In-Orbit Computation and Security Authentication for Satellite-Ground Twin Networks

Yongkang Gong, *Member, IEEE,* Xiaonan Liu, *Member, IEEE,* Haipeng Yao, *Senior Member, IEEE,* Xiuzhen Cheng, *Fellow, IEEE,* Arumugam Nallanathan, *Fellow, IEEE,* and George K. Karagiannidis, *Fellow, IEEE*

***Abstract*—Satellite-ground twin networks are considered a promising network structure, which can relieve network congestion and provide pervasive intelligent services. Since massive terrestrial users are not willing to share mobile data, it is necessary to design a novel security authentication method. In this paper, we consider a two-layer Stackelberg game model and propose a deep federated meta reinforcement learning (LST-DFMRL) framework based on Lyapunov stability theory to orchestrate the cycle frequency, channel assignment and block size. Simulation results confirm that the proposed LST-DFMRL framework outperforms existing baseline methods in terms of server profits, network throughput and security authentication overhead.**

***Index Terms*—Satellite-ground twin networks, in-orbit computation, security authentication, deep federated meta reinforcement learning.**

I. Introduction

TRADITIONAL terrestrial cellular networks [1] suffer from major challenges, such as severe network congestion, huge construction costs and small coverage area, which further affect the quality of service (QoS) for the sixth generation (6G) wireless networks. Fortunately, satellite-ground integrated twin networks (SGTN) can provide global coverage for terrestrial users, where digital twin can shorten the gap between physical unities and digital world. Specifically, Luo *et al.* [2] proposed a novel edge server scheduling algorithm to achieve a tradeoff between construction cost and network capacity. Cao *et al.* [3] maximized the network throughput via considering the ground-space and ground-air-space communication links for two task types. Guo *et al.* [4] introduced a comprehensive survey about computation offloading and privacy security in the SGTN network.

Nevertheless, the aforementioned works do not jointly consider related price profits and total throughput for satellite servers and terrestrial users. Moreover, when users offload tasks to remote servers, they may suffer channel interference from proximal areas. Hence, Lu *et al.* [5] proposed an adaptive edge association method to achieve digital placement and migration. Despite the improvement of the learning efficiency in digital twin systems, the lack of mutual trust among users affects the improvement of QoS. Thus, blockchain consensus protocols can be widely applied to ensure security authentication. Specifically, Qiu *et al.* [6] proposed a collective Q-learning mechanism to validate the proof of work in a cloud-edge-end network architecture. Wang *et al.* [7] designed a native edge intelligence framework to reduce the average system overhead while optimizing communication, computation and caching resources. Furthermore, Qu *et al.* [8] presented a blockchain-aided cognitive computation structure and handled the data island problem via a federated learning technique.

However, the aforementioned works only used a single learning mode and cannot quickly adapt to a small batch of samples. Hence, inspired by the above-mentioned challenges, we establish an SGTN network scenario to achieve task scheduling, reduce channel interference and improve security authentication functions in a dynamic network environment. The main contributions are as follows.

- First, we propose a two-layer Stackelberg game model to maximize network throughput and cloud server profits. Integrated with in-orbit computation and blockchain technology, it not only helps the network adapt to time-varying random tasks and dynamic satellite orbit locations, but also strengthens privacy authentication through transaction verification mechanisms. Furthermore, it decouples the long-term task queue and single-slot computation offloading variables.
- Next, a Lyapunov stability theory based deep federated meta reinforcement learning (LST-DFMRL) framework is proposed to optimize the local CPU cycle frequency, assign the optimal channel for each user to mitigate interference and choose the block size to perform transaction

Y. Gong and X. Cheng are with the School of Computer Science and Technology, Shandong University, 266237 China (e-mail: gokawa@sdu.edu.cn and xzcheng@sdu.edu.cn).

