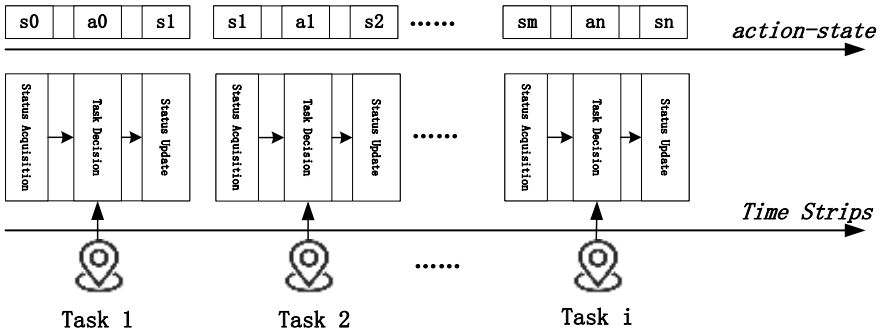

Figure 2: Multi-satellite online scheduling sequential decision.

For each decision, when the task arrives, acquire current environment state sm, then select decision action an according to sm, and execute the decision action to update the decision environment state to sn.

Factor analysis

As can be seen from the above description, when an observation task arrives, the subsequent system state only depends on the decision of the current decision-maker and the current environmental state, independent of previous decision conditions. Therefore, the online scheduling problem of satellite tasks is Markov, and the above satellite scheduling decision process can be expressed as Markov decision process. The state set S, action set A and return function R of the Markov decision process for multi-satellite online scheduling are defined as follows:

i. State set S:

In the existing solution to the online scheduling problem of satellite tasks, the state set S mainly includes the satellite resource state S_{sat}, and the observation task state S_D. Multi-satellite online scheduling requires different scheduling strategies at different stages of the scheduling cycle to ensure maximum task benefit. For example, in the early stage of the scheduling cycle, the probability of rejecting low-income tasks is high to ensure that subsequent possible high-income tasks can be completed. With the scheduling process, the probability of rejecting low-income tasks gradually decreases to ensure the full utilization of satellite resources. Therefore, this paper proposes the state factor of periodic time state S_T to describe the periodic progress when the observation task arrives. The state set S corresponding to the scheduling model includes satellite resource state S_{sat}, observation task state S_D, periodic time state S_T, and scheduling control state S_C.

- Satellite resource state S_{sat}

Defining the remaining storage space of satellite S_j at $t = A_i$ is $Stor_{S_j}(t)$, so at time t, the satellite resource state $S_{sat} = \bigcup_{n=1}^{Ls} Stor_{S_n}(t)$.

- Observation task state S_D

The observation task state mainly includes the storage space consumption and task income of the observation task, then the observation task state of the satellite at time t is $S_D = [Stor_i, W_i]$.

- Periodic time state S_T

The periodic time state S_T mainly represents the position of the arrival time A_i of the observation taskD_i in the whole scheduling cycle, so the value of S_T is between [0,1], define $S_T = A_i/T_e$.

- Scheduling control state S_C

Due to the scheduling process, the periodic time state S_T is infinitely close to 1, and the satellite resource state is infinitely close to 0. At this time, the scheduling decision is mainly affected by the income of the task, which is prone to the problem of low overall task completion due to the continuous rejection of low-income tasks, so a control state is introduced $S_C = \alpha\left(\bigcup_{n=1}^{Ls} Stor_{S_n}(t)\right) / ((T_e - A_i)/T_e)$, where α is a normalized parameter. This state considers both the scheduling progress and the available satellite resources, and considers high-yield tasks when the scheduling time is sufficient. In the case of emergency scheduling time, the scheduling task is prioritized to reduce the impact of income

In summary, in the process of satellite task online scheduling, at the scheduling time t = A_i, the state set S is expressed as follows:

$$S = [S_{sat}, S_D, S_T, S_C] \tag{2}$$

Since the range of each parameter in the state set S is different, in the reinforcement learning algorithm training, in order to make the weight of each parameter equal, it is necessary to normalize each parameter. The normalization formula used in this article is as follows:

$$X_n = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3}$$

The information of the task can only be determined when it arrives, so the maximum and minimum values in the normalization formula cannot be accurately estimated. Therefore, in specific training, the maximum and minimum values are calculated by pre-estimating the value range of the relevant information of the task. For example, in the simulation experiment, a single scheduling cycle node generates 100 random tasks, and the income of the task is generated into a random number of 1 – 20. Therefore, the minimum value is determined to be 1, and the maximum value is determined to be 20.

ii. Action set A

For acceptance or not, action set A includes two actions: receiving observation task and rejecting observation task. The formula is expressed as follow:

$$\pi(s_t, D_i) = \begin{cases} 1 & \text{accept } D_i \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Where $\pi(s_t, D_i)$ is the reinforcement learning decision policy for task D_i.

iii. Return function R

At the scheduling moment t = A_i, after the scheduling center makes a decision on the arriving observation task D_i, if D_i is accepted, the corresponding observation income W_i is obtained, otherwise the reward is 0. The return function can be expressed as follows:

$$R(s_t, \pi(s_t, D_i)) = \begin{cases} W_i & \text{accept } D_i \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

In summary, in multi-satellite online scheduling, the state of the scheduling moment $\mathrm{t} = \mathrm{A_i}$ is defined as S_t, the decision of the scheduling center is $a = \pi(s_t, D_i)$, and the immediate payoff of executing the decision a is $R(s_t, \pi(s_t, D_i))$, then the state-action function of the Markov decision process for multi-satellite online scheduling is defined as follows:

$$Q^{\pi}(s_t, a) = R(s_t, a) + \gamma \sum_{s' \in S} P(s_t, a, s') V^{\pi}(s') \tag{6}$$

Where $P(s_t, a, s')$ is the state transition function of the multi-satellite online scheduling problem and γ is the discount factor.

Constraint of problem

In the actual scheduling, the multi-satellite online scheduling will be constrained by multiple conditions such as satellite resources, observation tasks, and satellite time windows. The main constraints involved are as follows:

i. Satellite uniqueness observation constraint

In multi-satellite online scheduling, the same observation task is performed by a satellite at most once without redundant observations in order to use satellite resources efficiently. Then the following formula needs to be satisfied:

$$\sum_{i=1}^{L_D} \sum_{n=1}^{L_S} x_i^n \leq 1 \tag{7}$$

ii. Observation constraints of continuous tasks

The same observation satellite continuously observes two different ground targets, and the satellite remote sensor attitude needs to be adjusted to align with the next observation target between observations. Then, for two consecutive observation tasks D_i and D_j of the same satellite, if their observation time intervals are $[st_i, et_i]$ and $[st_j, et_j]$, the following formula need to be satisfied:

$$et_i + T_{trans} \leq st_j \tag{8}$$

T_{trans} represents the time of the posture adjustment.

