sensors

MDPI

Article

Inter-Satellite Cooperative Offloading Decision and Resource Allocation in Mobile Edge Computing-Enabled Satellite–Terrestrial Networks

Minglei Tong [1,2], Song Li [3], Xiaoxiang Wang [1,2,*] and Peng Wei [1,2]

[1] School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
[2] Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China
[3] School of Information and Control Engineering, China University of Mining and Technology, Xuzhou 221116, China
* Correspondence: cpwang@bupt.edu.cn

Abstract: Mobile edge computing (MEC)-enabled satellite–terrestrial networks (STNs) can provide task computing services for Internet of Things (IoT) devices. However, since some applications' tasks require huge amounts of computing resources, sometimes the computing resources of a local satellite's MEC server are insufficient, but the computing resources of neighboring satellites' MEC servers are redundant. Therefore, we investigated inter-satellite cooperation in MEC-enabled STNs. First, we designed a system model of the MEC-enabled STN architecture, where the local satellite and the neighboring satellites assist IoT devices in computing tasks through inter-satellite cooperation. The local satellite migrates some tasks to the neighboring satellites to utilize their idle resources. Next, the task completion delay minimization problem for all IoT devices is formulated and decomposed. Then, we propose an inter-satellite cooperative joint offloading decision and resource allocation optimization scheme, which consists of a task offloading decision algorithm based on the Grey Wolf Optimizer (GWO) algorithm and a computing resource allocation algorithm based on the Lagrange multiplier method. The optimal solution is obtained by continuous iterations. Finally, simulation results demonstrate that the proposed scheme achieves relatively better performance than other baseline schemes.

Keywords: mobile edge computing; satellite–terrestrial networks; inter-satellite cooperation; offloading decision; resource allocation

Citation: Tong, M.; Li, S.; Wang, X.; Wei, P. Inter-Satellite Cooperative Offloading Decision and Resource Allocation in Mobile Edge Computing-Enabled Satellite–Terrestrial Networks. *Sensors* **2023**, *23*, 668. https://doi.org/10.3390/s23020668

Academic Editor: Guanding Yu

Received: 17 November 2022
Revised: 25 December 2022
Accepted: 3 January 2023
Published: 6 January 2023

1. Introduction

With the development of space technology and communication technology, the number of satellites in space has increased dramatically. In the future 6G network, satellite networks can be used as supplements to terrestrial cellular networks to achieve seamless full-domain coverage [1]. In addition, the satellite mobile system also has high viability when disasters happen [2]. When terrestrial network entities are destroyed by natural disasters, such as earthquakes and typhoons, people can rely on satellites to access services. Therefore, for remote areas that are not covered by terrestrial networks or disaster areas where terrestrial networks are destroyed, satellite networks can replace or assist terrestrial networks to provide services to users. The integration of satellite networks and terrestrial networks, i.e., satellite–terrestrial networks (STNs), has received extensive attention from the academic world and the industrial world [3]. STNs have advantages such as wide coverage, strong robustness, and strong damage resistance.

Since the rise in popularity of smartphones and Internet of Things (IoT) devices, various applications have been emerging endlessly in people's daily life. Some of them,

such as interactive games, face recognition, and augmented reality, require large amounts of computing resources [4]. Due to the physical size limitation and energy supply constraints, the computing capacity of devices is limited [5]. IoT devices compute tasks of these applications locally, which significantly imposes delay and reduces the quality of the experience.

Mobile edge computing (MEC) provides information technology and cloud computing capabilities within the radio access networks, which are extremely close to users [6]. Edge servers are modular and miniaturized because they host cloud services that are scaled to the edge [7]. MEC servers can be deployed at multiple locations, such as base stations (BSs), access points, and satellites. Computation offloading technology makes it possible for users to utilize the computing capabilities at the edge of the network [8]. Due to MEC, the delay can be reduced and the quality of service can also be improved [9].

In this paper, we consider applying MEC technology to STNs, i.e., MEC-enabled STNs. Low earth orbit (LEO) satellite constellations can achieve seamless global coverage [10]. Moreover, LEO satellite networks have advantages of short satellite–terrestrial transmission distance and low construction cost. In addition, LEO satellite IoT can complement and extend terrestrial IoT to better achieve IoT applications, such as environmental monitoring, marine monitoring, and border control [11]. MEC servers can be deployed at LEO satellites to handle computing tasks from devices [12]. For remote areas with sparse users and insufficient terrestrial network coverage, such as mountainous areas and islands, offloading tasks to satellites is the only option. Compared to the computing resources of MEC servers deployed at BSs, the computing resources of MEC servers deployed at satellites are less due to the satellite payload constraints. Therefore, a single LEO satellite's MEC server may sometimes be insufficient to assist all IoT devices in computing tasks.

For sparsely populated areas where IoT devices are widely spaced, such as deserts and oceans, inter-satellite cooperation is an effective coping approach. Sometimes, the computing workload is distributed unevenly in different space, which leads to the fact that some MEC servers may need to handle tasks more than their computing capabilities, whereas some other MEC servers still have surplus computing resources [13]. The LEO satellites launched recently are equipped with laser terminals, which can establish inter-satellite links (ISLs) with neighboring satellites [14]. For example, in the BeiDou system, ISLs can be established between two satellites if they are visible [15]. Similar to BS cooperation in terrestrial networks, multiple LEO satellites can also achieve cooperation, which compensates for the insufficient computing resources of a single LEO satellite's MEC server.

The local satellite migrates some tasks to neighboring satellites, which can utilize the computing resources of neighboring satellites' MEC servers to compute tasks, thus reducing the task completion delay. Compared with local computing, task offloading introduces additional overhead, such as transmission delay. Moreover, the computing resources of MEC servers are relatively limited compared with the central cloud. Therefore, it is very important to make task offloading decisions and allocate computing resources properly. According to whether tasks are divisible, computation offloading can be divided into two categories, namely binary offloading and partial offloading [16]. Binary offloading involves integer programming, so the corresponding task offloading problem is generally NP-hard. Furthermore, assigning computing resources to tasks of different devices complicates the problem and increases the difficulty of algorithm design [17].

In this paper, we consider an STN consisting of IoT devices, a local satellite, and neighboring satellites. The tasks of the IoT devices within the coverage area of the local satellite are considered, and the neighboring satellites are regarded as computing nodes without considering the tasks of the IoT devices within their coverage areas. Neighboring satellites assist the local satellite in computing IoT devices' tasks by inter-satellite cooperation. We propose an inter-satellite cooperative task offloading and resource allocation scheme in an MEC-enabled STN, which minimizes the task completion delay for all IoT devices under the condition that the tasks are indivisible.

The main contributions of this paper are as follows:

- The system model of the MEC-enabled STN is established to provide MEC services in remote or disaster areas by utilizing inter-satellite cooperation. The joint task offloading and computing resource allocation problem is formulated to minimize the task completion delay and decomposed into a task offloading decision problem and a computing resource allocation problem.
- An inter-satellite cooperative joint offloading decision and resource allocation optimization scheme, which consists of a task offloading decision algorithm based on the Grey Wolf Optimizer (GWO) algorithm and a computing resource allocation algorithm based on the Lagrange multiplier method is proposed. The optimal task offloading decision and computing resource allocation are obtained by the proposed scheme.
- The performance of the proposed scheme is evaluated by comparing it with other baseline schemes with respect to the variation of some parameters. Simulation results demonstrate the performance gain of the proposed scheme.

The rest of this paper is organized as follows. Section 2 introduces related works. Section 3 describes the system model. The task completion delay minimization problem for all IoT devices is formulated in Section 4. In order to solve this problem, the corresponding algorithm is designed in Section 5. In Section 6, simulations and an analysis are presented. Section 7 concludes this paper.

