Towards Sustainable Satellite Edge Computing

Qing Li, Shangguang Wang, Xiao Ma, Ao Zhou, Fangchun Yang
State Key Laboratory of Networking and Switching Technology
Beijing University of Posts and Telecommunications
Beijing, China
{*q_li;sgwang;maxiao18;aozhou;fcyang*}*@bupt.edu.cn*

Abstract—**Recently, Low Earth Orbit (LEO) satellites experience rapid development and satellite edge computing emerges to address the limitation of bent-pipe architecture in existing satellite systems. Introducing energy-consuming computing components in satellite edge computing increases the depth of battery discharge. This will shorten batteries' life and influences the satellites' operation in orbit. In this paper, we aim to extend batteries' life by minimizing the depth of discharge for Earth observation missions. Facing the challenges of wireless uncertainty and energy harvesting dynamics, our work develops an online energy scheduling algorithm within an online convex optimization framework. Our algorithm achieves sub-linear regret and the constraint violation asymptotically approaches zero. Simulation results show that our algorithm can reduce the depth of discharge significantly.**

Keywords-**satellite edge computing; depth of discharge; online convex optimization.**

I. INTRODUCTION

Recently, Low Earth Orbit (LEO) satellites experience rapid development due to the reduced cost of both manufacturing and launching. Existing satellite systems operate under a bent-pipe architecture [1], where ground stations send control commands to orbits and satellites reply with raw data. This architecture relies heavily on the satellite-ground communication, which has limitations of high downlink latency, intermittent availability, and link unreliability [2]. Satellite edge computing equips satellites with computing resources and supports in-orbit data processing. Thus, it can reduce the downlink transmission load and provide scalability benefits when LEO satellite constellations scale up [2].

Introducing energy-consuming computing components in satellite edge computing brings a huge burden to energy systems, of which the capacities are inherently constrained due to strict volume and weight constraints [3]. Most LEO satellites have solar cells installed on their surface to harvest solar energy and store energy (usually in batteries) to keep functioning during the eclipse. Both energy harvesting and storage are constrained by physical sizes. For example, a common category of LEO satellites, CubeSats, are among [1, 15] kg in weight and of up to 12U in volume (1U = 10 cm × 10 cm × 10 cm) [4]. The harvested power of CubeSats ranges in [1, 7] watts because of the limited area of the solar arrays [4]. Besides, the capacities of batteries in

CubeSats range in dozens of watt-hour [3]. Introducing the computing components increases the depth of discharge (i.e., the amount of discharge energy) in each eclipse period. This will greatly shorten batteries' life, which further influences the satellites' operation in orbit [5]. Therefore, it is timely and important to extend batteries' life.

Extensive previous works focus on energy scheduling in ground energy harvesting systems involving many issues under different network contexts [6]–[9]. Applying these works directly to satellite systems is appealing, but they cannot address the challenges brought by high-speed satellite movement. Moreover, none of them attempts to extend batteries' life. One related work [10] focuses on extending batteries' life by transmission power control in LEO satellite networks. In our scenario, introducing edge computing to LEO satellite networks makes the problem more complex as it requires energy coordination between different energy-consuming components.

It is non-trivial to extend the batteries' life of LEO satellites due to the following challenges. The first challenge is how to optimize the depth of discharge in the unstable wireless environment. Each satellite orbits the Earth every ∼100 minutes, traveling at ∼27,000 kmph [11]. This high-speed movement of satellites creates high churn in satellite-ground links. Further, the bitrates of satellite-ground links are unpredictable due to the uncertainty of wireless environment such as weather conditions. The second challenge is how to adapt to energy harvesting dynamics. Typical low Earth orbits expose satellites to the Sun for about 66% of each ∼100 minutes orbit period [4]. The periodical satellite movement incurs significant energy harvesting dynamics when satellites show up in light and eclipse alternatively.

In this paper, we seek to extend the batteries' life in satellite edge computing by reducing the depth of discharge. We consider Earth observation missions, which consist of three energy-consuming processes: sensing, computing, and communication. We exploit the pattern information brought by periodical satellite movement and propose a novel optimal pattern-aware benchmark that generalizes state-of-the-art. Given the uncertainty of the wireless environment and the dynamics of energy harvesting, we propose a pattern-aware online energy scheduling algorithm within the online convex optimization framework. Our algorithm achieves

sub-linear regret (compared with the optimal pattern-aware benchmark) and the constraint violation asymptotically approaches zero. Simulation results show that our algorithm adapts to the energy harvesting dynamics and reduces the depth of discharge significantly.

II. RELATED WORK

Energy scheduling in ground energy harvesting systems. Energy harvesting systems have been investigated extensively on the ground. Most existing workS learn energy scheduling strategies online by reinforcement learning method to address the uncertainty in the energy harvesting process. Ortiz *et al.* learn a distributed energy allocation for both a transmitter and a relay with only partially observable system states [6]. Fraternali *et al.* aim to maximize the sensing quality of energy harvesting sensors for periodic and event-driven indoor sensing with available energy [7]. Huang *et al.* propose an adaptive processor frequency adjustment algorithm to plan the energy usage of energy harvesting edge servers [8]. Hatami *et al.* control sensors status update to minimize the energy cost considering the freshness requirement of the sensing information [9]. These works cannot be applied directly to the satellite edge computing systems for two reasons. First, they cannot address the challenges brought by the unique satellite movement, i.e., intermittent link availability. Second, they do not investigate the problem of extending the batteries' life.

