INFOCOM 2024 1570937358

SECO: Multi-Satellite Edge Computing Enabled Wide-Area and Real-Time Earth Observation Missions

Zhiwei Zhai, Liekang Zeng, Tao Ouyang, Shuai Yu, Qianyi Huang and Xu Chen
School of Computer Science and Engineering, Sun Yat-sen University, China
Email: {zhaizhw3, zenglk3, ouyt9}@mail2.sysu.edu.cn, {yushuai, huangqy89, chenxu35}@mail.sysu.edu.cn

***Abstract*—Rapid advances in low Earth orbit (LEO) satellite technology and satellite edge computing (SEC) have facilitated a key role for LEO satellites in enhanced Earth observation missions (EOM). These missions (*e.g.,* remote object detection) typically require multi-satellite cooperative observations of a large region of interest (RoI) area, as well as the observation image routing and computation processing, enabling accurate and real-time responsiveness. However, optimizing the resources of LEO satellite networks is nontrivial in the presence of its dynamic and heterogeneous properties. To this end, we propose SECO, a SEC-enabled framework that jointly optimizes multi-satellite observation scheduling, routing and computation node selection for enhanced EOM. Specifically, in the observation phase, we leverage the orbital motion and the rotatable onboard cameras of satellites, and propose a distributed game-based scheduling strategy to minimize the overall size of captured images while ensuring full (observation) coverage. In the sequent routing and computation phase, we first adopt image splitting technology to achieve parallel transmission and computation. Then, we propose an efficient iterative algorithm to jointly optimize image splitting, routing and computation node selection for each captured image. On this basis, we propose a theoretically guaranteed system-wide greedy-based strategy to reduce the total time cost (*i.e.,* transmission, computation and queuing delay) over simultaneous processing for multiple images. Extensive experiments based on real-world datasets demonstrate that SECO can achieve up to a 60.7% reduction in overall time cost compared to baselines.**

I. Introduction

Nowadays, the rapid advancements in low Earth orbit (LEO) satellite technology have significantly enhanced the capacity of LEO satellites in space, which are equipped with high-resolution cameras, communication modules and sophisticated processors [1]. This technological configuration enables LEO satellites to perform timely near-data processing at the edge of LEO satellite networks, thereby giving birth to the concept of satellite edge computing (SEC) [2]. Furthermore, LEO satellites with inter-satellite links (ISLs) [3] can connect with their adjacent satellites, thereby enhancing the utilization of inter-satellite resources to boost service performance. Serving as the centerpiece of SEC, LEO satellites have been extensively utilized in various Earth observation missions (EOM) [4]–[6]. These missions encompass a wide range of critical applications, including carbon emission monitoring [7] and emergency response to diverse natural weather events [8].

The EOM typically involves: i) observation phase, *i.e.,* making observations of a region of interest (RoI), and ii) processing phase, *i.e.,* computationally processing the captured RoI images and routing the results to the designated ground station. In terms of the observation phase, major work utilizes a single satellite to handle the observation of a RoI [6], [9], [10]. Nonetheless, in scenarios like earthquake warnings, observing a large RoI area is crucial for a more accurate and satisfactory response [5], [7], [11]. Therefore, as shown in Fig. 1, the multi-satellite cooperative observation becomes indispensable due to the limited observation range of a single satellite. Regarding the processing phase, EOM is often built upon advanced functionalities such as deep neural network (DNN) inference [12] or multi-stage image compression (MIC) [13], necessitating high efficiency to ensure real-time response for time-sensitive missions [14]. Given these attributes, we term this novel EOM paradigm as wide-area and real-time EOM.

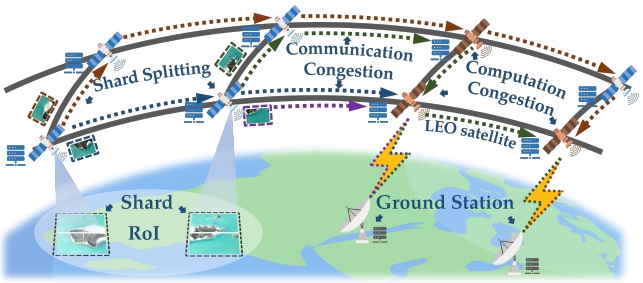


Fig. 1: Multi-satellite cooperative observation, routing and computation process of SECO.

To achieve wide-area and real-time EOM, several challenges must be addressed. First, as depicted in Fig. 2, cross-observation areas by ascending and descending satellites divide the RoI into shards. Notably, for the given boundary shard, the satellite capture properties lead to divergent capture sizes for ascending and descending satellites [5], [15]. For instance, for boundary shard A in Fig. 2, an ascending satellite only needs to capture the shadow area during its motion direction, while a descending satellite needs to capture the entire segment. Moreover, the observation regions for each satellite are time-varying, due to its dynamic property, *i.e.,* orbital motion and the rotatable onboard camera. In this way, efficient sustained cooperation for multiple satellites in the observation phase is nontrivial. Second, the computation of each shard in SEC involves tasks such as DNN inference or MIC that are computationally intensive. To fully leverage the resources within LEO satellite networks, these tasks can be divided into subtasks and assigned to satellite nodes [16]. However, the inherent dependencies within the computation workflow pose a challenge [17]. Additionally, the heterogeneous nature of satellite hardware resources further compli-

This work was supported in part by the National Science Foundation of China (No. U20A20159, No. 62272122); Guangdong Basic and Applied Basic Research Foundation (No. 2021B151520008, No. 2022B1515120002); Guangzhou Basic and Applied Basic Research Program (No. 2024A04J6367); the Guangdong Science and Technology Planning Project (No. 2018B030322004). Corresponding Author: Xu Chen.

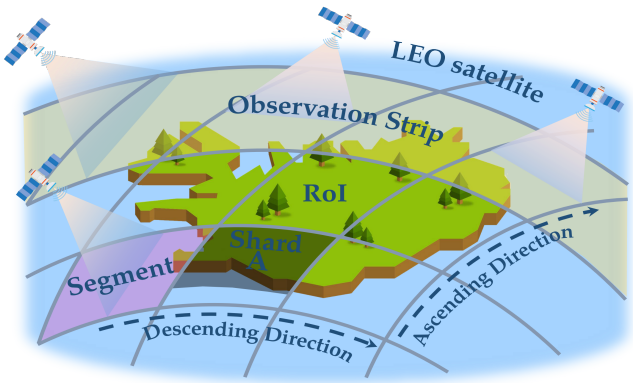


Fig. 2: Multi-satellite cooperative observation for a large RoI.

cates the embedding of subtasks within the LEO satellite networks. Finally, as shown in Fig. 1, since the satellite is in motion to capture shards, different satellites might capture shards simultaneously, potentially causing congestion (*i.e.,* queuing delays) on computation nodes and transmission links [18] when they are transmitting and computing. Hence, the complexity of optimizing the processing phase is significantly escalated while considering these queuing delays.

To address the above challenges, we propose SECO, a SEC-enabled framework for jointly optimizing multi-satellite observation scheduling, routing and computation node selection to facilitate wide-area and real-time EOM. In the observation phase, we strategically employ game theory to enable distributed inter-satellite interactions, thus achieving an efficient multi-satellite observation scheduling for a large RoI area. This approach considers the capture and dynamic properties of satellites, enabling the reduction of the overall captured image size while ensuring full RoI coverage. In the processing phase, to tackle the challenge presented by the dependent property in computation tasks (*i.e.,* DNN inference or MIC) and clarify the mapping of processing for each shard in LEO satellite networks, we model the LEO satellite networks as a layered graph to represent the routing and computation process for embedding computation tasks. We theoretically prove that this model scheme yields an integer optimal solution under linear relaxation, thus enabling efficient optimization of the processing for each shard while considering the queuing delay. Additionally, we augment processing efficiency via shard splitting, which allows for parallel transmission and computation. Supporting this, we design an iterative algorithm for jointly shard splitting, routing and computation node selection for each shard. Finally, due to the satellites capturing shards in motion, when multiple shards are captured simultaneously, a theoretically guaranteed system-wide greedy-based strategy is proposed to optimize the time cost at a system level.

In summary, the main contributions are as follows:

- We propose SECO, a comprehensive framework that optimizes observation scheduling, routing and computation node selection for wide-area and real-time EOM, which has great potential in boosting the real-time responses (*i.e.,* to enable efficient DNN inference or MIC, *etc*).
- We promote a distributed observation scheduling strategy by resorting to the potential game theory as the algorithm design tool based on the orbital motion and rotatable camera of each satellite. To our knowledge, it is the first strategy considering multi-satellite observations for the large RoI area and fine-grained boundary-side scheduling.
- In SECO, we employ shard splitting to facilitate parallel transmission and computation, thereby reducing time costs. Then an efficient iterative algorithm is proposed to jointly determine the optimal shard splitting, routing and computation node selection for each shard. Further, a system-wide greedy-based strategy is designed to optimize the time cost at the system level, providing performance guarantees through approximation ratio analysis.
- We conduct extensive experiments based on real-world datasets, showing that SECO can achieve up to a 60.7% time cost reduction compared to other baselines.

