Collaborative Edge Caching in LEO Satellites Networks: A MAPPO Based Approach

Mingzhou Wu*, Shiqi Dai*, Han Hu‡, Zhi Wang*†✉
* Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
†Tsinghua-Berkeley Shenzhen Institute, Tsinghua University, Shenzhen, China
‡Beijing Institute of Technology, Beijing, China
{wu-mz21, daisq21}@mails.tsinghua.edu.cn, hhu@bit.edu.cn, wangzhi@sz.tsinghua.edu.cn

2024 IEEE International Conference on Multimedia and Expo (ICME) | 979-8-3503-9015-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICME57554.2024.10687952

***Abstract*—Low Earth Orbit satellite networks, as a crucial component of global low-latency internet access, are expected to carry significant user traffic in the future. Caching frequently requested content, e.g., popular videos on short-video platforms, in satellite networks can significantly alleviate traffic congestion. However, the satellite's brief overhead passing time, which is less than ten minutes, makes it difficult for satellites to capture the content popularity distribution. And the changing relative position between satellites poses challenges for cooperation. To address the challenges, we propose a method called SEC_MAPPO for deploying cooperative edge caching in satellite networks. First, we model this novel scenario and transform it into a Partially Observable Markov Decision Process (POMDP). Then, we design a multi-agent reinforcement learning algorithm specifically tailored for this scenario. Trace-driven simulation using a real-world LEO satellite constellation and video request dataset demonstrated that our proposed algorithm could achieve a reduction in average video request latency ranging from 4.53% to 9.31% compared to the baseline solutions.**

***Index Terms*—LEO Satellite Network, Edge Caching, Multi-Agent Deep Reinforcement Learning**

I. Introduction

To accomplish the global seamless connection in the next generation of mobile communication, the Low Earth Orbit (LEO) satellite network is expected as the pivotal supplementary to 6G architecture [1]. However, the proliferation of content-rich applications, e.g., short-video platforms, has contributed to a global surge in the quantity of data being transmitted worldwide. Such a massive number of connections and volume of network traffic are challenging the limited communication resources in satellite networks. Edge caching is a common technique to reduce latency and save communication resources by storing frequently requested contents at the edge devices of the network. With the advancements in satellite hardware storage technology, deploying edge computing and caching capability on satellites is becoming feasible [2]. However, the high-speed movement of satellites and the dynamic nature of network topology present obstacles to implementing efficient edge caching on satellite networks. The high-speed movement of satellites makes it difficult to capture the content popularity distribution within its coverage area. Additionally, changing network topology poses a challenge to maintaining stable data transmission links. These new features make it infeasible to directly migrate terrestrial edge caching algorithms to satellite networks.

Conventional rule-based caching methods, e.g., LFU, LRU, ARC [3], have shown obvious performance degradation in the LEO satellite edge caching scenario. As demonstrated in works [4], [5], there exists a verifiable correlation between the popularity distribution of contents and geographical location. Consequently, the rapid motion of satellites makes it difficult to perceive the content popularity distribution as the coverage area remains changing. To improve edge caching performance on satellite networks, the work [6] considers the cache placement and content delivery strategy in satellite-terrestrial integrated cloud radio access networks composed of a single LEO satellite and multiple base stations. The work [7] proposes a three-layer cooperative caching model involving base stations, satellites, and the gateway to improve cache hit rate through cooperative caching but neglects the dynamic of the satellite networks and the cache efficiency. Due to the limited storage capacity of each individual satellite, the performance of edge caching methods based on a single satellite is often constrained. The work [8] takes the LEO satellite's movement into account and proposes a collaborative edge caching algorithm named CACVC to encourage cooperation among satellites in the same orbit. Although this work has considered the cooperation between satellites, the potential for cooperation between satellites across different orbit planes is ignored.