Satellite edge computing. Satellite edge computing is still in its infancy. Most existing works focus on the space-air-ground network. Boero *et al.* [12], Giambene *et al.* [13], and Shi *et al.* [14] design the space-air-ground network architecture based on SDN and NFV technologies. Tang *et al.* [15] manage resources for SDN-based satellite-terrestrial networks in an on-demand way. Chen *et al.* propose the time-varying resource graph to model resources in space-terrestrial integrated networks [16]. They also investigate how to dynamically place and assign controllers in LEO satellite networks to adapt to satellite mobility and traffic load fluctuation [17]. Besides, a few works focus on satellite edge computing frameworks [18]. Denby *et al.* [2] propose to support computing in nano-satellite constellations to address existing ben-pipe architecture limitations. Tsuchida *et al.* [10] also focus on extending battery life in transmission power control problems for LEO satellite networks. Introducing edge computing to LEO satellite networks makes energy scheduling more complex as it requires energy coordination between different energy-consuming components.

III. SYSTEM MODEL

As shown in Figure 1, we consider an electrical power subsystem model in satellites [19]. It encompasses efficient and reliable energy generation, storage, and distribution to various on-board subsystems. The electrical power subsystem consists of a solar module, a battery module, a payload
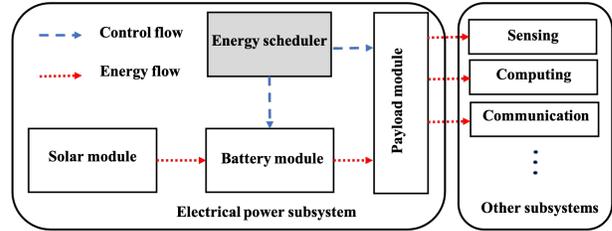

Figure 1: System model.

module, and an energy scheduler. The solar module is responsible for energy harvesting and transfer. Solar energy is the dominant energy source of LEO satellites, and about 85% of nano-satellites are equipped with solar panels [3]. The energy storage module provides energy supply when the solar energy is unavailable during on-orbit operations. The payload module regulates the energy to the other subsystems. On-board subsystems consume electrical energy to maintain satellite orbit motion or perform various missions. The energy scheduler decides how to allocate electrical energy to these subsystems. In this paper, we consider an energy scheduling problem for an Earth observation mission. To optimize the energy allocation among sensing, computing, and communication process, we abstract the energy harvesting, storage, and distribution as an energy queue model as shown in Figure 2. We divide time into slots indexed by t with duration T_{d}.

A. Energy Harvesting and Storage Model

Given the solar panel hardware, the energy harvesting rate $r_e(t)$ is dominated by two main factors: the light available to a satellite solar array and the projected surface area of the panels exposed to the Sun. The first factor varies with the inverse square of the distance from the Sun. The second factor varies with the angle between the solar panel and the Sun. We model the harvested energy in each time slot t as

$$E_{\mathrm{total}}(t) = r_e(t)T_{\mathrm{d}}. \tag{1}$$

The harvested energy varies intensely when satellites are in different light conditions. In the light, the harvested energy provides power to the energy-consuming subsystems directly and charges the batteries. In the eclipse, batteries provide energy to subsystems. The dynamics of energy buffered in the batteries can be modeled as an energy queue

$$E(t) = \min\{\max(E(t-1) + E_{\mathrm{in}}(t) - E_{\mathrm{out}}(t), 0), E_{\mathrm{max}}\}, \tag{2}$$

where $E_{\mathrm{in}}(t)$ is the amount of charge energy, $E_{\mathrm{out}}(t)$ is the amount of discharge energy, and E_{max} is the maximal battery capacity.

B. Energy Consumption Model

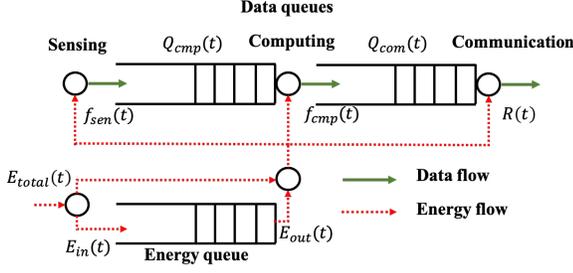The energy consumption can be divided into two categories: the mission level energy consumption (including

Figure 2: Queue model.

sensing energy $E_{\text{sen}}(t)$, computing energy $E_{\text{cmp}}(t)$, and communication energy $E_{\text{com}}(t)$) and the energy to perform the fundamental operations of the satellites denoted by $E_a(t)$. The total energy consumption satisfies

$$E_{\text{sen}}(t) + E_{\text{cmp}}(t) + E_{\text{com}}(t) + E_a(t) \leq \tag{3}$$
$$E_{\text{out}}(t) + E_{\text{total}}(t) - E_{\text{in}}(t).$$

1) Sensing Energy Model: The satellite collects the images of the Earth while moving along its orbit. The camera's frame rate $f_{\text{sen}}(t)$ (in frame/s) is adjustable to meet different missions demands. The sensing energy is

$$E_{\text{sen}}(t) = P_{\text{sen}}T_{\text{d}}, \tag{4}$$

where P_{sen} is the power of the camera.

2) Computing Energy Model: The sensed data (e.g., Earth imagery) can be hundreds of Gigabytes and its quality is fundamentally limited by the on-board cameras and orbit altitude [2]. It is unnecessary to transmit all the raw data to the ground. Besides, the satellite-ground downlink is not always available and the link bitrate is affected by many factors, e.g., orbit parameters, ground station capability, and location. These necessitate in-orbit data computing, which can significantly relieve the satellite-ground link pressure. A typical on-board computing example is to identify images of interest and separate them from raw data by CNN-based image classification, objective detection, or any other computation. The computing system adopts the dynamic voltage and frequency scaling technique [20] to adjust its CPU frequency denoted by $f_{\text{cmp}}(t)$ (in cycle/s). Let $0 \leq f_{\text{cmp}}(t) \leq f_{\text{cmp}}^{\max}(t)$, where $f_{\text{cmp}}^{\max}(t)$ is the maximal CPU frequency. The computation energy consumption is

$$E_{\text{cmp}}(t) = a f_{\text{cmp}}^3(t)T_{\text{d}}, \tag{5}$$

where a is the effective capacitance coefficient of computing hardware.