iii. Satellite time window constraint

If the observation task D_i is scheduled to execute in the time window of satellite S_j. The execution time interval of observation task is $[st^i, et^i]$, and the available time window set of satellite S_j for observation task D_i is $twSet_i^j$, which needs to satisfy the following formula:

$$[st^i, et^i] \cap twSet_i^j = [st^i, et^i], [st^i, et^i] \neq \emptyset \tag{9}$$

iv. Satellite storage capacity constraint

For any earth observation satellite S_j with storage capacity of $SatS_j$, if it accepts m observation tasks throughout the scheduling cycle, the storage space required is $\{Stor_1, Stor_2 \ldots \ldots Stor_m\}$, then the following formulas need to be satisfied:

$$\sum_{1 \leq i \leq m} Stor_i \leq SatS_j \tag{10}$$

v. Observation task arrival time constraint

In the multi-satellite online scheduling, if the arrival time of the observation task D_i is A_i, then A_i must be in the scheduling cycle, and the task D_i can be scheduled to perform. The formula is expressed as:

$$T_s \leq A_i \leq T_e \tag{11}$$

Objective Function

The goal of multi-satellite online scheduling is to maximize the task scheduling rewards at the end of the scheduling cycle after making decisions for each arriving observation task. The objective function can therefore be defined as follows:

$$max: \sum_{i=1}^{L_D} R(s_t, \pi(s_t, D_i)) \tag{12}$$

Model establishment

decision-making process

Since there are many and complex factors affecting the multi-satellite online scheduling, it is difficult to establish an accurate mathematical model for solving the problem. Whether the neural network accepts the current observation task needs to consider four state factors, namely, the observation satellite resource state S_{sat}, the observation task state S_D, the periodic time state S_T, and scheduling control state S_C. Many state factors contain multidimensional

information, and some state ranges are continuous intervals, which cannot be characterized by the look-up table (Q-Table). The neural network can well characterize the continuous state, and can well fit the mapping relationship between action decision and state. Therefore, this paper uses the model-free reinforcement learning algorithm to solve the problem, and uses the multilayer neural network to train the algorithm model.

According to the multi-satellite online scheduling process and constraints described, the multi-satellite online scheduling decision flow chart of satellite tasks can be drawn as follows:

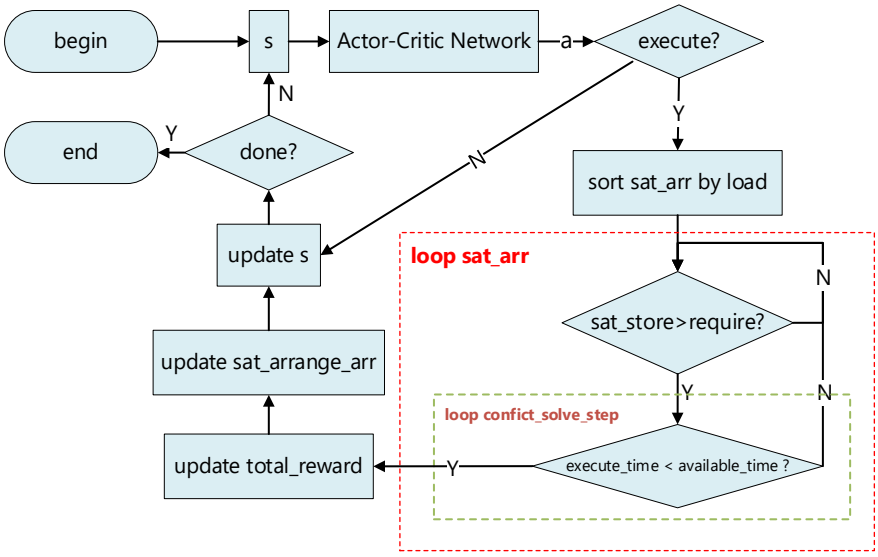

Figure 3: Multi-satellite online scheduling process.

i. Task data pre-processing

When the observation task arrives, it is necessary to pre-process the observation task and calculate the time window of each satellite for the observation task. Only when there is an available time window and the relevant constraints are met, the observation task can enter the next process, otherwise the observation task is directly rejected. In this process, the state set s is generated when the first task is executed, and the current state set is updated continuously before each task is executed.

ii. AC Network Decisions

The decision network makes a decision on the observation task according to the observation task state, the satellite resource state and the periodic time state. If accepted, the next process is performed, otherwise the processing of a task is initiated.

iii. Task execution time window arrangement

After receiving observation tasks, the decision-making network arranges observation satellites and time windows according to the following heuristic rules because of the high timeliness of observation tasks scheduled online by satellite tasks:

Observation satellites: the satellite resources with the highest remaining storage space ratio are preferred; defining satellite payload to describe satellite residual space ratio:

$$Load_l = \frac{t\_free_l}{T_l} * \frac{Stor_l}{SatS_l} \tag{13}$$

Where t_free_l denotes the available time window strip length of satellite l and T_l denotes the total time window strip length of satellite l.

Time window: Select the first observation window of the observation satellite.

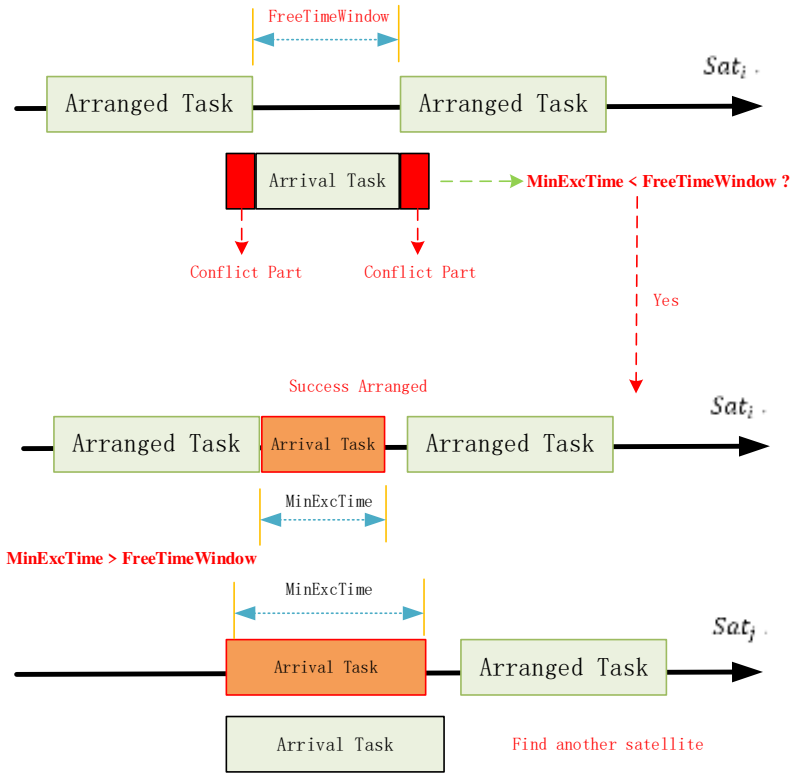

Figure 4: Time window arrangement.