To improve the collaborative edge caching between satellites in multiple orbit planes, this paper proposes an innovative satellite edge caching strategy, named SEC_MAPPO, based on the Multi-Agent Proximal Policy Optimization algorithm [9]. The algorithm follows a "centralized training, decentralized execution" framework. In the training phase, the global critic model assesses the joint actions of each agent to encourage collaboration. Specifically, our proposed algorithm aims to enhance storage capacity utilization by encouraging each satellite to cache different contents as far as possible. In the execution phase, each satellite independently makes decisions based on its local observations. At last, the efficacy of the proposed algorithm is validated through experiments using

Corresponding author of this paper is Zhi Wang. This research is funded by National Key Research and Development Project of China (Grant No. 2023YFF0905502), Shenzhen Science and Technology Program (Grant No. JCYJ20220818101014030). We would like to thank Xinjie Yuan, Rongwei Lu and Haojun Sun for their help in making this work possible.

real-world constellation information and widely used video request datasets. The principal contributions of this paper are threefold:

▷ To address the issue of redundant cached content among high-speed moving LEO satellites, we have designed a cooperative caching model for satellites. Aiming to minimize the average request latency, we have formulated the problem as a nonlinear integer programming problem.

▷ We transformed the original optimization problem into a Partially Observable Markov Decision Process (POMDP) to address the challenge of high dynamism in the scenario. Based on MAPPO, we propose the SEC_MAPPO method as the solution strategy to implement cooperative edge caching among satellites.

▷ We employ the configuration of 1,536 satellites from Starlink, the largest commercial LEO satellite constellation, to simulate the dynamics of the constellation. Testing with 53 million content requests collected from IQIYI, our algorithm exhibits substantial enhancements in cache hit rates and a notable decrease in average user request latency. These results corroborate the effectiveness of the proposed method in practical satellite network scenarios.

II. Low Earth Orbit Satellite Edge Caching System

A. System Model

In this work, we consider an LEO satellite collaborative edge caching system shown in Figure 1, which comprises three main elements:

▷ Multiple User Ends (UEs) with satellite communication capabilities $\mathcal{U} = \{u_1, u_2, ..., u_{N_1}\}$, including but not limited to mobile phones, vehicles, and drones. And N_1 is the total number of these diversified devices.

▷ A LEO satellite constellation consists of multiple satellites denoted by $\mathcal{S} = \{s_1, s_2, ..., s_{N_2}\}$, where N_2 is the total number of the satellite constellation.

▷ A set of existing Inter Satellite Links (ISLs) at time t, which is denoted by $\mathcal{E}^t = \{e_1, e_2, ..., e_{N_3^t}\}$. For each e_i in set $\mathcal{E}^t$, it consists three components as $e_i = (s_a, s_b, t_{a2b})$, where s_a and s_b are two ends of this ISL and t_{a2b} is the remaining establishment time of this link. With the fixed orbit parameters, such ISLs set at time t can be calculated in advance.

Each satellite has limited storage capability to cache popular contents. When UEs request content through the satellite network, the request latency can be significantly reduced if the content has been cached in the access satellite. Unfortunately, compared with the wide coverage area of the LEO satellite, its caching capability is relatively insufficient. To resolve this dilemma, collaborative caching is viewed as a viable approach, wherein cached content can be shared among neighboring satellites through ISL. By developing an efficient collaborative caching method, the storage capability of the whole system can be utilized and users' content request delay will be minimized. To better describe this system, detailed models are illustrated as follows.

B. Content Caching Model

Let $\mathcal{F} = \{1, 2, ..., N_F\}$ denote the set of contents that can be requested in this system, and all the contents are able to be derived from the remote cloud center but experience high latency. Within this caching system, without loss of generality, all contents are standardized to a uniform unit size of m. This is achieved by dividing files of varying sizes into multiple subfiles, each of equal size. As the cached content is refreshed periodically at each time slot indexed by $t \in \mathcal{T} = \{1, 2, .., T\}$, satellite s_i generates a cache decision vector at time t as $X_i^t = \{x_{i,1}^t, x_{i,2}^t, ..., x_{i,N_F}^t\}$, where $x_{i,k}^t = 1$ represents caching content k in satellite s_i at time slot t, and $x_{i,k}^t = 0$ otherwise. To satisfy the storage capability constraint, the cache decision vector needs to meet the following conditions:

$$\sum_{f \in \mathcal{F}} m x_{s,f}^t \leq M_s, \forall s \in \mathcal{S}, t \in \mathcal{T}, \tag{1}$$

where m is the size of each content and M_s is the storage capability of satellite s.

