Journal of Cloud Computing:
Advances, Systems and Applications

RESEARCH

Open Access

Economical revenue maximization in mobile edge caching and blockchain enabled space-air-ground integrated networks

Jianbo Du[1*], Jiaju Lv[1] and Guangyue Lu[1]

Abstract

In this paper, we study an edge caching and blockchain enabled space-air-ground integrated networking (SAGIN) network, where a low-earth-orbit (LEO) satellite serves as the content provider, and multiple edge caching enabled unmanned aerial vehicles (UAVs) will cache some contents to provide user equipments (UEs) with satisfactory content access services together with the satellite. Moreover, there's a blockchain system that is deployed on UAVs, to provide the network with trust mechanism without requiring a centralized authority. From the standpoint of the operator, we intend to maximize the long-term averaged economical revenue by providing UEs with satisfactory and secure content access services. To achieve this purpose, we will jointly optimize the content placement of each UAV, content replacement when each UAV is full, the access control of each UE, and the blockchain deployment strategy about each UAV. the concept of queues in Lyapunov optimization is utilized to represent the backlog of edge equipment, ensuring the stability of virtual queues on UAVs and satellites, while satisfying the caching capacity constraints for content caching and blockchain deployment. Due to the tight coupling of optimization in each time slot and the variables within each time slot, our problem, which involves stochastic optimization and binary integer programming, is challenging to solve. To address this issue, we initially employ Lyapunov optimization theory to transform and decouple the problem into individual time-slot optimization problems. Subsequently, we utilize an effective heuristic algorithm called the fireworks algorithm to solve these individual optimization problems. However, the original fireworks algorithm cannot be directly applied to our problem due to its binary characteristics and inter-coupling constraints. Therefore, we have redesigned the explosion and mutation operations to adapt them to our specific problem. Simulation results demonstrate that our proposed algorithm outperforms other baseline algorithms.

Keywords Space-air-ground integrated networks (SAGIN), Content caching, Blockchain deployment, Lyapunov optimization, Heuristic algorithm

Introduction

The world has witnessed a huge increasing number of mobile user equipments (UEs), such as smart phones, wearable devices, and Internet-of-Things (IoT) devices [1], etc., over the last decade [2], and meanwhile has spawned tremendous and ubiquitous high data rate applications, resulting in explosive growth in mobile data traffic. The unparalleled traffic growth brings conventional cellular networks with great challenges in offering high throughput content delivery services, and also result in frequent network congestion and growing content delivery latency [3]. However, there is a mass of redundancy content transmission over the network, and many popular contents are repeatedly requested and downloaded by most UEs.

*Correspondence:
Jianbo Du
dujianboo@163.com
[1] School of Communications and Information Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710121, China

To cope with this challenge, mobile edge caching (MEC) [4] has appeared as a hopeful supplement paradigm to the current existing wireless cellular networks and content delivery networks (CDNs). By prospectively caching the frequently requested Internet contents into adjacent edge caching servers [5], UEs can access and fetch these contents directly from their adjacent MEC servers, rather than from the remote content servers through backhaul links. As thus, the massive repeated content access can be significantly avoided, and backhaul pressure can be effectively alleviated. Meanwhile, users' content fetching latency can be dramatically reduced, and the quality of experience (QoE) is enhanced notably [6]. Therefore, mobile content caching has been a hot research topic recent years. In [7], the authors studied the optimization in an edge caching enabled vehicular networks. With the cooperation among cache-enabled macro base station, roadside units, and smart vehicles, they jointly optimized the content placement and content downloading process. In [8], the authors proposed to optimize the content caching locations, i.e., to cache the content to the cloud server or the base station, in a 5G wireless network, and the objective is to support the highest posibble content delivery data rates, subjecting to the content delivery latency requirement. In [9], the authors optimized cache replacement and which node to be cached in a device to device (D2D) assisted heterogeneous collaborative edge caching network, based on the cooperation among UEs, base station based edge caching server, and the cloud content server. Note that the above works mainly discussed about the edge caching enabled terrestrial networks.

However, the terrestrial infrastructures are not always exist, such as in remote rural regions or disaster areas [10]. Also, edge caching based terrestrial networks could not always satisfy the skyrocketing data increasing using the scarce spectrum resources of terrestrial networks. So as to provide global and massive connections, space-air-ground integrated networks (SAGINs) was regarded as a vital enabler and effective supplement to terrestrial networks for accommodating massive traffic and wide area ubiquitous coverage. SAGIN is constituted by three layers, where the space layer could offer broad coverage using satellites, the air layer can improve the flexibility and network capacity using unmanned aerial vehicles (UAVs) [11, 12], balloons, and high latitude platforms, etc. Combing edge caching with SAGIN, users' QoE can further be enhanced, especially for the UEs in remote areas or disaster areas. In [13], the authors studied a cache-enabling UAV-D2D network, where they proposed to jointly optimize UEs and UAV' caching placement, and UAVs' path, in order to maximize the cache utility in hot-spot areas. In [14], the authors jointly optimize the long-term cache placement policy, short-term base station clustering and satellite grouping, and short-term multi-cast beam-forming strategies, in order to increase the saved backhaul traffic in an integrated satellite-terrestrial network. In [15], the authors considered a edge caching based satellite-UAV-vehicle integrated network, where a satellite served as the content server, and UAVs acted as edge servers for content caching. The authors intended to decrease the total energy expenditure of the satellite and UAVs, through joint content placement, the satellite and the UAVs' transmit power allocation, and the number of UAVs to be deployed. The above works mainly considered the content placement and distribution in edge caching enabled integrated satellite and/or UAV ground networks, while did not consider the content replacement when the caching storage is full, and did not consider the privacy and secure issues.

Edge caching systems may confront some security and privacy problems, where edge nodes may send incorrect contents or even viruses to UEs, and UEs may also deliberately cheat and not pay after receiving their requested content. Also, privacy data such as user preferences may be leaked during data transmission. In [16], considering the content popularity is dynamic, unobservable, and private, the authors intended to maximize the cache hit ratio of UEs with privacy preservation constraints satisfied. The authors introduced a privacy-preserving based federated learning method to predict content popularity, and proposed a distributed deep reinforcement learning based algorithm for solving the distributed problems. In order to provide easy-deployment trust and privacy guarantee, blockchain is introduced into edge caching systems. In [17], the authors intended to optimize the content caching ratio, thus to maximize the benefits of both the edge servers and D2D UEs in a blockchain based edge caching system, where blockchain acts as a distributed shared ledger and database, to ensure the immutability and non-repudiation of system working process. In [18], a and trusted authentication guaranteed distributed edge architecture was presented for authentication assured information sharing among different IoT platforms, and an efficient content caching algorithm was presented to optimize the content placement strategy in order to minimize content downloading latency. The above works mainly concerned on the content placement issues, and did not jointly optimize the blockchain system. Also, the above works studied the caching optimization in ground edge caching enabled system.

Concluding the above works, it can be known that a comprehensive joint optimization about edge caching system and blockchain system is necessary in SAGIN networks, where we will confront great challenges for the following reasons. i) Since the network is high dynamic,

temporary optimization could not reflect the performance, we need to optimize the long-term averaged performance index and conduce long-term optimization. ii) Since the storage capabilities of UAVs are usually limited , it's unpractical to deploy all contents to each UAV, and moreover, while bringing certain economic benefits, the blockchain system also needs to occupy some capacities, so we need to optimize the placement of both contents and blockchain, and we also need to replace some content when the storage capacity is full. iii) Each UE may be covered by multiple UAVs, where some far UAVs may cache its requested content, and some adjacent UAVs may not cache its requested content, so we should determin where should each UE access and fetch its required content, i.e., from the satellite, or from a certain UAV? iv) Considering all the concerns, the question will be rather difficult to solve, so we need to design low-complexity algorithms to obtain feasible solutions. Considering all the above issues, we conclude the main contributions as follows.

- We consider the secure content deployment and delivery issues in an edge caching and blockchain enabled SAGIN system, where a LEO satellite acts as the content server, and multiple UAVs can cache some contents to provide UEs with QoE-guaranteed content accessing services. Meanwhile, a blockchain system is deployed on UAVs to stores authentication data and logs information, thus to provide trusted authentication, activity traceability, etc., for the edge caching system. Via optimizing the content placement and replacement strategies in content deployment stage, and the access control in content delivery stage of the edge caching system, and the blockchain deployment strategies of the blockchain system, we propose to maximize the long-term averaged economical revenue of the operator, with the virtual queue stability, storage capacity and blockchain performance constraints, etc., satisfied.
- We employ Lyapunov optimization technologies to address the global constraints of our formulated problem by deriving the upper bound of the drift-plus-penalty function, and our original long-term problem is decoupled into each-time-slot optimization problems, thus the solution can be determined by solving a determined problem at each time slot, and all virtual network queues can be stabilized meanwhile for all content providers including the satellite and all UAVs, i.e., to achieve finite content access delay for all UEs. The specific process of Lyapunov optimization theory involved in this paper is as follows: First,The Lyapunov function composed of the virtual queue of UAV and satellite is determined. As a basic element. Then, the Lyapunov drift function with single slot condition is derived. Finally, Based on stochastic optimization theory, minimizing this upper bound enables the realization of Lyapunov drift functions for content placement, content replacement, access control, and blockchain deployment optimization in time slot t. This approach simplifies the definition of the original optimization problem.
- In the decoupled problem of each time slot, there are four groups of binary variables need to be determined, i.e., the content placement, content replacement, access control, and blockchain deployment, and the four groups of binary variables are tightly intertwined by multiple constraints, making the each-time-slot optimization problem difficult to solve.Therefore, we resort to an effective heuristic algorithm called fireworks algorithm to solve it with low-complexity. However, the original fireworks is designed for simple continuous control problems with only a simple boundary constraints, it is powerless to solve the binary control problem with many tightly coupled constraints. In order to solve the problem, we improve the form of firework, the explosion and mutation operations to fit for our problem, i.e., to satisfy all constraints of our problem, and thus to generate feasible explosion and mutation sparks in the running process.

