# Dynamic Computation Offloading in Satellite Edge Computing

Lei Cheng*†, Gang Feng*, Yao Sun†, Mengjie Liu*, Shuang Qin*
*National Key Laboratory of Science and Technology on Communications,
and Yangtze Delta Region Institute(Huzhou),
University of Electronic Science and Technology of China
†James Watt School of Engineering, University of Glasgow
E-mail:fenggang@uestc.edu.cn

ICC 2022 - IEEE International Conference on Communications | 978-1-5386-8347-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICC45855.2022.9838943

***Abstract*—Satellite edge computing (SEC) has become a promising technology for future wireless networks to provide anywhere and anytime computing services. Different from terrestrial edge computing, the computing capacity at Low-Earth-Orbit (LEO) satellites is usually unstable, due to the limited and consistently changing energy supply of fast-orbiting LEO satellites. To well exploit the potentials of SEC, an optimal computation offloading strategy becomes imperative to determine when and how to offload computing tasks with respect to high dynamics of satellites. In this paper, we propose a dynamic offloading strategy to minimize the overall delay of tasks from terrestrial users in a SEC network, subject to the energy and computing capacity constraints of the LEO satellite. Based on Lyapunov optimization theory, a long-term stochastic problem with a time-varying energy constraint is converted into multiple deterministic one-slot problems parameterized by the current system state, where task offloading decisions, computing resource allocation and transmit power control are jointly optimized. Numerical results show that our algorithm achieves asymptotic optimality efficiently while maintaining the mean rate stable of the LEO satellite's energy queue, and has a lower delay compared with the other two comparison approaches with acceptable energy consumption.**

## I. Introduction

Satellite network has the prominent advantages of large capacity, long-distance communication, and immunity to disasters, and has shown great potential to assist existing terrestrial network to form the future satellite-terrestrial integrated network (STIN). More importantly, edge computing servers deployed at Low-Earth-Orbit(LEO) satellites can provide swift computing services for users in a large geographical area, which has become a hot computing paradigm, called satellite edge computing (SEC). Compared with conventional mobile edge computing (MEC) in ground networks, on the one hand, SEC can accommodate more computing tasks offloaded from a wider coverage area, even the place with no terrestrial infrastructure deployed [1]. On the other hand, by sinking abundant resources to LEO satellites, SEC significantly reduces end-to-end transmissions so that delay can be further reduced compared with transmitting to remote terrestrial cloud center.

This work has been supported by the National Natural Science Foundation of China under Grant number 61871099, and the Fundamental Research Funds for the Central Universities under Grant number ZYGX2020ZB044.

Before enjoying the great benefits of SEC, it is significant yet challenging to design an efficient SEC offloading strategy due to strong spatiotemporal constraints on STIN, which are brought by the inherent physical characteristics of satellite networks, such as topology dynamics, long propagation delay, limited resource, etc. Specially, with steadily increasing energy consumption of computation-intensive and delay-sensitive tasks, the energy budgets of LEO satellites have become one of the most crucial constraints. Typically, the energy supply for LEO satellites is provided by solar panels and battery cells, and the energy level consistently changes with energy harvesting and consumption along movement [2]. As a LEO satellite orbits earth, it may experience prolonged darkness with insufficient energy input, leading to unavailability to serve all offloading or transmission requests from users within its coverage [3]. Although the energy supply is abundant, the harvestable energy should be consumed in a cost-effective way, so as to save the operational expenditure for satellites [4]. Moreover, since the energy consumption associates with close coupled offloading decisions, computing and communication resource allocation, a joint optimization should be investigated to make energy-efficient and performance-optimization decisions subject to LEO satellite energy constraints in a long-term vision.

However, the energy constraint of LEO satellites was largely ignored in previous related works, despite the fact that the problem of minimizing total energy consumption and/or latency of tasks has been well addressed [5]–[7]. Most of the related work adopts a two-tier network architecture, in which both terrestrial networks (including users and base stations) and satellite networks are considered. With consideration of backhaul capacity constraints, the problem of joint user association and offloading decision in a SEC-enhanced STIN is studied in [5] to maximize the sum data rate via matching algorithms. By introducing rich resources of the cloud computing center, a three-tier computation architecture with both edge and cloud computing is proposed to provide multifarious services [6], [7]. A distributed algorithm based on the alternating direction method of multipliers (ADMM) is developed to minimize the sum energy consumption of ground users in [6] under the constraints of LEO coverage time and computing capability. To

deal with environmental dynamics, the authors of [7] present a model-free reinforcement learning (RL)-based approach to investigate computation offloading policy. Nevertheless, little attention has been paid to the energy budgets of satellites, which is indeed crucial in STIN.