For a certain observation task, due to the existence of multiple satellites, and for each satellite, it may pass through the observation point many times in a cycle, so the time window of the observation task may be multiple, and multiple observation tasks may be arranged on each satellite, so there is a time window conflict problem. For example, in the time window arrangement diagram, when a time window of the task conflicts with the time window of the scheduled task of the satellite Sat_i, the maximum available time window FreeTimeWindow can be obtained by interception. When the time window meets the minimum execution time MinExcTime of the task, the task can be scheduled to the earliest start time of FreeTimeWindows to reduce the fragmentation of the time window, if the minimum execution time is not satisfied, the other time windows of the selected task are compared. If the time windows that meet the conditions are still unable to be found, the available time windows of other satellites are searched. The above steps are repeated until a satisfied time window is found (the analysis here does not consider the transformation time in order to visually display the time window conflict solution).

Specific conflict resolutions are as follows:

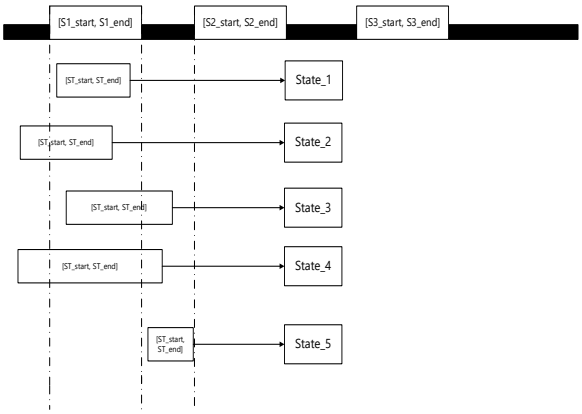

Figure 5: Time window conflict.

As shown in Figure 5, suppose a satellite strip has an allocated time window [S1_start, S1_end], [S2_start, S2_end], ...

When an observation task arrives, one of its observation time windows of the satellite is [ST_start, ST_end]. Then, there are five states. If it is at State_1, the observation task time window cannot be allocated. At State_2, the observation time window intercepts the free zone on the satellite strip, and the new observation time window is [ST_start, S1_start]. At State_3, the new observation time window becomes [S1_end, ST_end]. At State_4, the new observation time window becomes [S1_start, S1_start] and [S1_end, ST_end]. At State_5, the observation time window remains unchanged. Of course, there may still be conflicts with the newly generated observation time window, so it is necessary to continue to solve the subsequent conflicts. But the first half of the observation window will be no subsequent conflict.

Satellite Online Scheduling Network Based on PPO

At present, some mainstream model-free reinforcement learning algorithms, such as DQN algorithm, A3C algorithm, DDPG algorithm and D4PG algorithm, adopt the method of experience playback, and use random sampling method during each training, so that each 'experience' is independent, which satisfies the decision-making idea of Markov decision. However, since each sampling is selected in the experience pool, the existence of the experience pool consumes a lot of memory, and there are shortcomings such as hyperparameter sensitivity and algorithm performance instability. Among them, DQN[26] is only suitable for limited states, and the number of states for on-line immediate scheduling of satellites is often innumerable, and Q-learning cannot solve many simple problems, and the algorithm is even more difficult to understand [28-32]. The vanilla policy gradient algorithm has poor data utilization and poor robustness. Due to the same trajectory for multi-step optimization, it brings instability, and the TRPO algorithm has the problem that the hyperparameters are difficult to select. The proximal policy optimization algorithm-PPO (Proximal Policy Optimization) [25] uses the importance sampling method. The data samples generated by the same strategy are updated for the evaluation policy. The data samples are discarded and then sampled, so their consumption of memory is far less than that of the above algorithm. The data utilization rate is high, the model reliability is high, and the applicability is wide, which meets the actual needs of satellite online scheduling. And the stability of the algorithm is guaranteed by sampling the importance probability.

Therefore, this paper uses the PPO proximal policy optimization algorithm to solve the multi-satellite online scheduling problem, which largely makes up for the defects of the above reinforcement learning algorithm and performs well in tasks such as continuous control.

Table 1: PPO Algorithm.

PPO Algorithm：

for iteration = 1, 2, … do
for actor = 1, 2, …, N do
Run policy $\pi_{\theta_{old}}$ in environment for T timesteps
Compute advantage estimates $\hat{A}_1, \hat{A}_2, \ldots, \hat{A}_T$
end for
Optimize surrogate L wrt θ, with K epochs and minibatch size M $\leq$ NT
$\theta \rightarrow \theta_{old}$
end for

Two methods are proposed to limit the step size for each update in PPO papers. The first is to limit the importance sampling probability $\mathrm{r}(\theta) = \frac{\tilde{\pi}_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)}$ by setting a penalty hyperparameter β, so $\mathrm{r}(\theta_{old}) = 1$.

$$\mathrm{L}^{\mathrm{KLPEN}}(\theta) = \hat{\mathrm{E}}_\mathrm{t}\left[\frac{\tilde{\pi}_\theta(\mathrm{a_t}|\mathrm{s_t})}{\pi_{\mathrm{old}}(\mathrm{a_t}|\mathrm{s_t})}\hat{\mathrm{A}}_\mathrm{t} - \beta\mathrm{KL}\left[\pi_{\theta_{\mathrm{old}}}(\cdot\,|\mathrm{s}), \tilde{\pi}(\cdot\,|\mathrm{s})\right]\right] \tag{14}$$

Enter the same state, the probability distribution of the network can't be too different, in order to get similar action, the use of KL divergence as a limit, but the penalty hyperparameter β is difficult to set, so the PPO through the following rules adaptive selection β:

$$\mathrm{d} = \hat{E}_t\left[KL\left[\pi_{\theta_{old}}(\cdot\,|s), \tilde{\pi}(\cdot\,|s)\right]\right] \tag{15}$$

$$\beta = \begin{cases} \beta/2 & d < d_{targ}/1.5 \\ 2\beta & d > d_{targ} * 1.5 \end{cases} \tag{16}$$

The above algorithm approximately solves the KL constrained update such as TRPO, but penalizes the KL deviation in the objective function rather than makes it a hard constraint, and automatically adjusts the penalty coefficient in the training process to scale properly. Therefore, this is an improvement of TRPO algorithm, and through the first-order SGD method to optimize, so the speed is faster.

According to PPO, the second method can achieve better results. It is also through the variation of importance sampling probability.

$$L^{CPI}(\theta) = \hat{E}_t\left[\frac{\tilde{\pi}_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)}\hat{A}_t\right] \tag{17}$$

$L^{CPI}(\theta)$ is regarded as a conservative policy iteration method. If there is no constraint, $L^{CPI}(\theta)$ will lead to a huge policy update. Therefore, the first method introduces KL divergence to limit

it, and PPO finds another stage method. By Clip, the probability ratio is limited, so that the update strategy will not have too radical changes.

$$L^{CLIP}(\theta) = \hat{E}_t\left[\min\left(\mathrm{r}(\theta)\hat{A}_t, clip(\mathrm{r}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \tag{18}$$

Assuming that the current state-action pair advantage is positive, then in this case by Clip:

$$L^{CLIP}(\theta) = \hat{E}_t\left[\min\left(\mathrm{r}(\theta), 1+\epsilon\right)\hat{A}_t\right)\right] \tag{19}$$

Since the advantage is positive, if the possibility of action is greater, that is $\tilde{\pi}_\theta(a_t|s_t)$ increases, the goal will also increase. However, the total income caused by too large step size does not necessarily increase all the time, so it is necessary to limit the update step size. Therefore, the upper limit is $(1+\epsilon)\hat{A}_t$, which ensures that the new policy will not benefit from the old strategy. The same is true of negative advantages.