2. Related Work

Non-terrestrial networks, particularly satellite networks, were proposed to be integrated with terrestrial networks in the early 2030s. Moreover, the most anticipated function of the STN was to provide coverage to rural and remote areas [18]. The authors in [19] proposed satellite MEC. MEC servers deployed at LEO satellites enabled devices without a nearby MEC server to enjoy MEC services through satellite links. In addition, a cooperative computation offloading model was designed to realize parallel computation in STNs, which minimized user-perceived delay and system energy consumption by optimizing task scheduling. Through radio frequency (RF) communication or optical communication, ISLs can realize data communication, interconnection, and information relay. Moreover, compared with RF ISLs, optical ISLs have the advantages of broader bandwidth, lower energy consumption, and lower interference [20]. According to [21], inter-satellite laser technologies can establish ISLs and achieve transmission between satellites at a high speed. With the aid of optical or visible light communication systems, LEO satellites can achieve connection through intra-plane ISLs and inter-plane ISLs [22]. In [23], some satellites temporarily formed a task group, where the access satellite was responsible for task and resource scheduling, while others were responsible for computing tasks. The authors in [24] proposed a double-edge computing STN architecture, where an access satellite can send tasks to other satellites for computing. They also proposed a double-edge computation offloading algorithm to minimize system energy consumption and reduce task offloading delay. In [25], a double-edge intelligent integrated STN was designed. Satellite MEC servers were grouped by ISLs, and they can assist terrestrial MEC servers in computing tasks. A task migration strategy based on a greedy algorithm was proposed to achieve load balancing and reduce system delay. The authors in [26] proposed a hybrid cloud and edge computing LEO satellite network architecture, where the tasks of ground users can be completed in one of three places, i.e., the ground user themselves, the LEO satellite, or the terrestrial cloud server. They also came up with a distributed algorithm to minimize the sum energy consumption of the ground users. The authors in [27] proposed an MEC framework for the terrestrial–satellite IoT. With the assistance of terrestrial–satellite terminals, IoT mobile devices can offload tasks to LEO satellites for computing. They also proposed an energy-efficient algorithm for computation offloading and resource allocation, thereby minimizing the weighted sum of energy consumption of IoT mobile devices.

Most of the above works investigate the satellite–terrestrial cooperation or the inter-satellite cooperation in the presence of terrestrial networks, where satellites usually assist

terrestrial entities. However, there are few works that investigate inter-satellite cooperation to assist users in computing tasks in areas not covered by terrestrial networks.

Algorithms for computation offloading include heuristic algorithms, meta-heuristic algorithms, game theory-based algorithms, etc. [28]. Algorithms for resource allocation include convex optimization algorithms, heuristic algorithms, game theory-based algorithms, etc. Convex optimization algorithms are widely investigated and can easily obtain suboptimal solutions. However, they are complex and not easy to implement in actual systems. Heuristic algorithms are efficient and can quickly obtain solutions, but they are prone to fall into the local optimum. Meta-heuristic algorithms have too many parameters, which are not easy to adjust. Game theory-based algorithms are simple, flexible, and easy to implement. However, the obtained solution may not be globally optimal and the Nash equilibrium needs to be reached through continuous iterations [29]. The GWO algorithm is a meta-heuristic algorithm with few parameters. Therefore, we propose a task offloading decision algorithm based on the GWO algorithm.

3. System Model

In this section, the network model is proposed first. Then, the channel model is presented. Lastly, the computation model is introduced.

3.1. Network Model

As shown in Figure 1, a remote area beyond the reach of terrestrial BSs is considered in this paper. There is an LEO satellite 1 in space, called the local satellite, around which there are $M-1$ neighboring satellites, i.e., LEO satellite j, $j \in \mathcal{M}$, where $\mathcal{M} = \{2, 3, \ldots, M\}$. All satellites in the same orbit move in the same circular direction. We define a satellite within the communication range of the local satellite at the moment as a neighboring satellite. These LEO satellites are all deployed with MEC servers. N IoT devices are randomly distributed in the ground area covered by the local satellite, and the set composed of them is denoted by $\mathcal{N}$. These IoT devices are equipped with satellite antennas and can communicate with the local satellite directly. Each IoT device requests an indivisible task; for instance, IoT device i requests task i, denoted by (c_i, a_i), where c_i is the computation intensity, i.e., computing resources required for computing one bit of task i (cycles per bit), and a_i is the data size of task i (bits). Different IoT devices have different computing capabilities. MEC servers have more computing resources than IoT devices, but the computing resources of MEC servers are still limited. Task generation mechanisms may be periodic generation, random generation, and event-triggered generation. The problem we study is to build the corresponding task model for the set of tasks generated by the IoT devices in the network at a certain moment. We assume that each IoT device has only one task at the moment and perform computation offloading according to the task model. Based on the offloading decision, the local computing portion of the task of each IoT device is computed by the IoT device itself. The satellite allocates its computing resources to the offloading portion of the tasks of the IoT devices and concurrently computes the tasks offloaded by the IoT devices. There are three strategies to complete task i, $i \in \mathcal{N}$. IoT device i computes task i locally, i.e., local computing. IoT device i offloads task i to the local satellite, and task i is computed by the local satellite's MEC server, i.e., local satellite edge computing. IoT device i offloads task i to the local satellite, then task i is migrated to neighboring satellite j, $j \in \mathcal{M}$ and computed by the neighboring satellite's MEC server, i.e., neighboring satellite edge computing.

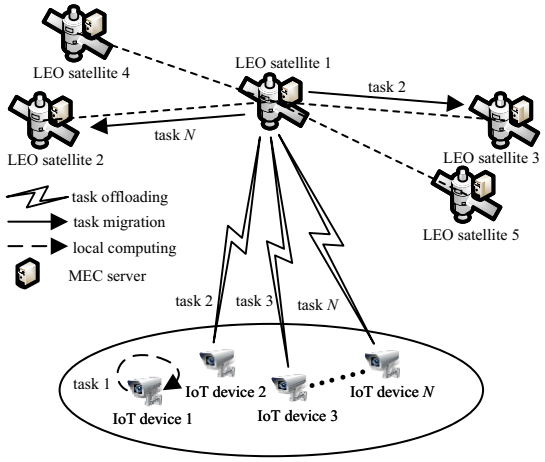

Figure 1. Network model.

3.2. Channel Model

In order to avoid the interference among IoT devices, orthogonal frequency division multiple access (OFDMA) technology is used. Subcarriers with equal bandwidth are allocated to IoT devices. The satellite–terrestrial transmission links between IoT devices and the local satellite use the Ka band [27] and have high communication rates. Since the line-of-sight (LOS) transmission is mainly considered in the link, the Rice channel is used. Free-space path loss (FSPL) and additive white Gaussian noise (AWGN) are taken into consideration. The achievable data rate of the link between IoT device i and the local satellite is:

$$r_{i,1} = B_{i,1}\log_2\left(1 + \frac{p_i g_i H_{i,1}}{N_{0_{i,1}}}\right), \tag{1}$$

where $B_{i,1}$ is the subchannel bandwidth, p_i is the transmission power of IoT device i, g_i is the antenna gain of IoT device i, $H_{i,1}$ is the channel gain of the link, and $N_{0_{i,1}}$ is the noise power of the link. $H_{i,1} = (h_{i,1})^2$, where the channel coefficient $h_{i,1} = \rho_{i,1}/(L^{\mathrm{f}}_{i,1})^{1/2}$, $\rho_{i,1}$ is the independent identically distributed Rice stochastic variable with a unit variance, $L^{\mathrm{f}}_{i,1}$ is FSPL, $L^{\mathrm{f}}_{i,1} = (4\pi d_{i,1} f/c)^2$, c is the light velocity (meter per second), $d_{i,1}$ is the distance between IoT device i and the local satellite (meter), and f is the carrier frequency (hertz). $d^{\min}_{i,1} = h_{\mathrm{s}}$, as Figure 2 depicted, $d^{\max}_{i,1} = [(R_{\mathrm{e}})^2 + (R_{\mathrm{e}} + h_{\mathrm{s}})^2 - 2R_{\mathrm{e}}(R_{\mathrm{e}} + h_{\mathrm{s}})\cos\theta_{\mathrm{ec}}]^{1/2}$, where R_{e} is the earth radius, h_{s} is the LEO satellite orbit altitude, and θ_{ec} is the covering geocentric angle. $\theta_{\mathrm{ec}} = \arccos\left[R_{\mathrm{e}}\cos\theta^{\min}_{\mathrm{u}}/(R_{\mathrm{e}} + h_{\mathrm{s}})\right] - \theta^{\min}_{\mathrm{u}}$, $\theta^{\min}_{\mathrm{u}}$ is the minimum elevation angle of the IoT devices on the ground to the satellite [30].