X. Liu is with the School of Natural and Computing Science, University of Aberdeen, Aberdeen AB24 3FX the United Kingdom (e-mail: xiaonan.liu@abdn.ac.uk).

H. Yao is with the School of Information and Communication Engineering, BUPT, Beijing, 100876, China (e-mail: yaohaipeng@bupt.edu.cn).

A. Nallanathan is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, London and also with the Department of Electronic Engineering, Kyung Hee University, Yongin-si, Gyeonggi-do 17104, Korea (mailto:a.nallanathan@qmul.ac.uk).

G. K. Karagiannidis is with Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, Greece and also with Artificial Intelligence & Cyber Systems Research Center, Lebanese American University (LAU), Lebanon (geokarag@auth.gr).

This research is partially funded by the National Key R&D Program of China (2022YFB2902500), the National Natural Science Foundation of China (62402292, 62325203, U22B2033), Shandong Postdoctoral Science Foundation (SDBX202302010), Qingdao Postdoctoral Science Foundation (QDBSH20240102027), the Program for Youth Innovative Research Team of BUPT (NO. 2024YQTD02). (Corresponding author: Yongkang Gong)

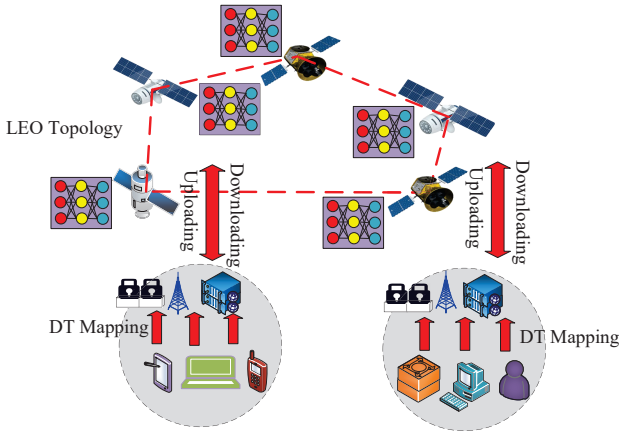


Fig. 1: The established SGTN network scenario.

verification. Specifically, the LST transformation is used to decompose the long-term task queues into single slot. The proposed DFMRL policy can achieve task scheduling, resource management, interference suppression and privacy protection.

- Finally, extensive simulation results show that the proposed LST-DFMRL framework outperforms existing baselines in terms of network throughput, cloud server profits and privacy overhead, which validates the efficacy and progressiveness of the LST-DFMRL framework.

The structure of this paper is shown as follows. Section II introduces the related SGTN network model. The corresponding LST-DFMRL algorithm is presented in Section III. Section IV conducts extensive simulation results. Finally, we conclude this paper in Section V.

II. System Model

A. SGTN Network Model

We show the SGTN network scenario in Fig. 1, which consists of a two-layer network structure, i.e., low earth orbit satellite (LEO) networks and terrestrial networks. Specifically, the LEO network includes massive satellites, whose groups are represented as $\mathbb{M} = \{1, 2, ..., m, ..., M\}$. Furthermore, the terrestrial network is composed of many macro base stations (MBS), whose sets are denoted as $\mathbb{N} = \{1, 2, ..., n, ..., N\}$. Meanwhile, for each MBS n, it overlays $\mathbb{L} = \{1, 2, ..., l, ..., L\}$ users and these users can map their data from physical unities to the digital world via the digital twin technology.

Next, as multiple terrestrial users are not willing to share privacy data in task offloading, we utilize the blockchain technology and federated learning method to achieve parameters aggregation and transaction verification, a detailed process is presented in Section IV.

B. Local Twin Model

In this subsection, we divide the Stackelberg game process into *follower* and *leader* stages. First, each terrestrial user aims to maximize the network throughput while minimizing privacy protection overhead. The local twin model is introduced as follows.