The remainder context is organized as follows. System model section exhibits the system model, and Problem formulation section presents our problem formulation. In Problem transformation using Lyapunov optimization section, we transform our problem into an each-time-slot optimization problem using Lyapunov optimization. In Using fireworks algorithm to solve the each-time-slot optimization problem ($\boldsymbol{\mathcal{P}_2}$) section, we present an coupling-constrained oriented binary fireworks algorithm to solve the each-time-slot optimization problem. Simulation results are elaborated in Simulation results and discussions section, and finally, we present the conclusions of this paper in Conclusions section.

System model

We consider a cache-enabled SAGIN system which is composed of a low earth orbit (LEO) satellite in the sky, which serves as the content provider, J cache-enabled UAVs in the air, and I ground UEs which will request content. Let the set of UEs and UAVs are denoted as $\mathcal{I} = \{1, 2, ..., i, ...I\}$ and $\mathcal{J} = \{1, 2, ..., j, ...J\}$, respectively, and the storage capacity of UAV is denoted as S_j. It is assumed that the satellite is stationary during each time slot, and the satellite hovers in the sky according to

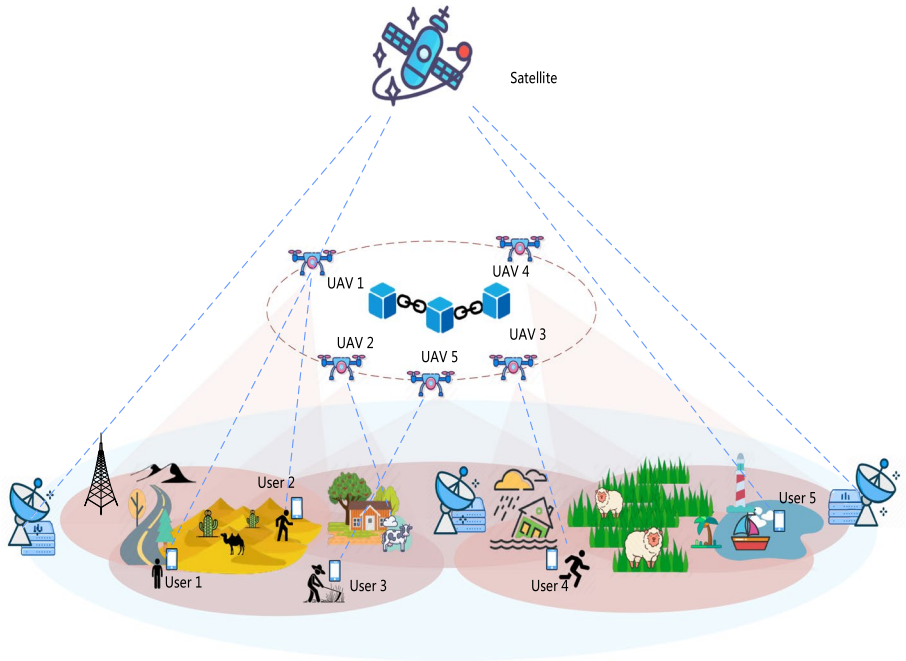

Fig. 1 The Concerned Scenario

certain rules, and they together provide wireless coverage and wireless communication services to the coverage area. We assume there's no macro base station to provide the traffic delivery, an example is the earthquake stricken area, remote area, or during a superstar concert. In these cases, the traffic will be served by UAVs and the satellite, they will serve as content provider to provide UEs with their requested content. The UAV starts from a given position, and will fly along a random trajectory to serve the ground UEs, and hovers within a certain range. The UEs can also acquire their desired content from the satellite,which is assumed to be stationary in the schedular time in this paper. We assume UAVs and the satellite are operated by the same operator, where UAVs can provide better transmit rate, and the transmit rate between UEs and the satellite is low. The UAVs and the satellite will charge UEs for money when proving content to UEs. There are F contents deployed on the satellite, and the set of the contents is denoted as $\mathcal{F} = \{1, 2, ..., f, ...F\}$, and the size of content f is s_f (in bits), which is limited by $0 \leq s_f \leq s^{max}$. Each content may be a short video clip, or a segment of a movie, etc. In fact, most downloadable Internet video content are in the form of video clips or segments, which usually continues a few seconds. The system works in a time-slotted manner, and we use $\mathcal{T} = \{1, 2, ..., t, ..., T\}$ and T to denote the set and number of time slots, respectively. The length of a time slot is denoted as $\triangle t$, which is assumed to be much longer than the delay caused by data transmission. Figure 1 depicts an example of the concerned scenario.

MEC subsystem

Content placement and replacement model

As mentioned, both UAVs and satellite have certain storage spaces for content caching, thus to provide proximal content to UEs. Denote the content placement of UAV j as $\delta_{f,j}(t) \in \{1, 0\}$, where $\delta_{f,j}(t) = 1$ mean to deploy content f at UAV j in time slot t, and $\delta_{f,j}(t) = 0$ otherwise. However, the caching capacity of each UAV j is limited, so it could not deploy all the contents in $\mathcal{F}$, and we have to keep optimizing the content by content placement and replacement. Denote the content replacement of UAV as $\rho_{f,j}(t) \in \{1, 0\}$, where $\rho_{f,j}(t) = 1$ mean to remove content f from the UAV j in time slot t, and $\rho_{f,j}(t) = 0$ otherwise.

We denote the set of existing contents at UAV j in time slot t as $\mathcal{N}_j(t)$, where $\mathcal{N}_j(t)$ is a $F \times 1$ vector, and each term takes a binary value, which is denoted as $n_{f,j}(t)$, i.e., $\mathcal{N}_j(t) = \{n_{f,j}(t)\}$, $f \in \mathcal{F}$. When $n_{f,j}(t) = 1$, content f exists on UAV j in time slot t after task placement and replacement, and $n_{f,j}(t) = 0$ otherwise. We denote $\mathcal{N}(t) = \{\mathcal{N}_j(t)\}_{F \times J}$, $j \in \mathcal{J}$, which is called the content existing matrix. Based on the above notations, we present the constraints on task placement and replacement. At this point we have not considered the impact of blockchain deployment on cache resource constraints. First, the finite caching capacity at UAV j can be given by

$$\sum_{f\in\mathcal{F}} n_{f,j}(t)s_f \le S_j. \tag{1}$$

Moreover, for content placement and replacement at UAV j, we have

$$\begin{aligned} n_{f,j}(t) = n_{f,j}(t-1) + \delta_{f,j}(t)\cdot\mathbb{I}\{n_{f,j}(t-1)=0\} \\ - \rho_{f,j}(t)\cdot\mathbb{I}\{n_{f,j}(t-1)=1\}, \end{aligned} \tag{2}$$

where $\mathbb{I}\{\cdot\}$ is the symbolic function, where $\cdot$ holds when $\mathbb{I}\{\cdot\}=1$, and otherwise, $\mathbb{I}\{\cdot\}=0$; the term $+\delta_{f,j}(t)\cdot\{n_{f,j}(t-1)$ means that we can deploy content f to UAV in time slot t, only when content f does not exist in UAV at time slot $t-1$; and the term $-\rho_{f,j}(t)\cdot\{n_{f,j}(t-1)$ means that we can remove content f from UAV in time slot t, only when content f have existed in UAV at time slot $t-1$.

Remark 1

Please note that, the situation that $\delta_{f,j}(t)=0$ and $\rho_{f,j}(t)=0$ may also exist, which means that we do not place or replace content f from UAV j, this means $n_{f,j}(t)=n_{f,j}(t-1)$.

We should also have the following constraint of task placement and replacement in UAV j,

$$\mathrm{All}\big(\delta_{\mathrm{f,j}}(\mathrm{t}),\rho_{\mathrm{f,j}}(\mathrm{t})\big)=0, \tag{3}$$

where $\mathrm{All}(\cdot)$ means that only when all the terms in bracket equal to 1, $\mathrm{All}(\cdot)=1$, and in other cases, we have $\mathrm{All}(\cdot)=0$. This constraint is used to restraint that the situation when $\delta_{f,j}(t)=1$ and $\rho_{f,j}(t)=1$ can not appear simultaneously, i.e., we can not place and remove a content form the UAV simultaneously.

UAV communication model

We consider a general three-dimensional Cartesian coordinate system, where the coordinate of each UE i is $\mathbf{q}_i=\{q_{i,x},q_{i,y}\}$, and the horizontal coordinate of each UAV j in time slot t is denoted as $\mathbf{u}_j(t)=\{x_j(t),y_j(t)\}$. It is assumed that the height of each UAV keeps fixed, and it is denoted by H.

Similar to many studies [19, 20], the wireless channels between UAVs and ground UEs are assumed to be mainly determined by LoS links. Denote the Euclidean distance between UAV j and UE i as $d_{i,j}(t)$, which is given by

$$d_{i,j}(t)=\sqrt{H^2+\|\mathbf{q}_i-\mathbf{u}_j(t)\|^2}. \tag{4}$$

Therefore, the channel gain between UAV j and UE i in time slot t can be given by

$$\begin{aligned} h_{i,j}(t) &= \beta_0 d_{i,j}^{-2}(t) = \frac{\beta_0}{H^2+\|\mathbf{q}_i-\mathbf{u}_j(t)\|^2} \\ &= \frac{\beta_0}{H^2+(q_{i,x}-x_j(t))^2+(q_{i,y}-y_j(t))^2}, \end{aligned} \tag{5}$$

where β_0 represents the channel power gain at a reference distance $d_0=1$ m. Then the downlink content delivery rate can be given by

$$R_{i,j}(t)=B_{i,j}(t)\log\left(1+\frac{P_{i,j}(t)h_{i,j}(t)}{\sigma^2+\varsigma_{i,j}(t)P_{i,j}^{LOS}(t)}\right), \tag{6}$$

where σ^2 denotes the white Gaussian noise power, $B_{i,j}(t)$ and $P_{i,j}(t)$ are the bandwidth and transmit power between UAV j and UE i, $P_{i,j}^{LOS}(t)$ denotes the penetration loss, and $\varsigma_{i,j}(t)$ represents whether there is block between UAV j and UE i in time slot t, where $\varsigma_{i,j}(t)=0$ denotes no blockage exists, and $\varsigma_{i,j}(t)=1$ represents there is blockage between them).