PPO-Clip pseudocode is as follows:

Table 2: PPO Algorithm.

PPO-Clip：

1: Input: initial policy parameters θ_0, initial value function parameters φ_0

2: for k = 0, 1, 2, … do

3: Collect set of trajectories $D_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.

4: Compute rewards-to-go $\hat{R}_t$,

5: Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{φ_k} :

$$\hat{A}_t(\pi_\theta) = -V(s_t) + r_t + \gamma\, r_{t+1} + \cdots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t}\, V(s_T)$$

6: Update the policy by maximizing the PPO-Clip objective:

$$\theta_{k+1} = \arg\max_{\varphi} \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^{T} \left(\frac{\tilde{\pi}_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)} \hat{A}_t - \beta KL\left[\pi_{\theta_{old}}(\cdot|s), \tilde{\pi}(\cdot|s)\right]\right)$$

Typically via stochastic gradient ascent with Adam.

7: Fit value function by regression on mean-squared error:

$$\varphi_k = \arg\min_{\varphi} \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^{T} \left(V_\varphi(s_t) - \hat{R}_t\right)^2$$

Typically via some gradient descent algorithm.

8: end for

However, experiments show that in multi-satellite scheduling problems, the model generated by ppo training with kl divergence as a constraint is more stable and better than the model generated by ppo-clip training, which will be verified by experiments below.

model training

The decision network of PPO algorithm uses Actor-Critic architecture, in which the strategy function Actor and the value function Critic are expressed by neural network.

The network model is built as a strategy network and a value network. The input S _t of the neural network is the state set of the system, including four state factors: the observation satellite resource state S_{sat}, the observation task state S_D, the periodic time state S_T, and scheduling control state S_C. The output of the network is the decision action D_i of the system,

that is, the probability value of receiving the observation task. Based on this probability value, this paper determines whether to accept the observation task. T is the scheduling cycle time axis. The hidden layer, hidden element number and activation function of each network are as follows:

Table 3: Parameters of actor-network.

layer	number of neurons	activation
FC_1	40	tanh
FC_2	20	tanh
FC_3	10	tanh
FC_4	1	tanh

Table 4: Parameters of critic-network.

layer	number of neurons	activation
FC_1	40	tanh
FC_2	14	tanh
FC_3	5	tanh
FC_4	1	∅

Note：obs_dim is the dimension of the input state，act_dim is the dimension of the input action. Strategy network learning rate set to 0.000045, evaluation network learning rate set to 1/1400. Activation function tanh：$\tanh x = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$.

In the multi-satellite online scheduling problem, the dimension of the output action should be 2, which represents the two states of acceptance and rejection. However, in fact, there is no need for two dimensions, only one dimension is needed to output the value between 0 and 1, and only the probability of executing the task is judged. When the training is sufficient, the output of the accepted action is infinitely close to 1, and the output of the rejected action is infinitely close to 0. As the probability of exploration, when the probability of action comes, a random number between 0 and 1 is output, and when the probability of action is higher than this random number, it is executed.

Other network training parameters are as follows:

Table 5: Parameters of training.

Parameters	description	value
gamma	Discount factor	0.995
lam	Lambda for Generalized Advantage Estimation	0.98
clip_ratio	Hyperparameter for clipping in the policy objective	0.2
β	Hyperparameter for kl penalty	1.0
pi_lr	Learning rate for policy optimizer	4.5e-5
vf_lr	Learning rate for value function optimizer	7e-4
total_steps	maximum training steps	8e4
evalute_steps	the step interval between two consecutive evaluations	5e2

Network updating process

In the training process of decision-making network for online scheduling of satellite tasks, in each training cycle (Episode), the agent makes decisions through the current state (observation satellite resource state, observation task state and periodic time state), selects actions (the probability of receiving observation tasks), and stores the obtained data samples into the experience pool after performing the actions, which is called step. The training will repeat the above process until the end of the round, the termination criteria are generally the end of the time cycle or the depletion of satellite resources.

In the process of satellite online immediate scheduling, the processing of an observation demand is set as a step, step. According to the step of strategy update, there are usually three types of model-free strategy update methods: Monte Carlo method, time series difference method, and n-steps Bootstrapping method.

The Monte Carlo method [33] only learns after each training completes the whole process, while the time-series difference method learns to update after each step, and the n-steps Bootstrapping method updates after n steps. From the perspective of satellite scheduling process, for each observation task, it is not fixed to reject the current observation task for subsequent income acquisition. Even if the income obtained by accepting the current step may be high, it may lead to higher income tasks that cannot be accepted due to the influence of constraints, resulting in lower final income value. Therefore, it is not appropriate to use the timing difference method, and the Monte Carlo method needs to be updated after the entire scheduling cycle. For the online scheduling problem, the observation task may be infinite, so the Monte Carlo method is not appropriate, and the use of n-steps Bootstrapping for multi-satellite online scheduling model is just consistent.

In order to simulate the continuous scheduling process, the training stipulates that each 100 tasks are a scheduling cycle node, and each 100 tasks calculates the income of the node and collects the data of the node. The income reset continues to train, and each 3 nodes integrate the data and update the model.

The training process is as follows:

Enter the current environmental state s to the policy network, including the observation satellite resource state S_{sat}, observation task state S_D, periodic time state S_T, scheduling control state S_C and the current node completes the identification. The strategy network obtains the environmental information, and obtains two values, namely, the mean mu and the variance sigma, and constructs the normal distribution of these two values. Through this normal distribution, an action is obtained. According to the action input into the environment, the reward and the next state s_ can be obtained, and s_ is input into the policy network, and cycle step 1.

Step 1. The policy network runs 3 episodes to collect the [r, s, a] obtained by each step of training, that is, the income, the environmental state, the action data information, and the next environmental state s_ obtained by each action. In this process, the strategy network is not updated, so this strategy network is also called the old network (Actor-old).

Step 2. All the s-combination data obtained in the second step are input into the Critic network to obtain the V value of the state, which is calculated according to the formula:

$$\hat{A}_t(\pi_\theta) = -V(s_t) + r_t + \gamma r_{t+1} + \cdots + \gamma^{T-t+1} r_{T-1} + \gamma^{T-t} V(s_T) \quad (20)$$

Step 3. Find the loss function loss of $\text{loss} = \text{mean}(\text{square}(\widehat{A}_t))$ and update the Critic network by back propagation.

Step 4. All s-combination data are input into the new policy network and the old policy network, and two normal distributions Normal1 and Normal2 are obtained. All a-combination data are input into the normal distributions Normal1 and Normal2, and prob1 and prob2 are calculated, and the ratio is obtained by dividing network is updated according to the formula:

$$\theta_{k+1} = \arg\max_{\varphi} \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^{T} \left(\frac{\tilde{\pi}_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)} \hat{A}_t - \beta KL\left[\pi_{\theta_{old}}(\cdot|s), \tilde{\pi}(\cdot|s)\right] \right) \quad (21)$$

Loop this step, update the old network with new policy network.