In this paper, we propose a drift-plus-penalty based dynamic offloading strategy in a LEO satellite edge computing network to minimize the overall delay of computing tasks, subject to the energy and computing capability constraints of terrestrial UDs and the LEO satellite. To deal with the randomness of task arrivals, time-varying channel condition, and consistent energy harvesting, we convert the formulated long-term stochastic optimization problem into multiple one-slot problems leveraging the framework of Lyapunov optimization, which helps optimize time averages without knowing the future system states, while maintaining mean rate stable of the energy queue. As the converted one-slot problem is nonconvex, we decompose it into multiple equivalent sub-problems, which can be iteratively solved with high efficiency. Simulation results demonstrate the asymptotic optimality and effectiveness of the proposed dynamic offloading algorithm along the movement of the LEO satellite, and the performance of the proposed algorithm in terms of overall delay and energy consumption outperforms the other two comparison approaches.

The rest of the paper is structured as follows. Section II presents the system model and problem formulation. Section III elaborates the proposed drift-plus-penalty algorithm and the corresponding one-slot solution. We evaluate the system performance in Section IV and finally conclude the paper in Section V.

## II. System Model and Problem Formulation

### A. Model of Satellite and Terrestrial Integrated Network

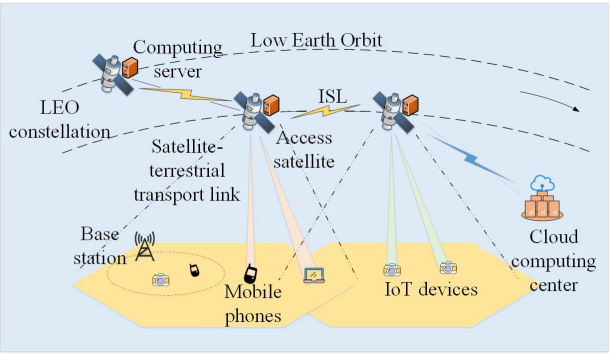



Fig. 1. Illustration of satellite and terrestrial integrated network architecture.

We consider a satellite-terrestrial integrated network shown in Fig. 1. The terrestrial user devices (UDs), such as mobile phones and IoT devices, are located in a wide area. Some UDs can offload their computing tasks via terrestrial networks, while the UDs without the support of ground communication infrastructure have to resort to satellite networks to fulfill their requirements. As the computing offloading problem in terrestrial networks has been intensively addressed and many efficient solutions have been provided in previous work, we focus on the UDs without the support of terrestrial communication facility in this work. Over the sky, multiple LEO satellites constitute constellations to achieve seamless global coverage. Each LEO satellite orbits the earth periodically and projects a large coverage area on the ground, in which multiple satellite-terrestrial communication connections can be established. Besides allowing multi-access, each LEO satellite is equipped with a computing server, and thus can provide computing service for UDs in its coverage. Furthermore, each LEO satellite is also connected to the cloud computing center via a high-speed backhaul link [6].

Without loss of generality, we consider a scenario that a LEO satellite, namely $\mathcal{L}$, provides access and computing complement to a set of UDs $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_N\}, N = |\mathcal{D}|$, within its coverage. The time is divided into $T$ slots, denoted by $\tau = \{1, 2, ..., T\}$, for ease of discussion of sequential offloading decision problem. Hypothetically, the number of computing task arrivals within the coverage of a LEO satellite in each slot $t$ obeys Poisson point progress (PPP), i.e., $N(t) \sim P(\lambda)$ [3]. We further assume that the tasks will be generated from a finite application set $\mathcal{A}$ , in which each type $a$ can be denoted by a tuple $\left(w_a^{in}, f_a, \tau_a^{\max}\right)$, where $w_a^{in}$, $f_a$ and $\tau$ represent data size (bits), required CPU cycles per bit, and maximum tolerable delay (s) respectively. A type-$a$ computing task generated by $\mathcal{D}_n \in \mathcal{D}$ in slot $t$ is denoted by $A_n^a(t)$.

Benefiting from code partitioning technology, we assume that computing tasks are separable in this paper, just like that in [8]. A task can be partially processed at different places simultaneously. We introduce an offloading vector $x_n(t) = \left[x_n^l(t), x_n^e(t), x_n^c(t)\right]$ to denote the partitioned portions of the task $A_n^a(t)$, where $x_n^l(t), x_n^e(t)$ and $x_n^c(t)$ denote the percentage of a task that processed locally, at the LEO server and at the cloud computing center, respectively. Due to the resource constraints on UDs and the LEO satellite, a task may be blocked and dropped, which is denoted by $x_n^d(t) = 1$, otherwise $x_n^d(t) = 0$. Therefore, these four variables are constrained by the following equations:

$$\begin{aligned} x_n^l(t) + x_n^e(t) + x_n^c(t) + x_n^d(t) = 1, \forall t \in \{1, 2, ..., T\}, \\ \text{where } x_n^l(t), x_n^e(t), x_n^c(t) \in [0, 1], x_n^d(t) \in \{0, 1\}. \end{aligned} \tag{1}$$

### B. Delay and Energy Consumption

In this subsection, we present the delay and energy consumption model. In this work, the total delay of each task consists of delay for transmission, propagation, and computation. As the size of computation results is usually much smaller than that of the input data, we ignore the delay on downlink transmission.

