Online Satellite Selection, Request Dispatching, and Service Provisioning in LEO Edge Constellations

Siru Chen, Lei Jiao, *Member, IEEE*, Konglin Zhu, *Member, IEEE*, and Lin Zhang, *Member, IEEE*

***Abstract*—Facing the growing demand for Low Earth Orbit (LEO) edge services, in order to address the manageability and economic issues in LEO edge computing, this study introduces an innovative two-timescale optimization approach designed to dynamically optimize satellite access selection, user request dispatching, and service replica placement. Integrating both online and offline optimizations, our method adapts to real-time fluctuations in user demand and satellite resources, effectively managing long-term decisions such as service migration and replica placement. We formalize this optimization challenge as a finite-horizon, integer-variable problem, taking into account both switching costs and resource utilization. Through extensive experimentation, our approach is proven to significantly balance performance enhancement and resource efficiency, and we prove the approximation ratio for each time slot the competitive ratio for the long-term cost. Our work contributes to the understanding of multi-timescale optimization in LEO edge computing and provides valuable insights for designing efficient control mechanisms in satellite-based systems.**

***Index Terms*—LEO satellite networks, edge computing, request dispatching, computation offloading.**

I. Introduction

In the rapidly evolving frontier of technological innovation, the assimilation of Low Earth Orbit (LEO) satellites with edge computing services [1], [2] heralds a new era of connectivity. These satellites, in their low-altitude orbits, provide a critical advantage—reduced signal latency, which is pivotal for a host of modern applications that demand real-time data exchange. The deployment of LEO satellites in close proximity to the Earth markedly enhances communication networks, extending the reach of edge services to previously underserved or inaccessible areas, and offering seamless integration with terrestrial networks. The establishment of a space backbone [3], [4] serves as a cornerstone, fortifying the underlying infrastructure. It provides a resilient framework for communication between satellites, further bolstering the deployment of sophisticated network topologies. The integration of the Walker Delta topology, in conjunction with the grid pattern framework [5], [6], significantly contributes to the overall efficiency and resilience of the satellite communication system.

This work was partially supported by the Fundamental Research Funds for the Central Universities (2024ZCJH07), the Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing, and the U.S. National Science Foundation (CNS-2047719 and CNS-2225949). (Corresponding authors: Lei Jiao, Konglin Zhu.)

Siru Chen, Konglin Zhu and Lin Zhang are with the School of Artificial Intelligence, Beijing University of Posts and Telecommunications, Beijing 100876, China (e-mail: chensiru@bupt.edu.cn; klzhu@bupt.edu.cn; zhanglin@bupt.edu.cn).

Lei Jiao is with the Center for Cyber Security and Privacy, University of Oregon, Eugene, OR 97403, USA (e-mail: ljiao2@uoregon.edu).

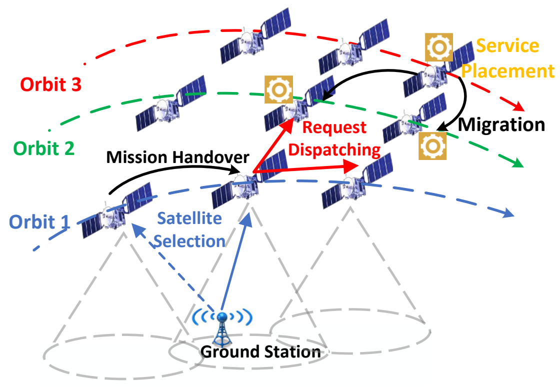


Fig. 1: System scenario

The imperative for service migration within this context can be traced to two fundamental factors. Firstly, the presence of stateful services [7] necessitates the dynamic relocation of services. This is crucial to accommodate the ever-evolving user requirements and ensure the continuous and optimal functionality of services. Secondly, the intrinsic limitation in resources on satellites [8] underscores the need for a strategic approach to managing and distributing services. This involves avoiding resource exhaustion, thus optimizing the overall system performance. With the rapid advancement of edge computing and Internet of Things (IoT) technologies, LEO satellites have emerged as a crucial component for delivering edge services. These satellites operate in close proximity to the Earth's surface, enabling low-latency and high-bandwidth connectivity to a wide range of applications and devices. Leveraging the capabilities of LEO satellites, edge services can be efficiently deployed to serve users in diverse environments.

The fact that a ground station is often covered by multiple satellites simultaneously means that, during specific time periods, the ground station can choose to establish communication with different satellites. Each satellite provides a window of visibility as it flies over the ground station, during which the ground station can establish a connection and perform data transmission. However, due to the orbital dynamics of the satellites, their visibility windows are typically limited [6]. Therefore, the ground station needs to intelligently select the appropriate satellite to establish a connection, ensuring communication reliability and efficiency.