In each time slot t, each user l in MBS n receives the task $B_{n,l}^t$, and we assume the second order is limited, i.e., $\mathbb{E}\left([B_{n,l}^t]^2\right) = c < \infty$, where c is obtained via interacting with historic information. Hence, the local processed number of bits is calculated as

$$D_{n,l}^{t1} = \frac{f_{n,l}}{\Phi}\tau, \tag{1}$$

where $f_{n,l}$ is the local CPU cycle frequency of each digital twin, Φ is the required number of CPU cycles while processing one bit task and τ is the duration among two time slots.

C. Task Scheduling Model

When each user offloads the task to remote servers, the time-varying random tasks, dynamic satellite orbit locations and corresponding channel interference among terrestrial users degrade the performance of computation offloading and security authentication. Furthermore, for the MBS n, the communication loss between the user l and the LEO m is represented as

$$\begin{aligned} S_{n,l}^t = {} & 20\log\left(4\pi f_c\sqrt{x_{n,l,m}^2 + y_{n,l,m}^2}/c\right) + p_{n,l,m}^{LoS}\alpha_{n,l,m}^{LoS} \\ & + \left(1 - p_{n,l,m}^{LoS}\right)\alpha_{n,l,m}^{NLoS}, \end{aligned} \tag{2}$$

where f_c and c represent the carrier frequency and the speed of light, respectively. Moreover, $x_{n,l,m}$ and $y_{n,l,m}$ denote the horizontal distance and vertical distance between the terrestrial user l and LEO m. Next, $\alpha_{n,l,m}^{LoS}$ and $\alpha_{n,l,m}^{NLoS}$ represent the additional path loss imposed on line of sight (LoS) and non line of sight (NLoS), respectively. Furthermore, the LoS propagation probability is denoted as

$$p_{n,l,m}^{LoS} = \frac{1}{1 + a_1\exp\left\{-a_2\left[\arctan\left({}^{y_{n,l,m}}/_{x_{n,l.m}}\right) - a_1\right]\right\}}, \tag{3}$$

where a_1 and a_2 are corresponding system parameters obtained via interacting with dynamic network environments. Furthermore, when massive terrestrial users transmit tasks to remote servers, more channel interference is caused. Assuming that the access scheme is orthogonal frequency division multiple access, the channel interference for the user l in the MBS n is denoted as

$$I_{n,l,r} = \sum_{z=1,z\neq n}^{N}\sum_{l=1}^{L}\lambda_{n,l,r}P_{n,l,r}\left|10^{-\frac{S_{n,l}^t}{10}}\right|^2, \tag{4}$$

where $\lambda_{n,l,r}$ denotes that the channel r is allocated to the user l. Meanwhile, $P_{n,l,r}$ is the transmission power and the total information transmission rate in the task scheduling mode is represented as

$$R_{n,l,r}^t = B_{n,l,r}^t\log\left(1 + \frac{b_{n,l,r}P_{n,l,r}\left|10^{\frac{-S_{n,l}^t}{10}}\right|^2}{\sigma^2 + I_{n,l,r}}\right), \tag{5}$$

where $B_{n,l,r}$ is the allocated channel bandwidth for the user n and $b_{n,l,r}$ represents the user l chooses the channel r as the offloaded server. Moreover, σ^2 is corresponding channel noise power and the total processed number of bits is calculated as

$$D_{n,l}^{t2} = R_{n,l,r}^t\tau. \tag{6}$$

D. Security Authentication Model

Since many terrestrial users are not willing to share data, the blockchain technology can be used to protect data privacy. Specifically, each block unit records these transaction information and they can be verified via corresponding users. Moreover, the security authentication model can be divided into three parts, i.e., parameters aggregation, transmission and verification overhead. The detailed process is presented below.