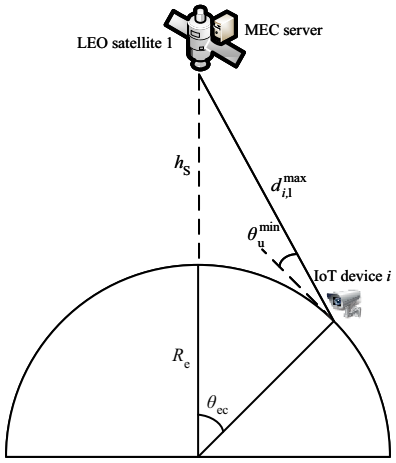

Figure 2. Illustration of $d^{\max}_{i,1}$.

The ISLs between the local satellite and neighboring satellites use the Ka band [31]. FSPL and AWGN are taken into account. The achievable data rate of the link between the local satellite and neighboring satellite j, $j \in \mathcal{M}$ is:

$$r_{1,j} = B_{1,j}\log_2\left(1 + \frac{PGH_{1,j}}{N_{0_{1,j}}}\right), \tag{2}$$

where $B_{1,j}$ is the subchannel bandwidth, P is the transmission power of the local satellite, G is the antenna gain of the local satellite, $H_{1,j}$ is the channel gain of the link, and $N_{0_{1,j}}$ is the noise power of the link. $G \approx (4D_{\mathrm{TA}}f/c)^2$, and D_{TA} is the diameter of the local satellite's transmitting antenna [31]. $H_{1,j} = \left(h_{1,j}\right)^2$, where the channel coefficient $h_{1,j} = 1/(L^{\mathrm{f}}_{1,j})^{1/2}$, $L^{\mathrm{f}}_{1,j}$ is FSPL, $L^{\mathrm{f}}_{1,j} = \left(4\pi l_{1,j}f/c\right)^2$, and $l_{1,j}$ is the distance between the local satellite and neighboring satellite j (meter).

Normally, each satellite has four neighboring satellites, two in the same orbit and the other two in the left and right orbits. The distance between the local satellite and the neighboring satellite in the same orbit is $l_{1,j} = \sqrt{2}R\sqrt{1-\cos\left(360^\circ/J\right)}$, where R is the orbital radius, $R = h_{\mathrm{s}} + R_{\mathrm{e}}$, and J is the number of satellites in each orbit. The distance between the local satellite and the neighboring satellite in different orbits is $l_{1,j} = \omega\cos(lat)$, $\omega = \sqrt{2}R\sqrt{1-\cos\left(360^\circ/2I\right)}$, where lat is the latitude where the ISL of the current satellite locate, and I is the number of orbits of satellite constellation [32].

3.3. Computation Model

To represent the task offloading strategies, we set indicator variables, i.e., $y_{i,0}$, $y_{i,1}$, $y_{i,j} \in \{0,1\}$, $\forall i \in \mathcal{N}$, and $\forall j \in \mathcal{M}$. When task i is completed by local computing, $y_{i,0} = 1$, otherwise, $y_{i,0} = 0$. When task i is completed by local satellite edge computing, $y_{i,1} = 1$, otherwise, $y_{i,1} = 0$. When task i is completed by neighboring satellite edge computing, $y_{i,j} = 1$, otherwise, $y_{i,j} = 0$.

When $y_{i,0} = 1$, the task computing delay of IoT device i is:

$$D_{i,0} = \frac{c_i a_i}{f_{i,0}}, \tag{3}$$

where $f_{i,0}$ is the computing capacity of IoT device i (cycles per second).

When $y_{i,1} = 1$, the task completion delay of IoT device i includes the delay with which IoT device i offloads task i to the local satellite, the propagation delay between IoT device i and the local satellite, and the delay that the local satellite's MEC server computes task i.

$$D_{i,1} = \frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{c_i a_i}{f_{i,1}}, \tag{4}$$

where $f_{i,1}$ is the computing resource allocated to task i by MEC server of the local satellite (cycles per second).

When $y_{i,j} = 1$, the task completion delay of IoT device i includes the delay that IoT device i offloads task i to the local satellite, the propagation delay between IoT device i and the local satellite, the delay with which the local satellite migrates task i to neighboring satellite j, the propagation delay between the local satellite and neighboring satellite j, and the delay with which neighboring satellite j's MEC server computes task i.

$$D_{i,j} = \frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{a_i}{r_{1,j}} + \frac{2l_{1,j}}{c} + \frac{c_i a_i}{f_{i,j}}, \tag{5}$$

where $f_{i,j}$ is the computing resource allocated to task i by the MEC server of neighboring satellite j (cycles per second).

In all of the cases above, the delay of IoT devices downloading task computing results from the local satellite or neighboring satellites is neglected, because the size of output resulting data obtained after task computation is usually much smaller than the size of input data [33].

4. Problem Formulation

The task completion delay of all IoT devices can be expressed as:

$$\begin{aligned} U &= \sum_{i\in\mathcal{N}} \left(y_{i,0}D_{i,0} + y_{i,1}D_{i,1} + \sum_{j\in\mathcal{M}} y_{i,j}D_{i,j} \right) \\ &= \sum_{i\in\mathcal{N}} \left[y_{i,0}\frac{c_i a_i}{f_{i,0}} + y_{i,1}\left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{c_i a_i}{f_{i,1}} \right) + \sum_{j\in\mathcal{M}} y_{i,j}\left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{a_i}{r_{1,j}} + \frac{2l_{1,j}}{c} + \frac{c_i a_i}{f_{i,j}} \right) \right]. \end{aligned} \tag{6}$$

The optimization objective of this paper is minimizing the task completion delay of all IoT devices. The optimization variables are $y_{i,0}$, $y_{i,1}$, $y_{i,j}$, $f_{i,1}$, and $f_{i,j}$. The problem $\mathcal{P}$ is as follows.

$$\mathcal{P}: \min_{\left\{ \substack{y_{i,0},y_{i,1},y_{i,j}, \\ f_{i,1},f_{i,j}} \right\}} U \tag{7a}$$

$$\text{s.t.} \sum_{i\in\mathcal{N}} y_{i,1}f_{i,1} \le F_1, \tag{7b}$$

$$\sum_{i\in\mathcal{N}} y_{i,j}f_{i,j} \le F_j, \forall j \in \mathcal{M}, \tag{7c}$$

$$0 \le f_{i,1} \le F_1, \forall i \in \mathcal{N}, \tag{7d}$$

$$0 \le f_{i,j} \le F_j, \forall i \in \mathcal{N}, \forall j \in \mathcal{M}, \tag{7e}$$

$$y_{i,0} + y_{i,1} + \sum_{j\in\mathcal{M}} y_{i,j} = 1, \forall i \in \mathcal{N}, \tag{7f}$$

$$y_{i,0}, y_{i,1}, y_{i,j} \in \{0,1\}, \forall i \in \mathcal{N}, \forall j \in \mathcal{M}. \tag{7g}$$

Equation (7a) is the objective function. Equation (7b) represents that computing resources allocated to task i cannot surpass the resources owned by the local satellite's MEC server, i.e., F_1. Equation (7c) shows that computing resources allocated to task i cannot exceed the resources of the MEC server of neighboring satellite j, i.e., F_j. Equations (7d) and (7e) mean that the MEC server's computing resource allocation for task i is non-negative. Equation (7f) denotes that only one strategy is selected to complete task i. Equation (7g) means that $y_{i,0}$, $y_{i,1}$, and $y_{i,j}$ are 0 or 1.

$\mathcal{P}$ consists of the IoT devices' task offloading decision and the computing resource allocation of the local satellite's MEC server and neighboring satellites' MEC servers. $\mathcal{P}$ includes discrete variables and continuous variables, so it is a mixed-integer nonlinear programming problem (MINLP), which is NP-hard.