II. System Model

As shown in Fig. 1, consider a multi-satellite cooperative system for joint observation, routing and computation for a large RoI area consisting of a set $\mathcal{N} = \{1, 2, \cdots, N\}$ of LEO satellites and a set $\mathcal{B} = \{1, 2, \cdots, B\}$ of ground stations. According to the dynamic and predictable properties of LEO satellite movement orbits, a given time horizon $\mathcal{T} = [t_0, t_T)$ can be divided into T consecutive time slots $\{\tau_1, \cdots, \tau_l, \cdots, \tau_T\}$, where $\tau_l = [t_{l-1}, t_l)$ and the network topology (*i.e.,* the connectivity among the satellites) is fixed during timeslot τ_l, but can be varying in different time slots [10]. Let $\mathcal{G}_u^{\tau_l} = \{\mathcal{V}_u^{\tau_l}, \mathcal{E}_u^{\tau_l}\}$ denote the topology of LEO satellite networks at time slot τ_l, where $\mathcal{V}_u^{\tau_l} = \mathcal{N} \cup \mathcal{B}$ and $\mathcal{E}_u^{\tau_l}$ is the set of communication links in LEO satellite networks.

A. Observation Scheduling

For RoI observation, as shown in Fig. 2, a satellite on each orbit can observe a stripe on the Earth's surface when its camera is pointing vertically downward. Particularly, when the camera angle is rotated within the allowed range, a different stripe can be directed. A stripe is composed of segments, corresponding to its intersections with the stripes in the opposite direction, and each portion of a RoI included in a segment is named **shard**. Denote that a set $\mathcal{N}'$ ($\mathcal{N}' \subset \mathcal{N}, |\mathcal{N}'| = V$) of satellites approaching the observation RoI are used as observation satellites. Moreover, consider a RoI consisting of M shards, where the shard set $\mathcal{M} = \{1, 2, \cdots, M\}$ need to be assigned to the satellites in $\mathcal{N}'$ for observing. Then the captured shards would be split into subshards and subsequently routed through satellites to reach the destination ground station, with the computation occurring during this routing process.

In existing implementations, observations of a large RoI area are made by satellites following their own trajectory over a complete observation stripe at the current camera angle [5]. However, when the ascending and descending satellites are observed together, this leads to a large redundancy of observations, resulting in unnecessary duplication of shard transmissions and computations. Since the satellite camera angle can be rotated [15], multiple satellite observations of shards in an RoI can be scheduled to optimize coverage and to reduce the duplication of observations over the same shard.

For each satellite $n \in \mathcal{N}'$, they observe the shard of RoI by rolling their cameras during their forward motion along the fixed orbit. Therefore, for two shards m and z, they can be sequentially observed by a satellite $n \in \mathcal{N}'$ only if they satisfy the following constraint,

$$\mathbf{C1}: \frac{|\theta_m^n - \sigma_z^n|}{\triangle t_{(m,z)}^n} \leq r_n^{\max}, \forall n \in \mathcal{N}', \tag{1}$$

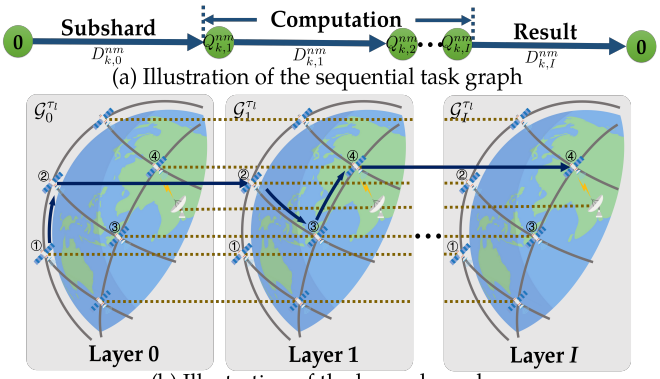
where θ_m^n represents the camera inclination of satellite n after observing shard m, σ_z^n denotes the camera inclination required for satellite n to start observing shard z, $r_n^{\max}$ refers to the maximum rolling velocity and $\triangle t_{(m,z)}^n$ is the time required for the satellite to orbit from shard m to z. To optimize the observable shard allocation, we define a directed acyclic graph (DAG) $\mathcal{D}_n = \{\mathcal{V}_n, \mathcal{E}_n\}$ for each satellite n that represents a feasible observation sequence, factoring in satellite orbital and camera rotation properties [9], [19]. $\mathcal{V}_n$ denotes the observable shard set of satellite n, with two additional virtual vertices a_n and b_n denoting the common source and destination, respectively. $\mathcal{E}_n$ denotes the edge set consisting of the observable edges connecting pairs of nodes in $\mathcal{V}_n$. Specifically, there exists an edge from a_n to each $m \in \mathcal{V}_n$ and an edge from $m \in \mathcal{V}_n$ to b_n. Moreover, based on constraint **C1**, the DAG $\mathcal{D}_n$ can be established for each satellite $n \in \mathcal{N}'$, the weight of each edge $(m,z) \in \mathcal{E}_n$ is denoted as $w_{(m,z)} \in \{0,1\}$, where $w_{(m,z)} = 1$ represent that the observation sequence from m to z is feasible, vice versa.

We denote by $x_{\tau_l}^{nm}$ the time slot τ_l at which satellite n observed shard m, where $x_{\tau_l}^{nm} \in \{0,1\}$. For a more fine-grained consideration of shard observations, as the shard A in Fig. 2, according to the movement orbits of the satellite, only the image size of the shadow area needs to be taken if the ascending satellite is selected for observation, and the image size of the complete segment needs to be taken if the descending satellite is used for observation. Therefore, by properly selecting satellites to capture the shards, the size of the image can be effectively reduced, thus reducing the transmission and computation overhead. To this end, we denote the size of the shard m as H_m and the image size for the satellite n to capture the shard m as W_{nm}. Then we denote the content percentage for the satellite n to capture the shard m as $R_{nm} = H_m/W_{nm}$. Higher content percentage enables encapsulation of the entire shard within a smaller image size.

B. Shard Splitting, Routing and Computation

Depending on the position of the shard $m \in \mathcal{M}$ in the RoI and the assigned observation satellite $n \in \mathcal{N}'$, shard m will start to transmit and compute (*e.g.,* DNN inference or MIC, *etc*) in LEO satellite networks at the corresponding time slot τ_l. The final result will arrive at the designated ground station. To boost the processing efficiency of each shard, we split the observed shard $m \in \mathcal{M}$ into K^{nm} subshards for more efficient transmission and computation over multiple paths. Let π_k^{nm} denote the k-th subshard of shard m captured by satellite n with size v_k^{nm}, where $k \in \{1, \cdots, K^{nm}\}$ and $\sum_{k=1}^{K^{nm}} v_k^{nm} = W_{nm}$.

As illustrated in Fig. 3(a), we model the computation of each subshard π_k^{nm} by a sequential task graph, which can represent such as the DNN inference or MIC process [5], [20]. The task



(a) Illustration of the sequential task graph

(b) Illustration of the layered graph

Fig. 3: Routing and computation process for the subshard π_k^{nm}.

graph is comprised of I layers, which correspond to the DNN model layers or image compression stages. Each vertex in the task graph represents a subtask, *i.e.,* one layer of DNN or one stage of MIC. Moreover, to reflect the fact that the input data originated from the observation satellite and the result is routed to the destination ground station, two virtual subtasks 0 and $I+1$ are introduced to represent the entry and exit subtasks, respectively. Each edge in the task graph denotes that the input data of subtask i is the output of the preceding subtask $i-1$. Let ρ_i denote the scaling factor, *i.e.,* the scaling rate of the data size after computing the subtask i. Then the input data size of subtask i for the subshard π_k^{nm} can be represented as $D_{k,i}^{nm} = \prod_{i=0}^{i} \rho_i v_k^{nm}$, where $i \in \{0, \cdots, I\}$. Besides, we denote the computation workload of subtask i for the subshard π_k^{nm} as $Q_{k,i}^{nm} = C_i \prod_{i=0}^{i-1} \rho_i v_k^{nm}$, where $i \in \{1, \cdots, I\}$ and C_i represents the CPU cycles required to compute 1-bit data of layer i. Note that C_i is dependent on the computation complexity of layer i [5]. Further, since subtasks 0 and $I+1$ represent the input and output of the DNN/MIC with I layers/stages, it can be obtained that $Q_{k,0}^{nm} = Q_{k,I+1}^{nm} = 0$ and $\rho_0 = 1$.

To better represent the processing phase of each subshard π_k^{nm} and the queuing delay in LEO satellite networks, we construct a layered graph $\mathcal{G}^{\tau_l} = (\mathcal{V}^{\tau_l}, \mathcal{E}^{\tau_l})$ to denote $I+1$ replications of the network topology $\mathcal{G}_u^{\tau_l} = \{\mathcal{V}_u^{\tau_l}, \mathcal{E}_u^{\tau_l}\}$ at time slot τ_l inspired by [21]. We will show in Subsection III-B that this modeling approach enables efficient routing and computation node selection while considering the queuing delay. As illustrated in Fig. 3(b), in layered graph $\mathcal{G}^{\tau_l}$, there are $I+1$ copies of $\mathcal{G}_u^{\tau_l}$, named $\mathcal{G}_0^{\tau_l}, \cdots, \mathcal{G}_I^{\tau_l}$ with $\mathcal{G}_i^{\tau_l} = (\mathcal{V}_i^{\tau_l}, \mathcal{E}_i^{\tau_l}), \forall i \in \{0, \cdots, I\}$. Let $e_i \in \mathcal{G}_i^{\tau_l}$ denote the replicated node of $e \in \mathcal{G}_u^{\tau_l}$, and hence, $(e_i, f_i) \in \mathcal{G}_i^{\tau_l}$ denote the replicated link of $(e,f) \in \mathcal{G}_u^{\tau_l}$. Further, there is an edge from node e_{i-1} to e_i for all $e \in \mathcal{G}^{\tau_l}$ and $i \in \{1, \cdots, I\}$. These edges are denoted by $\mathcal{E}_c^{\tau_l}$. Accordingly, it can be obtained that $\mathcal{V}^{\tau_l} = \mathcal{V}_0^{\tau_l} \cup \cdots \cup \mathcal{V}_I^{\tau_l}$ and $\mathcal{E}^{\tau_l} = \mathcal{E}_0^{\tau_l} \cup \cdots \cup \mathcal{E}_I^{\tau_l} \cup \mathcal{E}_c^{\tau_l}$.