The parameters aggregation overhead is denoted as

$$C_1 = \frac{|w_l|}{F_{MBS}}, \tag{7}$$

where w_l is related network model parameters for the user l and F_{MBS} is the CPU cycle frequency of each MBS. Subsequently, the transmission overhead is represented as

$$C_2 = \chi \log_2 N^{|w_l|}/_{r_{up}}, \tag{8}$$

where χ is the model transmission factor and r_{up} is the uplink transmission rate. Next, the verification overhead is denoted as

$$C_3 = \chi \log_2 NL \frac{S_{bc}}{r_{down}} + \max\left({}^{S_{bc}}/_{f_{n,l}}\right), \tag{9}$$

where S_{bc} is the block size and r_{down} is the download rate for each user l. Finally, the total security authentication overhead is represented as

$$C_{total} = C_1 + C_2 + C_3. \tag{10}$$

E. Problem Formulation

We formulate the Stackelberg game process as satellite cloud server and terrestrial users model and the corresponding service profits are denoted as

$$P1: \max_{\varphi_{n,l}} \sum_{n=1}^{N}\sum_{l=1}^{L} \varphi_{n,l} f_{n,l} - g f_{n,l} \tag{11}$$

$$s.t. \quad \varphi_{n,l} \geq 0, \tag{12}$$

where $\varphi_{n,l}$ indicates the cloud servers' price and g is the unit electronic energy consumption. After obtaining the cloud server pricing policy, the total processed number of bits is represented as

$$U_1 = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} b_{n,l}^t D_{n,l}^{t1} + \left(1 - b_{n,l}^t\right) D_{n,l}^{t2}. \tag{13}$$

Moreover, the loss of security authentication and computation resources is calculated as

$$U_2 = C_{total} + \varphi_{n,l} f_{n,l}. \tag{14}$$

Hence, the total profits for users are denoted as

$$P2: \max_{\{f_{n,l}, b_{n,l}^t, b_{n,l,r}, S_{bc}\}} \{U_1 - U_2\} \tag{15}$$

$$s.t. \quad \frac{f_{n,l}}{\Phi} \leq Q_{n,l}^t, \tag{16}$$

$$I_{n,l,r} \leq I_{\max}, \tag{17}$$

$$b_{n,l}^t \in \{0,1\}, \tag{18}$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[Q_{n,l}^t] < \infty, \tag{19}$$

$$S_{\min} \leq S_{bc} \leq S_{\max}, \tag{20}$$

where (16) indicates that the task bits cannot exceed the task queue $Q_{n,l}^t$, we will introduce the task queue in the next section. In (17) and (18), I_{max} and $b_{n,l}^t$ are the maximum channel interference and the task scheduling decision vector, respectively. Finally, (19) and (20) represent that task queue and block size are limited.

III. LST-DFMRL Algorithm

In this section, we propose an LST-DFMRL algorithm to resolve the computation offloading and security authentication in the two-stage Stackelberg game model. However, P2 implies that the variables coupling between long-term task queue and short-term computation offloading lead to an intractable resolving process. Hence, we first introduce Lyapunov based problem transformation to transfer the long-term multiple time slots to the single time slot subproblem. The task queue for each time slot can be denoted as $Q_{n,l}^t$.

A. Lyapunov Problem Transformation

Based on Lyapunov stability theory [9], the corresponding Lyapunonv function and Lyapunov drift function are denoted as

$$L\left(\vec{Q}\left(t\right)\right) = \frac{1}{2}\left(Q_{n,l}^t\right)^2, \tag{21}$$

$$\Delta\left(\vec{Q}\left(t\right)\right) = \mathbb{E}\left\{L\left(\vec{Q}\left(t+1\right)\right) - L\left(\vec{Q}\left(t\right)\right)|\vec{Q}(t)\right\}, \tag{22}$$

where $Q_{n,l}^t$ is the task queue. Then, the virtual task queue is denoted as

$$Q_{n,l}^{t+1} = \max\left\{Q_{n,l}^t - D_{n,l}^{t1/t2} + B_{n,l}^t, 0\right\}, \tag{23}$$