Step 5. Repeat the above steps.

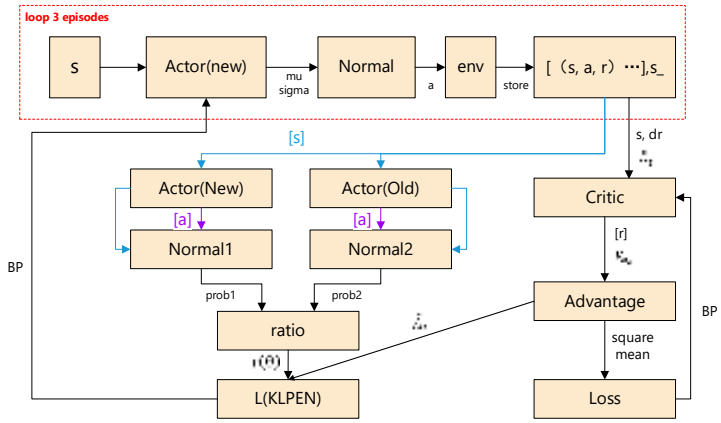

Figure 6: Network updating Algorithm (KLPEN).

Simulation and Result Analysis

In the simulation experiment of this paper, five actually on-orbit Earth observation satellites are selected, namely Spot5, Ikonos-2, Orbview1, Gaofen1 and Yaogan16A. Each satellite has different orbits, and its detailed orbit data can be directly imported from the AGI Server provided by the satellite tool software STK. The simulation experiment scheduling cycle is set to 24 h. Finally, the STK software is used to simulate the satellite task scheduling and calculate the available observation time window of each satellite for the observation task.

Parameter Setting

Table 6: Simulation Experimental Parameters of Multi - Star Online Scheduling.

Parameter	Value
S_n	5
Observation area	Longitude: [20° N, 40° N] latitude: [90° E, 120° E]
Task number	100
R_i	5G

$Store_l$	40G
Simulation period	Start: 10 Mar 2021 00:00:00.000 end: 11 Mar 2021 00:00:00.000
Reward for per task	Random between 10~20
inclination angle of ground	50

Case Study

In order to verify that the online scheduling model can effectively learn the online scheduling strategy of satellite tasks, in this section, we generate 5000 scheduling cycles to simulate the 5000-day online scheduling process of satellite tasks. The length of each scheduling cycle is 24h, and the total number of satellite tasks in each scheduling cycle is $L_D = 100$. The latitude and longitude range of the observation task, the shortest observation time of the task and the priority of the task execution are set according to the table above. The performance of the algorithm in the online learning phase of satellite task scheduling is evaluated by training indicators such as total task scheduling revenue, average task scheduling revenue and total task completion. The training results are as follows:

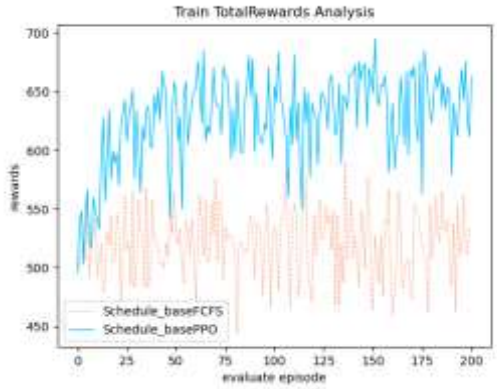

Figure 7: Total reward training.

In the above graph, the abscissa is the number of evaluation cycles, the ordinate is the total revenue of the observation task scheduling, the red dotted line is the total revenue of the FCFS algorithm, and the blue curve is the total revenue of the algorithm. It can be seen that in the online training process based on PPO algorithm, after the previous environmental exploration, the total income of the task is continuously improved and finally converged, which obviously exceeds the income value of FCFS algorithm, which proves that the algorithm can effectively learn the online scheduling strategy.

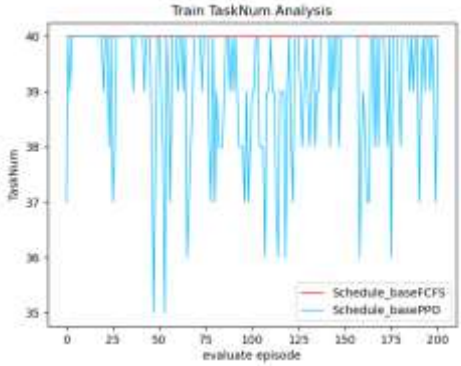

Figure 8: Number of observation tasks executed.

In the above figure, the abscissa is the number of scheduling cycles, the ordinate is the number of completed observation tasks, the red curve is the number of completed tasks of the first-come-first-serve algorithm (FCFS), and the blue curve is the number of completed tasks of the algorithm. In the early stage, the number of tasks completed by the algorithm in this paper is relatively small. At this time, because the algorithm itself is in a state of exploration, without any learning experience, it is infinitely close to the random algorithm, that is, the decision of the task itself is in a state of random decision, so the number of tasks completed is relatively small. However, with the continuous training, the algorithm selects tasks according to the current state. In order to maximize the benefit, the selection of tasks depends on the previous experience, so the number of tasks begins to rise, but does not reach the maximum (here is 40, limited by satellite resources). The first-come-first-serve algorithm is always the maximum number of tasks for any task as long as the conditions are satisfied.

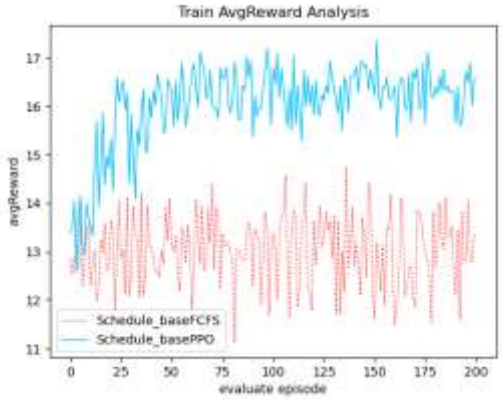

Figure 9: Average reward of observation task.

In Fig.4 above, the abscissa is the number of scheduling cycles, the ordinate is the average reward of observation tasks, the grey curve is the average income of observation tasks of FCFS, and the green curve is the average income of observation tasks of the algorithm in this paper. Since the task priority in this paper is randomly selected in the range of [1, 20], the average reward of the FCFS algorithm is always around 13, while the average reward of the PPO algorithm is continuously improved and finally converges to about 16.5 after previous environmental exploration, which is 27 % higher than that of the FCFS algorithm.

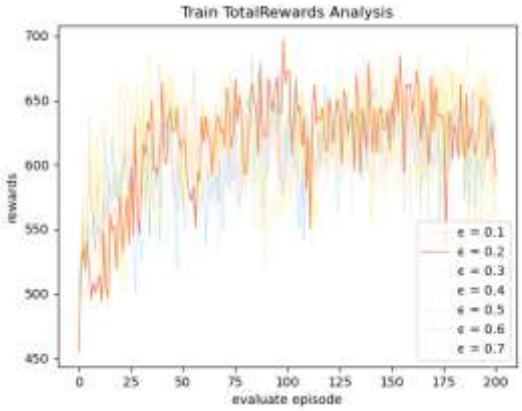

Figure 10: TotalRewards under different clip-ϵ.