We now discuss the routing and the computation node selection for each subshard π_k^{nm} in the layered graph $\mathcal{G}^{\tau_l}$. Depending on the observation scheduling strategy and the destination, we define the source and destination nodes for subshard π_k^{nm} are $s_k^{nm} \in \mathcal{V}^{\tau_l}$ and $t_k^{nm} \in \mathcal{V}^{\tau_l}$, respectively. Then we need to find a path from the $s_{k,0}^{nm}$ of layer 0 to the $t_{k,I}^{nm}$ of layer I in the layered graph $\mathcal{G}^{\tau_l}$, where the cross-layer edge

represents the computation of the corresponding layer in the task graph. Specifically, if the routing traverses edge (e_{i-1}, e_i), then layer i in the task graph is computed at node e. Moreover, if the routing traverses the edge (e_i, f_i), then the output data of layer i in the task graph is transmitted from node e to node f. We denote the transmission capacity of edge $(e, f) \in \mathcal{E}^{\tau_l}$ as ω_{ef} and the computation capacity of satellite e as μ_e.

To provide a clearer representation of the routing and computation process of subshard π_k^{nm} in the layered graph $\mathcal{G}^{\tau_l}$, we use Fig. 3 as an illustrative example. Note that Fig. 3(a) corresponds to Fig. 3(b), where the edges in Fig. 3(a) represent the layer i of the task transmission in the graph $\mathcal{G}_i^{\tau_l}$, and the vertexes in Fig. 3(a) represent the layer i of the task computation on the connecting edges of the graphs $\mathcal{G}_{i-1}^{\tau_l}$ and $\mathcal{G}_i^{\tau_l}$. In Fig. 3(b), shard m is captured by satellite ①. After the shard splitting is completed, subshard π_k^{nm} will be routed and computed in layered graph $\mathcal{G}^{\tau_l}$. Specifically, the subshard π_k^{nm} is first transmitted from satellite ① to satellite ②. Then the computation for the 1-th layer of the task is performed on satellite ②. Subsequently, the computation results are transmitted to satellite ④ via satellite ③. Finally, satellite ④ completes the computation of the remaining layers and transmits the computation results to the ground station.

Referring to the priority setting in [22], we set the priority with the subshard as the smallest unit and assume preemption scheduling on the transmission links and computation nodes. Specifically, priority is assigned to every subshard and uniformly applied to all the layers in $\mathcal{G}^{\tau_l}$ belonging to the same subshard. For instance, if subshards π_1^{nm} and π_2^{nm} utilize the same computation node or transmission link, with π_1^{nm} having higher priority, then π_2^{nm} must wait until π_1^{nm} completes its transmission or computation before it can proceed. Therefore, there is a queue for every transmission edge, and L_{ef} is the queue length at edge (e, f) representing the data size waiting to be transmitted. Further, L_e is the queue length of computation tasks waiting to be computed at node e. The above capacities and queue lengths in LEO satellite networks are reflected into the layered graph $\mathcal{G}^{\tau_l}$ as follows: 1) $L_{e_i,f_i} = L_{ef}$, $\omega_{e_i,f_i} = \omega_{ef}$, $\forall (e, f) \in \mathcal{E}_u^{\tau_l}$, $i \in \{0, \cdots, I\}$; 2) $L_{e_{i-1}e_i} = L_e$, $\mu_{e_{i-1}e_i} = \mu_e$, $\forall e \in \mathcal{V}_u^{\tau_l}$, $i \in \{1, \cdots, I\}$.

Let $x_{k,ef}^{nm}$ denote the routing or computation node selection for each subtask of subshard π_k^{nm}, with a value of 1 if it traverses the edge $(e, f) \in \mathcal{E}^{\tau_l}$ in $\mathcal{G}^{\tau_l}$, and 0 otherwise. Define $x_{k,e}^{nm}$ as the indicator to represent whether the node $e \in \mathcal{V}_u^{\tau_l}$ is selected to perform the computation for the subshard π_k^{nm}. Then the total time cost (*i.e.,* transmission, computation and queuing delay) for the subshard π_k^{nm} can be denoted as

$$T_k^{nm} = \sum_{i=0}^{I} \sum_{(e,f)\in\mathcal{E}_i^{\tau_l}} \frac{\prod_{i=0}^{i} \rho_i v_k^{nm}}{\omega_{ef}} x_{k,ef}^{nm} + \sum_{(e,f)\in\mathcal{E}_c^{\tau_l}} \frac{C_i \prod_{i=0}^{i-1} \rho_i v_k^{nm}}{\mu_{ef}} x_{k,ef}^{nm} + \sum_{i=0}^{I} \sum_{(e,f)\in\mathcal{E}_i^{\tau_l}} \frac{L_{ef}}{\omega_{ef}} x_{k,ef}^{nm} + \sum_{e\in\mathcal{V}_u^{\tau_l}} \frac{L_e}{\mu_e} x_{k,e}^{nm}. \quad (2)$$

For the subshard π_k^{nm}, since a path from the source (*i.e.,* observation satellite) s_k^{nm} to the destination (*i.e.,* ground station) t_k^{nm} needs to be determined, together with computation node selection, the following constraint need to be satisfied.

$$\mathbf{C2}: \sum_{f:(e,f)\in\mathcal{E}^{\tau_l}} x_{k,ef}^{nm} - \sum_{f:(f,e)\in\mathcal{E}^{\tau_l}} x_{k,fe}^{nm} = \begin{cases} 1, & if \quad e = s_k^{nm}, \\ -1, & if \quad e = t_k^{nm}, \\ 0, & otherwise, \end{cases} \quad (3)$$

where $\forall k \in \{1, \cdots, K^{nm}\}$, $n \in \mathcal{N}'$, $m \in \mathcal{M}$ and $e \in \mathcal{V}^{\tau_l}$.

C. Problem Formulation

Then the observation scheduling, routing and computation node selection problem can be formulated as follows:

$$\min_{\substack{x_{\tau_l}^{nm}, v_k^{nm}, \\ x_{k,ef}^{nm}, x_{k,e}^{nm}}} \sum_{l=1}^{T} \max_{\substack{n\in\{1,\cdots,V\} \\ m\in\{1,\cdots,M\}}} \{x_{\tau_l}^{nm} \max_{k\in\{1,\cdots,K^{nm}\}} T_k^{nm}\} \quad (\mathbf{P1})$$

$$\text{s.t.} \quad \mathbf{C1}, \mathbf{C2}, \mathbf{C3}: \sum_{k=1}^{K^{nm}} v_k^{nm} = W_{nm}, \forall n \in \mathcal{N}', m \in \mathcal{M},$$

$$\mathbf{C4}: v_k^{nm} \geq 0, \forall k \in \{1, \cdots, K^{nm}\}, n \in \mathcal{N}', m \in \mathcal{M},$$

$$\mathbf{C5}: x_{\tau_l}^{nm} \in \{0, 1\}, \forall n \in \mathcal{N}', m \in \mathcal{M}, l \in \{1, \cdots, T\},$$

$$\mathbf{C6}: x_{k,e}^{nm} \geq x_{k,e_{i-1}e_i}^{nm}, \forall k \in \{1, \cdots, K^{nm}\}, n \in \mathcal{N}', m \in \mathcal{M}, i \in \{1, \cdots, I\}, e \in \mathcal{V}^{\tau_l},$$

$$\mathbf{C7}: x_{k,ef}^{nm} \in \{0, 1\}, \forall k \in \{1, \cdots, K^{nm}\}, n \in \mathcal{N}', m \in \mathcal{M}, (e, f) \in \mathcal{E}^{\tau_l},$$

$$\mathbf{C8}: x_{k,e}^{nm} \in \{0, 1\}, \forall k \in \{1, \cdots, K^{nm}\}, n \in \mathcal{N}', m \in \mathcal{M}, e \in \mathcal{V}^{\tau_l}.$$

In the above, our objective is to minimize the time cost of routing and computing the observed shard in each time slot τ_l to achieve real-time response (*e.g.,* real-time object detection of each shard in a large RoI), where the inner max function represents the completion time of the subshard for the same shard, and the outer max function represents the completion time of the shard observed in the same time slot. Constraints **C3** and **C4** ensure that the size of the subshard after splitting is non-negative and sums to the size of the original shard. Constraints **C5**, **C6**, **C7** and **C8** ensure that $x_{k,ef}^{nm}$ and $x_{k,e}^{nm}$ are 0-1 integer variables and $x_{k,e}^{nm}$ is 1 in the optimal solution only if node e is selected for the computation.