*Local Computing:* If the task $A_n^a(t)$ is partially processed locally, and the CPU frequency is $z_n^D(t)$, the delay for local computing is

$$d_n^D(t) = \frac{x_n^l(t) \cdot \mathcal{CPU}_a}{z_n^D(t)}, \tag{2}$$

where $\mathcal{CPU}_a = w_a^{in} f_a$ is the total CPU cycles needed, and $z_n^D(t)$ is physically constrained by the local computing capacity

$z^D$, i.e., $0 \leq z_n^D(t) \leq z^D$. We denote local computing energy consumption by

$$e_n^{D,l}(t) = \kappa \cdot x_n^l(t) \cdot \mathcal{CPU}_a \cdot z_n^D(t)^2, \tag{3}$$

where $\kappa$ is the effective switched capacitance related to hardware structures [6].

*Offloading to the LEO satellite:* When a part of the task $A_n^a(t)$ is offloaded to the LEO satellite for the procession, it will first be transmitted via a satellite-terrestrial link. Assume that the bandwidth is shared among multiple terrestrial UDs, and there is no interference with each other due to geographical isolation. We denote by $\sigma_0$ the noise power spectral density at satellite receiver, and the transmission rate $r_n^D(t)$ of UD $\mathcal{D}_n$ associated with the LEO satellite according to Shannon's theorem is given by

$$r_n^D(t) = B\log_2\left(1 + \frac{g_n(t)\, p_n^D(t)}{\sigma_0 B}\right), \tag{4}$$

where $B$ represents the uplink bandwidth, and $p_n^D(t)$ denotes the transmit power. $g_n(t)$ is the channel gain composed of antenna gain, large-scale fading (determined by geographical distance $l_n$ and environmental attenuation, etc.) and shadowed Rician fading [6]. The task delay $d_n^e(t)$ consists of transmission delay, task computing delay and propagation delay, and can be expressed as

$$d_n^e(t) = x_n^e(t)\left(\frac{w_a^{in}}{r_n^D(t)} + \frac{\mathcal{CPU}_a}{z_n^S(t)}\right) + \frac{2l_n}{c}, \tag{5}$$

where $z_n^S(t)$ is the amount of computing resource allocated to task $A_n^a(t)$, and the maximum capacity that is allocated to each task is $z^S$, i.e., $0 \leq z_n^S(t) \leq z^S$.

In this way, the task energy consumption composed of UD's transmission consumption $e_n^{e,D}(t)$ and the satellite's computation consumption $e_n^{e,S}(t)$, can be respectively expressed as

$$e_n^{D,e}(t) = p_n^D(t)\frac{x_n^e(t)\, w_a^{in}}{r_n^D(t)}, \tag{6}$$

$$e_n^{S,e}(t) = \kappa \cdot x_n^e(t) \cdot \mathcal{CPU}_a \cdot \left(z_n^S(t)\right)^2. \tag{7}$$

*Offloading to the cloud server:* Similarly, we analyze the scenario if the task $A_n^a(t)$ is partially processed at the cloud computing center. As there is abundant computing resource at the cloud computing center, we assume that the task computation delay is negligible. Then the UD's transmission delay, satellite's transmission delay, and propagation delay constitute the task delay $d_n^c(t)$ as follows

$$d_n^c(t) = x_n^c(t)\, w_a^{in}\left(\frac{1}{r_n^D(t)} + \frac{1}{r_n^S(t)}\right) + \frac{2l_n}{c}, \tag{8}$$

where the backhaul transmission rate and power are $r_n^S(t)$ and $p_n^S(t)$ respectively. Furthermore, the task energy consumption including UD's transmission consumption $e_n^{c,D}(t)$ and the satellite's transmission consumption $e_n^{c,S}(t)$ can be respectively expressed as

$$e_n^{D,c}(t) = p_n^D(t)\frac{x_n^c(t)\, w_a^{in}}{r_n^D(t)}, \tag{9}$$

$$e_n^{S,c}(t) = p_n^S(t)\frac{x_n^c(t)\, w_a^{in}}{r_n^S(t)}. \tag{10}$$

In summary, a task $A_n^a(t)$ can be processed in parallel in different ways, and thus the execution delay $d_n(t)$ for $A_n^a(t)$ is expressed as

$$\begin{aligned} d_n(t) = \max(&1_{\{x_n^l(t)>0\}} \cdot d_n^l(t), 1_{\{x_n^e(t)>0\}} \cdot d_n^e(t), \\ &1_{\{x_n^c(t)>0\}} \cdot d_n^c(t)), \end{aligned} \tag{11}$$

where $1_{\{x\in A\}} = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{if } x \notin A \end{cases}$.