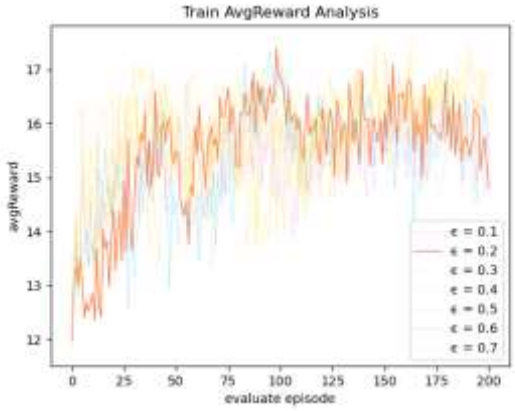

Figure 11: AvgRewards under different clip-ϵ.

Experiments show that under the conditions specified in this paper, in the training of satellite online scheduling model, the larger the ϵ is, the more unstable it is. The smaller the ϵ is, the less the training benefit is. In the later period, it is easier to fall into local optimal solution. Under the conditions of this paper, setting $\epsilon = 0.2$ can achieve the best effect.

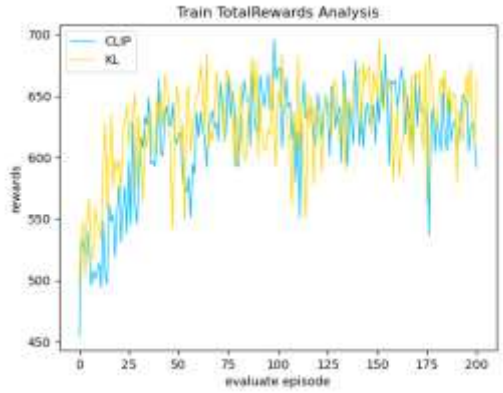

Figure 12: TotalRewards under kl and Clip.

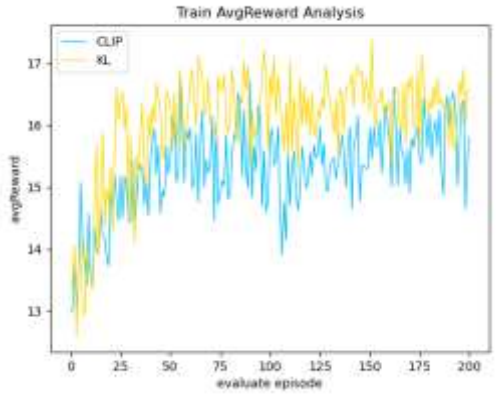

Figure 13: AvgRewards under kl and Clip.

Here, the intercept parameter of clip $\epsilon = 0.2$. From the total rewards figure (left) and the average reward figure (right), the effect of the model trained by kl penalty method is more stable and higher than that of clip method.

Adaptive result analysis of the model

It can be seen from the previous section that the satellite task online scheduling algorithm proposed in this paper can effectively learn the effective scheduling strategy in training, steadily improve the scheduling income of the observation task and eventually converge. However, in the actual satellite task scheduling, the number of observation tasks in the scheduling period is often uncertain. Therefore, we simulate this uncertainty by changing the total number of observation tasks in the scheduling period. The specific setting is: the total number of satellite tasks in each scheduling period is L_D, the first 1000 scheduling nodes $L_D = 50$, 1000 ~ 2000 scheduling nodes $L_D = 100$, the subsequent scheduling nodes $L_D = 50$, and the remaining parameters remain unchanged. Through this simulation experiment, the environmental adaptability of the satellite task online scheduling model is verified. The experimental results are as follows:

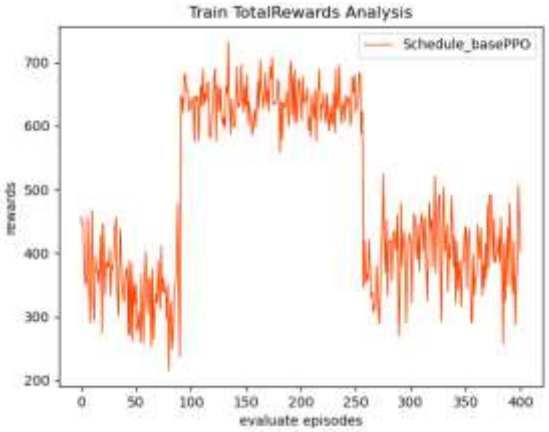

Figure 14: Adaptive testing-total rewards($\boldsymbol{L_D = 50{\sim}100{\sim}50}$).

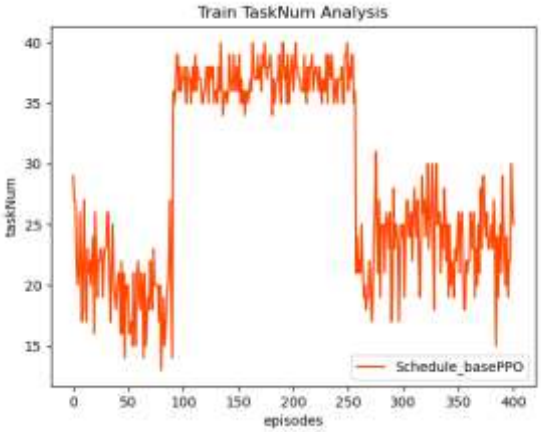

Figure 15: Adaptive executed tasks($\boldsymbol{L_D = 50{\sim}100{\sim}50}$).

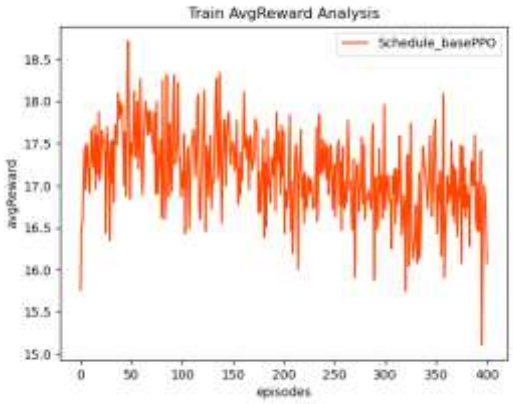

Figure 16: Adaptive testing-average reward($\boldsymbol{L_D = 50{\sim}100{\sim}50}$).

Through the analysis of the above Figures, it can be seen that in the model training of the first 1000 scheduling cycles ($\boldsymbol{L_D = 50}$), the profit after training is stable at about 350, and it is trained to the 1001 scheduling cycle ($\boldsymbol{L_D = 100}$). Due to the increase of the number of observation tasks in the cycle, the PPO algorithm gradually adjusts the decision strategy according to the change of the scheduling scene, and the total profit of the observation task is increased again, and gradually converges to a larger profit value. When the follow-up scheduling scene continues to control in 50 tasks, the algorithm is also constantly learning in the previous adaptive test, so the convergence is fast and stable. It is stable at about 400, and during the whole scheduling process, the average income of the task is relatively stable without much fluctuation. Therefore, it can be seen that the multi-satellite online scheduling model based on PPO algorithm in this paper has scene adaptive ability.