5. Algorithm Design

From $\mathcal{P}$, it can be seen that $y_{i,1}$ and $f_{i,1}$, $y_{i,j}$ and $f_{i,j}$ interact with each other, respectively. Since the variables are coupled, it is necessary to decouple them. Assuming $y^*_{i,0}$, $y^*_{i,1}$, and $y^*_{i,j}$, $\forall i \in \mathcal{N}$, $\forall j \in \mathcal{M}$ are given, the sets $\mathcal{N}^{\mathrm{D}} = \{i | y^*_{i,0} = 1\}$, $\mathcal{N}^{\mathrm{L}} = \{i | y^*_{i,1} = 1\}$, and $\mathcal{N}^{\mathrm{N}} = \{i | y^*_{i,j} = 1, \forall j \in \mathcal{M}\}$ can be obtained. $\mathcal{P}$ can be transformed into $\mathcal{P}1$.

$$\mathcal{P}1: \min_{\{f_{i,1},f_{i,j}\}} \sum_{i\in\mathcal{N}^{\mathrm{D}}} \frac{c_i a_i}{f_{i,0}} + \sum_{i\in\mathcal{N}^{\mathrm{L}}} \left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{c_i a_i}{f_{i,1}}\right) + \sum_{i\in\mathcal{N}^{\mathrm{N}}} \sum_{j\in\mathcal{M}} \left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{a_i}{r_{1,j}} + \frac{2l_{1,j}}{c} + \frac{c_i a_i}{f_{i,j}}\right), \tag{8a}$$

$$\text{s.t.} \sum_{i\in\mathcal{N}^{\mathrm{L}}} f_{i,1} \le F_1, \tag{8b}$$

$$\sum_{i\in\mathcal{N}^{\mathrm{N}}} f_{i,j} \le F_j, \forall j \in \mathcal{M}, \tag{8c}$$

$$0 < f_{i,1} \le F_1, \forall i \in \mathcal{N}^{\mathrm{L}}, \tag{8d}$$

$$0 < f_{i,j} \le F_j, \forall i \in \mathcal{N}^{\mathrm{N}}, \forall j \in \mathcal{M}. \tag{8e}$$

Equation (8a) is the objective function. The meanings of (8b)–(8e) are similar to the meanings of (7b)–(7e). From $\mathcal{P}1$, it can be seen that the delay of local computing is a constant. Moreover, the computing resource allocation problem of the local satellite's MEC server and the computing resource allocation problem of the neighboring satellites' MEC servers are independent of each other. Therefore, $\mathcal{P}1$ can be decomposed into two sub-problems, i.e., $\mathcal{P}1.1$ and $\mathcal{P}1.2$, and then $f_{i,1}^*$ and $f_{i,j}^*$ can be obtained by solving $\mathcal{P}1.1$ and $\mathcal{P}1.2$, respectively.

$\mathcal{P}1.1$ is the computing resource allocation problem for the local satellite's MEC server, which is represented as:

$$\mathcal{P}1.1: \min_{\{f_{i,1}\}} \sum_{i\in\mathcal{N}^{\mathrm{L}}} \left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{c_i a_i}{f_{i,1}}\right), \tag{9}$$

$$\text{s.t. (8b) and (8d)}.$$

The objective function in (9) is set as U^{L}. Then we can have $\partial U^{\mathrm{L}}/\partial f_{i,1} = -c_i a_i/(f_{i,1})^2$, which is less than 0. $\partial^2 U^{\mathrm{L}}/\partial (f_{i,1})^2 = 2c_i a_i/(f_{i,1})^3$, which is greater than 0. $\partial^2 U^{\mathrm{L}}/(\partial f_{i,1}\partial f_{j,1}) = 0$, $i, j \in \mathcal{N}^{\mathrm{L}}$, $i \neq j$. The Hessian matrix of U^{L} is positive definite, so U^{L} is convex. $\mathcal{P}1.1$ can be solved by using the Lagrange multiplier method. The Lagrangian function is constructed as:

$$L(f_{i,1},\theta) = \sum_{i\in\mathcal{N}^{\mathrm{L}}} \left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{c_i a_i}{f_{i,1}}\right) + \theta\left(\sum_{i\in\mathcal{N}^{\mathrm{L}}} f_{i,1} - F_1\right), \tag{10}$$

where θ is the Lagrange multiplier and $\theta \ge 0$. The KKT conditions of $\mathcal{P}1.1$ are listed as:

$$\frac{\partial L(f_{i,1},\theta)}{\partial f_{i,1}} = -\frac{c_i a_i}{(f_{i,1})^2} + \theta = 0, \tag{11}$$

$$\frac{\partial L(f_{i,1},\theta)}{\partial \theta} = \sum_{i\in\mathcal{N}^{\mathrm{L}}} f_{i,1} - F_1 = 0, \tag{12}$$

$$\theta\left(\sum_{i\in\mathcal{N}^{\mathrm{L}}} f_{i,1} - F_1\right) = 0, \tag{13}$$

$$\text{(8b) and (8d)},$$

$$\theta \ge 0. \tag{14}$$

$f_{i,1} = \sqrt{c_i a_i/\theta}$ is obtained from (11). Substituting it into (12), we can obtain $\sum_{i\in\mathcal{N}^{\mathrm{L}}} \sqrt{c_i a_i/\theta} - F_1 = 0$, i.e., $\sqrt{\theta} = \left(\sum_{i\in\mathcal{N}^{\mathrm{L}}} \sqrt{c_i a_i}\right)/F_1$. Thus, $f_{i,1} = F_1\sqrt{c_i a_i}/\left(\sum_{i\in\mathcal{N}^{\mathrm{L}}} \sqrt{c_i a_i}\right)$. Considering (8d), we can deduce:

$$f_{i,1}^* = \min\left[\max\left(\frac{F_1\sqrt{c_i a_i}}{\sum_{i\in\mathcal{N}^{\mathrm{L}}} \sqrt{c_i a_i}}, 0\right), F_1\right]. \tag{15}$$

$\mathcal{P}1.2$ is the computing resource allocation problem for neighboring satellites' MEC servers, which is expressed as:

$$\mathcal{P}1.2: \min_{\{f_{i,j}\}} \sum_{i\in\mathcal{N}^{\mathrm{N}}} \sum_{j\in\mathcal{M}} \left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{a_i}{r_{1,j}} + \frac{2l_{1,j}}{c} + \frac{c_i a_i}{f_{i,j}}\right), \quad \text{s.t. (8c) and (8e).} \tag{16}$$

Since the computing resource allocation of each neighboring satellite is independent of each other, $\mathcal{P}1.2$ can be decomposed into $M-1$ sub-problems, one of which is $\mathcal{P}1.3$.

$$\mathcal{P}1.3: \min_{\{f_{i,j}\}} \sum_{i\in\mathcal{N}^{\mathrm{N}}} \left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{a_i}{r_{1,j}} + \frac{2l_{1,j}}{c} + \frac{c_i a_i}{f_{i,j}}\right), \quad \text{s.t. (8c) and (8e).} \tag{17}$$

The objective function in (17) is set as U^{N}. Then, we can have $\partial U^{\mathrm{N}}/\partial f_{i,j} = -c_i a_i/\left(f_{i,j}\right)^2$, which is less than 0. $\partial^2 U^{\mathrm{N}}/\partial\left(f_{i,j}\right)^2 = 2c_i a_i/\left(f_{i,j}\right)^3$, which is greater than 0. $\partial^2 U^{\mathrm{N}}/\left(\partial f_{i,j}\partial f_{k,j}\right) = 0, i,k \in \mathcal{N}^{\mathrm{N}}, i \neq k$. The Hessian matrix of U^{N} is positive definite. Hence, U^{N} is convex. $\mathcal{P}1.3$ can be solved by using the Lagrange multiplier method. The Lagrangian function is constructed as:

$$L\left(f_{i,j},\lambda\right) = \sum_{i\in\mathcal{N}^{\mathrm{N}}} \left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{a_i}{r_{1,j}} + \frac{2l_{1,j}}{c} + \frac{c_i a_i}{f_{i,j}}\right) + \lambda\left(\sum_{i\in\mathcal{N}^{\mathrm{N}}} f_{i,j} - F_j\right), \tag{18}$$

where λ is the Lagrange multiplier and $\lambda \geq 0$. The KKT conditions of $\mathcal{P}1.3$ are listed as:

$$\frac{\partial L\left(f_{i,j},\lambda\right)}{\partial f_{i,j}} = -\frac{c_i a_i}{\left(f_{i,j}\right)^2} + \lambda = 0, \tag{19}$$

$$\frac{\partial L\left(f_{i,j},\lambda\right)}{\partial \lambda} = \sum_{i\in\mathcal{N}^{\mathrm{N}}} f_{i,j} - F_j = 0, \tag{20}$$

$$\lambda\left(\sum_{i\in\mathcal{N}^{\mathrm{N}}} f_{i,j} - F_j\right) = 0, \tag{21}$$

$$\text{(8c) and (8e)},$$

$$\lambda \geq 0. \tag{22}$$

Similar to solving $\mathcal{P}1.1$, $f_{i,j} = \sqrt{c_i a_i/\lambda}$ is obtained from (19). Substituting it into (20), we can obtain $\sum_{i\in\mathcal{N}^{\mathrm{N}}} \sqrt{c_i a_i/\lambda} - F_j = 0$, i.e., $\sqrt{\lambda} = \left(\sum_{i\in\mathcal{N}} \sqrt{c_i a_i}\right)/F_j$. So $f_{i,j} = F_j\sqrt{c_i a_i}/\left(\sum_{i\in\mathcal{N}^{\mathrm{N}}} \sqrt{c_i a_i}\right)$. Given (8e), we can obtain:

$$f_{i,j}^* = \min\left[\max\left(\frac{F_j\sqrt{c_i a_i}}{\sum_{i\in\mathcal{N}^{\mathrm{N}}} \sqrt{c_i a_i}}, 0\right), F_j\right]. \tag{23}$$