Nevertheless, some tasks may be dropped due to the energy and computing capacity constraints of UDs' and the LEO satellite. With this regard, we need to count the extra delay incurred, as the dropped task has to be retransmitted and executed. Therefore, the delay of task $A_n^a(t)$ is expressed as

$$\gamma_n(t) = d_n(t) + \beta \cdot 1_{\{x_n^d(t)=1\}}, \tag{12}$$

where $\beta$ represents the extra delay introduced by a re-executing dropped task.

Furthermore, the task needs to be accomplished in a given time,

$$\gamma_n(t) \leq \tau_a^{\max}. \tag{13}$$

### C. Problem Formulation

Typically, solar cells and batteries provide stable energy storage for the payload on LEO satellites. As handovers between sunlight and darkness frequently occur along the movement, the LEO satellite may not be able to serve all requests due to energy limitations. Therefore, in designing the SEC offloading strategies, we need to take into account the constraints of energy budgets and model the energy evolution progress of the LEO satellite. The harvestable energy of the satellite in slot $t$, denoted by $e^{h,S}(t)$, is a constraint given by the satellite's location, inclination, and solar panel configuration [4]. We denote the battery level of the satellite at the beginning of slot $t$ as $E^S(t)$, which evolves according to the following equation:

$$E^S(t+1) = E^S(t) - e^S(t) + e^{S,h}(t), \tag{14}$$

where $e^S(t) = \sum_{n=1}^{N}\left(e_n^{S,e}(t) + e_n^{S,c}(t)\right) + e^{S,r}(t) \leq E^S(t)$ is the energy consumption of the LEO satellite in slot $t$ for transmission, computation and nominal operation $e^{S,r}(t)$, and is denoted as $e^S(x_n(t), z_n(t), p_n(t))$. Besides, in each slot, the energy consumed is generally constrained by $e^S(t) \leq E^{S,\max}$ to eliminate the close coupling effect over multiple slots in constraint (14), where $E^{S,\max}$ is the maximum energy consumption determined by the discharge depth of the battery [2].

Different from the LEO satellite, UDs in the remote area usually have no additional energy supply, and its energy consumption $e_n^D(t)$ is used for task transmission, local task computation, and nominal operation $e_n^{D,r}(t)$, which can be expressed as

$$e_n^D(t) = e_n^{D,l}(t) + e_n^{D,e}(t) + e_n^{D,c}(t) + e_n^{D,r}(t) \leq E^{D,\max}, \tag{15}$$

where $E^{D,\max}$ is the energy capacity of UDs. Similarly, the energy consumption of $\mathcal{D}_n$ in slot $t$ is denoted as $e_n^D\left(x_n\left(t\right), z_n\left(t\right), p_n\left(t\right)\right)$.

In addition, as multiple users in the coverage area can simultaneously request a specific satellite for execution/transmission, task offloading and resource management need to be optimized jointly under the energy constraints of the satellite. With the aim to minimize the overall delay of UDs while satisfying the LEO satellite's and UDs' energy constraints, we formulate the joint task offloading and resource allocation optimization problem as

$$P_0: \min_{x_n(t), z_n^D(t), z_n^S(t), p_n^D(t), p_n^S(t)} \frac{1}{T}\sum_{t=1}^{T} E\left[\sum_{n=1}^{N(t)} \gamma_n(t)\right]$$

s.t.

$$C_1: e^S\left(x_n(t), z_n(t), p_n(t)\right) \le E^{S,\max},$$
$$C_2: e_n^D\left(x_n(t), z_n(t), p_n(t)\right) \le E^{D,\max}, n = 1, 2, \ldots, N(t)$$
$$C_3: 0 \le z_n^D(t) \le z^D \cdot 1_{\{x_n^l(t)>0\}},$$
$$0 \le z_n^S(t) \le z^S \cdot 1_{\{x_n^c(t)+x_n^c(t)>0\}}, n = 1, 2, \ldots, N(t)$$
$$C_4: 0 \le p_n^D(t) \le p^{D,\max} \cdot 1_{\{x_n^c(t)+x_n^c(t)>0\}},$$
$$0 \le p_n^S(t) \le p^{S,\max} \cdot 1_{\{x_n^c(t)>0\}}, n = 1, 2, \ldots, N(t)$$
(1) for task splitting,
(13) for task delay requirement,
(14)(15) for energy constraints.