C. Service Model

In each time slot t, we use R^t to represent the UEs request set. To be specific, $r_{b,f}^t = 1$ means UE b requests content f during the time slot t, and $r_{b,f}^t = 0$ otherwise. In the proposed collaborative caching system, UEs' requests can be satisfied in three ways according to the satellites' cache status:

- **Direct Hit Mode (DHM)**: In the direct hit mode, UE's requested content is cached in the access satellite, which means the satellite can deliver it back to the UE directly and with the lowest latency. The latency can be calculated as:

$$L_s^t(D) = \sum_{f \in \mathcal{F}} \frac{r_{b,f}^t x_{s,f}^t m}{v_{s2b}}, \tag{2}$$

  where v_{s2b} is the transmission rate between satellite and UE. v_{s2b} can be calculated according to Shannon Theory [10] which is related to transmit and receive power consumption at both ends. As power allocation is not addressed in this work, all UEs within the coverage area of satellite s enjoy the same transmission rate.
- **Neighbor Hit Mode (NHM)**: Neighbor hit mode is chosen only when the direct access satellite does not cache the corresponding content. Through the existing ISLs, content can be derived from the neighbor satellites of satellite s. Therefore, the total transmission latency is denoted as:

$$L_s^t(N) = \sum_{f \in \mathcal{F}} \left(\frac{r_{b,f}^t m}{v_{s2b}} + \frac{D_f r_{b,f}^t x_{s',f}^t m}{v_{s2s}}\right), \tag{3}$$

  where satellite s' is the nearest neighbor satellite cached content f, D_f is the number of hops between satellite s and s', v_{s2s} is the transmission rate between satellites through the ISL. Additionally, D_f can be calculated by the shortest path algorithm with the existing ISLs set $\mathcal{E}^t$

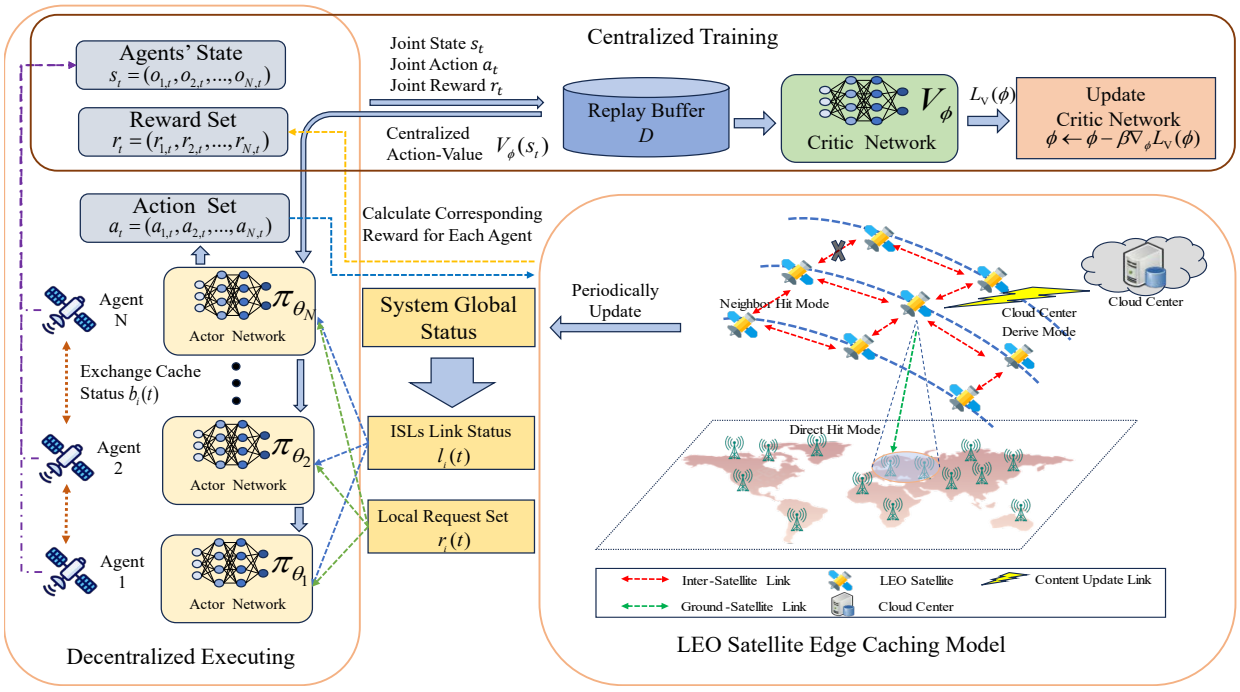


Fig. 1: An illustration of LEO satellite collaborative edge caching system. a) During the training phase, the algorithm encourages cooperation between satellites by evaluating the quality of actions from a global perspective. b) During the execution phase, each satellite makes caching decisions based on its local observation to adapt to the dynamic network.

mentioned above, and v_{s2s} can be calculated in the same way as v_{s2b}.