By solving $\mathcal{P}1.1$ and $\mathcal{P}1.3$ above, we can obtain $f_{i,1}^*$ and $f_{i,j}^*$, respectively. By taking them back to $\mathcal{P}$, $\mathcal{P}$ can be transformed into $\mathcal{P}2$.

$$\mathcal{P}2: \min_{\{y_{i,0},y_{i,1},y_{i,j}\}} \sum_{i\in\mathcal{N}} \left[y_{i,0}\frac{c_i a_i}{f_{i,0}} + y_{i,1}\left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{c_i a_i}{f_{i,1}}\right) + \sum_{j\in\mathcal{M}} y_{i,j}\left(\frac{a_i}{r_{i,1}} + \frac{2d_{i,1}}{c} + \frac{a_i}{r_{1,j}} + \frac{2l_{1,j}}{c} + \frac{c_i a_i}{f_{i,j}}\right)\right], \tag{24}$$

s.t. (7f) and (7g).

Equation (24) is the objective function. $\mathcal{P}2$ is an integer linear programming (ILP) problem, which is NP-hard. When the scale of a problem is large, it is difficult to obtain

the optimal solution in polynomial time. ILP problems can be solved by intelligence optimization algorithms, one of which is the GWO algorithm. The GWO algorithm is widely used and has few parameters and is easy to implement [34].

According to the GWO algorithm [35], the grey wolf population is highly hierarchical and each wolf represents a feasible solution. The top rank wolf is α, i.e., the optimal solution. The second rank wolf is β, i.e., the suboptimal solution. The third rank wolf is δ, i.e., the third optimal solution. The remaining wolves are ω, which are the lowest rank and form a set $\mathcal{N}^{\omega}$, i.e., the remaining solutions. The prey is the optimal solution to the problem. During each iteration, α, β, and δ command ω to encircle, hunt, and attack the prey to obtain the optimal solution.

We use N^{W} to represent the number of grey wolves in the population. The wolf l is encoded as $\vec{X}_l = [x_{l,1}, x_{l,2}, x_{l,3}, \ldots, x_{l,N}]$, where $x_{l,i}$ is the task offloading decision of task i of wolf l. To be specific, $x_{l,i} = 0$ represents local computing, $x_{l,i} = 1$ means local satellite edge computing, $x_{l,i} = j, j \in \mathcal{M}$ denotes neighboring satellite j edge computing. After such coding, Equations (7f) and (7g) are satisfied. The grey wolves' hunting of prey consists of three phases, i.e., encircling, hunting, and attacking.

During the encircling phase, the grey wolves scour the prey, and then approach and encircle it gradually. The distance between l and the prey p is denoted by $\vec{D} = \left|\vec{C} \cdot \vec{X}_p(t) - \vec{X}_l(t)\right|, \forall l \in \mathcal{N}^{\omega}$, and the updating of $\vec{X}_l$ is $\vec{X}_l(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D}$, where t is the number of iterations, $\vec{X}_l$ is the position of l, $\vec{X}_p$ is the position of p, and $\vec{A}$ and $\vec{C}$ are coefficient vectors. $\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a}$, $\vec{C} = 2 \cdot \vec{r}_2$, where $\vec{a}$ is the convergence factor, which reduces from 2 to 0 linearly as t increases, $\vec{a} = 2 \cdot (1 - t/t^{\max})$. Both $\vec{r}_1$ and $\vec{r}_2$ are random vectors in $[0, 1]$. $t^{\max}$ is the maximum number of iterations.

During the hunting phase, the grey wolves ω identify and track the location of prey according to the command of α, β, and δ. The distance between l and α, β, δ are expressed as $\vec{D}_\alpha = \left|\vec{C}_1 \cdot \vec{X}_\alpha - \vec{X}_l\right|$, $\vec{D}_\beta = \left|\vec{C}_2 \cdot \vec{X}_\beta - \vec{X}_l\right|$, and $\vec{D}_\delta = \left|\vec{C}_3 \cdot \vec{X}_\delta - \vec{X}_l\right|$ respectively, where $\vec{X}_\alpha$, $\vec{X}_\beta$, and $\vec{X}_\delta$ are the positions of α, β, and δ in that order. $\vec{C}_1$, $\vec{C}_2$, and $\vec{C}_3$ are all random vectors. The step length and direction that l moves forward to α, β, and δ are $\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot (\vec{D}_\alpha)$, $\vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot (\vec{D}_\beta)$, and $\vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot (\vec{D}_\delta)$ respectively. The updating of $\vec{X}_l$ is $\vec{X}_l(t+1) = (\vec{X}_1 + \vec{X}_2 + \vec{X}_3)/3$.

During the attacking phase, the prey stops moving and the grey wolves attack it. As the value of $\vec{a}$ decreases from 2 to 0 linearly, the value of $\vec{A}$ varies within the interval $[-|\vec{a}|, |\vec{a}|]$. When $|\vec{A}| < 1$, the grey wolves the prey. When $|\vec{A}| > 1$, the grey wolves move away from it to find a better one, which is conducive to jump out of the local optimum.

In addition to $\vec{A}$, another search coefficient, i.e., $\vec{C}$ also helps obtain the global optimal solution. $\vec{C}$ is composed of random values in $[0, 2]$, which can provide random weights for the prey, so that the search of prey by grey wolves has randomness, thus helping to avoid falling into the local optimum.

The fitnesses of l, α, β, and δ are U_l, U_α, U_β, and U_δ, respectively. During each iteration, the GWO algorithm compares U_l with U_α, U_β, and U_δ, and then updates $\vec{X}_\alpha$, $\vec{X}_\beta$, $\vec{X}_\delta$, U_α, U_β, and U_δ. When $t^{\max}$ is reached, $\vec{X}_\alpha$ is the optimal solution to the problem, and U_α is the optimal objective function value. According to $\vec{X}_\alpha$, $y^*_{i,0}$, $y^*_{i,1}$, and $y^*_{i,j}$ can be determined.

In this paper, we propose an inter-satellite cooperative task offloading scheme based on the GWO algorithm. We make some modifications to the GWO algorithm. Since $\vec{X}_l$ is a real number, which is inconsistent with the task offloading decision in integer form, we round it to convert the real number code into the integer code, so that the corresponding objective function value can be calculated and obtained.

The detailed step of the proposed joint offloading decision and resource allocation optimization scheme, which consists of a task offloading decision algorithm based on the GWO algorithm and a computing resource allocation algorithm based on the Lagrange multiplier method, and considers inter-satellite cooperation (GWOORAC), is expressed in Algorithm 1.

Algorithm 1 GWOORAC

Initialize the grey wolf population, $\vec{a}$, $\vec{A}$, and $\vec{C}$. Set $t = 1$. Set $t^{\max}$.
Obtain $f^*_{i,1}$ by (15), obtain $f^*_{i,j}$ by (23), thus calculating the fitness of each grey wolf. The top three wolves in fitness are identified as α, β, and δ, the other wolves are ω.
while $1 \leq t \leq t^{\max}$ **do**
  for each $l \in \mathcal{N}^{\omega}$ **do**
    Update $\vec{X}_l$.
    Update $\vec{a}$, $\vec{A}$, and $\vec{C}$.
    For $\vec{X}_l$, obtain $f^*_{i,1}$ by (15), obtain $f^*_{i,j}$ by (23), thus calculating U_l.
    if $U_l < U_\alpha$ **then**
      update $\vec{X}_\alpha = \vec{X}_l$, $U_\alpha = U_l$.
    else if $U_l > U_\alpha$ and $U_l < U_\beta$ **then**
      update $\vec{X}_\beta = \vec{X}_l$, $U_\beta = U_l$.
    else if $U_l > U_\alpha$ and $U_l > U_\beta$ and $U_l < U_\delta$ **then**
      update $\vec{X}_\delta = \vec{X}_l$, $U_\delta = U_l$.
    end if
  end for
  $t = t + 1$.
end while
return $\vec{X}_\alpha$ and U_α.