The objective function of $P_0$ is minimizing the long-term average of overall delay of tasks, for a specific system state (e.g., the LEO satellite's energy level, the terrestrial-satellite channel state, the number of arrival tasks, etc.) by determining the task offloading proportion $x_n\left(t\right)$, computation resource $z_n\left(t\right)$, and transmit power $p_n\left(t\right)$ in each slot. Constraints $C_1$ and $C_2$ indicate the energy limitation for UDs and the LEO satellite. Constraints $C_3$ and $C_4$ indicate the physical constraints of UDs' and the LEO server's computing capacity and transmit power. (1) limits the ranges of the offloading variables. (13) indicates that each task needs to be finished before its deadline. The energy constraint of the LEO satellite for adjacent slots is given in (14). $P_0$ is a long-term stochastic optimization problem containing continuous variables and integer variables.

## III. Lyapunov Optimization Based online Offloading Strategy

Leveraging the framework of Lyapunov optimization, we are able to convert this problem into multiple deterministic one-slot optimization problems $P_1$ at individual slots. Then we solve $P_1$ by iteratively solving the decomposed sub-problems.

### A. Optimization by minimizing Drift-plus-penalty

First, we set a virtual energy queue as $\widehat{E}^S(t) = E^S\left(t\right) - \phi$, which has the flavor of keeping the queue energy backlog near a non-zero auxiliary constant $\phi$ [9]. Then we set a backlogvector as $\Theta(t) = \widehat{E}^S(t)$, and the Lyapunov function is defined as $L(\Theta(t)) = \frac{1}{2}\widehat{E}^S(t)^2$. Accordingly, the Lyapunov drift function with respect to $L\left(\Theta\left(t\right)\right)$ is defined as follows:

$$\Delta\left(\Theta\left(t\right)\right) = E\left\{L\left(\Theta\left(t+1\right)\right) - L\left(\Theta\left(t\right)\right) | \Theta\left(t\right)\right\}. \quad (16)$$

**Algorithm 1** Drift-plus-penalty based Dynamic Offloading

**For** each time slot $t$

1: Obtain the task arrival distribution $N(t)$, the task type $A_n(t)$, the channel power gain $h_n(t)$, the satellite virtual energy queue length $\widehat{E}^S(t)$, the harvestable energy $e^{h,S}\left(t\right)$;

2: Determine task offloading vector $x_n(t)$, computing resource $z_n(t)$, transmit power $p_n(t)$ by solving $P_1$;

3: Update the energy queue according to (14);

4: $t = t + 1$;

**Until** $t > T$

Next, in order to minimize the objective function and stabilize the energy queue, we optimize the following Lyapunov drift-plus-penalty function in each slot:

$$\Phi_V\left(t\right) = V \cdot \gamma\left(t\right) + \Delta\left(\Theta\left(t\right)\right), \quad (17)$$

where $V \ge 0$ is a tuning factor with unit as $J^2 \cdot second^{-1}$ and $\gamma\left(t\right) = \sum_{n=1}^{N(t)} \gamma_n\left(t\right)$ is the sum of task delay in slot $t$.

*Lemma 1*: For arbitrary feasible decision variables $x_n\left(t\right), z_n\left(t\right), p_n\left(t\right)$ for $P_0$, $\Phi_V\left(t\right)$ is upper bounded as follows

$$\Phi_V\left(t\right) = V \cdot \gamma\left(t\right) + \Delta\left(\Theta\left(t\right)\right)$$
$$\le B^{'} + V \cdot \gamma\left(t\right) - \widehat{E}^S\left(t\right)\left(e^{h,S}\left(t\right) - e^S\left(t\right)\right).$$

where $B^{'} = \frac{1}{2}(E^{S,\max} + \Psi^{\max})^2$, and $\Psi^{\max}$ is the amount of maximum harvestable energy.

*Proof*: The proof can be achieved through equivalent transforms to (16)(17), and is omitted due to space limitation.

According to *Lemma 1*, we can obtain the approximate optimality of $\Phi_V\left(t\right)$ by minimizing the upper bound of $\Phi_V\left(t\right)$, and thus a stochastic optimization problem is converted to multiple one-slot problems. The one-slot problem for a specific slot $t$ is formulated as

$$P_1: \min_{x_n(t), z_n^D(t), z_n^S(t), p_n^D(t), p_n^S(t)} V \cdot \gamma(t) - \widehat{E}^S(t) e^S(t)$$
$$\text{s.t. } C_1 - C_4, (1), (13).$$

$P_1$ can be decomposed and then solved efficiently by using conventional methods such as the interior point method. Based on the solution of $P_1$ in each slot, we propose a drift-plus-penalty based dynamic offloading strategy, which can be summarized in Algorithm 1. In each slot, the current system state, i.e., the LEO satellite energy state $\widehat{E}^S(t)$, task distribution $N\left(t\right)$, arriving task type, and channel gain $h_n\left(t\right)$, can be observed. Next, the system decisions (task offloading proportion, computing resource allocation, transmit power control) are obtained by solving the one-slot optimization problem $P_1$ parameterized by the current system state, which yields an $\omega$-only policy. As the virtual energy queue of the LEO satellite satisfies the boundedness assumptions, the solution of the proposed algorithm can be arbitrarily closely to optimality with $O(\frac{1}{V})$ by setting $V$ to a sufficiently large value, while maintaining mean rate stable with $O(V)$ for the virtual energy queue [10].