Remark 2

In this paper, the bandwidth and transmit power of each UAV j is uniformly allocated among all its serving UEs [21].

At each time slot t, UAVs flies from position $\mathbf{u}_j(t)=\{u_{j,x}(t),u_{j,y}(t)\}$ to a new position $\mathbf{u}_j(t+1)=\{u_{j,x}(t+1),u_{j,y}(t+1)\}$, where

$$\begin{aligned} u_{j,x}(t+1) &= u_{j,x}(t)+\triangle x(t), \\ u_{j,y}(t+1) &= u_{j,y}(t)+\triangle y(t), \end{aligned} \tag{7}$$

and $\triangle x(t)$ and $\triangle y(t)$ is generated from $[-xy_{max},xy_{max}]$ randomly at each time slot.

Satellite communication model

With the development of microwave communications, direct ground-space transmission has been shown to be feasible [22]. The achievable wireless data rate R_i^S between ground IoT device i and the satellite is given as a constant, which is much smaller than the data rate between UEs and UAVs in general cases.

Content delivery model

At the beginning of each time slot t, each UE will request a desired content f, and the requests of all UEs will be collected by their attached UAVs. Assume the user request as $r_{i,f}(t)\in\{1,0\}$, where $r_{i,f}(t)=1$ means UE i request content f in time slot t, and $r_{i,f}(t)=0$ otherwise. Let $\alpha_{i,j}(t)$, $\alpha_i^S(t)$ denote access control policies of UE i, where $\alpha_{i,j}(t)=1$ means UE i access UAV j for content fetching, and $\alpha_{i,j}(t)=0$ otherwise; similarly, $\alpha_i^S(t)=1$ means UE

i access the satellite for content fetching, and $\alpha_i^S(t) = 0$ otherwise. Therefore, at each time slot, the transmitted data to UE i by the UAV j is given by

$$D_{i,j,f}(t) = r_{i,f}(t)\alpha_{i,j}(t)R_{i,j}(t)\triangle t, \tag{8}$$

and the transmitted data to UE i by the satellite is given by

$$D_{i,f}^S(t) = r_{i,f}(t)\alpha_i^S(t)R_i^S\triangle t, \tag{9}$$

and it can be easily known that $0 \le D_{i,j,f}(t), D_{i,f}^S(t) \le s_f$, combing $0 \le s_f \le s^{max}$, we have $0 \le D_{i,j,f}(t), D_{i,f}^S(t) \le D^{max}$, and $D^{max} = s^{max}$.

To describe the dynamics of content requests and delivery, we introduce two virtual queues especially for UAV j and the satellite, which are represented by $Q_j(t)$ and $Q^S(t)$, respectively, and virtual queues of UAV j and the satellite evolve according to

$$Q_j(t+1) = \left[Q_j(t) - \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} D_{i,j,f}(t)\right]^+ + \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} r_{i,f}(t)\alpha_{i,j}(t)s_f, \tag{10}$$

and

$$Q^S(t+1) = \left[Q^S(t) - \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} D_{i,f}^S(t)\right]^+ + \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} r_{i,f}(t)\alpha_i^S(t)s_f, \tag{11}$$

respectively, and where $[x]^+ = \max(x, 0)$. When the system starts to run, each user puts forward their own content requests and completes user tasks according to access control. At the beginning time t, there is no backlog problem in the queue, so we record all the initialization values of the whole queue as 0 and update the queue in the subsequent time slots. Please note that, UAV j and the satellite do not actually buffer the data of the virtual queues, since both the content provider will only need to cache a backup of a content, and each cached content can be delivered to multiple UEs that request the content. The meaning of introducing the two virtual queues is that, the backlog of the virtual queues can reflect the average latency of content delivery delay at UAV j and the satellite, respectively. Specifically, for the satellite, the model considered only one satellite, so the virtual queue of the satellite is a row vector. For UAVs, since J UAVs are considered, there are J row vectors in total.

Definition 1 A discrete time queue $Q(t), t \in \mathcal{T}$ is considered to be strongly stable if it satisfies $\bar{Q} = \lim_{T\to\infty} \sup \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\{Q(t)\} < \infty$ [23].

Definition 2 If all queues are stable, the system will be stable [23].

Remark 3

Based on Little's Theorem [23], given data arrival rate, it can be known that the average queuing delay is proportional to the average queue backlog. Definitions 1 and 2 tell us that, when the whole network is stable, then the average queuing delay will be finite.

Economical revenue obtained from content delivery

The operator intend to make more money by content caching and content delivery, and it will charge UEs for money according to the transmitted data. Since UAVs could generally provide better quality of services, it will charge UEs for higher price, and the price charged by the satellite is a little lower.

Denote the price charged by UAV j and the satellite as η_j (in \$/bit) and η^S (in \$/bit), respectively, the total economical revenue obtained by the operator from content delivery is given by

$$\begin{aligned} Rev^1(t) &= \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} \eta_j D_{i,j,f}(t) + \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} \eta^S D_{i,f}^S(t) \\ &= \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} \eta_j r_{i,f}(t)\alpha_{i,j}(t)R_{i,j}(t)\triangle t \\ &\quad + \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} \eta^S r_{i,f}(t)\alpha_i^S(t)R_i^S\triangle t. \end{aligned} \tag{12}$$

Blockchain subsystem

User information, such as content request records, request frequency, etc., are private and should be protected in content requesting and fetching. Moreover, the content provider may perform maliciously by transmitting UEs with tampered content, and UEs may perform maliciously by refusing pay for its requested content. Blockchain is a decentralized database, it can settle the above challenges and bring us with the following benefits. (i) Tamper-proof: All transactions are packaged using Merkel tree in block body, and the Merkel root is one element of block head, the block body and the block head is hashed, and the hash value is one element of the block head. The chain framework ensures the data can not be tampered with. (ii) Unforgeability: Each blockchain node will sign the transactions they send using its private key, thus prevent other malicious nodes to forge the data in the network. (iii) Non-repudiation: The algorithms are deployed on the smart contracts. When smart contract is triggered, the content provider will send the requested content to the

requester UE, and the UE will pay for the contract. The process will be enforced, so no one could play tricks and refuse to fulfill their obligations.

In our system, all UAVs will serve as the blockchain nodes. After our content fetching algorithms deployed at smart contract is executed, content provider will send the required contents, and requester UE will pay for the corresponding fees. And then, each UAV blockchain node will package these record as transactions, which will be shared to other UAVs in the network, and the block generator will package all its' received transactions into block. In blockchain system, there are two kinds of blockchain nodes, which are called the full nodes and the light-weighted nodes (which is shorted as light nodes). Full nodes will participate the block generation process and will gain rewards for successful packing, and the light nodes will only collect and forward information in consensus process. However, to serve as a full node, the UAV have to cache the whole blockchain, and a mount of caching spaces will be occupied, and the spaces for content caching will be reduced; to serve as a light-weighted node, the UAV only need to store the block head, and mass storage space will be saved for content caching, while however, the UAV will not obtain block generation reward. As many full nodes as possible is crucial to the security of the blockchain network. They are responsible for the broadcast and verification of transactions, thus maintaining the stable operation of the entire system. While light nodes do not make any contribution to the security of the network, because they do not download any copy of the blockchain, nor participate in any verification process and blockchain transaction authentication process, light nodes are just convenient to use the wallet, but also need to rely on other full nodes to provide the required information. Therefore, in the deployment process of blockchain, at least one full node is needed to provide necessary information for other light nodes. Nevertheless, since the block head will occupy some storage capacity, we assume it will be given with a little amount of reward for compensate in this paper. For this reason, we should make a decision for each UAV, whether to serve as a full node, or a light-weighted node. While full nodes are essential for the security and operation of the blockchain network, light nodes primarily serve as convenient wallets that rely on full nodes for necessary information. As such, during the deployment process of the blockchain, it is necessary to have at least one full node to support the light nodes by providing the required information.

Since full nodes will compete and all have the chance to serve as block generator, and we care for the total reward of the system, not a certain node, so we assume each full node will be awarded the same quality of reward r^{fu}, and each light-weighted node will be awarded with r^{li}, and $r^{fu} >> r^{li}$. Also, we assume the size of the blockchain is s^{fu}, and the size of block head is s^{li}, and $s^{fu} >> s^{li}$ (generally, $s^{fu} \approx 1000 s^{li}$). Let $c_j(t) \in \{1, 0\}$ denote the blockchain deployment strategy, where $c_j(t) = 1$ means UAV j serves as the full node at time slot t, and $c_j(t) = 0$ otherwise.

Remark 4

We do not take the blockchain deployment delay of full nodes and the light-weighted nodes into consideration in this paper.

Based on the above definitions, the economical revenue obtained from blockchain system can be given by

$$Rev^2(t) = \sum_{j \in \mathcal{J}} \Big(c_j(t) r^{fu} + (1 - c_j(t)) r^{li} \Big). \tag{13}$$

Our objective

In this paper, we intend to provide privacy and QoS guaranteed content delivery services to UEs, thus to maximize the total averaged long-term economical revenue of the operator. The economical revenue that the operator obtained at each time slot is given by

$$Rev(t) = Rev^1(t) + Rev^2(t). \tag{14}$$

Remark 5

Please note that, the operator can obtain economical revenue from content delivery and block mining. However, there are some conflicts. For the MEC system, by caching content on UAV, it can provide UEs with their desired content, and charge UEs with money. From the standpoint of MEC system, caching more contents will probably charge more money. However, more spaces will be required. For blcokchain system, by deploying more full nodes, more reward will be obtained. However, more spaces will also be required. Therefore, we need to carefully design the content placement, replacement and blockchain deployment strategies to deal with the contradiction.