To sum up, the $\vec{X}_\alpha$ that is obtained finally is the optimal task offloading decision, i.e., $y^*_{i,0}$, $y^*_{i,1}$, and $y^*_{i,j}$. The corresponding $f^*_{i,1}$ and $f^*_{i,j}$ are the optimal computing resource allocation. U_α is the minimum task completion delay. The complexity of Algorithm 1 is $\mathcal{O}(t^{\max} N^{\mathrm{W}} N)$.

6. Simulation and Analysis

In this section, the delay is evaluated by performing our scheme and other baseline schemes. The results are also analyzed.

6.1. Simulation Parameters

The simulation parameters of this paper are shown in Table 1. The variables a_i and $f_{i,0}$ follow random distribution within the intervals of [0.5, 5] Mbits and [1, 2] Gcycles/s, respectively. K is the Rice factor. The main references for simulation parameter settings are [16,24,25,27,30,31]. To validate our scheme, GWOORAC was compared with other baseline schemes described below. The first scheme uses the GWO algorithm to determine the offloading decision, allocates the resources randomly, and considers inter-satellite cooperation (GWORandRAC). The second scheme obtains the offloading decision randomly, optimizes the resource allocation, and considers inter-satellite cooperation (RandORAC). The third scheme uses the GWO algorithm to make the offloading decision, optimizes the resource allocation, and takes no account of inter-satellite cooperation (GWOORANC). The fourth scheme only has local computing, and the rest is the same as our scheme (OLC). The fifth scheme uses the particle swarm optimization (PSO) algorithm to determine the offloading decision, optimizes the resource allocation, and considers inter-satellite cooperation (PSOORAC). The number of particles is 20. The maximum number of iterations is 200. The inertia weight is 0.8 and both learning factors are 1.5. All simulation results were obtained by averaging 300 times simulations.

Table 1. Simulation parameters.

Parameter	Value	Parameter	Value
N	50	M	5
c_i	1000 cycles/bit	a_i	[0.5, 5] Mbits
$f_{i,0}$	[1, 2] Gcycles/s	p_i	2 W
g_i	43.3 dBi	$B_{i,1}$	100 MHz
$B_{1,j}$	100 MHz	P	50 W
F_1	12 Gcycles/s	F_j	12 Gcycles/s
D_{TA}	2.2 m	$N_{0_{i,1}}$	-203 dBm/Hz
$N_{0_{1,j}}$	-203 dBm/Hz	R_{e}	6371 km
$\theta^{\mathrm{U}}_{\min}$	16°	lat	60°
h_{s}	780 km	c	3×10^8 m/s
f	30 GHz	K	7
I	6	J	11
N^{W}	20	$t^{\max}$	200

6.2. Simulation Results and Analysis

As shown in Figure 3, as the number of IoT devices, i.e., N, increases, the delay of all schemes increases gradually. GWOORAC has the lowest delay, and the more IoT devices, the more the delay reduction. Compared with GWORandRAC, the delay is relatively close when $N = 10$. However, when $N > 10$, the delay reduction increases notably, which demonstrates that our optimal resource allocation is effective. The delay of GWOORAC is lower than that of RandORAC. Moreover, the delay reduction increases notably when $N > 15$, which proves the effectiveness of the GWO algorithm. Compared with GWOORANC, there is little difference in the delay when $N \leq 25$. By contrast, when $N > 25$, the difference between the delay of GWOORAC and that of GWOORANC increases gradually, which indicates that the inter-satellite cooperation is of great significance in reducing the delay. The computing resources of MEC servers are much larger than those of IoT devices, and each task can be allocated more computing resources than the computing capacity of IoT devices when tasks are offloaded to satellites for computing; thus, the delay of OLC is much higher than that of GWOORAC. Moreover, although the delay increases as N increases, the increasing speed of GWOORAC is much smaller than that of OLC.

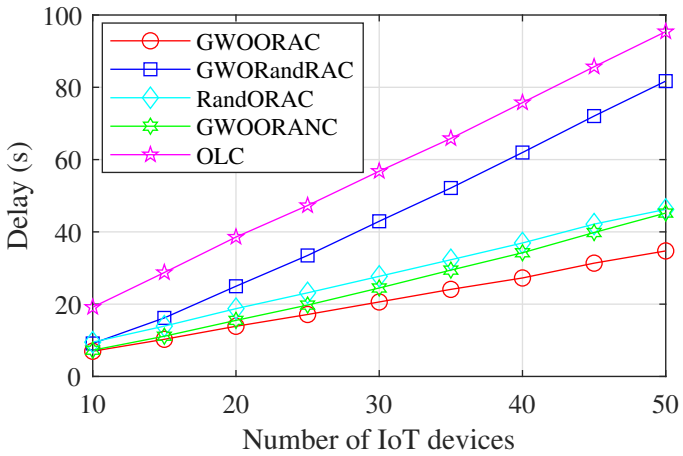

Figure 3. Delay vs. number of IoT devices.

Figure 4 shows that GWOORAC has the lowest delay. As the number of satellites, i.e., M increases, the delay of GWOORAC, GWORandRAC, and RandORAC decreases gradually, because more neighboring satellites bring more computing resources, and more computing resources are available for each offloaded task. Neither GWOORANC nor OLC takes into account neighboring satellites, so their delay almost does not change with the variation of M. Due to the optimal resource allocation, the delay of GWOORAC is much

lower than that of GWORandRAC. The delay of GWOORAC is much lower than that of RandORAC, which indicates that the GWO algorithm is appropriate to solve the problem. When $M \geq 2$, the delay of GWOORAC is lower than that of GWOORANC, and the gap gets larger, which demonstrates that the inter-satellite cooperation reduces the delay. Compared with OLC, the delay of GWOORAC is much lower, and the delay reduction becomes larger as M increases, because a large number of computing resources can be allocated to each offloaded task, far more than the computing capacity of IoT devices.

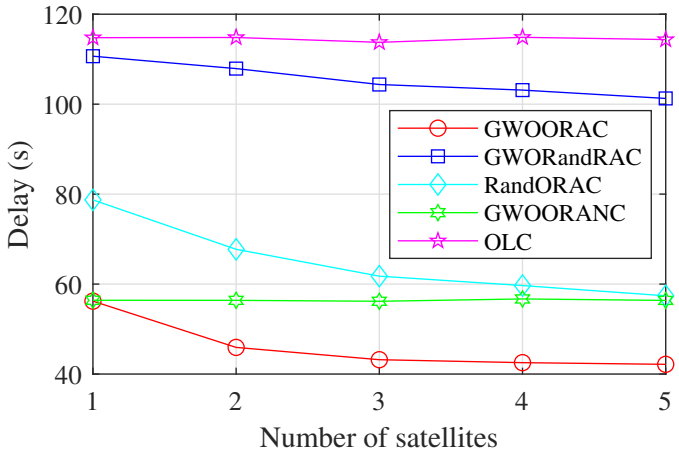

Figure 4. Delay vs. number of satellites.

As Figure 5 shows, the delay of all schemes except OLC shows downward trends as the computing resources of each satellite, i.e., F_1 and F_j, increase. The reason is that the more computing resources a satellite has, the more computing resources are allocated to offloaded tasks, which reduces the delay of computing tasks. OLC does not consider computation offloading, so its delay is not affected by F_1 and F_j. Due to reasonable resource allocation, the delay of GWOORAC is much lower than that of GWORandRAC. The delay of GWOORAC is lower than that of RandORAC, which indicates that the GWO algorithm can obtain an appropriate task offloading decision, thereby reducing the delay. The delay of GWOORAC is lower than that of GWOORANC, which proves that the inter-satellite cooperation reduces the delay effectively. The delay of GWOORAC is much lower than that of OLC, and the delay reduction becomes larger gradually. This demonstrates that a combination of suitable task offloading decisions and reasonable computing resource allocation can reduce the delay significantly.