### B. One-slot solution

We solve the one-slot problem $P_1$ in this subsection. To deal with the coupling issue of variables and the non-convexity of $P_1$, we further decompose $P_1$ into the following equivalent convex sub-problems and solve them iteratively:

*1) Task offloading (TO) Problem:* When the transmit power and computing resource are given, i.e., $z_n(t) = z_n, p_n(t) = p_n$, we try to obtain the optimal $x_n(t)^*$. If $x_n^d(t), n = 1, 2, ..., N(t)$ are all relaxed to continuous variables ranging in [0, 1], the task offloading problem will be convex with respect to the offloading vector $x_n(t)$. Then the TO problem can be efficiently solved by using some mature algorithms, such as the interior point method.

*2) Resource Allocation (RA) Problem:* When the task offloading vectors are given, i.e., $x_n(t) = x_n$, we aim to obtain the optimal solution of $z_n(t)^*, p_n(t)^*$. Observing that $e^S(t)$ is only relevant to $z_n^S(t), p_n^S(t)$, we can further decompose the RA problem into $N$ local RA problems and an edge RA problem as follows:

*Local Resource Allocation:* For each terrestrial UD, we get the optimal $z_n^D(t), p_n^D(t)$ by solving the following problem:

$$\begin{aligned} &P_{\mathrm{LRA}} \quad : V \cdot \gamma_n(t) \\ &\underset{z_n^D(t), p_n^D(t)}{\text{s.t. } C_2 - C_4, (13).} \end{aligned}$$

The difficulty of $P_{\mathrm{LRA}}$ lies on the cross term $\frac{p_n^D(t)}{r_n^D(t)}$ in $C_1$, as it is nonconvex with respect to $p_n^D(t)$. To solve this problem, we introduce a new variable $y_n^D(t)$,

$$y_n^D(t) = y_n^D\left(p_n^D(t)\right) = \frac{1}{B\log_2\left(1 + \frac{g_n(t)p_n^D(t)}{\sigma_0 B}\right)} = \frac{1}{r_n^D(t)}. \tag{18}$$

Next, by means of variable substitution, we define $\xi\left(y_n^D(t)\right)$ as follows:

$$\xi\left(y_n^D(t)\right) = \frac{p_n^D(t)}{r_n^D(t)} = \frac{\sigma_0 B}{g_n(t)} y_n^D(t)\left(2^{\frac{1}{y_n^D(t)B}} - 1\right). \tag{19}$$

As $\xi\left(y_n^D(t)\right)$ is convex with respect to $y_n^D(t)$, $P_{\mathrm{LRA}}$ can be solved by solving the optimal $z_n^D(t)^*$ and $y_n^D(t)^*$.

*Edge Resource Allocation:* By means of variable substitution, we can optimize $y_n^S(t)$ and $\xi\left(y_n^S(t)\right)$ defined similarly to (19), (20) and try to solve the following problem:

$$\begin{aligned} &P_{\mathrm{ERA}}^{'} : \quad V \cdot \gamma_n(t) - \widehat{E}^S(t)\, e^S(t) \\ &\underset{z_n^S(t), y_n^S(t)}{\text{s.t. } C_1, C_3, C_4, (13).} \end{aligned}$$

On the one hand, when $\widehat{E}^S(t) \leq 0$, it is intuitive that the objective function and constraints are convex with respect to $y_n^S(t)$ and $\xi\left(y_n^S(t)\right)$. However, when $\widehat{E}^S(t) > 0$, the objective function is nonconvex due to the negative-definiteness of $-\widehat{E}^S(t)e^S(t)$, which cannot be directly solved. Observing the monotony property of the objective function with respect to $z_n^S(t)$ and $y_n^S(t)$, we consider two cases:

1) The optimum is not achieved at the boundary of constraint $C_1$. Then the optimal solution must be $z_n^S(t)^* = z^S \cdot 1_{\{x_n^e(t)>0\}}$, and $p_n^S(t)^* = p^{S,\max} \cdot 1_{\{x_n^c(t)>0\}}$.