where $D_{n,l}^{t1/t2} = b_{n,l}^t D_{n,l}^{t1} + \left(1 - b_{n,l}^t\right) D_{n,l}^{t2}$. Furthermore, we derive the task queue via taking squares from both sides

$$\begin{aligned}\left(Q_{n,l}^{t+1}\right)^2 = \left(Q_{n,l}^t\right)^2 + 2Q_{n,l}^t\left(B_{n,l}^t - D_{n,l}^{t1/t2}\right) \\ + \left(B_{n,l}^t - D_{n,l}^{t1/t2}\right)^2.\end{aligned} \tag{24}$$

Next, the formulation (24) is derived as

$$\begin{aligned}0.5\left(Q_{n,l}^{t+1}\right)^2 - 0.5\left(Q_{n,l}^t\right)^2 = Q_{n,l}^t\left(B_{n,l}^t - D_{n,l}^{t1/t2}\right) \\ + 0.5\left(A_{n,l}^t - D_{n,l}^{t1/t2}\right)^2,\end{aligned} \tag{25}$$

Subsequently, the Lyapunov drift function is represented as

$$\begin{aligned}&\triangle\left(\vec{Q}(t)\right) = \mathbb{E}\left\{L(\vec{Q}(t+1)) - L(\vec{Q}(t))\right\} \\ &= 0.5\left(B_{n,l}^t - D_{n,l}^{t1/t2}\right)^2 + Q_{n,l}^t\left(B_{n,l}^t - D_{n,l}^{t1/t2}\right) \\ &\leq X + Q_{n,l}^t\left(B_{n,l}^t - D_{n,l}^{t1/t2}\right),\end{aligned} \tag{26}$$

where

$$\begin{aligned}&0.5\left(B_{n,l}^t - D_{n,l}^{t1/t2}\right)^2 \leq 0.5\left(B_{n,l}^t\right)^2 + \left(D_{n,l}^{t1/t2}\right)^2 \\ &\leq 0.5c + \left(D_{n,l}^{t1/t2/\max}\right)^2 = X.\end{aligned} \tag{27}$$

$D_{n,l}^{t1/t2/\max}$ is the maximum of $D_{n,l}^{t1/t2}$, and it denotes the maximum number of task scheduling. Meanwhile, the drift-plus-penalty function is represented as

$$L_d = \Delta\left(\vec{Q}\left(t\right)\right) - V\mathbb{E}\left\{U_1 - U_2\right\}, \tag{28}$$

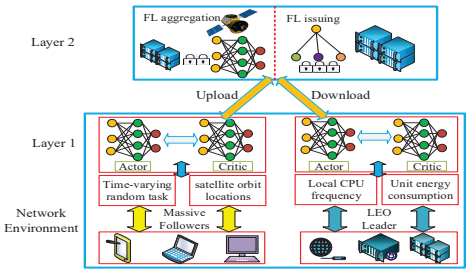


Fig. 2: The proposed two-layer LST-DFMRL framework.

where V is the corresponding Lyapunov control parameter. Hence, through transforming the drift-plus-penalty function (28), the original problem P2 is transformed into

$$P2^{'}: \quad \max_{\{\forall l\}} \quad Q_{n,l}^{t} D_{n,l}^{t1/t2} + V\mathbb{E}(U_1 - U_2). \tag{29}$$

B. DFMRL Implmentation

As depicted in Fig. 2, we propose a two-layer DFMRL framework to resolve local twin, task scheduling, and security authentication model. Moreover, each user can adapt to time-varying random tasks ($B_{n,l}^{t}$), dynamic satellite orbit locations ($x_{n,l,m}$, $y_{n,l,m}$), and optimize the CPU cycle frequency $f_{n,l}$, channel selection $b_{n,l,r}$, task scheduling unit $b_{n,l}^{t}$ and block size S_{bc}. Subsequently, each leader can determine the cloud server price and unit energy consumption in terms of each user's scheduling decisions. The detailed DFMRL process is shown as follows.