Working process of the whole system

At the beginning of each time slot, each UE sends a content access request to its attached UAV. Based on the requests of all its served UEs, each UAV first perform content placement and replacement, based on the popularity and content size of content, the remain storage spaces of the UAV, the channel state information of its served UEs, etc. And then, each UAV decide which

access point should serve each request, i.e., provide the corresponding UE with their requested content. Then the access point sends contents to their served UEs. After content transmission, the blockchain system deployed on UAVs will work, it will record the information related in content transmission as transactions, the information includes which access point serve which UE, whether the UE have paid fees to the content provider, etc., in order to guarantee the happened transactions will not be tampered with, and thus to provide evidence when there are some disputes. The transactions will be packaged by the block generator into a block, and will be added to the blockchain after verification.

Problem formulation

Problem formulation

We propose to maximize the long-termed averaged economical revenue of the operator, by jointly optimizing the content placement $\boldsymbol{\delta}(t) = \{\delta_{f,j}(t)\},\ f \in \mathcal{F}, j \in \mathcal{J}, t \in \mathcal{T}$, content replacement $\boldsymbol{\rho}(t) = \{\rho_{f,j}(t)\}, f \in \mathcal{F}, j \in \mathcal{J}, t \in \mathcal{T}$, the access control of UAVs $\boldsymbol{\alpha}(t) = \{\alpha_{i,j}(t), \alpha_i^S(t)\}$, $i \in \mathcal{I}, j \in \mathcal{J}, t \in \mathcal{T}$, and the blockchain deployment strategy $\mathbf{c}(t) = \{c_j(t)\},\ j \in \mathcal{J}, t \in \mathcal{T}$. Our problem is formulated as

$$
\begin{aligned}
(\boldsymbol{\mathcal{P}_1}): \quad & \max_{\boldsymbol{\delta}(t),\boldsymbol{\rho}(t),\boldsymbol{\alpha}(t),\mathbf{c}(t)} \overline{Rev} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\{Rev(t)\} \\
\text{s.t. (C1)}: \quad & \delta_{f,j}(t), \rho_{f,j}(t) \in \{0,1\}, f \in \mathcal{F}, j \in \mathcal{J}, \\
\text{(C2)}: \quad & \alpha_{i,j}(t), \alpha_i^S(t) \in \{0,1\}, i \in \mathcal{I}, j \in \mathcal{J}, \\
\text{(C3)}: \quad & c_j(t) \in \{0,1\}, j \in \mathcal{J}, \\
\text{(C4)}: \quad & \delta_{f,j}(t) \cdot \mathbb{I}\{n_{f,j}(t-1) = 0\} = \delta_{f,j}(t), f \in \mathcal{F}, j \in \mathcal{J}, \\
\text{(C5)}: \quad & \rho_{f,j}(t) \cdot \mathbb{I}\{n_{f,j}(t-1) = 1\} = \rho_{f,j}(t), f \in \mathcal{F}, j \in \mathcal{J}, \\
\text{(C6)}: \quad & \mathrm{All}\big(\delta_{\mathrm{f,j}}(\mathrm{t}), \rho_{\mathrm{f,j}}(\mathrm{t})\big) = 0, \mathrm{f} \in \mathcal{F}, \mathrm{j} \in \mathcal{J}, \\
\text{(C7)}: \quad & \sum_{f\in\mathcal{F}} n_{f,j}(t) s_f + c_j(t) s^{fu} + (1 - c_j(t)) s^{li} \le S_j,\ j \in \mathcal{J}, \\
\text{(C8)}: \quad & \sum_{j\in\mathcal{J}} \alpha_{i,j}(t) + \alpha_i^S(t) = 1, i \in \mathcal{I}, \\
\text{(C9)}: \quad & 1 \le \sum_{j\in\mathcal{J}} c_j(t) \le J, j \in \mathcal{J}, \\
\text{(C10)}: \quad & \textit{All virtual queues are mean rate stable.}
\end{aligned}
\tag{15}
$$

In problem $(\boldsymbol{\mathcal{P}_1})$, (C1)-(C3) are the integer constraints on content placement, content replacement, access control, and blockchain deployment strategies; (C4) is the constraint on task deployment on UAVs, which requires that we can only perform content placement only when the content does not exist on the UAV; (C5) is the constraint on task replacement on UAVs, which requires that we can only perform content replacement when the content have existed on the UAV; (C6) requires that we can not conduct content placement and replacement simultaneously on a UAV; (C7) requires that the occupied storage space of the cached contents and the blockchain should not exceed the storage capacity of each UAV; (C8) requires each UE can only access one access point, i.e., the satellite, or a UAV; (C9) requires that the count of full nodes cannot exceed the number of drones since each drone can deploy at most one full node. Therefore, the summation of the binary variables for blockchain deployment needs to be less than or equal toJ (the number of UAVs).On the other hand, full nodes are essential in the blockchain system. Ensuring an adequate number of full nodes is crucial for the performance of the blockchain system. Hence, the summation result also needs to be greater than or equal to 1; and (C10) guarantees the network is stable, i.e., all required contents can be delivered within finite time, and thus guarantees the QoS of content fetching for UEs.

Remark 6

Please note that we omit the $t \in \mathcal{T}$ in each constraint for concise [24].

Remark 7
We assume the satellite, all UAVs and UEs work in an ideal synchronization state.

The formulated problem ($\boldsymbol{\mathcal{P}_1}$) is rather hard to solve, since it is a stochastic optimization problem and designed for long-term optimization, the decisions in each time slot is strongly intertwined with each other, so we can not perform optimization in each time slot under unknown future time slots' information, such as real-time channel gains, and the stochastic user content requests, etc.

Problem transformation using Lyapunov optimization

Lyapunov optimization theory is designed for long-term optimization, through employing multi-slot drift analysis, it can capture the time-scale temporal diversity, and can solve long-term optimization problems through settling deterministic problems of each time slot, without needing any information about the future time slots. In the following, we will employ Lyapunov theory to design efficient online algorithms to solve our problem.

Let $\mathbf{Q}(t) = \{Q_1(t), ..., Q_J(t)\}$, and we denote $\boldsymbol{\Theta}(t) = \{\mathbf{Q}(t), Q^S(t)\}$, then the Lyapunov function can be given by

$$L(\boldsymbol{\Theta}(t)) \triangleq \frac{1}{2}\sum_{j\in\mathcal{J}} Q_j^2(t) + \frac{1}{2}Q^{S2}(t). \tag{16}$$

Then the one-time-slot conditional Lyapunov drift function $\Delta(\boldsymbol{\Theta}(t))$ can be given by

$$\Delta(\boldsymbol{\Theta}(t)) \triangleq \mathbb{E}\{L(\boldsymbol{\Theta}(t+1)) - L(\boldsymbol{\Theta}(t))|\boldsymbol{\Theta}(t)\}. \tag{17}$$

Next, we deduce the upper bound of $L(\boldsymbol{\Theta}(t+1)) - L(\boldsymbol{\Theta}(t))$.

Theorem 1 *For any $Q \geq 0, b \geq 0, A \geq 0$, we have*

$$(max[Q-b,0]+A)^2 \leq Q^2 + A^2 + b^2 + 2Q(A-b). \tag{18}$$

According to Theorem 1, *we know*

$$\begin{aligned} &L(\boldsymbol{\Theta}(t+1)) - L(\boldsymbol{\Theta}(t)) \\ &\leq \sum_{j\in\mathcal{J}} Q_j(t) \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} \left(r_{i,f}(t)\alpha_{i,j}(t)s_f - D_{i,j,f}(t)\right) \\ &+ Q^S(t) \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} \left(r_{i,f}(t)\alpha_i^S(t)s_f - D_{i,f}^S(t)\right) + B, \end{aligned} \tag{19}$$

where

$$\begin{aligned} B = &\frac{1}{2}\sum_{j\in\mathcal{J}}\left[\Big(\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} D^{max}\Big)^2 + \Big(\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} r_{i,f}(t)\alpha_{i,j}(t)s^{max}\Big)^2\right] \\ &+ \frac{1}{2}\left[\Big(\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} D^{max}\Big)^2 + \Big(\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}} r_{i,f}(t)\alpha_i^S(t)s^{max}\Big)^2\right]. \end{aligned} \tag{20}$$

On the basis of Lyapunov optimization, so as to maximize $\overline{Rev}$ in our formulated problem ($\boldsymbol{\mathcal{P}_1}$), the drift-plus-penalty function $\Delta(\boldsymbol{\Theta}(t)) - V\mathbb{E}\{U(t)|\boldsymbol{\Theta}(t)\}$ need to be considered, which is bounded by (21), where the parameter V is a non-negative constant which is used for striking a balance between the revenue of the operator and the queue backlog.

$$\begin{aligned} &\Delta(\boldsymbol{\Theta}(t)) - V\mathbb{E}\{U(t)|\boldsymbol{\Theta}(t)\} \\ &\leq \sum_{j\in\mathcal{J}}\mathbb{E}\left\{Q_j(t)\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\left(r_{i,f}(t)\alpha_{i,j}(t)s_f - D_{i,j,f}(t)\right)\Big|\boldsymbol{\Theta}(t)\right\} + \mathbb{E}\left\{Q^S(t)\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\left(r_{i,f}(t)\alpha_i^S(t)s_f - D_{i,f}^S(t)\right)\Big|\boldsymbol{\Theta}(t)\right\} \\ &\quad - V\mathbb{E}\left\{Rev(t)\big|\boldsymbol{\Theta}(t)\right\} + B \\ &= B + \sum_{j\in\mathcal{J}}\mathbb{E}\left\{Q_j(t)\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\left(r_{i,f}(t)\alpha_{i,j}(t)s_f - D_{i,j,f}(t)\right)\Big|\boldsymbol{\Theta}(t)\right\} + \mathbb{E}\left\{Q^S(t)\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\left(r_{i,f}(t)\alpha_i^S(t)s_f - D_{i,f}^S(t)\right)\Big|\boldsymbol{\Theta}(t)\right\} \\ &\quad - V\mathbb{E}\left\{\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\eta_j r_{i,f}(t)\alpha_{i,j}(t)R_{i,j}(t)\triangle t + \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\eta^S r_{i,f}(t)\alpha_i^S(t)R_i^S\triangle t + \sum_{j\in\mathcal{J}}\left(c_j(t)r^{fu} + (1-c_j(t))r^{li}\right)\Big|\boldsymbol{\Theta}(t)\right\} \\ &= B + \mathbb{E}\left\{\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\left(Q_j(t)r_{i,f}(t)\alpha_{i,j}(t)s_f - Q_j(t)D_{i,j,f}(t) - V\eta_j r_{i,f}(t)\alpha_{i,j}(t)R_{i,j}(t)\triangle t\right)\right. \\ &\quad \left. + \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\left(Q^S(t)r_{i,f}(t)\alpha_i^S(t)s_f - Q^S(t)D_{i,f}^S(t) - V\eta^S r_{i,f}(t)\alpha_i^S(t)R_i^S\triangle t\right) - \sum_{j\in\mathcal{J}}V\left(c_j(t)r^{fu} + (1-c_j(t))r^{li}\right)\Big|\boldsymbol{\Theta}(t)\right\}. \end{aligned} \tag{21}$$