III. Solution of the Problem

Solving problem **P1** faces several challenges. First of all, since problem **P1** is a mixed integer programming problem, it is NP-hard to be solved in general. Then the routing of different subshards would affect each other. Finally, since queuing delay is considered, it is difficult to solve directly for the shard splitting. To address the above challenges, we decouple the original problem into a two-stage optimization problem. The first stage is the optimization of multi-satellite observation scheduling. However, given the NP-hard nature of such combinatorial optimization problem, a centralized approach is difficult for optimization. Hence, we propose an efficient iterative algorithm relying on distributed inter-satellite interactions leveraging potential game theory [23]. The second stage is to optimize the processing of each shard. Specifically, for the smallest transmission unit (*i.e.,* subshard) in LEO satellite networks, we first prove that the routing and computation node selection has a theoretically optimal solution under linear programming relaxation for the consideration of queuing delay under our modeling framework. In addition, we derive the theoretically optimal solution for shard splitting, and then we design an efficient iterative algorithm to jointly

determine the shard splitting, routing and computation node selection. Finally, we design a theoretically guaranteed system-wide greedy-based strategy to optimize the time cost for the case where multiple shards are captured in the same time slot, taking into account the queuing delay at the system level.

A. Solution for the Observation Scheduling

For shard observation scheduling, we first define the set of feasible observation policies for each satellite. Specifically, according to the DAG $\mathcal{D}_n$ of each satellite $n \in \mathcal{N}'$, a feasible observation sequence from the start node a_n to the end node b_n forms a policy $\mathbf{p}_n^f$. All policies in $\mathcal{D}_n$ comprise the policy set of satellite n, denoted by $\mathcal{P}_n$=$\{\mathbf{p}_n^1, \mathbf{p}_n^2, \cdots\}$. Denote $\mathbf{p}_n^a$ as the longest sequence in $\mathcal{P}_n$ and $\mathcal{O}(\mathbf{p}_n)$ as the set of observation shards on policy $\mathbf{p}_n$. Each satellite $n \in \mathcal{N}'$ selects a policy $\mathbf{p}_n \in \mathcal{P}_n$ to observe shards and these selected policies consist of the policy profile denoted by $\mathbf{P}$=$(\mathbf{p}_1, \cdots, \mathbf{p}_V)$, where $\mathbf{P} \in \mathcal{P}_1 \times \cdots \times \mathcal{P}_V$. Note that the value of $x_{\tau_l}^{nm}$ can be obtained based on $\mathbf{P}$, *i.e.,* it is available through $\mathbf{P}$ at which time slot τ_l which shard m is observed by the satellite n. Hence, we convert the decision variable for observation scheduling to $\mathbf{P}$. Define $\sum_{l=1}^{T}\sum_{n=1}^{V} x_{\tau_l}^{nm} = |\{n | n \in \mathcal{N}', \mathbf{p}_n \in \mathbf{P}, m \in \mathcal{O}(\mathbf{p}_n)\}|$ as the number of satellites observing the same shard m.

The goal of observation scheduling is that multiple satellites work jointly to maximize the RoI coverage with the least repeated shards. To this end, the group of observation satellites needs to cover the maximum RoI while reducing the subsequent transmission and computation overhead. Accordingly, the reward for the satellite n to capture the shard m can be defined as the content percentage R_{nm} (*i.e.,* effective RoI coverage in an observed image). Hence, we define the global utility function $\Phi\colon \mathcal{P}_1 \times \cdots \times \mathcal{P}_V \to \mathbb{R}$ as

$$\Phi(\mathbf{P}) = \sum_{n=1}^{V} \sum_{m \in \mathcal{O}(\mathbf{p}_n)} \frac{R_{nm}}{\sum_{l=1}^{T}\sum_{n=1}^{V} x_{\tau_l}^{nm}}, \tag{4}$$

where $\sum_{m \in \mathcal{O}(\mathbf{p}_n)} \frac{R_{nm}}{\sum_{l=1}^{T}\sum_{n=1}^{V} x_{\tau_l}^{nm}}$ denotes that each observation satellite whose observation sequence $\mathbf{p}_n$ include shard m contributes $\frac{R_{nm}}{\sum_{l=1}^{T}\sum_{n=1}^{V} x_{\tau_l}^{nm}}$ to the global utility at m, which promotes more content percentages but discourage repeated shards in RoI coverage. Then the optimal observation sequence to maximize the global utility needs to be found, such that

$$\mathbf{P}^* = \underset{\mathbf{P} \in \mathcal{P}_1 \times \cdots \times \mathcal{P}_V}{\arg\max} \ \Phi(\mathbf{P}). \tag{5}$$

To achieve the above goal, we solve the complicated observation scheduling problem by casting it as a strategic game $G = (\mathcal{N}', \mathcal{P}_1 \times \cdots \times \mathcal{P}_V, \{U_1, \cdots, U_V\})$ and leverage potential game theory as the algorithm design tool, where $\mathcal{P}_n$ and U_n are the action space and the utility for each satellite $n \in \mathcal{N}'$. The key point then lies in formulating the local utility function U_n, such that the satellites try to iteratively improve their local utilities to boost the global utility. This formulation ensures that the game G operates as an exact potential game, *i.e.,* there is a guarantee of a Nash Equilibrium (NE), achievable by iterative and asynchronous better response update process (*i.e.,* one player improves its local utility in each iteration) [23]. Before designing, we provide Definition 1 based on [23].

Definition 1. *Let* $\mathbf{p}_{-n} = \{\mathbf{p}_1, \cdots, \mathbf{p}_{n-1}, \mathbf{p}_{n+1}, \cdots, \mathbf{p}_V\}$ *denote the action profile of opponents for satellite* $n \in \mathcal{N}'$. *Each satellite* $n \in \mathcal{N}'$ *aims to select the best action* $\mathbf{p}_n^*$ *to maximize its utility no matter what the* $\mathbf{p}_{-n}$ *are. In game* G*, a solution reaches a NE if each satellite* $n \in \mathcal{N}'$ *optimizes its local utility and no satellite can improve its utility by solely changing its actions. Further, game* G *is called an exact potential game if there exists a potential function* $\Phi\colon \{\mathcal{P}_n\}_{n \in \mathcal{N}'} \to \mathbb{R}$ *such that,* $\mathbf{p}_n', \mathbf{p}_n'' \in \mathcal{P}_n, \mathbf{p}_{-n} \in \{\mathcal{P}_i\}_{i \neq n}, \forall n \in \mathcal{N}'$*:*

$$U_n(\mathbf{p}_n', \mathbf{p}_{-n}) - U_n(\mathbf{p}_n'', \mathbf{p}_{-n}) = \Phi(\mathbf{p}_n', \mathbf{p}_{-n}) - \Phi(\mathbf{p}_n'', \mathbf{p}_{-n}). \tag{6}$$

In exact potential game G*, NE (i.e.,* $\mathbf{P}^*$*) is guaranteed to exist.*

We then construct the local utility function for each satellite $n \in \mathcal{N}'$ to form a potential game as follows.

Theorem 1. *The game* G *is an exact potential game when the local utility function of each satellite* $n \in \mathcal{N}'$ *is defined as*

$$U_n(\mathbf{P}) = |\mathcal{O}(\mathbf{p}_n)| - \sum_{m \in \mathcal{O}(\mathbf{p}_n)} \left(\sum_{l=1}^{T}\sum_{n=1}^{V} x_{\tau_l}^{nm} - \frac{\sum_{n=1}^{V} R_{nm}}{\sum_{l=1}^{T}\sum_{n=1}^{V} x_{\tau_l}^{nm}} \right). \tag{7}$$

Proof. The key idea is to very that the defined local utility of each satellite $n \in \mathcal{N}'$ satisfies the condition in (6). The proof is given in Appendix A in [24]. □

We have that the local utility of satellite $n \in \mathcal{N}'$ increases with the shard observation numbers $|\mathcal{O}(\mathbf{p}_n)|$ and local reward R_{nm}, and decreases with the repeated observation numbers for the same shard according to Theorem 1. Based on the property of potential game [23], we then design an iterative algorithm for observation scheduling in Algorithm 1, such that in each iteration a satellite improves its observation sequence to boost its local utility, which can lead to the optimization of the global utility in (4) until the equilibrium point (*e.g.,* no satellites can improve for a period of iterations) is reached. Note that the optimization process of observation scheduling only needs to be conducted once before the satellites carry out the actual EOM task operations until a new task is input.

Algorithm 1 Iterative Algorithm for Observation Scheduling

1: **Initialization:** $T = 0$, $A_n(0) = \mathbf{p}_n^a, \forall n \in \mathcal{N}'$ and $A(0) = \{A_n(0)\}_{n \in \mathcal{N}'}$.
2: **while** $A(T)$ is not NE **do**
3: Select an observation satellite $n \in \mathcal{N}'$ randomly.
4: Satellite n selects an improved strategy $A_n(T+1) = \mathbf{p}_n^i$, where $\mathbf{p}_n^i \in \mathcal{P}_n$.
5: Update $A_{-n}(T+1) = A_{-n}(T)$.
6: $A(T+1) = \{A_n(T+1), A_{-n}(T+1)\}$.
7: $T = T + 1$.
8: **end while**
9: **return** $\mathbf{P}^* = \{A_n(T)\}_{n \in \mathcal{N}'}$.

B. Solution for the Shard Splitting, Routing and Computation Node Selection of Each Shard

1) Solution for the routing and computation node selection: We now perform the solution for the optimal routing and computation node selection of each subshard π_k^{nm}. We first assume that the splitting size of the shard is fixed and the subproblem is hence formulated as follows:

$$\min_{x_{k,ef}^{nm}, x_{k,e}^{nm}} \quad T_k^{nm} \tag{$\mathbf{P_{sub1}}$}$$

$$\text{s.t.} \quad \mathbf{C2}, \mathbf{C6} - \mathbf{C8}.$$

Though the problem $\mathbf{P_{sub1}}$ is NP-hard to be solved in general, we show in the following Theorem that it has 0-1 integer optimal solutions, *i.e.,* the optimal solution can also be

found by relaxing constraints **C7** and **C8** to $0 \leq x_{k,ef}^{nm} \leq 1$ and $0 \leq x_{k,e}^{nm} \leq 1$. According to the above, the problem $\mathbf{P_{sub1}}$ can be solved in polynomial time.