3) Communication Energy Model: The satellite-ground connection is intermittent because the satellite moves fast with respect to ground stations and the downlink session can only last for less than ten minutes in one single pass [2]. The satellite stores the processed data before connecting to a ground station. In each time slot t, the satellite-ground connection $I_c(t)$ is known as a prior, where $I_c(t) = 1$ means

that the satellite-ground connection is available, and vice versa. We model the transmission rate $R(t)$ (in bit/s) as

$$R(t) = B \log_2(1 + \frac{P_{\text{com}}(t)h(t)}{N_0}), \tag{6}$$

where $P_{\text{com}}(t)$ is the transmit power of the satellite, N_0 is the received noise power, and $h(t)$ is the channel gain. Let $0 \leq R(t) \leq R^{\max}I_c(t)$, where $R^{\max}$ is the maximal transmission rate. The communication energy consumption is

$$E_{\text{com}}(t) = P_{\text{com}}(t)T_{\text{d}}. \tag{7}$$

C. Data Queue Model

To describe the data buffers dynamics, we construct two data queues: a waiting-for-computing data queue $Q_{\text{cmp}}(t)$ between sensing and computing processes and a waiting-for-transmitting data queue $Q_{\text{com}}(t)$ between computing and communication processes. The arrival rate of the queue is the sensed data amount and the departure rate is the processed data amount in the current slot. We can update the waiting-for-computing data queue $Q_{\text{cmp}}(t)$ as

$$Q_{\text{cmp}}(t) = \max\{Q_{\text{cmp}}(t-1) + A_s(t) - \kappa f_{\text{cmp}}(t)T_{\text{d}}, 0\}, \tag{8}$$

where each CPU cycle executes κ bits of data known as a prior, $A_s(t) = D f_{\text{sen}}(t)T_{\text{d}}$ is the amount of sensed data, and D is the data size of each image frame.

After the mission completion, all sensed information is required to be downloaded to the ground for further use. Hence, we should have $Q_{\text{cmp}}(T+1) = 0$ with the initial queue length $Q_{\text{cmp}}(1) \geq 0$ by the long-term constraint

$$\sum_{t=1}^{T}(D f_{\text{sen}}(t) - \kappa f_{\text{cmp}}(t)) \leq 0. \tag{9}$$

The waiting-for-transmitting data queue $Q_{\text{com}}(t)$ buffers results of interest that need to be transmitted to the ground. Similarly, we can update the waiting-for-transmitting data queue as

$$Q_{\text{com}}(t) = \max\{Q_{\text{com}}(t-1) + \rho A_p(t) - R(t)T_{\text{d}}, 0\}, \tag{10}$$

where $\rho \in [0, 1]$ is the effective data proportion, the processed data amount $A_p(t) = \min\{Q_{\text{cmp}}(t-1) + A_s(t), \kappa f_{\text{cmp}}(t)T_{\text{d}}\}$. Similar to (9), the long-term constraint on $Q_{\text{com}}(t)$ is

$$\sum_{t=1}^{T}(\rho \kappa f_{\text{cmp}}(t) - R(t)) \leq 0. \tag{11}$$

D. Problem Formulation

In this paper, we aim to extend the battery life by minimizing the depth of discharge, i.e., the amount of discharge energy over the time horizon T. We formulate the problem as

$$\min_{\{E_{\text{out}}, f_{\text{cmp}}, R\}} \sum_{t=1}^{T} E_{\text{out}}(t) \qquad (12)$$

$$s.t. \quad (3), (9), (11).$$

It requires a holistic optimization of discharge energy amount, CPU frequency, and transmission rate to adapt to system dynamics such as energy arrival rate and satellite mobility, subject to real-time and long-term constraints.

IV. ALGORITHM DESIGN

If the full information of energy arrival rate and wireless channel state information over the whole time horizon T is known, the problem is a convex optimization, which can be solved offline. In our scenario, the energy arrival rate can be calculated precisely. However, the wireless channel state information of the satellite-ground link is hard to predict in a long run by complex prediction methods. If we can predict the wireless channel state information in the current slot, the problem is a stochastic network optimization problem. We can adopt Lyapunov optimization theory to transform the long-term problem into real-time ones, which are convex optimization problems [21]. However, the channel state information in the current slot is unknown before making decisions. Hence, the scheduler should adapt its decisions based on the results of previous slots. A typical setting for such online optimization and learning is online convex optimization [22].

A. Optimal Pattern-Aware Benchmark

Although the channel state information is unavailable, we can exploit the pattern information when the satellite moves around the Earth. For example, the periodical satellite movement generates periodical-like pattern or trend information, i.e., the satellite-ground connection is periodical as satellites move around the Earth and satellites show up in the light and eclipse alternately. We design a pattern-aware benchmark to adapt to the different characteristics of the environment and energy harvesting across time slots. Given the time horizon T, we first create a partition which splits the pattern space Ω into $K = (L)^N$ hypercubes of identical size $\frac{1}{L} \times \frac{1}{L} \times \cdots \times \frac{1}{L}$. These hypercubes correspond to the different environment and energy harvesting characteristics. Then we define the time window $\mathcal{W}_k$ contains all the time slots whose pattern $\mathbf{c}(t)$ belongs to the k_{th} hypercube, i.e.,

$$\mathcal{W}_k = \{t : \mathbf{c}(t) \in \Omega_k\}, \forall k = \{1, \cdots, K\}. \qquad (13)$$

The partitioning of the time horizon captures a general pattern information.