- **Cloud Center Derive Mode (CCDM)**: When all the satellites do not cache the content f or the target satellite is far away from the access satellite, i.e., $D_f > K$, the system adopts the cloud center derive mode. Compared with other modes, the cloud center derive mode needs to fetch the content from the remote cloud center, which results in much higher latency [11]. Therefore, we use a constant number to represent the latency of CCDM:

$$L_s^t(C) = \overline{L}. \tag{4}$$

Each request needs to be satisfied in one mode, we use a_D, a_N and a_C as indicators. The average request latency, which is the optimization objective, is defined as follows:

$$L_{avg} = \frac{\sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{S}}\sum_{b\in C_s(t)} a_D L_s^t(D) + a_N L_s^t(N) + a_C L_s^t(C)}{\sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{S}}\sum_{b\in C_s(t)}\sum_{f\in\mathcal{F}} r_{b,f}^t}, \tag{5}$$

and $C_s(t)$ is the coverage area of satellite s at time t.

D. Problem Formulation

To minimize the average content request latency in this LEO satellite system, we can formulate the objective function with several constraints as follows:

$$\min\ L_{avg} \tag{6}$$

$$\text{s.t.} \sum_{f\in\mathcal{F}} m x_{s,f}^t \le M_s, \forall s \in \mathcal{S}, t \in \mathcal{T}, \tag{7}$$

$$x_{s,f}^t \in \{0,1\}, \forall t \in \mathcal{T}, \forall s \in \mathcal{S}, \forall f \in \mathcal{F}, \tag{8}$$

$$a_D + a_N + a_C = 1, \tag{9}$$

$$D_f \le K, \tag{10}$$

where (7) ensures the total size of cached contents does not exceed the satellite's maximum allowed cache space; (8) guarantees the legitimacy of cache decision vector; (9) ensures each request will be handled in exactly one mode and (10) prevents users from requesting files from distant satellites. The optimization objective is a classic nonlinear integer programming problem, known to be NP-hard, which renders exact solutions intractable within polynomial time. And single-agent reinforcement learning method cannot handle high-dimensional environments well. Consequently, we employ multi-agent reinforcement learning to efficiently approximate optimal solutions in this complex and dynamic environment.

III. MAPPO-BASED ALGORITHM FOR LEO SATELLITE COLLABORATIVE EDGE CACHING

Due to the satellites' mobility, limited cache capacities, and the uneven distribution of user requests, traditional offline optimization methods are not suitable for solving this problem. Meanwhile, the single-agent deep reinforcement learning algorithm cannot effectively learn the optimal strategy with complex observation and action space. Multi-Agent Proximal

Policy Optimization (MAPPO), as an extension of the PPO algorithm tailored for multi-agent systems, is particularly well-suited for addressing problems in scenarios characterized by high-dimensional action and state spaces. Therefore, to address these challenges, this paper proposes an innovative collaborative edge caching strategy named SEC_MAPPO.

A. POMDP Formulation

Because each satellite cannot observe the entire system's state while making caching decisions, we reformulate this problem as a POMDP. Typically, an MDP contains three important components, a state space $\mathcal{S}$, an action space $\mathcal{A}$, and a reward function $\mathcal{R}$. In the multi-agent DRL system, each agent is able to observe the partial state of the whole system. Therefore, the global state $\mathcal{S}$ is the Cartesian product of all observation spaces for each agent, $\mathcal{S} = \mathcal{O}_1 \times \mathcal{O}_2 \times \ldots \times \mathcal{O}_N$. Similarly, action space is the Cartesian product of all agents' action space, $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \ldots \times \mathcal{A}_N$, where N is the number of agents. The entire agent set is denoted as $\mathcal{I} = \{1, 2, ..., N\}$, and each satellite is considered as an independent agent.

Observation Space: At each time slot t, we define the state of agent i as:

$$\mathcal{O}_i = (c_i(t), r_i(t), b_i(t), l_i(t)), \tag{11}$$

where $c_i(t)$ is the cache status of satellite i at time t, $r_i(t)$ is the requests collected by all UEs within the service area of satellite i, $b_i(t)$ is the cache status of neighbor satellites (at most four) and $l_i(t)$ represents the link status (at most four), specifically, the remaining service time of each link.