Theorem 2. *For the routing and computation node selection problem of subshard π_k^{nm}, it has integer optimal solutions after linear programming (LP) relaxation (i.e., **C7** and **C8** are relaxed to $0 \leq x_{k,ef}^{nm} \leq 1$ and $0 \leq x_{k,e}^{nm} \leq 1$).*

Proof. The key idea is to prove that the formulation of $\mathbf{P_{sub1}}$ has a special structure called **total unimodularity**, which ensures the integer optimal solutions after LP relaxation [25]. The proof is given in Appendix B in [24]. □

2) Solution for the shard splitting: To solve the shard splitting optimally, we first fix the routing and computation node selection policies for each subshard of shard $m \in \mathcal{M}$ and define the set of the selected transmission links and computation nodes as $\mathcal{X}_k^{nm}$. Then without considering the queuing delay, the unit transmission and computation delay for the subshard π_k^{nm} can be denoted as

$$t_k^{nm}=\sum_{i=0}^{I}\sum_{(e,f)\in\mathcal{E}_i^{\tau_l}\cap\mathcal{X}_k^{nm}}\frac{\prod_{i=0}^{i}\rho_i}{\omega_{ef}}+\sum_{(e,f)\in\mathcal{E}_c^{\tau_l}\cap\mathcal{X}_k^{nm}}\frac{C_i\prod_{i=0}^{i-1}\rho_i}{\mu_{ef}}. \quad (8)$$

Due to the preemptive scheduling setting in Subsection II-B, the lower priority subshard needs to wait for the higher priority subshard during transmission and computation. So the lower priority subshard will incur queuing delay if it uses the same transmission link and computation node as in the higher priority subshard. Here we set the priority for the subshard as the index order. Later we will show that the initial setting of different priorities for subshards in a shard does not affect the final result. Let $\mathcal{X}_{j,k}^{nm}=\mathcal{X}_j^{nm}\cap\mathcal{X}_k^{nm}$ denote the intersection of the policies with subshard π_j^{nm} and π_k^{nm}. Then for subshard π_k^{nm}, the unit queuing delay caused by the j-th subshard with higher priority than itself can be denoted as

$$t_{j,k}^{nm}=\sum_{i=0}^{I}\sum_{(e,f)\in\mathcal{E}_i^{\tau_l}\cap\mathcal{X}_{j,k}^{nm}}\frac{\prod_{i=0}^{i}\rho_i}{\omega_{ef}}+\sum_{(e,f)\in\mathcal{E}_c^{\tau_l}\cap\mathcal{X}_{j,k}^{nm}}\frac{C_i\prod_{i=0}^{i-1}\rho_i}{\mu_{ef}}, \quad (9)$$

where $1 \leq j < k$. Then we define a diagonal matrix $\mathbf{D}^{nm}$ as

$$\mathbf{D}^{nm}\triangleq\left(t_1^{nm},t_2^{nm}+\frac{v_1^{nm}}{v_2^{nm}}t_{1,2}^{nm},\cdots,t_{K^{nm}}^{nm}+\sum_{j=1}^{K^{nm}-1}\frac{v_j^{nm}}{v_{K^{nm}}^{nm}}t_{j,K^{nm}}^{nm}\right). \quad (10)$$

Since the priority is ordered by the index order of the subshard, each diagonal element in $\mathbf{D}^{nm}$ represents the time cost of each subshard π_k^{nm} (*i.e.,* transmission and computation delay plus queuing delay for occupying the same transmission link and computation node). Denote $\mathbf{V}^{nm}\triangleq[v_1^{nm},\cdots,v_{K^{nm}}^{nm}]^\top\in\mathbb{R}^{K^{nm}}$ as the variables regarding the splitting size of shard m. Then, the subproblem for shard splitting can be denoted as

$$\begin{aligned}\min_{\mathbf{V}^{nm}}\quad & \|\mathbf{D}^{nm}\mathbf{V}^{nm}\|_\infty \qquad (\mathbf{P_{sub2}})\\ \text{s.t.}\quad & \mathbf{D1}:\mathbf{1}^\top\mathbf{V}^{nm}=W_{nm},\\ & \mathbf{D2}:\mathbf{V}^{nm}\geq 0.\end{aligned}$$

$\mathbf{P_{sub2}}$ is an infinity norm minimization problem (*i.e.,* the inner min-max function in $\mathbf{P1}$), the slack form of which is feasible and its optimal objective value is finite. Therefore, the simplex method and interior point method can be applied to obtain the optimal solution efficiently. To reduce the time complexity, we derive the analytical expression of the optimal solution for $\mathbf{P_{sub2}}$. The result is as the following theorem.

Algorithm 2 Iterative Algorithm for the Optimal Shard Splitting, Routing and Computation Node Selection for Each Shard

Initialization: $E=0$, $L=0$, $\mathcal{X}_k^{nm}(0)$ and $v_k^{nm}(0)=W_{nm}/K^{nm},\forall k\in\{1,\cdots,K^{nm}\}$.
while $\mathcal{X}_k^{nm}(E-1)\neq\mathcal{X}_k^{nm}(E)$, $\forall k\in\{1,\cdots,K^{nm}\}$ **do**
  $E=E+1$.
  Solve problem $\mathbf{P_{sub}}$ for subshard π_k^{nm}, $\forall k\in\{1,\cdots,K^{nm}\}$.
  Obtain policies $\mathcal{X}_k^{nm}(E)$, $\forall k\in\{1,\cdots,K^{nm}\}$ from line 4.
  Update $\mathbf{D}^{nm}(E)$ according to the line 5.
  while $\mathbf{V}^{nm}(L-1)\neq\mathbf{V}^{nm}(L)$ **do**
    $L=L+1$.
    Calculate the value in equation (12).
    Update $v_k^{nm}(L),\forall k\in\{1,\cdots,K^{nm}\}$ based on equation (13).
  end while
end while
return $v_k^{nm*}=v_k^{nm}(L)$ and $\mathcal{X}_k^{nm*}=\mathcal{X}_k^{nm}(E)$, $\forall k\in\{1,\cdots,K^{nm}\}$.

Theorem 3. *The optimal objective value of $\mathbf{P_{sub2}}$ is*

$$\min_{\mathbf{V}^{nm}}\|\mathbf{D}^{nm}\mathbf{V}^{nm}\|_\infty=\frac{W_{nm}}{\sum_{k=1}^{K^{nm}}1/D_{k,k}^{nm}}, \quad (11)$$

if

$$D_{i,i}^{nm}v_i^{nm}=D_{j,j}^{nm}v_j^{nm},1\leq i\neq j\leq K^{nm}, \quad (12)$$

where v_i^{nm} is the i-th element of vector $\mathbf{V}^{nm}$ and $D_{i,i}^{nm}$ is the i-th diagonal element of $\mathbf{D}^{nm}$.

Proof. For $\mathbf{P_{sub2}}$, since $D_{k,k}^{nm}>0$, $v_k^{nm}\geq 0$, $\forall k$, we have

$$\|\mathbf{D}^{nm}\mathbf{V}^{nm}\|_\infty\triangleq\max_k\{|D_{k,k}^{nm}v_k^{nm}|\}=\lim_{x\to\infty}\sqrt[x]{\sum_{k=1}^{K^{nm}}\left(D_{k,k}^{nm}v_k^{nm}\right)^x}, \quad (13)$$

According to the AM-GM inequality [26], the following inequality always holds:

$$\frac{\sum_{k=1}^{K^{nm}}\left(D_{k,k}^{nm}v_k^{nm}\right)^x}{K^{nm}}\geq\sqrt[K^{nm}]{\prod_{k=1}^{K^{nm}}\left(D_{k,k}^{nm}v_k^{nm}\right)^x}, \quad (14)$$

if (12) is satisfied, for $\forall x>0$, we can get that

$$\sqrt[x]{\frac{\sum_{k=1}^{K^{nm}}\left(D_{k,k}^{nm}v_k^{nm}\right)^x}{K^{nm}}}\geq\sqrt[K^{nm}]{\prod_{k=1}^{K^{nm}}D_{k,k}^{nm}v_k^{nm}}. \quad (15)$$

Multiply both sides by $\sqrt[x]{K^{nm}}$ and take the limit, we have

$$\|\mathbf{D}^{nm}\mathbf{V}^{nm}\|_\infty\geq\lim_{x\to\infty}\sqrt[x]{K^{nm}}\cdot\sqrt[K^{nm}]{\prod_{k=1}^{K^{nm}}D_{k,k}^{nm}v_k^{nm}}. \quad (16)$$

Combining with $\mathbf{P_{sub2}}$ and (12), the right side of (16) is actually a constant. Then we can obtain that

$$\begin{aligned}\min_{\mathbf{V}^{nm}}\|\mathbf{D}^{nm}\mathbf{V}^{nm}\|_\infty&=\lim_{x\to\infty}\sqrt[x]{K^{nm}}\cdot\sqrt[K^{nm}]{\prod_{k=1}^{K^{nm}}D_{k,k}^{nm}v_k^{nm}}\\&=\sqrt[K^{nm}]{\prod_{k=1}^{K^{nm}}D_{k,k}^{nm}v_k^{nm}}=\frac{W_{nm}}{\sum_{k=1}^{K^{nm}}1/D_{k,k}^{nm}}. \quad (17)\end{aligned}$$

It can be obtained from the above that (11) and (12) are the optimal objective value and corresponding optimal condition of $\mathbf{P_{sub2}}$, respectively. This completes the proof. □

According to Theorem 3, since the final optimal solution is that each subshard has the same time cost considering the queuing delay, it will eventually converge to the same time cost regardless of the initial priority setting of different subshards. Based on Theorems 2-3, we design the iterative Algorithm 2 to solve the size of the optimal shard splitting and obtain the solution for the optimal routing and computation node selection policies for each subshard π_k^{nm}.