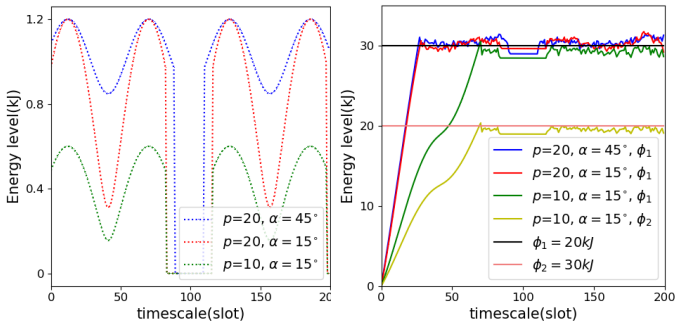
TABLE I: SIMULATION PARAMETERS

| Parameter | Value |
|---|---|
| LEO satellite altitude | 1500 km |
| Uplink/Backhaul bandwidth | 1 MHz/20 MHz |
| $E^{D,max}$ | 5 mJ |
| $E^{S,max}$ | 10 kJ |
| $z^D$ | 0.1 Gcycles/s |
| $z^S$ | 3 Gcycles/s |
| $P^{D,max}$ | 24 dBm |
| $P^{S,max}$ | 46 dBm |
| $\mathcal{A}$ | $\{(100k, 1000, 10), (400k, 1000, 10)\}$ |
| $\beta$ | $= \tau_a^{\max}$ = 10s |
| $\sigma_0$ | -174dBm/Hz |
| $\kappa$ | $10^{-25}$ |

2) The optimum is achieved at the boundary of constraint $C_1$. Then $-\widehat{E}^S(t)e^S(t)$ in objective function will be eliminated by constraint $C_1$, and $P_{\mathrm{ERA}}^{'}$ can then be solved.

## IV. Performance Evaluation

In this section, we conduct simulations to implement the dynamic offloading algorithm under different environmental settings, evaluate the performance of the proposed algorithm in terms of total task delay and required energy capacity, and compare it with the other two algorithms. In simulations, we use the LEO satellite power supply model used in [4], and other parameters for the STIN are shown in Table 1.

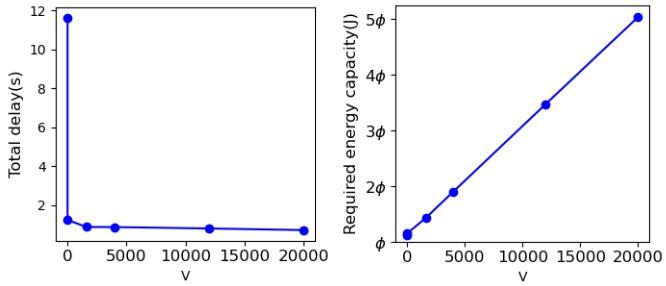



(a) Harvestable energy level vs. $t$. (b) Energy level of the LEO satellite vs. $t$.

Fig. 2. Energy evolution at the LEO satellite.

In the first experiment, we verify the feasibility of the proposed drift-plus-penalty algorithm under different absorption power of LEO satellite and perturbed index $\phi$. Fig. 2(a) shows the harvested energy as a function of time, where $p$ and $\alpha$ in the legend denote the absorption power and the angle between the orbital plane of the LEO satellite and the sunlight. It can be observed that the harvested energy varies even if the satellite is exposed to the sun. And the eclipse period with no energy supply is also shown. Fig. 2(b) presents the energy evolution corresponding to the harvested energy in Fig. 2(a). Please note that the energy consumption in each slot is optimized according to the solution given in Section. III. First, we observe that the energy level increases rapidly in the beginning, and eventually stabilizes around the perturbed energy level $\phi$ for all four settings. This is achieved through the minimization of the upper bound of the Lyapunov drift-plus-penalty function in the proposed algorithm, as is indicated by *Lemma 1*. The

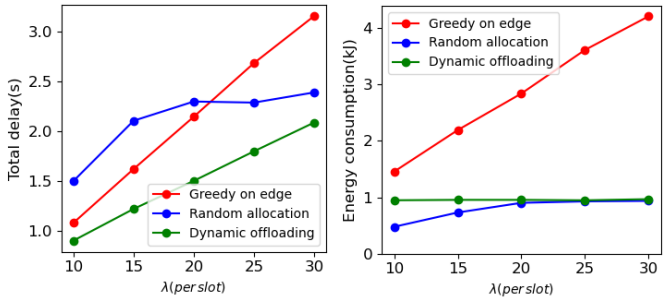
energy evolution curve grows faster to the perturbation index with a greater $p$. The eclipse period is also reflected in each curve, where the energy level nearly remains unchanged. This is due to the fact that local computing and offloading to the cloud center are largely chosen, as these two offloading strategies only consume little energy of the LEO satellite while guaranteeing a satisfactory performance in the current setting of the simulation.




(a) Overall delay vs. $V$. (b) Required energy capacity vs. $V$.