For notation simplicity, as shown in Eq. (22), we make the following formula abbreviation.

$$\begin{aligned} U(t) = & \sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\left(Q_j(t)r_{i,f}(t)\alpha_{i,j}(t)s_f - Q_j(t)D_{i,j,f}(t) - V\eta_j r_{i,f}(t)\alpha_{i,j}(t)R_{i,j}(t)\triangle t\right) \\ & + \sum_{i\in\mathcal{I}}\sum_{f\in\mathcal{F}}\left(Q^S(t)r_{i,f}(t)\alpha_i^S(t)s_f - Q^S(t)D_{i,f}^S(t) - V\eta^S r_{i,f}(t)\alpha_i^S(t)R_i^S\triangle t\right) - \sum_{j\in\mathcal{J}}V\left(c_j(t)r^{fu} + (1-c_j(t))r^{li}\right) \end{aligned} \tag{22}$$

According to stochastic optimization theory, the joint task placement, task replacement, access control, and blockchain deployment optimization at time slot t could be realized by minimizing the upper bound of the drift-plus-penalty function as defined in (23) as follows[1]

$$\begin{aligned} &(\boldsymbol{\mathcal{P}_2}): \min_{\boldsymbol{\delta}(t),\boldsymbol{\rho}(t),\boldsymbol{\alpha}(t),\mathbf{c}(t)} U(t) \\ &\text{s.t. (C1)} - \text{(C9)}. \end{aligned} \tag{23}$$

By conducting the above transformations, the backlog of queues $Q_j(t)$ and $Q^S(t)$ can be controlled at small values, and the queue stability constraint (C10) in problem $(\boldsymbol{\mathcal{P}_1})$ can be satisfied and thus be removed from $(\boldsymbol{\mathcal{P}_1})$ [23]. Therefore, our originally formulated temporal coupling problem $(\boldsymbol{\mathcal{P}_1})$ is converted into a deterministic each-time-slot optimization problem, i.e., Problem $(\boldsymbol{\mathcal{P}_2})$. However, $(\boldsymbol{\mathcal{P}_2})$ is a binary integer programming problem with many constraints which coupling the binary variables tightly, making it still hard to solve [25, 26].

Using fireworks algorithm to solve the each-time-slot optimization problem $(\boldsymbol{\mathcal{P}_2})$

Heuristic algorithms are effective for solving troublesome problems with low complexity, and sub-optimal feasible solutions can be obtained. Fireworks algorithm [27] is an effective heuristic algorithm that performs well both in convergence speed and convergence accuracy. In this section, we use fireworks algorithm to settle the each-time-slot optimization problem $(\boldsymbol{\mathcal{P}_2})$. We first present the preliminaries of fireworks algorithm, and then we adopt it in our problem.

Preliminaries of fireworks algorithm

When we set off a firework, a large wave of sparks will be generated around the firework in its adjacent local space. While we use fireworks algorithm in solving optimization problems, the explosion process of a firework can be regarded as conducting local search around the solution that the firework stands for (which is shorted as firework solution), and find out some feasible solutions adjacent to the firework. To improve the global searching capabilities of fireworks algorithm, mutation sparks are introduced, which will generate feasible solutions far away from the firework solution. The process of using fireworks algorithm to solve $f(x) = y$, i.e., to determine an x that satisfies $f(x) = y$ can be conducted like this. We continually set off a set of fireworks in potential space to generate some explosion and mutation sparks, until the algorithm converges, i.e., the performance of a spark or firework is fairly near the point x. Imitating the process of igniting fireworks, fireworks algorithm is presented in [27].

Using fireworks algorithm to solve $(\boldsymbol{\mathcal{P}_2})$

Resorting to the original fireworks algorithm, we will propose efficient algorithm to solve our problem. However, the original fireworks algorithm is very simple, which is designed for continuous solution problems with very simple constraint, where the optimization variable is a continuous scalar, and there is only one constraint on the value range of the scalar. Therefore, the original fireworks algorithm can not be directly used for our problem for several reasons. First, the variables of our problem are all binary; second, our variables contain four set of matrices, which is much complex than the original scalar; and third, our variables are tightly coupled with many constraints, which is also much complicated then that of the original fireworks algorithm. Therefore, we need to make some improvement to reform the original fireworks algorithm to adapt to our problem as follows.

The form of fireworks and sparks

In this paper, a firework or a spark stands for a feasible solution to problem $(\boldsymbol{\mathcal{P}_2})$, i.e., a solution to joint cache placement, cache replacement, access control, and blockchain deployment strategy, with the constraints (C1)-(C9) guaranteed. Suppose there are totally L fireworks in each iteration, and the set of all fireworks of

[1] By the principle of opportunistic optimization theory [23], to minimize $f(t)$ could guarantee that $\mathbb{E}\{f(t)|\boldsymbol{\Theta}(t)\}$ is minimized.

each iteration as $\mathcal{L} = \{1, 2, ..., L\}$. For notation simplicity, we call a firework or a spark as an individual, and the form of an individual l is an array, whose structure is given by

$$\begin{aligned} \mathbf{\Pi}_l = & \\ & \text{(i) content existing} : [n_{f,j}(t)]_{F\times J} \\ & \text{(ii) access control} : [\alpha_{i,j}(t)]_{I\times(J+1)} \\ & \text{(iii) blockchain deployment} : [c_j(t)]_{J\times 1}, \end{aligned} \tag{24}$$

where the the index $j \in 1, ..., J$ refers to UAV, and $j = J + 1$ means the satellite.

Fitness function and fitness value

Based on ($\boldsymbol{\mathcal{P}_2}$), we consider the objective function $U(t)$ to be the fitness function, and its value corresponding to a solution is called fitness value.

1: **Input**: The selected firework l.
2: *Step 1*: Randomly select a zero element from the content existing matrix $\mathcal{N}(t-1)$, and let it equal to 1, and the new content existing matrix is $\mathcal{N}(t)$; For example, we select a zero element $n_{2,1}(t-1)$, and let $n_{2,1}(t) = 1$, i.e., we put content 2 on UAV 1 in time slot t. Please note that constraints (C4) and (C6) can be satisfied by the action.
3: *Step 2*: Judge whether constraint (C7) is satisfied. If satisfied, the explosion spark is generated successfully. Please end.
4: *Step 3*: If (C7) is not met, perform the above steps $Step$ 1 and $Step$ 2 at most 3 times, if there's a success explosion spark, please end.
5: *Step 4*: If there's no success explosion spark, this means the caching storage of most UAVs have been full, we let the explosion spark be the same as firework l.
6: **Output**: An explosion spark of firework l, i.e., a feasible solution to problem ($\mathcal{P}_2$).

Algorithm 1 Improved Explosion Algorithm for Case 1

Explosion

There are some points for explosion in each iteration.

i) In explosion, each firework $\mathbf{\Pi}_l$ will generate χ_l explosion sparks, and χ_l can be determined by

$$\chi_l = \hat{\chi}\frac{f_{max} - f(\mathbf{\Pi}_l) + \epsilon}{\sum_{l=1}^{L}(f_{max} - f(\mathbf{\Pi}_l)) + \epsilon}, \tag{25}$$

where $\hat{\chi}$ represents the upper limit of the number of explosion sparks of a firework, and $f_{max} = max(f(\mathbf{\Pi}_l)), l \in \mathcal{L}$ is the worst fitness value of all the L fireworks, i.e., the maximum fitness value in this paper, and ϵ is a very tiny number for avoiding zero-division-error. A balance needs to be struck between the number of explosion sparks and the computational burden of the algorithm. For several randomly generated fireworks, their explosion spark numbers are determined by comparing their fitness values. Specifically, the higher the fitness value of a firework, the more explosion sparks it will have. Moreover, we define the following constraints to bound the number of explosion sparks, so the actual number of explosion sparks $\hat{\chi}_l$ of firework l can be determined by

$$\hat{\chi}_l = \begin{cases} round(a\hat{\chi}), \ if \ \chi_l < a\hat{\chi} \\ round(b\hat{\chi}), \ if \ \chi_l > b\hat{\chi}, a < b < 1, \\ round(\chi_l), \ \ otherwise \end{cases} \tag{26}$$

where $round(\cdot)$ stands for the rounding-off function, and the two parameters a and b are given constants.

1: **Input**: The selected firework l.
2: *Step 1*: Randomly select an element $c_j(t-1)$ from the blockchain deployment matrix, and change it from 0 to 1, or from 1 to 0.
3: *Step 2*: If we change $c_j(t-1)$ from 0 to 1, judge whether constraint (C7) is satisfied.
4: *Step 3*: If satisfied, an explosion spark can be generated successfully. Please end.
5: *Step 4*: If not satisfied, this means the storage capacity of the UAV is not enough to deploy a full node, then choose another UAV. If (C7) is satisfied, please end.
6: *Step 5*: If there's no success explosion spark, this means the caching storage of most UAVs have been full, we then let the explosion spark be the same as firework l. Please end.
7: *Step 6*: If we change $c_j(t-1)$ from 1 to 0, judge whether constraint (C9) is satisfied. If satisfied, an explosion spark can be generated successfully. Please end.
8: *Step 7*: If not met, this means the number of full nodes is too less to be reduced, we let the explosion spark be the same as firework l. Please end.
9: **Output**: An explosion spark of firework l, i.e., a feasible solution to problem ($\mathcal{P}_2$).