C. System-wide Greedy-based Routing and Computation Node Selection Policies

In this section, we focus on the case of multiple shard arrivals in the same time slot τ_l. To this end, we develop a corresponding greedy-based scheme for minimizing the time cost while considering the queuing delay at the system level. We denote $\mathcal{L}^{\tau_l} = \{L_{ef}^{\tau_l}, \forall (e,f) \in \mathcal{E}^{\tau_l}\}$ as the set of queuing tasks in LEO satellite networks at time slot τ_l. Let $s \in \mathcal{S}^{\tau_l}$ denote the shard captured in the time slot τ_l, where $\mathcal{S}^{\tau_l} = \{1, \cdots, S^{\tau_l}\}$ and $S^{\tau_l} = \sum_{n=1}^{V} \sum_{m=1}^{M} x_{\tau_l}^{nm}$. Note that the initialization of set $\mathcal{S}^{\tau_l}, \forall l \in \{1, \cdots, T\}$ can be obtained according to the Algorithm 1. As illustrated in Algorithm 3, based on the dynamic property of LEO satellite networks, we solve for routing and computation node selection for shards arriving at each time slot τ_l in time sequence. Note that if only one shard is captured in a time slot, the shard splitting number K^{nm} can be determined through a preset experiment. Specifically, as will be shown in Experiment (5) of Section IV, a large shard splitting number leads to dominating network queuing delay and increased time cost. Therefore, the optimal shard splitting number for each shard can be determined via preset experiments. Further, the routing and computation node selection policy can be directly obtained from Algorithm 2.

If more than one shard is captured in the same time slot, we propose a greedy-based scheme to determine the priority for each shard. Denote s_j as the j-th shard that performs the routing and computation node selection policies under the greedy scheme, where $j \in \{1, \cdots, J\}$ and $J = S^{\tau_l}$. Additionally, let K^{s_j} represent the splitting number for the j-th shard, the optimal value of which can be obtained through a preset experiment. Further, we define $U_{K^{s_j}}^{\tau_l}(\mathcal{L}^{\tau_l})$ as the time cost for shard s_j arrived in the time slot τ_l when it is routed on the basis of the formulation in the presence of queuing tasks $\mathcal{L}^{\tau_l}$ under the shard splitting number is K^{s_j}. Then, in each round of our proposed algorithm, the shard with the smallest time cost is selected to give the highest priority in time slot τ_l. After that, the set of queuing tasks is updated, and the routing and computation node selection for the remaining shards in the same time slot is based on the updated $\mathcal{L}^{\tau_l}$. The above procedure is repeated until all the shards in the same time slot have completed solving the routing and computation node selection policies. Notably, the ground station can function as the coordinator to perform the Algorithm 3 and broadcast the results to LEO satellites for distributed execution.

We further analyze the performance (*i.e.,* approximation ratio) of Algorithm 3 regarding the optimal solution as follows.

Theorem 4. *We denote Z_s^H and Z_s^L as the longest and shortest path lengths in hop count between the source node and destination node, respectively. Further, let $Z^H = \max_s Z_s^H$ and $Z^L = \max_s Z_s^L$. Suppose each satellite in LEO satellite networks has A antennas (i.e., $\mathcal{G}_u^{\tau_l}$ is A-edge-connected). The time cost under the greedy algorithm in Algorithm 3 is at most $\alpha T^{\tau_l *}$, where $T^{\tau_l *}$ is the optimal time cost in time slot τ_l and*

$$\alpha = \left(2 - \frac{1}{|\mathcal{V}_u^{\tau_l}| + |\mathcal{E}_u^{\tau_l}|}\right) \max\left\{\left(1 + \frac{|\mathcal{E}_u^{\tau_l}|}{|\mathcal{V}_u^{\tau_l}|}\right) \frac{\max\limits_e \mu_e}{\min\limits_e \mu_e}, \right.$$

Algorithm 3 System-wide Greedy-based Routing and Computation Node Selection Algorithm

Input: Observation scheduling strategy $x_{\tau_l}^{nm}$, where $n \in \mathcal{N}'$, $m \in \mathcal{M}$ and $l \in \{1, \cdots, T\}$.
Initialization: $j = 0$, $L_{ef}^{\tau_l} = 0$, $\forall (e,f) \in \mathcal{E}^{\tau_l}$, $\mathcal{S}^{\tau_l}$, where $l \in \{1, \cdots, T\}$.
function QUEUING DELAY COMPUTING( )
  Obtain v_k^{nm}, $\mathcal{X}_k^{nm}$, $\forall k \in \{1, \cdots, K^{nm}\}$ from Algorithm 2.
  for $k = 1, \cdots, K^{nm}$ **do**
    $L_{ef}^{\tau_l} \leftarrow L_{ef}^{\tau_l} + x_{k,ef}^{nm} D_{k,i}^{nm}$, $\forall (e,f) \in \mathcal{E}_i^{\tau_l}$, $i \in \{0, \cdots, I\}$.
    $L_e^{\tau_l} \leftarrow L_e^{\tau_l} + x_{k,ef}^{nm} Q_{k,i}^{nm}$, $\forall (e,f) \in \mathcal{E}_c^{\tau_l}$, $i \in \{1, \cdots, I\}$.
  end for
end function
for $l = 1, \cdots, T$ **do**
  if $S^{\tau_l} = 0$ **then** continue.
  else if $S^{\tau_l} = 1$ **then**
    Obtain K^{nm} through the preset experiment.
    Obtain v_k^{nm}, $\mathcal{X}_k^{nm}$, $\forall k \in \{1, \cdots, K^{nm}\}$ from Algorithm 2.
  else
    $j = 0$.
    while $\mathcal{S}^{\tau_l} \neq \emptyset$ **do**
      $j = j + 1$.
      Obtain K^s by the preset experiment, $\forall s \in \mathcal{S}^{\tau_l}$.
      $s_j = \arg\min_{s \in \mathcal{S}^{\tau_l}} U_{K^s}^{\tau_l}(\mathcal{L}^{\tau_l})$.
      $\mathcal{L}^{\tau_l} \leftarrow$ QUEUING DELAY COMPUTING( ).
      $\mathcal{S}^{\tau_l} \leftarrow \mathcal{S}^{\tau_l} \backslash \{s_j\}$.
    end while
  end if
end for
return v_k^{nm} and $\mathcal{X}_k^{nm}$, $\forall k \in \{1, \cdots, K^{nm}\}, n \in \mathcal{N}', m \in \mathcal{M}$.

$$\left. \left(2, \frac{2(I+1)(|\mathcal{V}_u^{\tau_l}| + |\mathcal{E}_u^{\tau_l}|)}{A}\right) \frac{Z^H \max\limits_{s,k,i} v_k^s \max\limits_{(e,f)} \omega_{ef}}{Z^L \min\limits_{s,k,i} v_k^s \min\limits_{(e,f)} \omega_{ef}} \right\}. \quad (18)$$

Proof. We introduce the Minimum Service Time (MST) policy, which achieves the lowest transmission and computation time for each shard s with the time cost F_s^{MST}, and we can obtain that $\frac{1}{|\mathcal{V}_u^{\tau_l}| + |\mathcal{E}_u^{\tau_l}|} \sum_{s=1}^{S^{\tau_l}} F_s^{MST} \leq T^{\tau_l *}$. Moreover, we introduce the Minimum Queuing Time (MQT) policy, which means that $J-1$ shards have been optimized in τ_l by Algorithm 3, the J-th shard is assigned to a node e^* with the minimum computation queuing time. Note that the time cost of Algorithm 3 will not exceed MQT. Define $\alpha_{tran} = \frac{Z^H \max_{s,k,i} v_k^s \max_{(e,f)} \omega_{ef}}{Z^L \min_{s,k,i} v_k^s \min_{(e,f)} \omega_{ef}}$ and $\alpha_{comp} = \frac{\max_e \mu_e}{\min_e \mu_e}$. We can get the upper bound for the transmission time plus computation time in MQT is $\max(2\alpha_{tran}, \alpha_{comp}) T^{\tau_l *}$ and the upper bound for the queuing time in MQT is $\max(\frac{2(I+1)\alpha_{tran}}{A}, \frac{\alpha_{comp}}{|\mathcal{V}_u^{\tau_l}|}) \sum_{j=1}^{J-1} F_{s_j}^{MST}$. Combining the above, we can prove Theorem 4. The detailed proof is given in Appendix C in [24]. □

IV. PERFORMANCE EVALUATION

In this section, we will present experiment results based on real-world datasets to verify the performance enhancement of our proposed framework. Specifically, we consider a Starlink constellation of 200 satellites evenly distributed over 20 orbits by observing Hong Kong and routing the observed shards and computing results to the ground station in Shanghai. The satellite dynamic characteristics are captured by Satellite Tool Kit (STK) simulation software. For the observation, as shown in Fig. 4, we assigned 10 satellites to observe Hong Kong, the green ones are ascending satellites and the orange ones are

TABLE I: Simulation Parameters.