Action Space: In many related works, researchers prefer to set the action space as a vector $p = (p_1, p_2, ..., p_F)$, where p_i is the possibility of caching content i. Such settings combine the dimension of action space to the dimension of the entire content library $\mathcal{F}$, which may be extremely large. Therefore, when agent i encounters a "miss" content, we formulate the action space of agent i as:

$$\mathcal{A}_i \in \{1, 2, ..., M_s\} \cup \{0\} \cup \{M_s+1, M_s+2, M_s+3, M_s+4\}, \tag{12}$$

where M_s is the cache capability of the satellite i. The action $a_i \in \{1, 2, ..., M_s\}$ means the satellite will evict the $a_i{}^{th}$ content and store the new content, the action $a_i \in \{0\}$ means do nothing and the action $a_i \in \{M_s+1, M_s+2, M_s+3, M_s+4\}$ means satellite i will send the content to exact one of the neighbor satellite for caching. Thus, the dimension of action space is only related to its storage capability, which is much smaller than the dimension of the entire content library.

Reward Function: To minimize the average request delay, we wish that the network can satisfy as many requests as possible. Therefore we use the satisfaction of requests between two adjacent misses to measure the rewards, e.g., $\mathbf{R} = \{r_1^{Miss}, r_2^{Hit}, r_3^{Hit}, ..., r_k^{Miss}\}$. Additionally, direct satellite hits should obtain larger rewards than neighbor satellite hits:

$$\mathcal{R}_i = \sum_{r \in \mathbf{R}} \alpha_1 \times a_D + \alpha_2 \times a_N \times e^{-D_r}, \tag{13}$$

where α_1, α_2 are constants, a_D, a_N are mode indicators, and D_r is the number of hops to deliver the required content for request r.

B. SEC_MAPPO Train Framework

As shown in Algorithm 1, SEC_MAPPO is built based on the multi-agent extension version of the PPO algorithm. The algorithm adopts an actor-critic framework, where the actor network is responsible for selecting actions based on the current policy and the critic network evaluates the potential value of the actions taken by the actors. For the agent set $\mathcal{I} = \{1, 2, ..., N\}$, we use $\pi_\theta = \{\pi_{\theta_1}, \pi_{\theta_2}, ..., \pi_{\theta_N}\}$ to represent the actor network for approximating the policy and V_ϕ to represent the shared critic network to approximate the value function. θ and ϕ are the parameters for actor and critic networks. At each time step t of the centralized training phase, each agent i gets the local observation $o_{i,t}$ from the environment and takes action $a_{i,t}$ according to its actor network. Then the environment returns the reward $r_{i,t}$ of the action. The global critic network collects all agents' actions and gets the new state s_{t+1}. After collecting enough transitions (s_t, a_t, r_t, s_{t+1}) for the critic network and $(o_{i,t}, a_{i,t}, r_{i,t}, o_{i,t+1})$ for the actor networks in replay buffer $\mathcal{D}$, a mini-batch set of data is sampled for updating actor networks and the critic network.

Following the multi-agent training paradigm, the clipped objective function for each agent i is:

$$L^{\text{CLIP}}(\theta_i) = \hat{\mathbb{E}}_t \left[\min(v_t(\theta_i)\hat{A}_t, \text{clip}(v_t(\theta_i), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right], \tag{14}$$

where ϵ is the hyper-parameter for clip fraction to avoid the excessive adjustment of the objective function value and $v_t(\theta_i)$ is the probability ratio of taking action in new policy and old policy:

$$v_t(\theta_i) = \frac{\pi_{\theta_{i,t}}(a_{i,t}|o_{i,t})}{\pi^{old}_{\theta_{i,t}}(a_{i,t}|o_{i,t})}, \tag{15}$$

General advantage estimation $\hat{A}_t$ measures how good are current actions regarding the baseline critic value as the following equation:

$$\hat{A}_t = \sum_{t=0}^{\infty} (\gamma\lambda)^t \delta^V_{t+l}, \tag{16}$$

where γ and λ are the discount factor and trade-off factor between bias and variance in the advantage estimate.