Algorithm 2 Improved Explosion Algorithm for Case 4

Input: The L fireworks.
Step 1: Randomly select a firework from the L fireworks, as mentioned, each firework is a feasible solution,which is composed of three sub-matrices, including the content existing sub-matrix, the access control sub-matrix, and the blockchain placement sub-matrix.
Step 2: Randomly choose a zero element from the content existing matrix $\mathcal{N}(t-1)$, and let it equal to 1, and constraints (C4) can be satisfied by the action.
Step 3: Randomly select an element with value 1 from the content existing matrix $\mathcal{N}(t-1)$, and let it equal to 0, meanwhile constraints (C5) is satisfied. Please note that, after conducting the above steps *Step 1* and *Step 2*, constraints (C6) is also satisfied, and the content existing matrix becomes $\mathcal{N}(t)$.
Step 4: Judge whether space constraint (C7) is satisfied. If satisfied, go to the following *Step 7*.
Step 5: If (C7) is not satisfied, please conduct the steps *Step 1*-*Step 3* for at most 3 times, if there's a success (C7), go to the following step *Step 7*.
Step 6: If (C7) is still not satisfied when the above steps *Step 1*-*Step 3* are conducted for 3 times, this means the caching storage of most UAVs have been full, we keep the content existing sub-matrix the same as that of the firework l, and go to step *Step 7*.
Step 7: Randomly select a zero element from the access control sub-matrix, and let it equal to 1. For example, we select a zero element $\alpha_{3,2}(t-1)$, and let $\alpha_{3,2}(t)=1$, i.e., we let UE 3 to access UAV 2. Meanwhile, we let the former access control variable of UE 3 equal to 0, thus constraint (C8) is satisfied.
Step 8: Randomly select an element $c_j(t-1)$ from the blockchain deployment matrix, and change it from 0 to 1, or from 1 to 0.
Step 9: If we change $c_j(t-1)$ from 0 to 1, judge whether constraint (C7) is satisfied. If satisfied, a mutation spark can be generated successfully. Please end.
Step 10: If not satisfied, this means the storage capacity of the UAV is not enough to deploy a full node, then choose another UAV. If (C7) is still not satisfied after two tries, this means the caching storage of most UAVs have been full, we keep the blockchain placement sub-matrix the same as that of firework l. Please end.
Step 11: If we change $c_j(t-1)$ from 1 to 0, judge whether constraint (C9) is satisfied. If satisfied, an mutation spark can be generated successfully. Please end.
Step 12: If not met, this means the number of full nodes is too less to be reduced, we keep the blockchain placement sub-matrix the same as that of firework l. Please end.
Output: An mutation spark, i.e., a feasible solution to problem $(\mathcal{P}_2)$.

Algorithm 3 Improved Mutation Algorithm

ii) Perform explosion operation for each firework $\mathbf{\Pi}_l$ and generating $\hat{\chi}_l$ explosion sparks. Each explosion spark is produced as follows.

Randomly choose a number from $\{1,2,3,4\}$, there are 4 cases.

♠ **Case 1:** When we choose 1, we will generate explosion sparks according to Algorithm 1.

♠ **Case 2:** When we choose 2, we will generate explosion sparks as follows.

Randomly select an element with value 1 from the content existing matrix $\mathcal{N}(t-1)$, and let it equal to 0, and the new content existing matrix is $\mathcal{N}(t)$; Please note that constraints (C5) and (C6) and (C7) can be simultaneously satisfied by the action, and an explosion spark can be generated successfully.

♠ **Case 3:** When we choose 3, we will generate explosion sparks as follows.

Randomly select a zero element from the access control sub-matrix, and let it equal to 1. For example, we select a zero element $\alpha_{3,2}(t-1)$, and let $\alpha_{3,2}(t)=1$, i.e., we let UE 3 to access UAV 2. Meanwhile, we let the former access control variable of UE 3 equal to 0, thus constraint (C8) is satisfied, and an explosion spark can be generated successfully.

♠ **Case 4:** When we choose 4, we will generate explosion sparks according to Algorithm 2.

Mutation

So as to increase diversity of the firework swarm, and thereby to improve the global searching performance, mutation is introduced and a mutation spark is generated according to Algorithm 3.

Selection

After conducting explosion and mutation operations, the population is composed by different kinds of individuals, i.e., the primary fireworks, the new generated explosion sparks and mutation sparks. Based on the population, we will generate the fireworks of the next generation according to the following steps. First, the individual whose fitness value is the best (i.e., the smallest) is considered as a firework for the following next generation, thus to improve the performance of the swarm. The rest $L-1$ fireworks are chosen according to roulette wheel method and thus to improve the diversity of the population, where the probability of selecting an individual $\mathbf{\Pi}_l$ is

$$p(\mathbf{\Pi}_l)=\frac{D(\mathbf{\Pi}_l)}{\sum\limits_{j\in\mathcal{K}} D(\mathbf{\Pi}_j)}, \tag{27}$$

where $D(\mathbf{\Pi}_l)=\sum_{j\in\mathcal{K}}||\mathbf{\Pi}_l-\mathbf{\Pi}_j||$ denotes the Euclidean distance between two individuals, and $\mathcal{K}$ represents the collection of the current population, i.e., the population without the optimal fitness value individual. In the selection process, the individuals with longer Euclidean distance

away from the others will easily be chosen as the firework of the following generation. By this way, we use multiple criteria to select the firework population of the next generation, instead of only relying on a single fitness value standard, which can improve the diversity of the population, and improve the global searching performance of the algorithm.

FA based each-time-slot joint optimization algorithm

To employ FA to solve our each-time-slot optimization problem ($\boldsymbol{\mathcal{P}_2}$), we will first initialize the necessary parameters and some feasible solutions as the first-generation fireworks, and then we start iterations. In each generation of iteration, we perform the following steps in sequence: compute each firework's fitness value, calculate the number of explosion sparks and generate explosion sparks for each firework, generate a mutation spark, calculate the fitness value of all sparks, and select L fireworks for the following generation from the population. We perform the above steps until the last generation, where the best individual of the last population is considered as the joint solution to our problem ($\boldsymbol{\mathcal{P}_2}$). Detailed process is given in Algorithm 4.

Initialization:

Set the number of UEs, UAVs, and contents as I, J and F, the parameters of firework algorithm, including number of fireworks L, the amplitude upper bound $\hat{\chi}$ of explosion sparks, the number of iteration U, etc;

Initialize a swarm with L fireworks, i.e., L feasible solutions to ($\mathcal{P}_2$).

Iteration:

while 1 or $u <= U$ **do**

  for $l <= L$ **do**

    Calculate the fitness value of each firework $\mathbf{\Pi}_l$.

    Calculate $\hat{\chi}_l$ based on (25) and (26).

    for $p <= \chi_l$ **do**

      Conduct explosion operation, and generate explosion sparks, where each one is generated according to a case among the four cases, and Algorithm 1 and Algorithm 2 may be involved.

      Calculate the fitness values of each explosion spark.

    end for

  end for

  Generate a mutation spark via Algorithm 3, and calculate its fitness value.

  Consider the best individual as a firework of the following generation, and select the other $L-1$ fireworks from the rest individuals randomly according to the probability in (27).

end while

Pick out the best individual of the last generation of iteration as the solution $\mathbf{\Pi}^*$ to ($\mathcal{P}_2$).

Output: $\mathbf{\Pi}^*$, i.e., the joint content placement, content replacement, access control, and blockchain deployment strategies to each-time-slot problem ($\mathcal{P}_2$).

Algorithm 4 Improved Constrained Fireworks Algorithm based Each-time-slot Joint Optimization Algorithm

Simulation results and discussions

Next, we will verify the performance of our proposed algorithms through simulation. The specific parameters are detailed in a three-column Table 1, including the representation, meaning, and default values of each parameter and the parameters will keep unchanged in the following simulations except for specifical illustration. Please note that, the LoS link indicator $\varsigma_{i,j}(t)$ is generated randomly from {0, 1}.

In the following, our main innovation is the constrained binary explosion and mutation algorithms, to evaluate the performance of our algorithm, we compare it with the following algorithms.

i) OneES: which is shorted for "One-explosion-spark". In this algorithm, only one explosion spark is generated in the explosion process of each firework in the each-time-slot optimization solving, and others are the same with our joint algorithm.

ii) No mutation: In this algorithm, only explosion is conducted under our proposed four cases using our proposed algorithms, and no mutation operation, and others are the same with our joint algorithm. This algorithm can show the performance of our proposed algorithm. For notation simplicity, we denote our proposed joint algorithm as "Proposed" in the following sections. Please note that, in the following figures, each point of the figure is plotted based on the an average of 2000 runs.

Performance evaluation of Algorithm 1

In Figs. 2, 3 and 4, we demonstrate the economical revenue, the average queue backlog of all UAVs, and the queue of the satellite versus the number of UEs, respectively. As can be seen from Fig. 2, with the number of UEs increase, the total revenue of the three algorithms all increase, this is in line with our intuition, since when there are more UEs, and each UE will request a content either from a UAV or the satellite, and all UAVs and the satellite are all operated by the same operator, so more economical revenue will be made. It can also be seen that, the revenue that the proposed algorithm obtains is the most under different user numbers. In Figs. 3 and 4, we can see the average queue backlog of the UAVs and the queue length of the satellite also increase, this is also easy to understand, when there are more UEs, longer queue it will be. Also, the the proposed algorithm performs the best, whose queue backlog is always the minimum under different user numbers.