Training comparison

DQN algorithm [26] belongs to the off-policy category, which adopts the method of empirical playback. Each training adopts the random sampling method, so that each "experience" is independent. However, DQN is only suitable for limited states, and the number of states of satellite online immediate scheduling is often innumerable. However, in order to facilitate comparison, the number of tasks defined in each scheduling cycle is only 100, and the

constraints are simplified, so DQN algorithm can also be applied. This comparison is to compare the efficiency of training, the comparison is as follows:

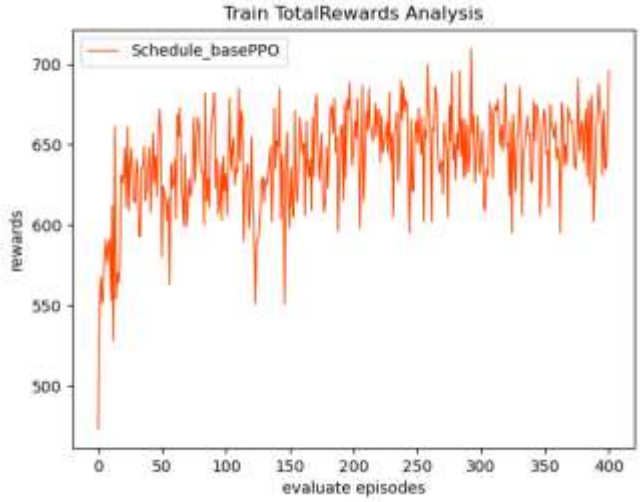

Figure 17: PPO algorithm training (Kl penalty), 11 minutes 58 seconds.

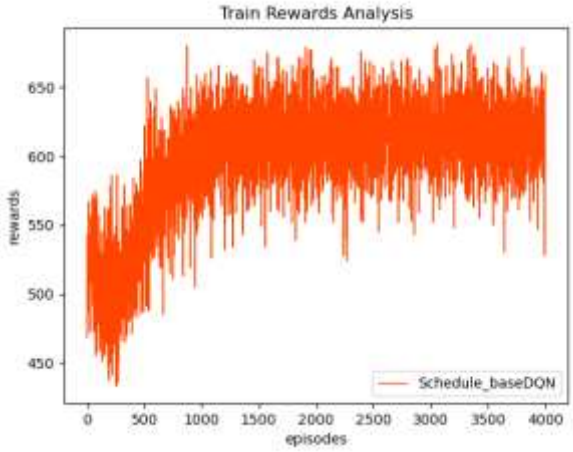

Figure 18: DQN algorithm training, 18 minutes 28 seconds.

It can be seen from the figure that when the number 1000 is used, the training benefit of DQN is convergent, that is, the training is convergent in the 4th–5th minutes, and the PPO algorithm reaches the convergence state in the 80th evaluation (once every 500 steps), that is, it reaches the convergence in 2 minutes and 30 seconds. Therefore, it is easy to obtain that the learning efficiency of PPO is much higher than that of DQN, and the training speed is also much faster than that of DQN. Since the sampling is discarded and re-sampling is completed, the memory consumption is also lower than that of DQN algorithm.

A3C (Asynchronous Advantage Actor-Critic) [27] algorithm uses the Actor-Critic architecture, which is an on-policy algorithm. It does not use empirical playback or sampling methods. It maintains a master node and multiple child nodes, each node has a learning program, each child node trains its own, does not interfere with each other, and then sends the trained parameters to the master node. After the master node is responsible for integration, it is sent to each child

node to continue training. This multi-threaded approach greatly improves the update rate, but it does not take the way of sampling, that is, only from the sub-network learning decision-making experience is also doomed to its strategy is not as good as off-policy algorithm and PPO sampling algorithm. The training results are as follows:

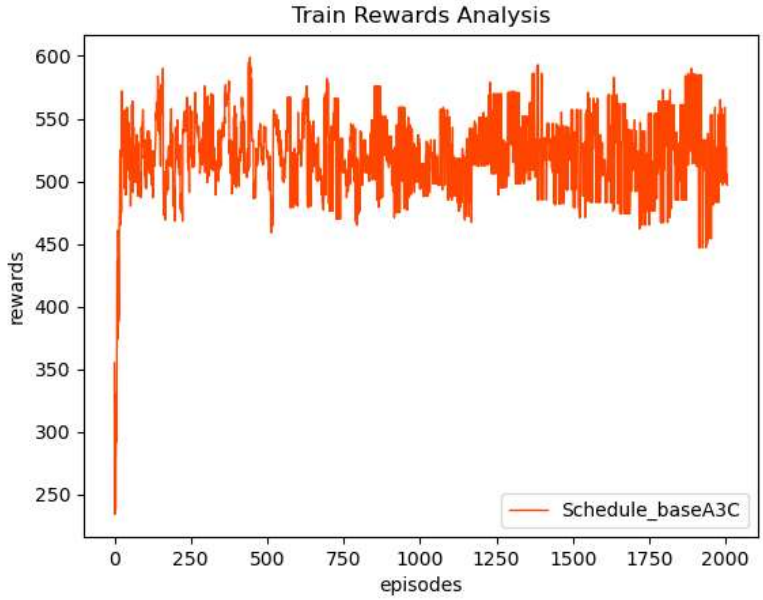

Figure 19: A3C algorithm training, 2 minutes 4 seconds.

The training speed of A3C algorithm is greatly accelerated by multithreading operation. However, in terms of the results, the training effect of A3C algorithm is far inferior to that of DQN algorithm and PPO algorithm. In the training of scheduling model in this paper, the model trained by A3C algorithm has no obvious improvement compared with that of FCFS algorithm.

Conclusion

Firstly, this paper studies and analyzes the multi-satellite online scheduling problem. The basic scheduling process of satellite task online scheduling problem is analyzed and the corresponding Markov decision process is established. Considering the influence of periodic schedule on online scheduling strategy in online scheduling, the state factor of periodic time state is added to establish the corresponding satellite task online scheduling model. By analyzing the advantages and disadvantages of different reinforcement learning algorithms, this paper selects the proximal strategy algorithm as the training algorithm of the model in this paper. Through simulation experiments, the multi-satellite online scheduling model in this paper is evaluated, and the self-learning ability and adaptability of the model are proved. Through the adjustment of different truncation parameters, the rationality of the parameter selection of the model in this paper is proved. The training of the model in this paper by different reinforcement learning training algorithms proves that the proximal strategy optimization algorithm has achieved the greatest improvement in the training of the model in this paper.

Acknowledgements

The publication of this research has been supported by National Natural Science Foundation of China, under grant No. 62172438.

Funding

Not applicable.

Contributions

Both authors have participated in conception and design of the framework, Xuefei Li, Jia Chen, and Ningbo Cui carried out the experimental work and results analysis. The work revised by Prof. Xiantao Cai and Shaohua Wan. Both authors read, edited, and approved the final manuscript.