(1) *Meta Learning Adaptation Mechanism*: The goal of meta learning is to learn a small batch of samples to speed up the convergence rate. We assume that these old tasks and new tasks for meta training and testing are subject to distribution $p(\Gamma)$. Specifically, meta learning can update corresponding network weight parameters, which is represented as

$$\min_{\theta} \mathbb{E}\left[L\left(E_{\Gamma}^{test}, \theta^{'}\right)\right] \tag{30}$$

$$s.t. \quad \theta^{'} = \nabla L\left(E_{\Gamma}^{train}, \theta\right), \tag{31}$$

where E_{Γ}^{train} and E_{Γ}^{test} denote the training tasks and testing tasks from the distribution $p(\Gamma)$, respectively. Moreover, $L\left(E_{\Gamma}^{test}, \theta^{'}\right)$ indicates the testing set loss function for new tasks. Thus, we can separate the meta learning algorithm to two modules, i.e., inner loop training samples and outer testing samples. Specifically, the inner training samples can be utilized to update network parameters $\theta^{'}$, and then $\theta^{'}$ can adapt to testing tasks.

(2) *DFMRL Process*: We utilize the proposed actor-critic network structure to optimize the task scheduling, resource allocation and block size. For multiple terrestrial users l, there are time-varying random tasks $B_{n,l}^{t}$, dynamic satellite orbit locations ($x_{n,l,m}$, $y_{n,l,m}$). Thus, the specific state, action, and reward function are denoted as follows.

State Space: Each user l in MBS n outputs the local state $S_{n,l} = \left(B_{n,l}^{t}, x_{n,l,m}, y_{n,l,m}\right)$, and these state information cannot exchange with each other due to strict privacy protection.

Action Space: According to Fig. 2, each actor network selects the CPU cycle frequency $f_{n,l}$, task scheduling decision $b_{n,l}^{t}$, channel selection $b_{n,l,r}$ and block size S_{bc}. Thus, the action space for each user is denoted as $A_{n,l} = \left(f_{n,l}, b_{n,l}^{t}, b_{n,l,r}, S_{bc}\right)$.

Reward Function: The instant reward function is represented as

$$R_{n,l} = \sum_{n=1}^{N}\sum_{l=1}^{L} Q_{n,l}^{t} D_{n,l}^{t1/t2} + V\mathbb{E}\left(U_1 - U_2\right), \tag{32}$$

MDP Transformation: For the SGTN network scenario, it is hard to find a fixed transformation policy to represent network states. Hence, we denote the set $\mho = \left\{S_{n,l}^{'} | S_{n,l}, A_{n,l}, R_{n,l}\right\}$ to represent MDP transformation process.

Similar to multiple terrestrial users, the state space for satellite cloud servers is represented as $S_s = \{f_{n,l}, g\}$. Subsequently, after receiving the state space, each actor neural network from satellite server outputs action space $A_s = \{\varphi_{n,l}\}$. Moreover, the corresponding reward function is $R_s = \sum_{n=1}^{N}\sum_{l=1}^{L} \varphi_{n,l} f_{n,l} - g f_{n,l}$. Once the neural network reaches convergence, it can obtain the optimal reward function. Finally, the MDP transformation process is represented as $\Upsilon = \left\{S_s^{'} | S_s, A_s, R_s\right\}$.