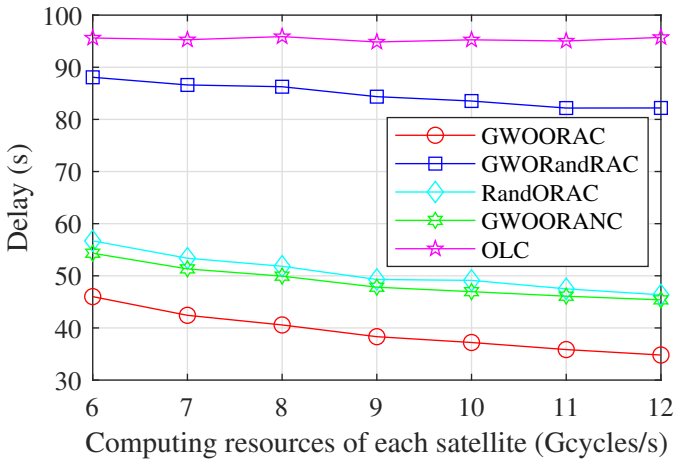

Figure 5. Delay vs. computing resources of each satellite.

As shown in Figure 6, the delay of all schemes shows upward trends as the computation intensity, i.e., c_i, increases, because of the increased task computation delay. The delay of GWOORAC is much lower than that of GWORandRAC due to the optimal computing resource allocation. The delay of RandORAC is higher than that of GWOORAC, which validates that the GWO algorithm can obtain the proper task offloading decision, thus reducing the delay. Thanks to the inter-satellite cooperation, the delay of GWOORAC is lower than that of GWOORANC. The computing capacity of IoT devices is weaker than that of MEC servers, so the delay of OLC is higher than that of GWOORAC, and its growth speed is much faster than that of GWOORAC. When $c_i \leq 400$ cycles/bit, the delay of GWOORAC and other schemes is fairly close. When $c_i > 400$ cycles/bit, the gap becomes larger, which validates that both the appropriate task offloading decision and the proper computing resource allocation are necessary.

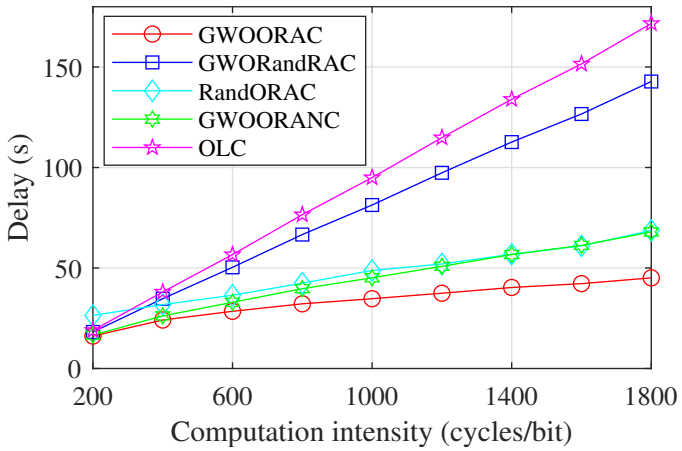

Figure 6. Delay vs. computation intensity.

As shown in Figure 7, the data size of each task, i.e., a_i is assumed to be the same. As a_i increases, the delay of all schemes increases. The reason is that the data size is bigger, and the corresponding task transmission delay and task migration delay are higher. Additionally, the computing resources required by a task are proportional to its data size, which means that big data sizes bring high task computation delay. Compared with GWORandRAC, the delay of GWOORAC is much lower due to the optimal resource allocation. As can be seen from the fact that the delay of GWOORAC is lower than that of RandORAC, the task offloading decision obtained by the GWO algorithm can reduce the delay. The gap between the delay of GWOORAC and that of GWOORANC shows the contribution of the inter-satellite cooperation. The delay of OLC is much higher than that of GWOORAC, which validates the effectiveness of GWOORAC.

As Figure 8 shows, the delay of both schemes increases with the increase of N. The delay of GWOORAC is lower than that of PSOORAC. This is because the values of the parameters, such as $\vec{A}$, are adaptive, allowing for a better balance between exploration and exploitation, thus more effectively avoiding falling into the local optimum.

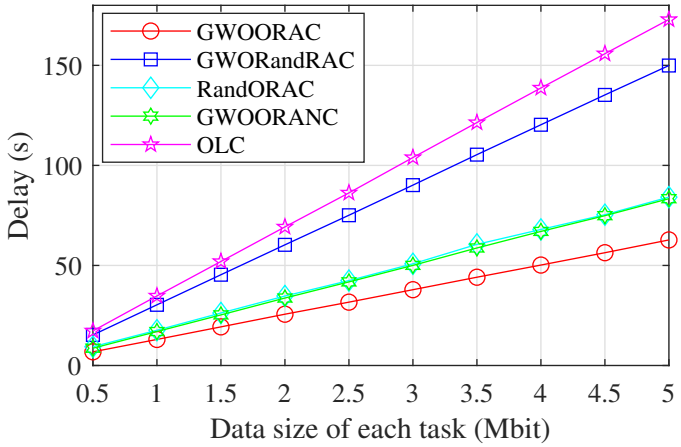

Figure 7. Delay vs. data size of each task.

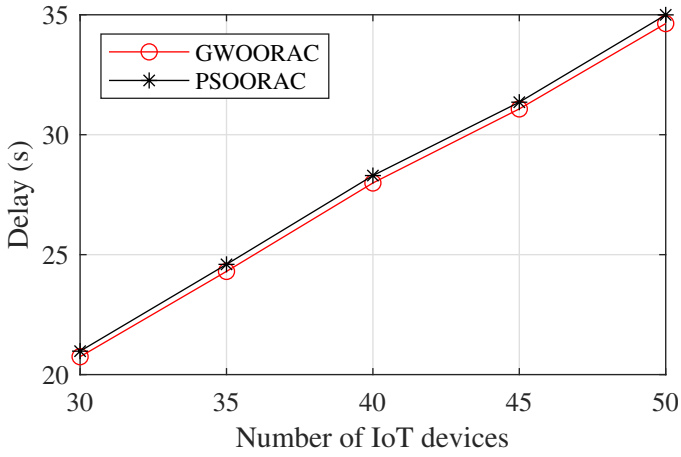

Figure 8. Delay vs. number of IoT devices.

7. Conclusions

In this paper, we investigate inter-satellite cooperative task offloading and resource allocation in an MEC-enabled STN. The task completion delay optimization problem for all IoT devices when the tasks are indivisible is formulated and decomposed. Then, we propose a joint offloading decision and resource allocation optimization scheme, which consists of a task offloading decision algorithm based on the GWO algorithm and a computing resource allocation algorithm based on the Lagrange multiplier method, thus obtaining the optimal task offloading decision and allocating optimal computing resources of the local satellite's MEC server and neighboring satellites' MEC servers. Simulation results validate that the proposed scheme performs better than the other baseline schemes in the following cases: the number of IoT devices is large, the number of satellites is large, each satellite has numerous computing resources, the computation intensity is high, or the data size is big. Much work remains to be done in the future. For example, we will investigate inter-satellite cooperation, where multiple satellites with MEC servers can simultaneously receive tasks offloaded by IoT devices and assist IoT devices in computing tasks in areas not covered by terrestrial networks.