Data Availability

The data included in this paper are available without any restriction.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] Wertz, J.R. and Larsen, W.J., Space Mission Analysis and Design, 3rd Edition, Microcosm Press, El Segundo, CA and Kluwer Academic Publishers, Dordrecht, The Netherlands, pp. 694, 1999.

[2] W Wei, R Yang, H Gu, et al, Multi-objective Optimization for Resource Allocation in Vehicular Cloud Computing Network, IEEE Transactions on Intelligent Transportation Systems, 2021.

[3] C Chen, L Liu, S Wan, et al, Data Dissemination for Industry 4.0 Applications in Internet of Vehicles Based on Short-term Traffic Prediction, ACM Transactions on Internet Technology (TOIT), vol.22, no.1, pp.1-18, 2021.

[4] Bingyi Liu, Dongyao Jia, Jianping Wang, et al, Cloud-assisted safety message dissemination in VANET–Cellular heterogeneous wireless network, IEEE Systems Journal, vol 11, no.1, pp.128-139, MAR 2017.

[5] Chen H, Wu J, Shi W, Coordinate scheduling approach for EDS observation tasks and data transmission jobs, Journal of Systems Engineering and Electronics, 2016.

[6] Song Y J, Zhang Z S, Song B Y, et al, Improved Genetic Algorithm with Local Search for Satellite Range Scheduling System and its Application in Environmental monitoring, Sustainable Computing, 21(MAR.), pp. 19-27, 2019.

[7] C Chen, Y Zeng, H Li, Y Liu, et al, A Multi-hop Task Offloading Decision Model in MEC-enabled Internet of Vehicles, IEEE Internet of Things Journal, 2022.

[8] C Chen, J Jiang, R Fu, et al, An Intelligent Caching Strategy Considering Time-Space Characteristics in Vehicular Named Data Networks, IEEE Transactions on Intelligent Transportation Systems, 2021.

[9] Berger J, Lo N, Noel M, et al, DynaQUEST: A New Approach to the Dynamic Multi-satellite Scheduling Problem, In Proceedings of the 9th International Conference on Operations Research and Enterprise Systems (ICORES 2020), pp. 194-201, 2020.

[10] Li Y Q, Wang R X, Liu Y, et al, Satellite range scheduling with the priority constraint: An improved genetic algorithm using a station ID encoding method. China Journal of Aeronautics, vol.28, no.3, pp.789-803, 2015.

[11] Wang J M, Li J F, Tan Y J, Study on Heuristic Algorithm for Dynamic Scheduling Problem of Earth Observing Satellites, Study on Heuristic Algorithm for Dynamic Scheduling Problem of Earth Observing Satellites, 2007.

[12] S Wan, J Hu, C Chen, A Jolfaei, et al, Fair-Hierarchical Scheduling for Diversified Services in Space, Air and Ground for 6G-Dense Internet of Things, IEEE Transactions on Network Science and Engineering, 2020.

[13] Cui J, Zhang X, Application of a multi-satellite dynamic mission scheduling model based on mission priority in emergency response.Sensors (Switzerland), 19, 1430, 2019.
[14] Niu X N, Tang H, Wu L X, Multi-satellite observation scheduling for large area disaster emergency response, International Society for Photogrammetry and Remote Sensing, pp.1327-1331, 2018.
[15] L. Zhao, H. Li, N. Lin, et al, Intelligent Content Caching Strategy in Autonomous Driving Towards 6G, IEEE Transactions on Intelligent Transportation Systems (T-ITS), 2021, DOI: 10.1109/TITS.2021.3114199.
[16] Hoel C J, Wolff K, Laine L, Automated Speed and Lane Change Decision Making using Deep Reinforcement Learning, In:2018 21st International Conference on Intelligent Transportation Systems (ITSC), 2018.
[17] Han T, Nageshrao S, Filev D P, et al, An online evolving framework for advancing reinforcement-learning based automated vehicle control, 2020.
[18] L. Zhao, G. Han, Z. Li, et al, "Intelligent Digital Twin-based Software-Defined Vehicular Networks", IEEE Network, 2020, DOI: 10.1109/MNET.011.1900587.
[19] Zhang X, Jiang H, Automated optimal control in energy systems: the reinforcement learning approach, New Technologies for Power System Operation and Analysis, 2021.
[20] Haijiao W, Zhen Y, Wugen Z, Dalin L., Online scheduling of image satellites based on neural networks and deep reinforcement learning. Chinese Journal of Aeronautics, vol. 32, no.4, pp.1011-1019, 2019.
[21] He L, Liu X L, Chen Y W, et al, Hierarchical scheduling for real-time agile satellite task scheduling in a dynamic environment, Advances in Space Research 63, pp.897–912, 2019.
[22] Zhang Z, Wang W, Zhong S, et al, FLOW SHOP SCHEDULING WITH REINFORCEMENT LEARNING, Asia Pacific Journal of Operational Research, vol.30, no.5, pp.1350014-1-1350014-25, 2013.
[23] AJ Kleywegt, JD Papastavrou, The Dynamic and Stochastic Knapsack Problem. Operations Research, vol.46, no.1, 1998.
[24] Liang Zhao, Tong Zheng, Mingwei Lin, Ammar Hawbani, Jiaxing Shang, Chunlong Fan "SPIDER: A Social Computing Inspired Predictive Routing Scheme for Softwarized Vehicular Networks," IEEE Transactions on Intelligent Transportation Systems (T-ITS), 2021.
[25] Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O, Proximal policy optimization algorithms, https://arxiv.org/abs/1707.06347, 2017.
[26] Mnih V, Kavukcuoglu K, Silver D, et al, Human-level control through deep reinforcement learning, Nature, vol. 518, pp.529-533, 2015.
[27] Mnih V, Badia AP, Mirza M, et al, Asynchronous Methods for Deep Reinforcement Learning. Proceedings of the 33rd International Conference on Machine Learning, vol 48, pp.1928-1937, 2016.
[28] Si, W., Mburano, B., Zheng, W.X. and Qiu, T., 2022. Measuring Network Robustness by Average Network Flow. IEEE Transactions on Network Science and Engineering.
[29] Zhu, X., Qiu, T., Qu, W., Zhou, X., Wang, Y. and Wu, O., 2021. Path Planning for Adaptive CSI Map Construction with A3C in Dynamic Environments. IEEE Transactions on Mobile Computing.
[30] Chen, N., Qiu, T., Lu, Z. and Wu, D.O., 2021. An Adaptive Robustness Evolution Algorithm With Self-Competition and Its 3D Deployment for Internet of Things. IEEE/ACM Transactions on Networking.
[31] Fu, S., Atiquzzaman, M., Ma, L. and Lee, Y.J., 2005. Signaling cost and performance of SIGMA: A seamless handover scheme for data networks. Wireless Communications and Mobile Computing, 5(7), pp.825-845.
[32] Qiu, T., Wang, X., Chen, C., Atiquzzaman, M. and Liu, L., 2018. TMED: A spider-Web-like transmission mechanism for emergency data in vehicular ad hoc networks. IEEE Transactions on Vehicular Technology, 67(9), pp.8682-8694.
[33] Leonard T, Monte Carlo and bust. (Letter), RSS News, vol.23, no.8, 1996.