C. Security Authentication

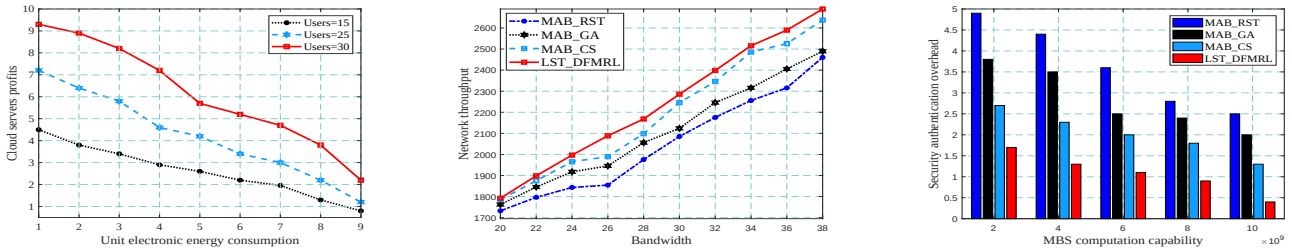
After each actor network generates task scheduling decision, channel selection and CPU cycle frequency, we need to evaluate the action values. Nevertheless, the state information exchange among multiple users increases the risk of privacy disclosure. Specifically, all terrestrial users receive the global network model $W(t)$ from the satellite cloud server, and then they update local network model $W_{n,l}(t)$, which is illustrated as $I_{n,l}(t) = W(t) - W_{n,l}(t)$. After that, the satellite cloud server receives the global model and calculates it as $W(t+1) = W(t) + \eta I(t)$, where η is the federated aggregation rate and $I(t)$ is represented as $I(t) = \frac{|B_{n,l}^{t}|+|G_{n,l}^{t}|}{|B_{total}|+|G_{total}|} I_{n,l}(t)$, where $B_{n,l}^{t}$ and B_{total} are corresponding task sets of each user and total task sets of all users, respectively. Accordingly, $L_{n,l}^{t}$ and L_{total} are the corresponding distance from LEO to each terrestrial user and the sum of distance for all terrestrial users.

When each user transmits network weight parameters to LEO, detailed transaction security authentication is shown as follows. For instance, each user stores the task information to block units. Next, each user broadcasts transaction information to other users and transmits them to the satellite server. Subsequently, the LEO server packages them to block S_{bc}. Accordingly, each user will download block S_{bc} from LEO servers, whose transaction process is verified via delegated stakes proof protocols [5]. Once the transaction verification is achieved via other users, the reward coins are returned back to the initial users. If there is fraud or tamper behavior in the transaction process, the mechanism closes corresponding transaction.

IV. Simulation Results

A. Simulation Parameters

By refering [10], we consider that $N = 4$, $L = 12$ and $M = 4$. The distribution of random tasks $B_{n,l}^{t}$ is uniformly distributed at (15, 30) MB, the required number of CPU



(a) (b) (c)

Fig. 3: The simulation results in terms of cloud servers profits, network throughput and security authentication overhead.

cycle for processing one bit task is uniformly distributed at (2500, 3500) cycle/bit and the blockchain throughput is 100 transactions per second. Meanwhile, the horizontal distance $x_{n,l,m}$ and vertical distance $y_{n,l,m}$ are uniformly distributed at (1000, 1800) KM and (500, 1500) KM. Accordingly, the light speed c and carrier frequency f_c are represented as $3 * 10^8$ m/s and $0.3 * 10^9$ HZ. Furthermore, $\alpha_{n,l,m}^{LoS}$ and $\alpha_{n,l,m}^{NLoS}$ are uniformly subject to $(0, 1)$ and $(15, 25)$. Finally, the noise power σ is defined as 10^{-13} W.

B. Performance Analysis

We compare the proposed LST-DFMRL algorithm with some advanced benchmarks, such as multi-agent-based random task scheduling (MAB_RST), multi-agent-based greedy algorithm (MAB_GA) and multi-agent-based channel selection (MAB_CS). Meanwhile, we demonstrate the performance gains in terms of satellite cloud servers' gains, network throughput and security authentication overhead.