The critic network is learned using gradient descent with the loss function as follows:

$$L_V(\phi) = \frac{1}{|\mathcal{D}|n} \sum_{\tau \in \mathcal{D}} \sum_{i=1}^{n} \left(V_\phi(s_{i,t}) - \hat{R}_t \right)^2, \tag{17}$$

where $\hat{R}_t$ is the discounted return so far.

IV. Performance Evaluation

In this section, we evaluate the performance of our proposed algorithm based on both the synthetically generated Zipf request dataset and the real dataset from IQIYI with several baseline algorithms.

Algorithm 1: SEC_MAPPO: MAPPO-Based Satellite Collaborative Edge Caching Alogrithm

Output: Policy networks π_{θ_i} for $i \in \mathcal{I}$

Initialize critic network $V_\phi(s_t)$, actor network π_θ and replay buffer $\mathcal{D}$

for *Episode* $e = 1, \ldots, E$ **do**

  for *Time Step* $t = 1, \ldots, T$ **do**

    Calculate satellites' coverage area $C(t)$ and ISLs set $\mathcal{E}^t$

    for *Each Agent* $i = 1, \ldots, N$ **do**

      Obtain observation $o_{i,t}$

      Choose action $a_{i,t} \sim \pi_{\theta_i}(a_{i,t}|o_{i,t}, \theta_i)$

      Obtain returned reward $r_{i,t}$ and new observation $o_{i,t+1}$ Store transition $\mathcal{D} \leftarrow \mathcal{D} \cup \{(o_{i,t}, a_{i,t}, r_{i,t}, o_{i,t+1})\}$

    end

    Obtain state s_t and new state s_{t+1}

    Collect the global reward $r_t = \sum_{i \in N} r_{i,t}$

    Store transition $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r_t, s_{t+1})\}$

  end

  for *Update Phase* $u = 1, \ldots, U$ **do**

    Sample mini-batch $\mathcal{B}$ from $\mathcal{D}$

    Update critic network $\phi \leftarrow \phi - \beta \nabla_\phi L_{\mathrm{V}}(\phi)$

    Update actor networks $\theta_i \leftarrow \theta_i + \alpha \nabla_{\theta_i} L^{\mathrm{CLIP}}(\theta_i)$

  end

end

TABLE I: Experiments Setting Parameters

Parameter	Value	Training Parameter	Value
Number of orbits	3	Learning rate	0.0003
Satellites per orbit	3,4,3	Discount factor	0.95
Half cone angle ψ	62°	Entropy coefficient	0.01
Orbit inclination	53°	Iterations	20000
Orbit height	550km	Clip parameter	0.3
Number of video	5000,400000	Batch size	16
Unit size	20 Mb	KL coefficient	0.2

A. Experimental Settings

To better understand our proposed algorithm in the real world, we use Simulation Tool Kits (STK) and Matlab to build the StarLink constellation as the testbed. We select three adjacent orbits and 10 satellites in total to simulate the cooperation between satellites. 936 UEs are distributed along the satellite trajectory to generate content requests. According to [12], we set the ISL data transmission rate to 2Gbps and the GSL data transmission rate to 800Mbps. Other experimental settings are shown in the Table I.

To fully evaluate the performance of our proposed algorithm, we choose four baseline algorithms: **RANDOM** (Random cache replacement algorithm), **LFU** (Least frequently used), **LRU** (Least recently used) and **DQN** (Deep Q-learning based algorithm).

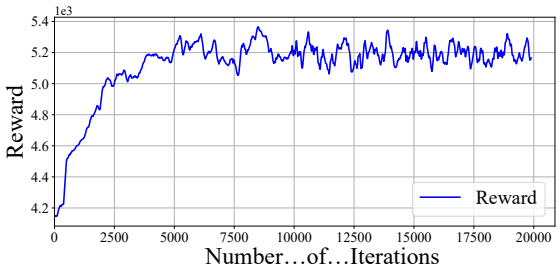
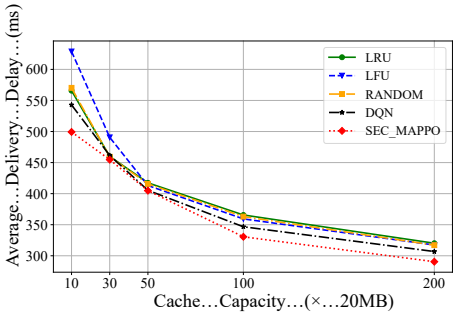


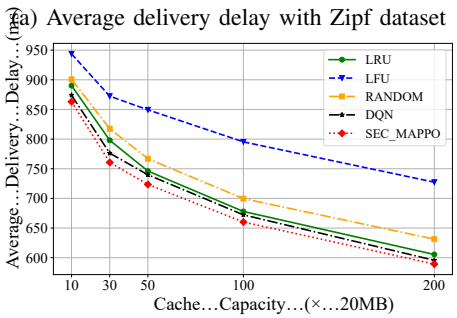
Fig. 2: The training reward curve of SEC_MAPPO algorithm.