Let's further analyze the reason. Each explosion spark is a feasible solution close to the firework, since there's only one term, and may be zero term, different from the original firework. So an explosion spark represents the local searching capability of the fireworks algorithm. More explosion sparks, and stronger local searching capability. Since in OneES algorithm, only one

Table 1 Simulation Parameter Settings

Symbol	Definition	Value
I	Number of UEs	10
J	Number of UAVs	4
F	Number of contents	20
Δt	Length of a time slot	0.1 s
s_f	Size of each content	[0,30] Mb
R_i^S	Data rate between UEs and satellite	[20,60] Kbps
S_j	Storage capacity of each UAV	20 G
B_j	Bandwidth of each UAV	(0.1,2) GHz
$q_{i,x}$, $q_{i,y}$, $x_j(t)$, $y_j(t)$	Horizontal area of UEs and UAVs	[-1000, 1000] m
H	Flight height of UAV	100 m
xy_{max}	Max distance varifiaction bound	25 m
β_0	Channel power gain at 1 m	-50dB
$P_{i,j}^{LOS}(t)$	Penetration loss	20dB
s^{fu}	Size of blockchain	20 Mbit
s^{li}	Size of blockchain head	20 Kbit
η_j	Price of UAV communication	$[1, 20] \times 10^{-6}$ \$/bps
η^S	Price of satellite communication	1×10^{-9} \$/bps
L	Number of fireworks	4
$\hat{\chi}$	Maximum number of explosion sparks	6
a and b	Parameters	0.2, 0.8
V	Parameter in Lyapunov Optimization	$1 * 10^{10}$

explosion spark is generated, making its local searching capability much worse than the Proposed, so its performance will degrade. On the other hand, each mutation spark is a feasible solution far away from the firework, and there at most four terms different from the original firework. Therefore, a mutation spark represents the global searching capability of the fireworks algorithm. In No mutation algorithm, we do not conduct mutation operation, so there's no global searching capability, and the algorithm will easy to fall in local optimal solution, making it performs worse. Therefore, the gap between the Proposed and No mutation demonstrate the performance gain brought by the proposed mutation algorithm.

In Figs. 5 and 6, we show how the number of mutation sparks affect the total profit and the queue backlog, respectively. Since mutation sparks represent the global searching performance of the fireworks algorithm, more mutations sparks, stronger global searching capabilities, and better performance fireworks algorithm will gain, and this can be demonstrated in Figs. 5 and 6. In Fig. 5, the obtained profit of both Proposed and OneES generally grow with the number of mutation sparks increase. Since there's no mutation operation in No mutation algorithm, its obtained profit generally keeps in the same level. We can also find that, the Proposed algorithm can gain the maximum economical profit among the three algorithms in different number of mutation sparks, and the performance gap between Proposed and the other two algorithm demonstrate the performance gain brought by the proposed explosion and mutation algorithms. On the other hand, when the performance of fireworks algorithm increases, less data stays in the queue. For this reason, we can also find in Fig. 6 that the queue backlog in both the UAVs and the satellite decrease with the number of mutation sparks increase, so Fig. 6 further demonstrate the performance of our proposed mutation algorithm works well.

In Fig. 7, we show the relationship between the economical revenue and the bandwidth of each UAV. As can be seen from Fig. 7, the economical revenue of the three algorithms increase with the increase of the bandwidth of each UAV, which is in line with our intuition, because when the UAV has a larger bandwidth, the content it can transmit also increases, and the number of users it can serve increases. That leads to an increase in average economical revenue. At the

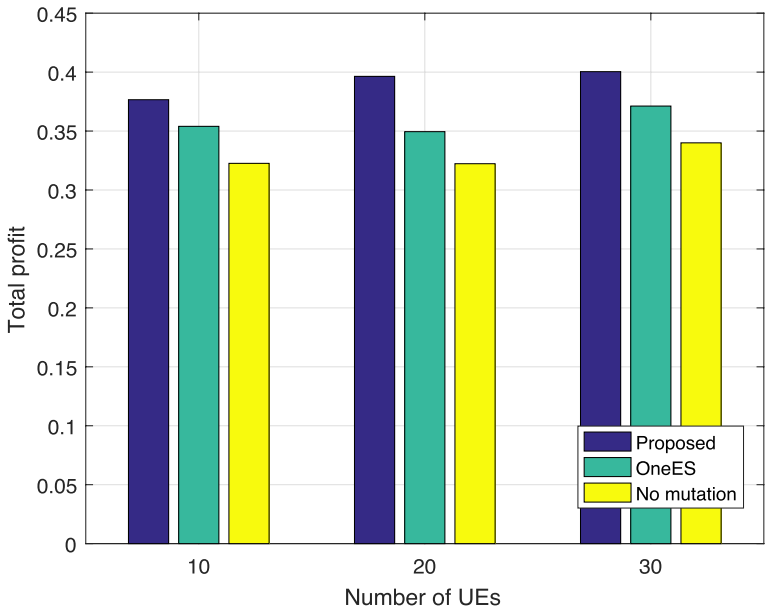

Fig. 2 Profit vs. *N*

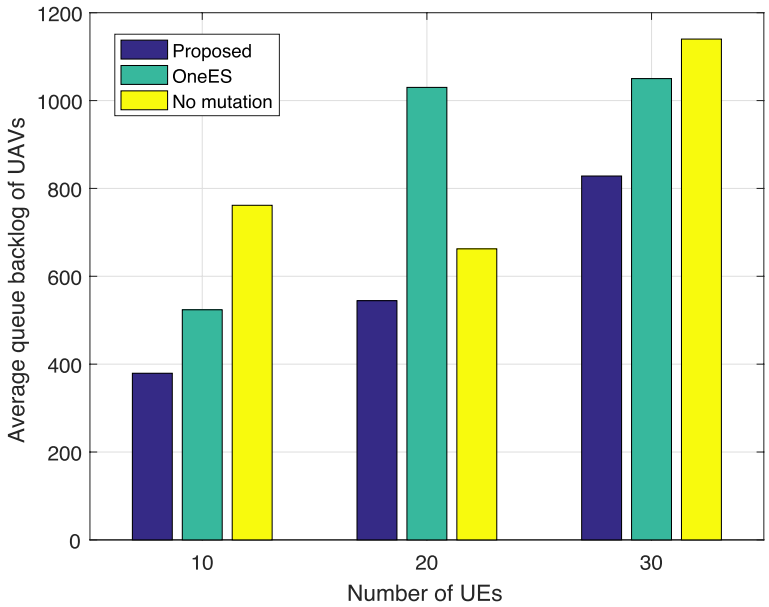

Fig. 3 Queue backlog of UAVs vs. *N*

same time, it is not difficult to see that the proposed algorithm gains more revenue than the baseline algorithms, and the gap increases with the increase of bandwidth.

Conclusions

In this paper, we discussed trust content delivery issues in an edge caching and blockchain enabled SAGIN network. We considered to optimize both mobile edge caching and blockchain systems. In edge caching system, we investigated the content placement and replacement in content deployment stage, and decided whether each UE should access a UAV or the satellite for content fetching in content delivery stage. For blockchain system, we considered the blockchain placement optimization, where we optimized whether to deploy the full blockchain to each UAV to be a full node for more revenue, or to deploy the block head to be a light-weighted node for saving more

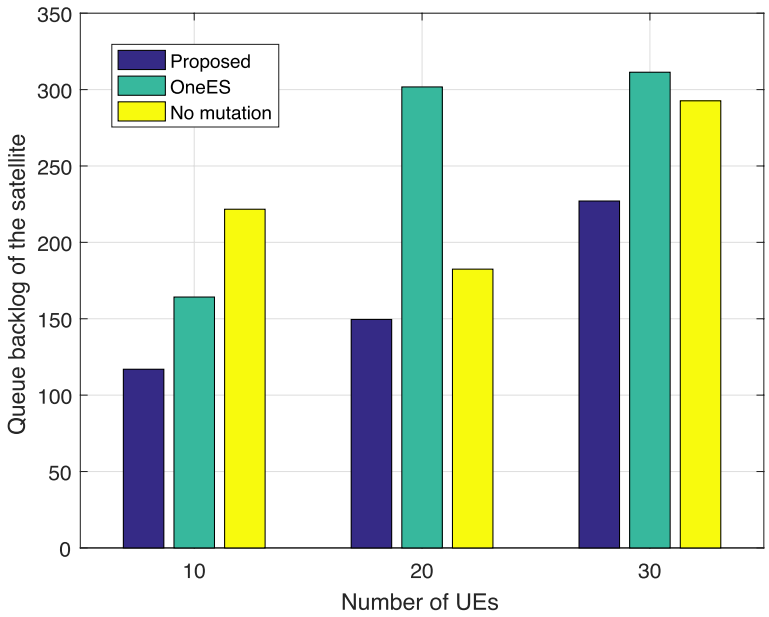

Fig. 4 Queue backlog of the satellite vs. *N*

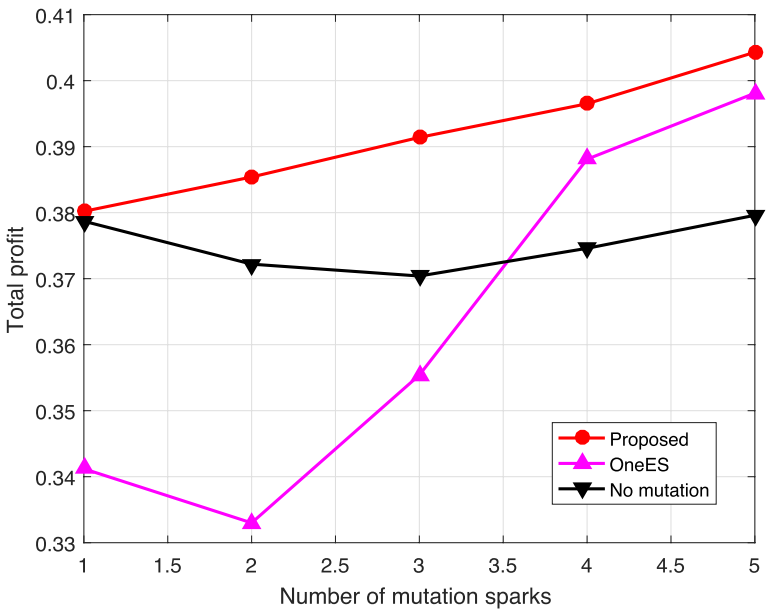

Fig. 5 Profit vs. number of mutation sparks

storage spaces for edge caching system. By the above optimization of the two systems, we intended to improve the long-term averaged economical revenue about the operator. Our problem is rather troublesome to solve, since the optimization is tightly coupled among different time slots, and the variables is also strongly intertwined