As shown in Fig. 3(a), we explore the impact of unit electronic energy consumption on satellite servers profits. Obviously, as the number of terrestrial users increases, the satellite server obtains higher service profits. When the number of users is 30, the cloud servers' profits have approximately 38% performance gains compared with "Users=25". However, when the unit energy consumption increases, the profits of satellite servers decrease. This is because it consumes more energy while serving terrestrial users. Hence, it is essential to balance the satellite servers profits and unit energy consumption to guarantee QoS for more terrestrial users.

As shown in Fig. 3(b), when the transmission bandwidth is 28 MHZ, the proposed LST_DFMRL algorithm has approximately 4%, 6.6% and 10.1% performance gains compared with MAB_CS, MAB_GA and MAB_RST. This is because it can not only better adapt to dynamic network environments, but also achieve better task scheduling and resource orchestration for each terrestrial user. However, the transmission bandwidth is limited in practical scenarios, so the LST-DFMRL algorithm can also accelerate the training process via meta learning methods.

Finally, as shown in Fig. 3(c), when the MBS computation capability is $6 * 10^9$, the LST_DFMRL algorithm has approximately 42% and 52% performance gains compared with MAB_CS, MAB_GA. Furthermore, as MAB-GA and MAB-CS algorithms only greedily choose the CPU cycle frequency and assign a wireless channel for each terrestrial user, which cannot fulfil the optimal resource orchestration mechanism. Hence, the proposed LST-DFMRL algorithm can achieve lower privacy overhead, because it can upload model parameters to remote satellite servers instead of local tasks, and the block-chain-based authentication protocols strengthen the transaction security.

V. Conclusions

In this paper, we propose a two-layer Stackelberg game model to maximize the network throughput and LEO server profits while minimizing the security authentication overhead. Furthermore, the proposed LST-DFMRL algorithm can not only adapt to dynamic network environments, but also resolves variables coupling for long-term task queues and short-term resource orchestration. Meanwhile, a block-chain-based authentication protocols strengthen the transaction security. Finally, extensive simulation results corroborate that the proposed LST-DFMRL algorithm has superior performance gain in terms of cloud servers profits, network throughput and security authentication overhead compared with some advanced benchmarks.

References

[1] Y. Gong, H. Yao, J. Wang, M. Li, and S. Guo, "Edge intelligence-driven joint offloading and resource allocation for future 6g industrial internet of things," *IEEE Transactions on Network Science and Engineering, (DOI: 10.1109/TNSE.2022.3141728)*, 2022.

[2] R. Luo, H. Jin, Q. He, S. Wu, and X. Xia, "Cost-effective edge server network design in mobile edge computing environment," *IEEE Transactions on Sustainable Computing, Early Access*, 2022.

[3] X. Cao, B. Yang, C. Yuen, and Z. Han, "HAP-reserved communications in space-air-ground integrated networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 8, pp. 8286–8291, Aug. 2021.

[4] H. Guo, J. Li, J. Liu, N. Tian, and N. Kato, "A survey on space-air-ground-sea integrated network security in 6G," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 53–87, 1st Quart., 2021.

[5] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Low-latency federated learning and blockchain for edge association in digital twin empowered 6G networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 5098–5107, Jul. 2020.

[6] C. Qiu, X. Wang, H. Yao, J. Du, F. R. Yu, and S. Guo, "Networking integrated cloud–edge–end in IoT: A blockchain-assisted collective Q-learning approach," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 694–12 704, Aug. 2020.

[7] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge ai: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Network*, vol. 33, no. 5, pp. 156–165, Sept.-Oct. 2019.

[8] Y. Qu, S. R. Pokhrel, S. Garg, L. Gao, and Y. Xiang, "A blockchained federated learning framework for cognitive computing in industry 4.0 networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2964–2973, Apr. 2020.

[9] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[10] Y. Gong, H. Yao, J. Wang, L. Jiang, and F. R. Yu, "Multi-agent driven resource allocation and interference management for deep edge networks," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 2018–2030, Feb. 2022.