Parameter	Value
Number of LEO satellites	200
Number of observation satellites	10
Number of antennas for each LEO satellite	4
Inter-satellite communication rate	$[1, 1.5]\,Gbps$
Satellite-to-ground communication rate	$[0.8, 1]\,Gbps$
Satellite computation capability	$[2.5, 6]\,MHz$
Ground station computation capability	$[8, 20]\,MHz$
Image resolution for captured shard	$200\,Mb$

descending satellites. The observation area of these satellites cut Hong Kong into 21 shards and the observation area of the satellite can be switched between three adjacent strips. According to the settings in [27], the important simulation parameters are listed in Table I. In addition, we consider two computation tasks, which are described as follows.

- **Multi-stage image compression (MIC) [28]:** We consider a five-stage image compression task, where the compression rate for each stage is 0.8,0.7,0.5,0.3 and 0.1 of the original captured image, respectively.
- **Object detection by YOLO (OD) [20]:** We utilize YOLO as the detection network, which has 24 convolutional layers followed by 2 fully connected layers. The detailed network parameters can be found in [20].

To evaluate the performance of the proposed framework, the following baselines are compared.

- **NoSplit:** Instead of splitting the shard, this scheme directly transmits and computes the captured shard.
- **AvgSplit [16]:** This scheme evenly splits the captured shard for transmission and computation (*i.e.,* the size of each subshard is identical for a shard.).
- **NoQueue [29]:** This scheme performs our framework without considering queuing delay optimization.
- **Longest [5]:** In this scheme, each satellite captures its own entire strip without switching strips.

1) Observation Scheduling: Fig. 5 and Fig. 6 illustrate the coverage and observation size for different schemes under different amounts of observation satellites. To validate the scalability of our proposed framework, we set up two observation cases. The first one is that the satellite can continuously capture shards, *i.e.,* the rolling velocity of the satellite can be infinite, called case 1. The other one is that if the satellite needs to switch the stripe after capturing the current shard, the maximum rolling velocity of the satellite is limited to $\frac{\pi}{6} rad/s$, called case 2. In addition, for the scenario with 6 satellites observing, the satellites with the serial numbers 0, 2, 4, 5, 7 and 9 in Fig. 4 are assigned to observe. It should be noted that given the limited number of satellites, no observation scheduling can achieve complete coverage of the RoI. It can be observed from Fig. 5 and Fig. 6 that SECO reduces the number of repeated shards and thus the total captured image size with improved coverage compared to Longest. Moreover, SECO shows superior performance for any observation cases and amounts of observation satellites, demonstrating its robustness.

2) Impact of image resolution on the time cost: In Fig. 7 and Fig. 8, we can observe that the time cost increases with higher image resolutions. This is primarily due to the increased load on both transmission and computation when dealing with larger image sizes. Moreover, the absence of shard splitting results in the highest time cost, clearly highlighting the indispensability of this technique. Furthermore, we can observe that the adoption of evenly splitting is non-optimal when considering queuing delay, which proves the effectiveness of the comprehensive optimization strategy adopted by SECO. Additionally, SECO surpasses all baseline methods in both MIC and OD tasks, demonstrating its superior performance.

Fig. 4: Illustration of observing Hong Kong by Starlink.

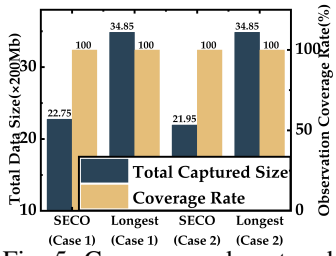


Fig. 5: Coverage and captured size (10 observation satellites).

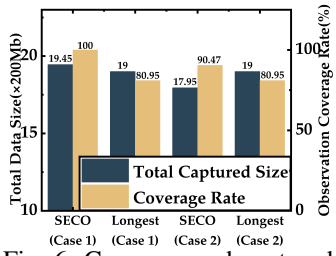


Fig. 6: Coverage and captured size (6 observation satellites).

3) Impact of computation capacity on the time cost: It can be observed in Fig. 9 and Fig. 10 that the time cost exhibits a clear inverse relationship with satellite computation capacity, indicating that an increase in computation capacity leads to a reduction in time cost for both the MIC and OD tasks. This can be attributed to the fact that higher computation capacity enhances system performance, resulting in decreased time costs. Moreover, the figures provide further evidence that the time cost without queuing delay optimization is significantly higher compared to our framework, thus underscoring the imperative nature of optimizing queuing delay.

4) Impact of communication capacity on the time cost: From Fig. 11 and Fig. 12, it can be observed that there is a reduction in time cost as the communication capacity increases. This phenomenon can be attributed to the enhanced hardware capacity, which improves the efficiency of both queuing and service, consequently reducing the overall time cost. Notably, the optimization effect of our framework is more pronounced in the OD task compared to the baseline, thereby validating the robust optimization performance of our framework in complex computation tasks.

5) Impact of the shard splitting number on time cost: We can observe from Fig. 13 that the time cost shows a decreasing

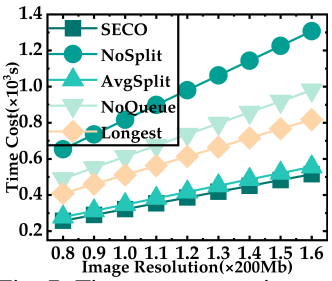


Fig. 7: Time cost versus image resolutions (MIC).

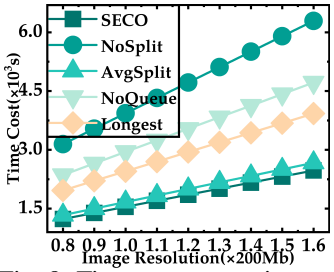


Fig. 8: Time cost versus image resolutions (OD).

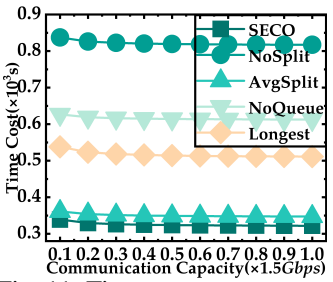


Fig. 11: Time cost versus communication capacities (MIC).

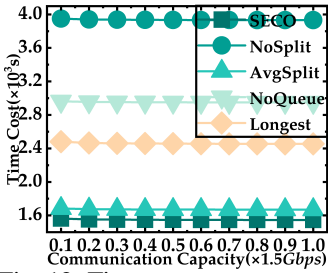


Fig. 12: Time cost versus communication capacities (OD).

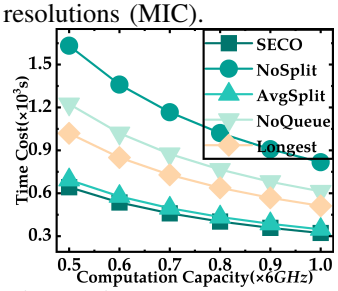


Fig. 9: Time cost versus computation capacities (MIC).

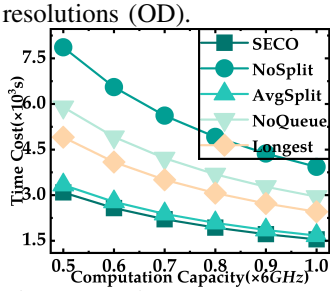


Fig. 10: Time cost versus computation capacities (OD).

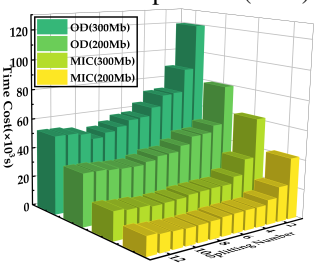


Fig. 13: Time cost versus splitting numbers for a shard.

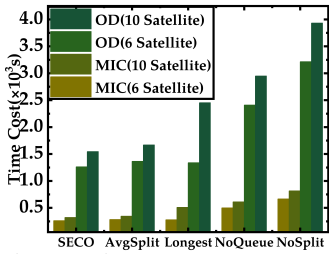


Fig. 14: Time cost versus number of observation satellites.

trend initially, followed by an increase as the shard splitting number increases, both for the MIC and OD tasks. This can be attributed to the occurrence of significant queuing phenomena in the LEO satellite network when the shard splitting number becomes excessively large, leading to a dominant queuing delay. Therefore, a preliminary experiment can be conducted to determine the optimal shard splitting number.

6) Impact of the number of observation satellites on time cost: According to Fig. 14, our framework exhibits superior performance compared to all baseline models in scenarios involving different numbers of observation satellites. Remarkably, our framework achieves a substantial reduction in time cost, up to 60.7%, while ensuring 100% observation coverage. Moreover, our framework delivers significant optimization outcomes for both the MIC and OD tasks, thereby validating the effectiveness of observation scheduling, shard splitting and optimization strategies that consider queuing delay.

V. Related Works

Recent advancements in free-space optical (FSO) communications technology [30] have enabled the establishment of ISLs within LEO satellite networks. Consequently, complete SEC services can be realized within the LEO satellite networks. Several research efforts have explored the potential of SEC technology, particularly in addressing computation challenges for both terrestrial and space missions [2], [27], [31]–[33].

Current Earth observation mission (EOM) advancements facilitate various tasks such as disaster monitoring and resource exploration [8]. Traditional observation scheduling approaches typically focus on capturing multiple RoIs, with each RoI assigned to a single satellite for imaging [6], [9], [10]. Lv *et al.* [6] presented a holistic solution that precisely models the observation service of a single LEO satellite and allocates several Earth observation tasks to a group of LEO satellites. An iterative algorithm is proposed in [9] to coordinate satellite observation resources to schedule observations for multiple RoIs. However, the above efforts do not allow for the scheduling of fine-grained observations for a large RoI area. In [5], a distributed compression scheme is designed for a large RoI area with high resolution. However, this scheme does not consider the scheduling of observations within the RoI, and each satellite captures a complete shot based on its trajectory.