Then, we introduce an optimal pattern-aware benchmark to measure the performance of pattern-aware algorithms. We denote the decision variable with $\mathbf{x} = (E_{\text{out}}(t), f_{\text{cmp}}(t), R(t))$ and the objective function with $f^t(\mathbf{x}(t))$ for simplicity. Given a sequence of channel state information $\{h(1), \cdots, h(T)\}$ over the time horizon T, the optimal pattern-aware benchmark finds K energy scheduling strategies $[\mathbf{x}^*(1), \cdots, \mathbf{x}^*(K)]$ for each time window $\mathcal{W}_k$ as

$$\mathbf{x}^*(k) = \arg\min_{\mathbf{x}} \sum_{t \in \mathcal{W}_k} f^t(\mathbf{x}), \qquad (14)$$

which is optimal regarding information only in the respective time window $\mathcal{W}_k$. The optimal pattern-aware benchmark is unavailable because it requires the full knowledge of the $h(t)$ in each time window $\mathcal{W}_k$. However, it captures energy harvesting and satellite ground connection patterns.

A performance metric to evaluate the learning performance of online algorithms is *regret*: the difference between the online algorithm and a benchmark. The accumulative regret of an algorithm with respect to the optimal pattern-aware benchmark over the whole T time horizon is defined as

$$Reg_A(T, K) = \sum_{t=1}^{T} f^t(\mathbf{x}(t)) - \sum_{k=1}^{K} \sum_{t \in \mathcal{W}_k} f^t(\mathbf{x}^*(k)). \quad (15)$$

Our regret definition is different from the common metrics of static regret and dynamic regret [23] because all three compare with different benchmarks. The static regret compares with an optimal static benchmark which finds the best fixed strategy $\mathbf{x}^*$ in hindsight over the T time horizon. The dynamic regret compares with an optimal dynamic benchmark which finds the best strategy $\mathbf{x}^*(t)$ for each time slot t. The pattern-aware regret is general as it reduces to the static regret when $K = 1$ and to the dynamic regret when $K = T$ and each time slot has unique pattern information. Note that the optimal dynamic benchmark has the best performance compared with both the optimal static benchmark and the optimal pattern-aware benchmark when the underlying system optima is inherently changing [24]. We can infer that the optimal pattern-aware benchmarks with larger K perform better in our scenario with the dynamics of energy harvesting and the uncertainty of the wireless environment. We aim to is find a sequence $\mathbf{x}(t)$ such that the regret $Reg_A(T, K)$ grows sub-linearly with respect to T.

B. Online Energy Scheduling with No Regret

In Lyapunov optimization, we introduce virtual queues to control the long-term constraint violations. Hence, we introduce a low-complexity virtual queue based algorithm that addresses our online convex optimization problem with long-term constraints [25]. To describe the algorithm more clear, we first introduce some notations. We denote the

gradient of the objective function as $\nabla f^t(\mathbf{x}(t))$. The convex set $\mathcal{X}_0$ is defined by instantaneous constraints, i.e.,

$$\mathcal{X}_0 = \{\mathbf{x} : 0 \le f_{\mathrm{cmp}}(t) \le f_{\mathrm{cmp}}^{\max}(t), \tag{16}$$
$$0 \le R(t) \le R^{\max} I_c(t), E_{\mathrm{out}} \ge 0\}, \forall t.$$

We denote the long-term constraint functions as $\mathbf{g}(\mathbf{x}(t)) = (g_1(x(t)), g_2(x(t)))$ where $g_1(x(t)) = Df_{\mathrm{sen}}(t) - \kappa f_{\mathrm{cmp}}(t)$, $g_2(x(t)) = \rho\kappa f_{\mathrm{cmp}}(t) - R(t)$. Let $\tilde{\mathbf{g}}(\mathbf{x}(t)) = \gamma\mathbf{g}(\mathbf{x}(t))$, where γ is a positive constant. We introduce a virtual queue vector $\mathbf{Q}^V(t) = (Q_1^V(t), Q_2^V(t))$ for the long-term constraint vector $\mathbf{g}(\mathbf{x}(t))$. The algorithm first chooses arbitrary feasible $\mathbf{x}(1) \in \mathcal{X}_0$ and then chooses $\mathbf{x}(t+1)$ that solves the following problem

$$\min_{\mathbf{x}\in\mathcal{X}_0} \{[\nabla f^t(\mathbf{x}(t))]^{\mathbf{T}}[\mathbf{x} - \mathbf{x}(t)] \tag{17}$$
$$+ [\mathbf{Q}^V(t) + \tilde{\mathbf{g}}(\mathbf{x}(t))]^{\mathbf{T}}\tilde{\mathbf{g}}(\mathbf{x}) + \alpha||\mathbf{x} - \mathbf{x}(t)||^2\},$$

where α is a positive constant and the virtual queues

$$Q_i^V(t) = \max\{-\tilde{g}_i(\mathbf{x}(t)), Q_i^V(t-1) + \tilde{g}_i(\mathbf{x}(t))\}. \tag{18}$$

Note that the virtual queues are different from the data queues and the energy queue in the system model. The value of $\mathbf{Q}^V(t)$ is a queue backlog of constraint violations. By introducing the virtual queue vector $\mathbf{Q}^V(t)$, the algorithm transforms the long-term constraints into queue length variation and solve the problem in (17) in each time slot t.

Linear constraints. We observe that our $\mathbf{g}(\mathbf{x}(t))$ is affine i.e, $\mathbf{g}(\mathbf{x}(t)) = \mathbf{Ax} - \mathbf{b}$, where $\mathbf{A} = [0, -\kappa, 0; 0, \rho\kappa, -1]$ and $\mathbf{b} = [-Df_s(t), 0]^{\mathbf{T}}$, then the update of $\mathbf{x}(t+1)$ can be solved by a projection onto a convex set as in Lemma 1.