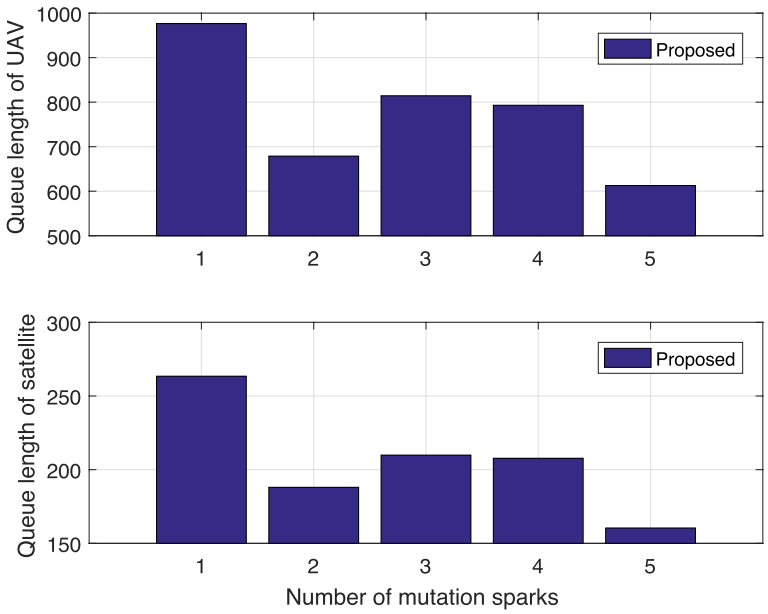

Fig. 6 Queue backlog vs. number of mutation sparks

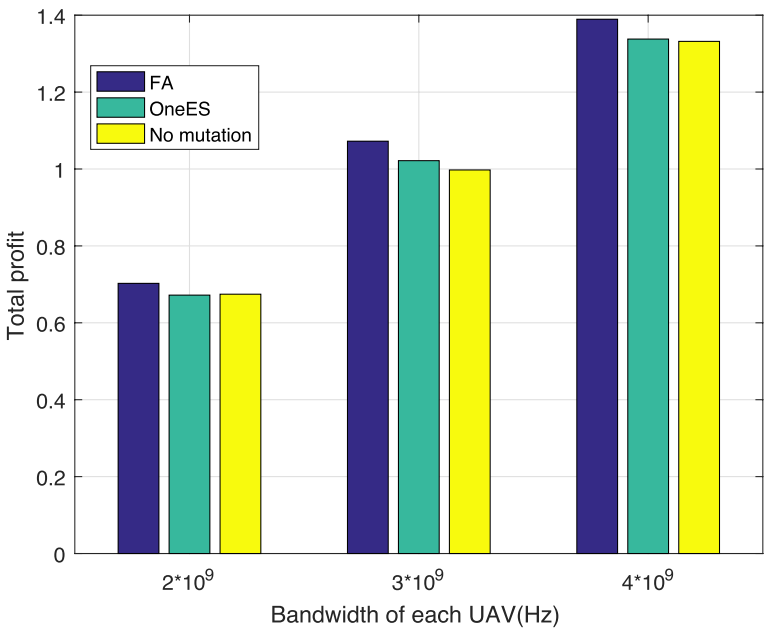

Fig. 7 Profit vs. bandwidth of each UAV

within the optimization of each time slot. To effectively solve the problem with low-complexity, we first employed Lyapunov optimization to decouple the problem into an each-time-slot optimization, and then proposed an improved constrained fireworks algorithm to solve the each-time-slot optimization problem, where optimized content placement and replacement, access control, and blockchain deployment is obtained. Our simulation results showed that the proposed algorithm performed well in operator economical revenue improving.

Acknowledgements
This work was supported in part by the Natural Science Foundation of China under Grant 62271391, in part by the Serving Local Special Scientific Research Project of Education Department of Shaanxi Province under Grant 21JC032.

Authors' contributions
Jianbo Du conduced system model and algorithm design, and writing the main manuscript text; Jiaju Lv performed simulation; and Guangyue Lu conducted performance analysis.

Availability of data and materials
Not applicable.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 31 March 2023 Accepted: 6 June 2023
Published online: 29 June 2023

References

1. Kang J, Li X, Nie J, Liu Y, Xu M, Xiong Z, Niyato D, Yan Q (2022) Communication-efficient and cross-chain empowered federated learning for artificial intelligence of things. IEEE Trans Netw Sci Eng 1–1. https://doi.org/10.1109/TNSE.2022.3178970
2. Du J, Yu FR, Lu G, Wang J, Jiang J, Chu X (2020) Mec-assisted immersive vr video streaming over terahertz wireless networks: A deep reinforcement learning approach. IEEE Internet Things J 7(10):9517–9529
3. Kang J, Du H, Li Z, Xiong Z, Ma S, Niyato D, Li Y (2022) Personalized saliency in task-oriented semantic communications: Image transmission and performance analysis. IEEE J Sel Areas Commun 41(1):186–201
4. Liu J, Zhang X, Zhang R, Huang T, Yu FR (2022) Reliable and low-overhead clustering in leo small satellite networks. IEEE Internet Things J 9(16):14844–14856
5. Yuan X, Chen J, Yang J, Zhang N, Yang T, Han T, Taherkordi A (2022) Fedstn: Graph representation driven federated learning for edge computing enabled urban traffic flow prediction. IEEE Trans Intell Transp Syst. Early Access
6. Zhang R, Liu J, Liu F, Huang T, Tang Q, Wang S, Yu FR (2021) Buffer-aware virtual reality video streaming with personalized and private viewport prediction. IEEE J Sel Areas Commun 40(2):694–709
7. Qiao G, Leng S, Maharjan S et al (2019) Deep reinforcement learning for cooperative content caching in vehicular edge computing and networks. IEEE Internet Things J 7(1):247–257
8. Kwak J, Kim Y, Le LB et al (2018) Hybrid content caching in 5g wireless networks: Cloud versus edge caching. IEEE Trans Wirel Commun 17(5):3030–3045
9. Wang X, Li R, Wang C et al (2020) Attention-weighted federated deep reinforcement learning for device-to-device assisted heterogeneous collaborative edge caching. IEEE J Sel Areas Commun 39(1):154–169
10. Zhou D, Sheng M, Wu J, Li J, Han Z (2022) Gateway placement in integrated satellite-terrestrial networks: Supporting communications and internet of remote things. IEEE Internet Things J 9(6):4421–4434
11. Zhai D, Wang C, Cao H et al (2022) Deep neural network based uav deployment and dynamic power control for 6g-envisioned intelligent warehouse logistics system. Futur Gener Comput Syst 137:164–172
12. Zhang Z, Zhang Q, Miao J, Yu FR, Fu F, Du J, Wu T (2021) Energy-efficient secure video streaming in uav-enabled wireless networks: A safe-dqn approach. IEEE Trans Green Commun Netw 5(4):1892–1905
13. Zhang T, Wang Y, Yi W et al (2022) Joint optimization of caching placement and trajectory for uav-d2d networks. IEEE Trans Commun 70(8):5514–5527
14. Han D, Liao W, Peng H et al (2021) Joint cache placement and cooperative multicast beamforming in integrated satellite-terrestrial networks. IEEE Trans Veh Technol 71(3):3131–3143
15. Gu S, Sun X, Yang Z et al (2021) Energy-aware coded caching strategy design with resource optimization for satellite-uav-vehicle-integrated networks. IEEE Internet Things J 9(8):5799–5811
16. Liu S, Zheng C, Huang Y et al (2022) Distributed reinforcement learning for privacy-preserving dynamic edge caching. IEEE J Sel Areas Commun 40(3):749–760
17. Yang Y, Liu Z, Liu Z et al (2022) Joint optimization of edge computing resource pricing and wireless caching for blockchain-driven networks. IEEE Trans Veh Technol 71(6):6661–6670
18. Guo S, Hu X, Guo S et al (2019) Blockchain meets edge computing: A distributed and trusted authentication system. IEEE Trans Ind Inf 16(3):1972–1983
19. Fu F, Kang Y, Zhang Z, Yu FR, Wu T (2021) Soft actor-ccritic drl for live transcoding and streaming in vehicular fog-computing-enabled iov. IEEE Internet Things J 8(3):1308–1321
20. Li M, Si P, Yang R, Wang Z, Zhang Y (2020) Uav-assisted data transmission in blockchain-enabled m2m communications with mobile edge computing. IEEE Netw 34(6):242–249
21. Xu D, Yu K, Ritcey JA (2022) Cross-layer device authentication with quantum encryption for 5g enabled iiot in industry 4.0. IEEE Trans Ind Inf 18(9):6368–6378
22. Zhou D, Sheng M, Wang Y, Li J, Han Z (2021) Machine learning based resource allocation in satellite networks supporting internet of remote things. IEEE Trans Wirel Commun 20(10):6606–6621
23. Neely MJ (2010) Stochastic network optimization with application to communication and queueing systems. Synth Lect Commun Netw 1(1):1–211
24. Xu D, Liu L, Zhang N et al (2023) Nested hash access with post quantum encryption for mission-critical iot communications. IEEE Internet Things J. https://doi.org/10.1109/JIOT.2023.3245360
25. Sutton RS, Barto AG (2018) Reinforcement learning: An introduction. MIT press, One Rogers Street, Cambridge, MA 02142, USA
26. Zhai D, Wang C, Zhang R et al (2022) Energy-saving deployment optimization and resource management for uav-assisted wireless sensor networks with noma. IEEE Trans Veh Technol 21(6):6609–6620
27. Tan Y (2015) Fireworks algorithm, vol 10(978-3). Springer, Heidelberg

Publisher's Note