Author Contributions: Conceptualization, M.T.; methodology, M.T. and S.L.; software, M.T.; validation, M.T.; formal analysis, M.T.; investigation, M.T.; resources, X.W.; data curation, M.T.; writing—original draft preparation, M.T.; writing—review and editing, M.T., S.L., X.W. and P.W.; visualization, M.T.; supervision, S.L.; project administration, S.L. and X.W.; funding acquisition, X.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program of China grant number 2018YFC1504502.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the editors and the reviewers for their helpful suggestions and constructive comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. IMT-2030 6G Promotion Group. *The 6G Overall Vision and Potential Key Technologies White Paper*, 1st ed.; IMT-2030 6G Promotion Group: Beijing, China, 2021; pp. 1–32. (In Chinese)
2. Chen, S.; Sun, S.; Kang, S. System Integration of Terrestrial Mobile Communication and Satellite Communication-The Trends, Challenges and Key Technologies in B5G and 6G. *China Commun.* **2020**, *17*, 156–171. [CrossRef]
3. Tirmizi, S.B.R.; Chen, Y.; Lakshminarayana, S.; Feng, W.; Khuwaja, A.A. Hybrid Satellite–Terrestrial Networks toward 6G: Key Technologies and Open Issues. *Sensors* **2022**, *22*, 8544. [CrossRef] [PubMed]
4. Zhang, F.; Wang, M.M. Stochastic Congestion Game for Load Balancing in Mobile-Edge Computing. *IEEE Internet Things J.* **2021**, *8*, 778–790. [CrossRef]
5. Zhao, T.; Zhou, S.; Song, L.; Jiang, Z.; Guo, X.; Niu, Z. Energy-Optimal and Delay-Bounded Computation Offloading in Mobile Edge Computing with Heterogeneous Clouds. *China Commun.* **2020**, *17*, 191–210. [CrossRef]
6. Patel, M.; Naughton, B.; Chan, C.; Sprecher, N.; Abeta, S.; Neal, A. *Mobile-Edge Computing-Introductory Technical White Paper*, 1st ed.; ETSI: Sophia Antipolis, France, 2014; pp. 1–36.
7. Liu, X.; Zhao, X.; Liu, G.; Huang, F.; Huang, T.; Wu, Y. Collaborative Task Offloading and Service Caching Strategy for Mobile Edge Computing. *Sensors* **2022**, *22*, 6760. [CrossRef] [PubMed]
8. Liu, J.; Zhang, Q. Code-Partitioning Offloading Schemes in Mobile Edge Computing for Augmented Reality. *IEEE Access* **2019**, *7*, 11222–11236. [CrossRef]
9. Huang, S.; Zhang, J.; Wu, Y. Altitude Optimization and Task Allocation of UAV-Assisted MEC Communication System. *Sensors* **2022**, *22*, 8061. [CrossRef]
10. Zou, C.; Wang, H.; Chang, J.; Shao, F.; Shang, L.; Li, G. Optimal Progressive Pitch for OneWeb Constellation with Seamless Coverage. *Sensors* **2022**, *22*, 6302. [CrossRef]
11. Jin, L.; Wang, L.; Jin, X.; Zhu, J.; Duan, K.; Li, Z. Research on the Application of LEO Satellite in IOT. In Proceedings of the 2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI), Changchun, China, 27–29 May 2022; pp. 739–741. [CrossRef]
12. Xie, R.; Tang, Q.; Wang, Q.; Liu, X.; Richard Yu, F.; Huang, T. Satellite-Terrestrial Integrated Edge Computing Networks: Architecture, Challenges, and Open Issues. *IEEE Netw.* **2020**, *34*, 224–231. [CrossRef]
13. Xiao, Z.; Dai, X.; Jiang, H.; Wang, D.; Chen, H.; Yang, L.; Zeng, F. Vehicular Task Offloading via Heat-Aware MEC Cooperation Using Game-Theoretic Method. *IEEE Internet Things J.* **2020**, *7*, 2038–2052. [CrossRef]
14. Yang, Z.; Liu, H.; Jin, J.; Tian, F. A Cooperative Routing Scheme Using Inter-Satellite Links to Assist Data Downloading for LEO Satellite Networks. *Sensors* **2022**, *22*, 7986. [CrossRef] [PubMed]
15. Liu, S.; Guo, X.; Lai, J.; Yang, J. Distributed Timekeeping in BeiDou Inter-satellite Link Network. *IEEE Commun. Lett.* 2022, *in press*. [CrossRef]
16. Li, S.; Sun, W.; Sun, Y.; Huo, Y. Energy-Efficient Task Offloading Using Dynamic Voltage Scaling in Mobile Edge Computing. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 588–598. [CrossRef]
17. Zhang, G.; Zhang, S.; Zhang, W.; Shen, Z.; Wang, L. Joint Service Caching, Computation Offloading and Resource Allocation in Mobile Edge Computing Systems. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 5288–5300. [CrossRef]
18. Zhu, X.; Jiang, C. Integrated Satellite-Terrestrial Networks Toward 6G: Architectures, Applications, and Challenges. *IEEE Internet Things J.* **2022**, *9*, 437–461. [CrossRef]
19. Zhang, Z.; Zhang, W.; Tseng, F.H. Satellite Mobile Edge Computing: Improving QoS of High-Speed Satellite-Terrestrial Networks Using Edge Computing Techniques. *IEEE Netw.* **2019**, *33*, 70–76. [CrossRef]
20. Al Homssi, B.; Al-Hourani, A.; Wang, K.; Conder, P.; Kandeepan, S.; Choi, J.; Allen, B.; Moores, B. Next Generation Mega Satellite Networks for Access Equality: Opportunities, Challenges, and Performance. *IEEE Commun. Mag.* **2022**, *60*, 18–24. [CrossRef]
21. Cao, X.; Li, Y.; Xiong, X.; Wang, J. Dynamic Routings in Satellite Networks: An Overview. *Sensors* **2022**, *22*, 4552. [CrossRef]
22. Pi, J.; Ran, Y.; Wang, H.; Zhao, Y.; Zhao, R.; Luo, J. Dynamic Planning of Inter-Plane Inter-Satellite Links in LEO Satellite Networks. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 3070–3075. [CrossRef]
23. Wang, H.; Han, J.; Cao, S.; Zhang, X. Computation Offloading Strategy of Multi-satellite Cooperative Tasks Based on Genetic Algorithm in Satellite Edge Computing. In Proceedings of the 2021 International Conference on Space-Air-Ground Computing (SAGC), Huizhou, China, 23–25 October 2021; pp. 22–28. [CrossRef]

24. Wang, Y.; Zhang, J.; Zhang, X.; Wang, P.; Liu, L. A Computation Offloading Strategy in Satellite Terrestrial Networks with Double Edge Computing. In Proceedings of the 16th IEEE International Conference on Communication Systems (IEEE ICCS), Chengdu, China, 19–21 December 2018; pp. 450–455. [CrossRef]
25. Zhang, J.; Zhang, X.; Wang, P.; Liu, L.; Wang, Y. Double-Edge Intelligent Integrated Satellite Terrestrial Networks. *China Commun.* **2020**, *17*, 128–146. [CrossRef]
26. Tang, Q.; Fei, Z.; Li, B.; Han, Z. Computation Offloading in LEO Satellite Networks With Hybrid Cloud and Edge Computing. *IEEE Internet Things J.* **2021**, *8*, 9164–9176. [CrossRef]
27. Song, Z.; Hao, Y.; Liu, Y.; Sun, X. Energy-Efficient Multiaccess Edge Computing for Terrestrial-Satellite Internet of Things. *IEEE Internet Things J.* **2021**, *8*, 14202–14218. [CrossRef]
28. Abu-Taleb, N.A.; Hasan Abdulrazzak, F.; Zahary, A.T.; Al-Mqdashi, A.M. Offloading Decision Making in Mobile Edge Computing: A Survey. In Proceedings of the 2022 2nd International Conference on Emerging Smart Technologies and Applications (eSmarTA), Ibb, Yemen, 25–26 October 2022; pp. 1–8. [CrossRef]
29. Luo, Q.; Hu, S.; Li, C.; Li, G.; Shi, W. Resource Scheduling in Edge Computing: A Survey. *IEEE Commun. Surv. Tutorials* **2021**, *23*, 2131–2165. [CrossRef]
30. Deng, R.; Di, B.; Zhang, H.; Kuang, L.; Song, L. Ultra-Dense LEO Satellite Constellations: How Many LEO Satellites Do We Need? *IEEE Trans. Wirel. Commun.* **2021**, *20*, 4843–4857. [CrossRef]
31. Lee, Y.; Choi, J.P. Connectivity Analysis of Mega-Constellation Satellite Networks with Optical Intersatellite Links. *IEEE Trans. Aerosp. Electron. Syst.* **2021**, *57*, 4213–4226. [CrossRef]
32. Ekici, E.; Akyildiz, I.F.; Bender, M.D. A distributed routing algorithm for datagram traffic in LEO satellite networks. *IEEE ACM Trans. Netw.* **2001**, *9*, 137–147. [CrossRef]
33. Tang, L.; Hu, H. Computation Offloading and Resource Allocation for the Internet of Things in Energy-Constrained MEC-Enabled HetNets. *IEEE Access* **2020**, *8*, 47509–47521. [CrossRef]
34. Hou, Y.; Gao, H.; Wang, Z.; Du, C. Improved Grey Wolf Optimization Algorithm and Application. *Sensors* **2022**, *22*, 3810. [CrossRef]
35. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]