B. Convergence Analysis

As shown in Fig.2, the proposed algorithm experienced a fast learning period at the beginning of the whole process and became stable after 5000 iterations. After 5000 iterations, the training reward has consistently maintained at a high level with slight fluctuations, indicating that the algorithm is still actively exploring additional strategies within the context of a good strategy guarantee.



(a) Average delivery delay with Zipf dataset

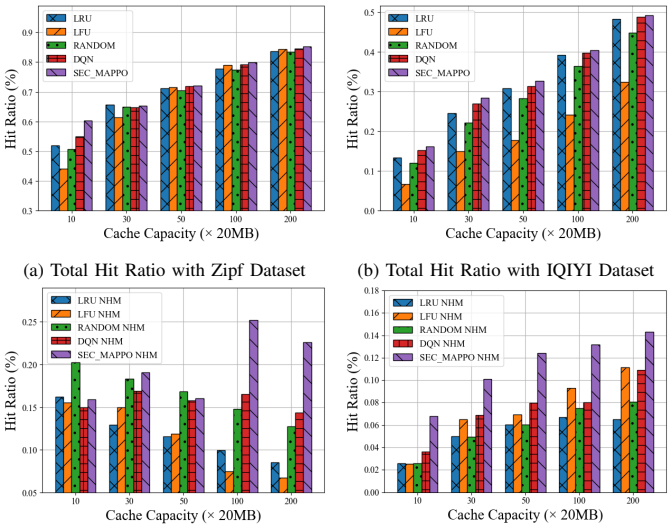


(b) Average delivery delay with IQIYI dataset

Fig. 3: Average delivery delay.

C. Numerical Result

Figure 3 shows the average delivery delay with different satellites' storage capacities from 200MB to 4GB. The results demonstrate that no matter in the synthetic Zipf dataset or IQIYI real-world dataset, our proposed SEC_MAPPO algorithm outperforms other baseline solutions. Referencing Figure 3a, it becomes evident that our algorithm shows enhanced performance with the Zipf dataset, particularly when dealing with either very small (200MB) or relatively large (4GB) storage capacities. The efficiency of the proposed caching



(a) Total Hit Ratio with Zipf Dataset (b) Total Hit Ratio with IQIYI Dataset

(c) Neighbor Hit Ratio with Zipf Dataset (d) Neighbor Hit Ratio with IQIYI Dataset

Fig. 4: The performance of SEC_MAPPO algorithm. (a) & (b) show SEC_MAPPO obtains the highest total hit ratio in both Zipf and IQIYI datasets. (c) & (d) show SEC_MAPPO facilitates collaboration between satellites through a higher neighbor hit ratio.

method can be quantified by the notable reduction in average latency. Specifically, it achieved a reduction of 20.6% (for 200MB storage capacity) and 8.5% (for 4GB storage capacity) compared to LFU. Similarly, when compared to LRU, RANDOM, and DQN methods, the reduction in average latency was 11.7%, 12.4%, and 8.1% (for 200MB), and 9.3%, 8.4%, and 5.3% (for 4GB) respectively.

In the IQIYI real-world dataset, our algorithm has demonstrated significant superiority over other baseline algorithms in all scenarios. Specifically, when the cache capacity is set at 1GB per satellite, our algorithm achieves a remarkable reduction in average latency compared to LFU, LRU, RANDOM, and DQN. The reductions are measured at 14.8%, 3.0%, 5.7%, and 2.2% respectively. Notably, our findings indicate that LFU performs the poorest among all solutions when facing dynamic and complex video request patterns in real-world scenarios.

Lastly, we use the total cache hit ratio and neighbor hit mode (NHM) cache hit ratio to measure the level of collaboration between satellites. The overall cache hit ratio exhibits a direct correlation with the average delivery delay, indicating that a higher cache hit ratio typically results in lower delivery latency. As shown in Figure 4a and 4b, the SEC_MAPPO algorithm achieves the highest hit ratio in both Zipf and IQIYI datasets. Although the margin of superiority diminishes as storage capacity increases, the results still demonstrate the effectiveness of our algorithm.