Lemma 1. *[25] If $\mathbf{g}(\mathbf{x}(t))$ is affine, i.e, $\mathbf{g}(\mathbf{x}(t)) = \mathbf{Ax} - \mathbf{b}$ for some matrix $\mathbf{A}$ and vector $\mathbf{b}$, then we can update $\mathbf{x}(t+1)$ as*

$$\mathbf{x}(t+1) = \mathcal{P}_{x\in\mathcal{X}_0}\left[\mathbf{x} - \left(\mathbf{x}(t) - \frac{1}{2\alpha}\mathbf{d}(t)\right)\right], \tag{19}$$

where

$$\mathbf{d}(t) = \nabla f^t(\mathbf{x}(t)) + \sum_{i=1}^2 [Q_i^V(t) + \tilde{g}_i(\mathbf{x}(t))]\nabla\tilde{g}_i(\mathbf{x}(t)). \tag{20}$$

We design our pattern-aware online energy scheduling algorithm as in Algorithm 1. Our algorithm exploits the periodical energy harvesting and connection information then it learns to make decisions based on the historical information in each time window $\mathcal{W}_k$. We use t_k^1, t_k^τ to denote the first time slot and the τ_{th} time slot in each time window $\mathcal{W}_k$, respectively. At $t = t_k^1$ of $\mathcal{W}_k$, our algorithm chooses any feasible energy scheduling strategy because it has no previous information to rely on. At $t = t_k^\tau$ of $\mathcal{W}_k$, it updates the energy scheduling strategy by solving the projection in (21). After receiving the channel state information of the current slot, it updates virtual queues,

---

Algorithm 1 Pattern-Aware Online Energy Scheduling

Input:
  Constant parameter α, γ, β, time slot duration T_d, energy related parameters: $E_{\mathrm{total}}(t)$, $E_a(t)$, $E_{\max}$, sensing related parameters D, $P_{\mathrm{sen}}(t)$, $f_{\mathrm{sen}}(t)$, computing related parameters $\kappa, \rho, f_{\mathrm{cmp}}^{\max}$, communication related parameter $B, N_0, I_c(t), R^{\max}$.

Output:
  Energy scheduling strategies $\mathbf{x}(t), t = \{1, \cdots, T\}$.
1: Initialization: $Q_{\mathrm{com}}(0), Q_{\mathrm{cmp}}(0), E(0), Q_i^V(0)$ as 0.
2: **for** Each time slot $t = 1, \cdots, T$ **do**
3:    Identify time window $\mathcal{W}_k \ni t$
4:    **if** $t = t_k^1$ for $\mathcal{W}_k$ **then**
5:       Choose arbitrary $\mathbf{x}(t_k^1) \in \mathcal{X}_0$.
6:    **else**
7:       Choose $\mathbf{x}(t_k^\tau)$ by solving the projection

$$\mathcal{P}_{x\in\mathcal{X}_0}\left[\mathbf{x} - \left(\mathbf{x}(t_k^{\tau-1}) - \frac{1}{2\alpha}\mathbf{d}(t_k^{\tau-1})\right)\right]. \tag{21}$$

8:    **end if**
9:    Observe actual $h(t)$.
10:   Update the virtual queue vector $\mathbf{Q}^V$ via

$$Q_i^V(t) = \max\{-\tilde{g}_i(\mathbf{x}(t)), Q_i^V(t_k^{\tau-1}) + \tilde{g}_i(\mathbf{x}(t))\}. \tag{22}$$

11:   Update the energy queue $E(t)$, data queues $Q_{\mathrm{com}}(t)$ and $Q_{\mathrm{cmp}}(t)$ according to (2), (8), (10).
12: **end for**

---

the energy queue, and data queues via (25), (2), (8), and (10), respectively. Our algorithm is simple as it either selects a strategy arbitrarily or performs an easy gradient descent, incurring neglected energy consumption.

Performance Analysis. We analyze the objective function and constraints then make a mild assumption for the theoretical proof. The feasible region $\mathcal{X}_0$ is bounded in our problem, then there exists a constant G_1 such that $||\mathbf{x} - \mathbf{y}|| \le G_1$ and a constant G_2 such that $||\mathbf{g}(\mathbf{x})|| \le G_2$, for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}_0$. The objective function has bounded gradient on the feasible region $\mathcal{X}_0$, i.e., $||\nabla f^t(\mathbf{x}(t))|| = 1, \forall\mathbf{x}, t$. There exists a constant β such that $||\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{y})|| \le \beta||\mathbf{x} - \mathbf{y}||$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}_0$.

Assumption 1. *Assume that there exist ϵ and $\hat{\mathbf{x}} \in \mathcal{X}_0$ such that $\mathbf{g}_i(\hat{\mathbf{x}}) \le \epsilon$ for all $i = 1, 2$.*

The slater condition is mild for convex optimization. We characterize the regret and constraint violations for Algorithm 1 through the analysis of a drift plus penalty expression following the idea of [25] as in Lemma 2.