Fig. 3. Delay and required energy capacity vs. tuning factor.

Next, we examine the relationship between the average overall delay of tasks/the required LEO satellite energy capacity and tuning factor $V$, when the task arrival rate is set to be 10 per slot in Fig. 3. In Fig. 3(a), it can be observed that when $V$ approaches 0, the average total task delay reaches maximum. As $V$ increases, the average overall delay reduces sharply to around 1, and eventually converges to the optimality of $P_1$, which confirms the asymptotic optimality in afore analysis. However, as is indicated in Fig. 3(b), for a certain perturbation index, the required battery capacity increases linearly with $V$, since it takes a longer time for the energy queue to achieve mean rate stable, which implies a larger battery capacity is needed. For instance, with only $5\%$ performance improvement, more than four times energy capacity is needed for $V = 2\times10^4$ compared to $V = 40$. Thus, $V$ should be chosen for a good tradeoff between the overall delay and the required battery capacity.




(a) Total task delay vs. $\lambda$. (b) Averge energy consumption vs. $\lambda$.

Fig. 4. Total task delay and energy consumption vs. task arrival rate.

Finally, we compare the average total task delay and energy consumption when the task arrival rate $\lambda$ increases from 10 to 30 per slot in Fig. 4. Following the comparison algorithms developed in [7], the first strategy for comparison is the greedy on edge strategy, in which tasks are incessantly offloaded to the LEO server as long as there is enough energy. The other one is the random allocation strategy, in which the power and computing resources are randomly allocated. In Fig. 4(a), with the assistance of computing resources of the cloud center and UDs, dynamic offloading strategy achieves a lower delay than greedy on edge strategy. With optimal resource allocation, the delay can be reduced by $46.2\%$ on average compared with the random allocation strategy. In Fig. 4(b), it is noticeable that the energy consumption of greedy on edge strategy remains the highest, hence the available energy may be exhausted when the harvestable energy is insufficient. The energy consumption of the random allocation strategy is the lowest, which indicates that a higher energy capacity is needed, or the unused energy will be discarded. Since the dynamic offloading strategy makes full use of available energy, it has a slightly higher energy consumption than the random allocation strategy to achieve the minimum delay. As task arrival rate increases, more tasks are offloaded to the cloud computing center in the dynamic offloading strategy, and as a result, the energy consumption nearly remains at the same level.

## V. Conclusions

In this paper, we have proposed a dynamic offloading strategy by leveraging the framework of Lyapunov optimization in a LEO satellite edge computing network. We jointly optimize task offloading, computing resource allocation, and transmission power control by iteratively solving the decomposed subproblems, while maintaining energy queue mean rate stable. The near-optimality and effectiveness of the proposed algorithm are verified through simulations. By choosing appropriate control parameters, the overall delay of terrestrial users can be minimized with an acceptable energy capacity. The proposed algorithm has significantly reduced the overall delay when compared with greedy on edge, and random allocation strategy.

## References


[1] J. Liu, X. Du, J. Cui, M. Pan, and D. Wei, "Task-oriented intelligent networking architecture for space-air-ground-aqua integrated network," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2020.

[2] R. Liu, M. Sheng, K. S. Lui, X. Wang, Y. Wang, and D. Zhou, "An analytical framework for resource-limited small satellite networks," *IEEE Communications Letters*, vol. 20, no. 2, pp. 1–1, 2016.

[3] A. Fu, E. Modiano, and J. Tsitsiklis, "'optimal energy allocation and admission control for communications satellites," *IEEE/ACM Transactions on Networking*, 2003.

[4] Y. Yuan, M. Xu, W. Dan, and W. Yu, "Towards energy-efficient routing in satellite networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3869–3886, 2016.

[5] B. Di, H. Zhang, L. Song, Y. Li, and G. Y. Li, "Ultra-dense leo: Integrating terrestrial-satellite networks into 5g and beyond for data offloading," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2018.

[6] Q. Tang, Z. Fei, B. Li, and Z. Han, "Computation offloading in leo satellite networks with hybrid cloud and edge computing," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2021.

[7] N. Cheng, F. Lyu, W. Quan, C. Zhou, H. He, W. Shi, and X. Shen, "Space/aerial-assisted computing offloading for iot applications: A learning-based approach," *IEEE Journal on Selected Areas in Communications*, vol. PP, no. 5, pp. 1–1, 2019.

[8] Y. Deng, Z. Chen, X. Yao, S. Hassan, and A. Ibrahim, "Parallel offloading in green and sustainable mobile edge computing for delay-constrained iot system," *IEEE Transactions on Vehicular Technology*, vol. PP, no. 99, pp. 1–1, 2019.

[9] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, 2016.

[10] M. Neely, *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan and Claypool, 2010.