Various studies have aimed to tackle routing and computation node selection in resource-limited edges for complex inter-task dependent functions. Liu *et al.* [34] proposed a framework that forms such functions as a service function chain, and the computation node selection for each function is obtained by minimizing the total time cost. In [17], a dependent function embedding algorithm has been designed that considers both routing and computation node selection. However, the congestion in the network is not considered in the above work. Jung *et al.* [22] proposed a framework that optimizes the routing for DNN inference jobs in distributed networks. Nonetheless, there is still a need to incorporate congestion considerations and explore the potential benefits of raw image splitting for parallel transmission and computation.

VI. Conclusion

In this paper, we propose a SEC-enabled framework that jointly optimizes observation scheduling, routing and computation node selection for wide-area and real-time EOM (SECO). SECO incorporates a distributed observation scheduling strategy based on game theory. This approach enables fine-grained boundary-side scheduling and multi-satellite cooperative observations of a large RoI area. To enhance efficiency, shard splitting is introduced to facilitate parallel transmission and computation. Regarding routing and computation node selection, SECO considers queuing delay within the LEO satellite networks during the optimization process. Further, we propose a theoretically guaranteed system-wide greedy-based strategy to minimize the time cost while considering queuing delay at a system level for simultaneous shard capturing.

REFERENCES

[1] Renchao Xie, Qinqin Tang, Qiuning Wang, Xu Liu, F. Richard Yu, and Tao Huang. Satellite-terrestrial integrated edge computing networks: Architecture, challenges, and open issues. *IEEE Network*, 34(3):224–231, 2020.

[2] Xinyuan Zhang, Jiang Liu, Ran Zhang, Yudong Huang, Jincheng Tong, Ning Xin, Liang Liu, and Zehui Xiong. Energy-efficient computation peer offloading in satellite edge computing networks. *IEEE Transactions on Mobile Computing*, pages 1–15, 2023.

[3] Jehyun Heo, Seungwoo Sung, Hyunwoo Lee, Incheol Hwang, and Daesik Hong. Mimo satellite communication systems: A survey from the phy layer perspective. *IEEE Communications Surveys & Tutorials*, 2023.

[4] M. Harris. Tech giants race to build orbital internet [news]. *IEEE Spectrum*, 55(6):10–11, 2018.

[5] Israel Leyva-Mayorga, Marc Martinez-Gost, Marco Moretti, Ana Peŕez-Neira, Miguel Ángel Vázquez, Petar Popovski, and Beatriz Soret. Satellite edge computing for real-time and very-high resolution earth observation. *IEEE Transactions on Communications*, 2023.

[6] Mingsong Lv, Xuemei Peng, Wenjing Xie, and Nan Guan. Task allocation for real-time earth observation service with leo satellites. In *2022 IEEE Real-Time Systems Symposium (RTSS)*, pages 14–26. IEEE, 2022.

[7] Camile Sothe, Alemu Gonsamo, Joyce Arabian, and James Snider. Large scale mapping of soil organic carbon concentration with 3d machine learning and satellite observations. *Geoderma*, 405:115402, 2022.

[8] Ivan Franch-Pardo, Brian M Napoletano, Fernando Rosete-Verges, and Lawal Billa. Spatial analysis and gis in the study of covid-19. a review. *Science of the total environment*, 739:140033, 2020.

[9] Yu Wang, Min Sheng, Weihua Zhuang, Shan Zhang, Ning Zhang, Runzi Liu, and Jiandong Li. Multi-resource coordinate scheduling for earth observation in space information networks. *IEEE Journal on Selected Areas in Communications*, 36(2):268–279, 2018.

[10] Wei Liu, Huiting Yang, and Jiandong Li. Multi-functional time expanded graph: A unified graph model for communication, storage and computation for dynamic networks over time. *IEEE Journal on Selected Areas in Communications*, 2023.

[11] Qiang Zhao, Le Yu, Zhenrong Du, Dailiang Peng, Pengyu Hao, Yongguang Zhang, and Peng Gong. An overview of the applications of earth observation satellite data: impacts and future trends. *Remote Sensing*, 14(8):1863, 2022.

[12] Zhishu Shen, Jiong Jin, Cheng Tan, Atsushi Tagami, Shangguang Wang, Qing Li, Qiushi Zheng, and Jingling Yuan. A survey of next-generation computing technologies in space-air-ground integrated networks. *ACM Computing Surveys*, 2023.

[13] Van-Phuc Bui, Thinh Q Dinh, Israel Leyva-Mayorga, Shashi Raj Pandey, Eva Lagunas, and Petar Popovski. On-board change detection for resource-efficient earth observation with leo satellites. *arXiv preprint arXiv:2305.10119*, 2023.

[14] Bing Zhang, Yuanfeng Wu, Boya Zhao, Jocelyn Chanussot, Danfeng Hong, Jing Yao, and Lianru Gao. Progress and challenges in intelligent remote sensing satellite systems. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15:1814–1822, 2022.

[15] Roberto Cordone, Federico Gandellini, and Giovanni Righini. Solving the swath segment selection problem through lagrangean relaxation. *Computers & operations research*, 35(3):854–862, 2008.

[16] Wuyang Zhang, Zhezhi He, Luyang Liu, Zhenhua Jia, Yunxin Liu, Marco Gruteser, Dipankar Raychaudhuri, and Yanyong Zhang. Elf: accelerate high-resolution mobile deep vision with content-aware parallel offloading. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 201–214, 2021.

[17] Shuiguang Deng, Hailiang Zhao, Zhengzhe Xiang, Cheng Zhang, Rong Jiang, Ying Li, Jianwei Yin, Schahram Dustdar, and Albert Y Zomaya. Dependent function embedding for distributed serverless edge computing. *IEEE Transactions on Parallel and Distributed Systems*, 33(10):2346–2357, 2021.

[18] Mario Minardi, Youssouf Drif, Thang X Vu, Ilora Maity, Christos Politis, and Symeon Chatzinotas. Sdn-based testbed for emerging use cases in beyond 5g ntn-terrestrial networks. In *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*, pages 1–6. IEEE, 2023.

[19] Virginie Gabrel, Alain Moulet, Cécile Murat, and Vangelis Th Paschos. A new single model and derived algorithms for the satellite shot planning problem using graph theory concepts. *Annals of Operations Research*, 69(0):115–134, 1997.

[20] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[21] Jianan Zhang, Abhishek Sinha, Jaime Llorca, Antonia M Tulino, and Eytan Modiano. Optimal control of distributed computing networks with mixed-cast traffic flows. *IEEE/ACM Transactions on Networking*, 29(4):1760–1773, 2021.

[22] Sehun Jung and Hyang-Won Lee. Optimization framework for splitting dnn inference jobs over computing networks. *Computer Networks*, page 109814, 2023.

[23] Dov Monderer and Lloyd S Shapley. Potential games. *Games and economic behavior*, 14(1):124–143, 1996.

[24] Zhiwei Zhai, Liekang Zeng, Tao Ouyang, Shuai Yu, Qianyi Huang, and Xu Chen. SECO: Multi-satellite edge computing enabled wide-area and real-time earth observation missions. 2023. Available online at: https://1drv.ms/b/s!An3CC4kPGOzbhHb0MRUFa9Gjp0mD?e=gtFYJc.

[25] Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.

[26] Zdravko Cvetkovski. *Inequalities: theorems, techniques and selected problems*. Springer Science & Business Media, 2012.

[27] Qingze Fang, Zhiwei Zhai, Shuai Yu, Qiong Wu, Xiaowen Gong, and Xu Chen. Olive branch learning: A topology-aware federated learning framework for space-air-ground integrated network. *IEEE Transactions on Wireless Communications*, 2022.

[28] Xiaoyang Li, Changsheng You, Sergey Andreev, Yi Gong, and Kaibin Huang. Wirelessly powered crowd sensing: Joint power transfer, sensing, compression, and transmission. *IEEE Journal on Selected Areas in Communications*, 37(2):391–406, 2018.

[29] Jiaqi Cao, Shengli Zhang, Qingxia Chen, Houtian Wang, Mingzhe Wang, and Naijin Liu. Computing-aware routing for leo satellite networks: A transmission and computation integration approach. *arXiv preprint arXiv:2211.08820*, 2022.

[30] Peng Hu. A cross-layer descent approach for resilient network operations of proliferated leo satellites. In *2023 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2023.

[31] Lei Cheng, Gang Feng, Yao Sun, Mengjie Liu, and Shuang Qin. Dynamic computation offloading in satellite edge computing. In *ICC 2022-IEEE International Conference on Communications*, pages 4721–4726. IEEE, 2022.

[32] Francesco Valente, Francesco G Lavacca, and Vincenzo Eramo. A resource allocation strategy in earth observation orbital edge computing-enabled satellite networks to minimize ground station energy consumption. In *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*, pages 1–6. IEEE, 2023.

[33] Xuelin Cao, Bo Yang, Yulong Shen, Chau Yuen, Yan Zhang, Zhu Han, H Vincent Poor, and Lajos Hanzo. Edge-assisted multi-layer offloading optimization of leo satellite-terrestrial integrated networks. *IEEE Journal on Selected Areas in Communications*, 41(2):381–398, 2022.

[34] Liuyan Liu, Haisheng Tan, Shaofeng H-C Jiang, Zhenhua Han, Xiang-Yang Li, and Hong Huang. Dependent task placement and scheduling with function configuration in edge computing. In *Proceedings of the International Symposium on Quality of Service*, pages 1–10, 2019.