Lemma 2. *Consider online convex optimization with long-term constraints that satisfy Assumption 1. Let $\mathbf{x}^* \in \mathcal{X}_0$ be any fixed solution that satisfies $\mathbf{g}(\mathbf{x}^*)$, e.g., $\mathbf{x}^* =$*

$\arg\min_{\mathbf{x}\in\mathcal{X}_0}\sum_{t=1}^T f^t(\mathbf{x})$. *Let $\gamma > 0, \eta > 0$ be arbitrary.*

1. If $\alpha \geq \frac{1}{2}(\gamma^2\beta^2 + \eta)$ in Algorithm 1, then for all $T \geq 1$, Algorithm 1 in [25] has

$$\sum_{t=1}^T f(\mathbf{x}(t)) \leq \sum_{t=1}^T f(\mathbf{x}^*) + \alpha||((\mathbf{x}^* - \mathbf{x}(1))||^2 + \frac{T}{2\eta}. \quad (23)$$

2. For all $T \geq 1$, the constraint violations of Algorithm 1 in [25] is bounded as

$$\sum_{t=1}^T g_i(\mathbf{x}(t)) \leq 2G_2 + \frac{\alpha G_1^2 + G_1}{\gamma^2\epsilon} + \frac{2G_2^2}{\epsilon}, \forall i = 1, 2. \quad (24)$$

We extend theoretical results of [25] to fit the context of our pattern-aware scenario as follows.

Theorem 1. *If $\gamma = T^{1/4}, \eta = \sqrt{T}$ and $\alpha = \frac{1}{2}(\beta^2 + 1)\sqrt{T}$, for all $T \geq 1$, our algorithm has sublinear regret as*

$$\sum_{k=1}^K \sum_{t\in\mathcal{W}_k} f(\mathbf{x}(t)) \leq \sum_{k=1}^K \sum_{t\in\mathcal{W}_k} f(\mathbf{x}^*(k)) + O(\sqrt{T}). \quad (25)$$

For all $T \geq 1$, the constraint violations are bounded as

$$\sum_{k=1}^K \sum_{t\in\mathcal{W}_k} g_i(\mathbf{x}(t)) \leq 2KG_2+ \quad (26)$$

$$\frac{K(\frac{1}{2}(\beta^2 + 1)G_1^2 + G_1 + 2G_2^2)}{\epsilon} + \frac{KG_1}{\epsilon\sqrt{T}}.$$

Proof: We adopt Lemma 2 for each time window $\mathcal{W}_k$, separately. If $\alpha \geq \frac{1}{2}(\gamma^2\beta^2 + \eta)$ in Algorithm 1, then for all $|\mathcal{W}_k| \geq 1$, we have

$$\sum_{t\in\mathcal{W}_k} f(\mathbf{x}(t)) \leq \sum_{t\in\mathcal{W}_k} f(\mathbf{x}^*) + \alpha||((\mathbf{x}^*(k) - \mathbf{x}(t_k^1))||^2 + \frac{|\mathcal{W}_k|}{2\eta}. \quad (27)$$

For the whole time horizon T, we have

$$\sum_{k=1}^K \sum_{t\in\mathcal{W}_k} f(\mathbf{x}(t)) \leq \sum_{k=1}^K \sum_{t\in\mathcal{W}_k} f(\mathbf{x}^*) \quad (28)$$

$$+ \sum_{k=1}^K \alpha||((\mathbf{x}^*(k) - \mathbf{x}(t_k^1))||^2 + \sum_{k=1}^K \frac{|\mathcal{W}_k|}{2\eta}.$$

If $\gamma = T^{1/4}, \eta = \sqrt{T}$ and $\alpha = \frac{1}{2}(\beta^2 + 1)\sqrt{T}$, we have (25).

The constraint violation bound is irrelevant to the time horizon. Thus, for each time window $\mathcal{W}_k$, we have

$$\sum_{t\in\mathcal{W}_k} g_i(\mathbf{x}(t)) \leq 2G_2 + \frac{\alpha G_1^2 + G_1}{\gamma^2\epsilon} + \frac{2G_2^2}{\epsilon}, \forall i = 1, 2. \quad (29)$$

For the whole time horizon T, we have

$$\sum_{k=1}^K \sum_{t\in\mathcal{W}_k} g_i(\mathbf{x}(t)) \leq K(2G_2 + \frac{\alpha G_1^2 + G_1}{\gamma^2\epsilon} + \frac{2G_2^2}{\epsilon}), \forall i = 1, 2. \quad (30)$$

If $\gamma = T^{1/4}, \eta = \sqrt{T}$ and $\alpha = \frac{1}{2}(\beta^2 + 1)\sqrt{T}$, we have (26). ∎

Theorem 1 implies that if we choose $\gamma = T^{1/4}$ and $\alpha = \frac{1}{2}(\beta^2 + 1)\sqrt{T}$ in Algorithm 1 then we can achieve $O(\sqrt{T})$ regret and $O(\frac{1}{\sqrt{T}})$ constraint violation. Note that the theoretical results are based on the parameters G_1, G_2, ϵ while it only requires β to implement Algorithm 1, which is known since the constraint functions $\mathbf{g}(\mathbf{x}(t))$ do not change.

V. SIMULATION RESULTS

A. Simulation Setting

We simulate a scenario where an LEO satellite orbits the Earth and captures Earth images according to the mission demand. We select a 6am Sun-synchronous orbit and the orbit altitude is set to 550km. We simulate two ground stations, one in the light and the other in the eclipse. The satellite harvests energy as it orbits the Earth. We calculate the energy harvesting rate $r_e(t)$ (in J/min) by $r_e(t) = E_s^{\max}\cos(\theta_t)$, where E_d is the maximal harvest power and θ_t is the angle between the Sun and the normal vector of the solar panel mentioned in Section III-A. We set $E_d = 30$ J/min and θ_t varies between $[0, \pi/2]$. We set the main simulation parameters according to [3] and [26]. The energy consumption of other subsystems $E_a = 0$ for simplicity. Battery capacity $E_{\max} = 10800$ J, and the battery is full of charge at the beginning of the mission. The data size of each frame is set to 60 Mbits. The sensing power is set to 2 J/min. Sensing frame rate $f_{\text{sen}}(t)$ is randomly generated in $[0, 4]$. The data to computation ratio κ is set to 0.1 bit/cycle. The effective data proportion $\rho = 0.25$. Maximal CPU frequency $f_p^{\max} = 4$ GHz. Bandwidth $B = 80$ MHz. Parameter h/N_0 is randomly generated in $[15, 20]$. Maximal transmission bitrate $R_{\max}$ is set to 500 Mbit/s. Algorithm Parameters are set to $\beta = 14$ and $\gamma = 1440^{1/4}$. In the simulation, we schedule the energy every one minute over 24 hours. Hence, the whole time horizon $T = 1440$. We divide the whole time horizon into $K = 4$ time windows.