In the Zipf dataset, each satellite experiences distinct content request patterns. Excluding the RANDOM algorithm, our algorithm demonstrates a higher level of collaboration between satellites, for example, SEC_MAPPO achieves 25.2% compared to LFU with 7.47%, LRU with 9.94%, and DQN with 16.55% when cache capacity is 1GB per satellite. The reason why the RANDOM algorithm can obtain a higher NHM cache hit ratio with a smaller cache capacity is due to the randomness of its cache replacement, while other algorithms strive to ensure the performance of cache hits by sensing the request pattern, so it is difficult to achieve efficient cooperation between regions with completely different request patterns. While with the IQIYI dataset, our proposed algorithm maintains a significant lead in the NHM cache hit ratio metric, corroborating the effectiveness of our approach in enhancing cache collaboration among neighboring satellites.

V. Conclusion

In this paper, we formulate the edge caching problem in the LEO satellite network. To minimize the average content request latency, we further transmit the original optimization problem into a multi-agent decision process. To obtain the optimal solution, we propose an algorithm SEC_MAPPO using multi-agent reinforcement learning. The results obtained from both synthetic and real-world datasets validate our method's ability to reduce the average video content request latency.

References

[1] Wei Jiang, Bin Han, Mohammad Asif Habibi, and Hans Dieter Schotten, "The road towards 6g: A comprehensive survey," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 334–366, 2021.

[2] Jin Tang, Jian Li, Xianhao Chen, Kaiping Xue, Lan Zhang, Qibin Sun, and Jun Lu, "Cooperative caching in satellite-terrestrial integrated networks: A region features aware approach," *IEEE Transactions on Vehicular Technology*, pp. 1–15, 2024.

[3] Nimrod Megiddo and Dharmendra S. Modha, "ARC: A self-tuning, low overhead replacement cache," in *USENIX FAST*. 2003, USENIX.

[4] Ge Ma, Zhi Wang, Miao Zhang, Jiahui Ye, Minghua Chen, and Wenwu Zhu, "Understanding performance of edge content caching for mobile video streaming," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1076–1089, 2017.

[5] Zhi Wang, Lifeng Sun, Xiangwen Chen, Wenwu Zhu, Jiangchuan Liu, Minghua Chen, and Shiqiang Yang, "Propagation-based social-aware replication for social video contents," in *ACM MM*. 2012, pp. 29–38, ACM.

[6] Dairu Han, Haixia Peng, Huaqing Wu, Wenhe Liao, and Xuemin Sherman Shen, "Joint cache placement and content delivery in satellite-terrestrial integrated c-rans," in *IEEE ICC*. 2021, pp. 1–6, IEEE.

[7] Xiangming Zhu, Chunxiao Jiang, Linling Kuang, and Zhifeng Zhao, "Cooperative multilayer edge caching in integrated satellite-terrestrial networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 5, pp. 2924–2937, 2022.

[8] Ruili Zhao, Yongyi Ran, Jiangtao Luo, and Shuangwu Chen, "Towards coverage-aware cooperative video caching in LEO satellite networks," in *IEEE GLOBECOM*. 2022, pp. 1893–1898, IEEE.

[9] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and YI WU, "The surprising effectiveness of ppo in cooperative multi-agent games," in *NeurIPS*. 2022, vol. 35, pp. 24611–24624, Curran Associates, Inc.

[10] Aaron D. Wyner, "Recent results in the shannon theory," *IEEE Transactions on Information Theory*, vol. 20, no. 1, pp. 2–10, 1974.

[11] Zeqi Lai, Hewu Li, Qi Zhang, Qian Wu, and Jianping Wu, "Cooperatively constructing cost-effective content distribution networks upon emerging low earth orbit satellites and clouds," in *IEEE ICNP*. 2021, pp. 1–12, IEEE.

[12] Mohamed I Shehata, M Asmaa Zaki, Heba A Fayed, Ahmed Abd El Aziz, and Moustafa H Aly, "Optical inter-satellite link over low earth orbit: Enhanced performance," in *Journal of Physics: Conference Series*. IOP Publishing, 2021, vol. 2128, p. 012001.