Baseline Approaches. We compare our algorithm with three baselines as follows.

1) *Optimal dynamic benchmark.* The scheduler knows channel state information $\{h(1), \cdots, h(T)\}$ over the time horizon T and finds optimal energy scheduling strategies $[\mathbf{x}^*(1), \cdots, \mathbf{x}^*(T)]$. This benchmark provides a performance upper bound for the evaluated algorithms. We achieve the optimal dynamic benchmark directly by the convex optimization tool, CVX [27].

2) *Optimal pattern-aware benchmark.* The scheduler knows channel state information $\{h(1), \cdots, h(T)\}$ over the time horizon T and finds K energy scheduling strategies $[\mathbf{x}^*(1), \cdots, \mathbf{x}^*(K)]$ for each time window $\mathcal{W}_k$ as defined in (14), which is optimal only regarding energy arrival rate in each time window $\mathcal{W}_k$.

3) *Greedy algorithm.* The scheduler knows channel state information of the current time slot $h(t)$ and allocate min-
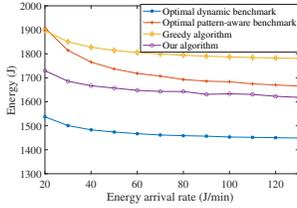
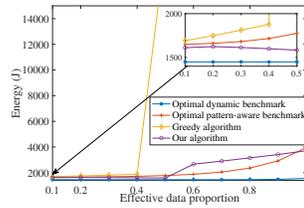Figure 3: Impact of harvested energy density.

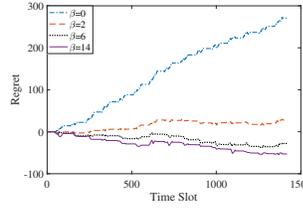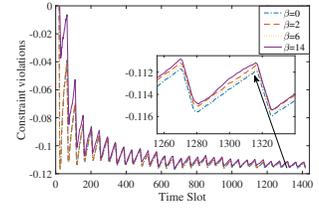Figure 4: Impact of effective data proportion.

Figure 5: Algorithm regret.

Figure 6: Constraint violations.

imal energy to sensing, computing, and communication to satisfy the constraints in (9) and (11) in each time slot t.

B. Simulation Results

In Figure 3, the depth of discharge of our algorithm is less than both the optimal pattern-aware benchmark and the greedy algorithm over the time horizon when the energy harvesting capability varies in the range of [20, 130] J. The optimal dynamic benchmark achieves the minimal depth of discharge. With the full information, it allocates the least energy only for sensing and no energy for computation and communication in the eclipse. And it allocates as much energy (directly from the solar panels) as possible to finish the accumulative workloads in the light. As expected, the optimal pattern-aware benchmark performs worse than the optimal dynamic benchmark because it only optimizes for each time window without global scheduling and gets 4 fixed solutions for each orbit period, which cannot track the system dynamics. Our algorithm can adapt to the system dynamics by learning from historical information. In our algorithm, the depth of discharge decreases slowly with the harvested energy density. It implies that our algorithm is robust to the variation of maximal harvesting energy and can work well with limited harvesting capability. The optimal dynamic benchmark is also robust to the variation of maximal harvesting energy.

In Figure 4, we aim to evaluate how the effective data proportion impacts our algorithm. We set the parameter ρ in [0.1, 1] and record the sum of the amount of discharge energy over the whole time horizon T. We can observe that our algorithm performs better than both the optimal pattern-aware benchmark and the greedy algorithm when $\rho \leq 0.5$ and its amount of discharge is larger than the optimal pattern-aware benchmark when $\rho > 0.5$. The effective data proportion influences the data amount to be transmitted to the ground. The transmission rate will increase with ρ, and the energy increases exponentially with transmission rate, which results in deeper battery discharge in the downlink session of the eclipse. The greedy algorithm performance deteriorates most sharply with ρ. The two optimal benchmarks are robust to the variation of ρ because they can optimize in larger timescale and keep the workload to be processed or transmitted in the light to reduce the depth of discharge

in the eclipse.

In Figure 5 and Figure 6, our algorithm has no regret against the optimal pattern-aware benchmark and the constraint violation asymptotically approaches zero as proved in Theorem 1. We plot the accumulative regret and constraint violations with time under different values of parameter β. Despite the energy harvesting dynamics and the lack of channel gain information, our algorithm can achieve a near-optimal solution. When $\beta = 0$ (not a validate value), the accumulative regret grows linearly with the time horizon because it leads to a large step size $\frac{1}{2\alpha}$ in (19) and the learning performance is poor. The regret gets smaller with the increase of β. Increasing β will result in smaller step size $\frac{1}{2\alpha}$, the algorithm will fail to adapt to the system dynamics, thus incurring many constraint violations as in Figure 6. We find that our algorithm can achieve the minimal depth of discharge without constraint violations when $\beta = 14$.

VI. CONCLUSION

In this paper, we extend the batteries' life of LEO satellites by minimizing the depth of discharge for Earth observation missions. We propose a pattern-aware online energy scheduling algorithm that decides the depth of discharge, CPU frequency, and transmission rate under lack of wireless channel state information. Our algorithm has a theoretical guarantee of learning loss and reduces the depth of discharge significantly. Even our solution cannot fundamentally extend batteries' life like inventing new battery materials, it can work well together with those advanced battery technologies. In future work, we are interested to optimize the depth of discharge for large-scale constellations with satellites collaboration.

VII. ACKNOWLEDGEMENT

REFERENCES

[1] W. J. Larson and J. R. Wertz, "Space mission analysis and design," Torrance, CA (United States); Microcosm, Inc., Tech. Rep., 1992.

[2] B. Denby and B. Lucia, "Orbital edge computing: Nanosatellite constellations as a new class of computer system," in *proc. of International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020, p. 939–954.

[3] B. Yost, "State-of-the-art small spacecraft technology," 2020. [Online]. Available: https://www.nasa.gov/sites/default/files/atoms/files/soa2020_final3.pdf

[4] F. Davoli, C. Kourogiorgas, M. Marchese, A. Panagopoulos, and F. Patrone, "Small satellites and cubesats: Survey of structures, architectures, and protocols," *International Journal of Satellite Communications and Networking*, vol. 37, pp. 343–359, 2019.

[5] C. Wang, N. Lu, S. Wang, Y. Cheng, and B. Jiang, "Dynamic long short-term memory neural-network-based indirect remaining-useful-life prognosis for satellite lithium-ion battery," *Applied Sciences*, vol. 8, no. 11, 2018.

[6] A. Ortiz, T. Weber, and A. Klein, "Multi-agent reinforcement learning for energy harvesting two-hop communications with a partially observable system state," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 1, pp. 442–456, 2021.

[7] F. Fraternali, B. Balaji, Y. Agarwal, and R. K. Gupta, "Aces: Automatic configuration of energy harvesting sensors with reinforcement learning," *ACM Transactions on Sensor Networks*, vol. 16, no. 4, pp. 1–31, 2020.

[8] T. Huang, W. Lin, Y. Li, X. Wang, Q. Wu, R. Li, C.-H. Hsu, and A. Y. Zomaya, "Adaptive processor frequency adjustment for mobile edge computing with intermittent energy supply," *arXiv preprint arXiv:2102.05449*, 2021.

[9] M. Hatami, M. Jahandideh, M. Leinonen, and M. Codreanu, "Age-aware status update control for energy harvesting IoT sensors via reinforcement learning," in *proc. of IEEE Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, 2020, pp. 1–6.

[10] H. Tsuchida, Y. Kawamoto, N. Kato, K. Kaneko, S. Tani, S. Uchida, and H. Aruga, "Efficient power control for satellite-borne batteries using Q-learning in low-earth-orbit satellite constellations," *IEEE Wireless Communications Letters*, vol. 9, no. 6, pp. 809–812, 2020.

[11] D. Bhattacherjee and A. Singla, "Network topology design at 27,000 km/hour," in *proc. of International Conference on Emerging Networking Experiments And Technologies*, 2019, pp. 341–354.

[12] L. Boero, R. Bruschi, F. Davoli, M. Marchese, and F. Patrone, "Satellite networking integration in the 5G ecosystem: Research trends and open challenges," *IEEE Network*, vol. 32, no. 5, pp. 9–15, 2018.

[13] G. Giambene, S. Kota, and P. Pillai, "Satellite-5G integration: A network perspective," *IEEE Network*, vol. 32, no. 5, pp. 25–31, 2018.

[14] Y. Shi, Y. Cao, J. Liu, and N. Kato, "A cross-domain SDN architecture for multi-layered space-terrestrial integrated networks," *IEEE Network*, vol. 33, no. 1, pp. 29–35, 2019.

[15] F. Tang, "Dynamically adaptive cooperation transmission among satellite-ground integrated networks," in *proc. of IEEE Conference on Computer Communications*, 2020, pp. 1559–1568.

[16] L. Chen, F. Tang, Z. Li , L. T. Yang, J. Yu, and B. Yao, "Time-varying resource graph based resource model for space-terrestrial integrated networks," in *proc. of IEEE Conference on Computer Communications*, 2021.

[17] L. Chen, F. Tang, and X. Li, "Mobility- and load-adaptive controller placement and assignment in LEO satellite networks," in *proc. of IEEE Conference on Computer Communications*, 2021.

[18] D. Bhattacherjee, S. Kassing, M. Licciardello, and A. Singla, "In-orbit computing: An outlandish thought experiment?" in *proc. of ACM Workshop on Hot Topics in Networks*, 2020, p. 197–204.

[19] T. M. Lim, A. M. Cramer, J. E. Lumpp, and S. A. Rawashdeh, "A modular electrical power system architecture for small spacecraft," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1832–1849, 2018.

[20] E. Le Sueur and G. Heiser, "Dynamic voltage and frequency scaling: The laws of diminishing returns," in *proc. of the International Conference on Power Aware Computing and Systems*, 2010, pp. 1–8.

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

[22] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[23] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *proc. of the International Conference on Machine Learning*, 2003, pp. 928–936.

[24] X. Cao, J. Zhang, and H. V. Poor, "A virtual-queue-based algorithm for constrained online convex optimization with applications to data center resource allocation," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 4, pp. 703–716, 2018.

[25] H. Yu and M. J. Neely, "A low complexity algorithm with $O(\sqrt{T})$ regret and $O(1)$ constraint violations for online convex optimization with long term constraints," *arXiv preprint arXiv:1604.02218*, 2016.

[26] T. Zhu, J. Li, Z. Cai, Y. Li, and H. Gao, "Computation scheduling for wireless powered mobile edge computing networks," in *proc. of IEEE Conference on Computer Communications*, 2020, pp. 596–605.

[27] CVX, 2020. [Online]. Available: http://cvxr.com/cvx/