The scenario of a ground station being covered by multiple satellites is prevalent in LEO satellite communication systems, this scenario is illustrated in Fig. 1. This situation can be attributed to factors such as satellite orbit designs, increased satellite density, and requirements of the communication network [9]–[11]. Having a ground station covered by multiple satellites allows for better utilization of satellite resources and a wider service coverage [12]. However, the coverage of multiple satellites also presents some challenges. Firstly, the ground station needs to select the most suitable satellite for communication among the multiple visible satellites. This involves considering factors such as satellite positions, transmission quality, signal strength, and the distance between the ground station and the satellites. Secondly, the ground station may need to switch connections between different satellites to meet user demands and manage system resource allocation. This can result in connection switching overhead and impact on service quality [13], [14].

To respond to constantly changing environments in a cost-efficient manner, we advocate for dynamically controlled LEO edge services with multi-timescale flexibility [15]. This approach enables intertwined decisions regarding access satellite selection for ground stations, user request dispatching from access satellites to service satellites, and service replica placement in satellite constellations. User requests from different regions often fluctuate, necessitating continuous migration of services across satellites. However, in satellite networks with rapidly changing topology, a myopic approach that optimizes decisions in isolation can be counterproductive. While this may seem optimal in the short term, it overlooks the aggregate impact of frequent handovers, leading to service degradation due to interruptions and delays [16]. Excessive migration incurs high performance costs, while conservative approaches may force users to suboptimal satellites, hindering timely adaptation to request dynamics [9], [11], [17]. Moreover, multiple satellites are visible with varying time windows, complicating access satellite selection [18]. Choosing the satellite with the longest visibility may be impractical due to capacity restrictions, and frequent switching can incur handover overhead, damaging perceived service quality. Thus, synchronizing service migration and satellite selection poses a significant challenge, underscoring the need for a strategic approach that balances immediate access with long-term stability and sustainability [19], [20]. The problem involves multiple objective functions, such as minimizing total costs and maximizing service quality. These objectives may conflict, necessitating careful trade-offs and optimization strategies. Handling multi-objective optimization requires developing algorithms to identify optimal or approximately optimal solutions, which is particularly challenging given the previous complexities.

To the best of our knowledge, our work is the first to (1) investigate the three dimensions of satellite selection, request dispatching, and service provisioning jointly in LEO edge computing, and (2) model and solve this problem from a two-timescale online optimization perspective [21].

In this paper, we initiate our exploration by formulating the problem of social cost minimization as a non-convex mixed-integer program spanning the entire time horizon. Our goal is to devise effective solutions to this challenging problem, and to this end, we propose a series of polynomial-time online algorithms. To address the intricacies of the problem, we introduce two innovative algorithms, namely primal-dual-based Algorithm 1 and Algorithm 2. These algorithms are specifically designed to tackle the one-shot problem, involving the strategic placement of offline classifiers, data dispatching, and inference aggregation. It is important to note that our algorithmic approach assumes that all other control decisions have been pre-determined. An additional facet of our approach is the derivation of a parameterized-constant competitive ratio for the total cost concerning the offline optimum. This ratio is established under the assumption that all inputs over the entire time horizon are observed simultaneously beforehand. This implies that our algorithms are not only efficient in their online execution but also yield competitive results when compared to the optimal solution derived from full knowledge of inputs across the entire temporal span.

Through these algorithmic innovations and competitive ratio considerations, our methodology presents a comprehensive and efficient framework for addressing the social cost minimization problem. In the subsequent sections, we delve into the specifics of our algorithms, their theoretical underpinnings, and the empirical evaluations conducted to validate their efficacy in real-world scenarios.

II. Related Work

In this section, we provide an overview of the existing studies related to LEO satellite edge computing and two-timescale optimization approaches for edge services. We categorize the related work into two groups and highlight their limitations compared to our proposed approach.

LEO Satellite Edge Computing:

Research on LEO satellite edge computing has focused on various aspects, including server placement, controller placement, resource placement. Li et al. [10] studied how to efficiently deploy services on satellite edge computing nodes. Zhang et al. [22] reduced transmission costs by integrating multiple access edge computing in the LEO network and use decision variables to schedule requests. Yan et al. [23] conducted research on edge computing server placement based on various system delays. Tang et al. [24], [25] introduced novel strategies and algorithm for controller placement problem and load balancing in satellite networks. Both Pfandzelter and Lai' teams built content delivery networks (CDN) to reduce system latency and bandwidth usage in satellite networks [5], [26].

However, these existing studies have certain limitations when compared to our work. Firstly, they often overlook the interdependencies and interactions among the placement decisions, failing to capture the holistic optimization perspective required for efficient edge service management. Moreover, the nonlinear relationship between handover penalties and service migration costs poses a significant challenge that has not been adequately addressed in the literature [27]. Lastly, most existing approaches focus solely on online control decisions, neglecting the benefits of incorporating multi-timescale flexibility into the optimization framework.

Service Migration:

A significant and noteworthy extension within the realm of related work revolves around the thoughtful consideration of service migration. The exploration of this aspect involves an in-depth analysis of service migration, a process that dynamically relocates services to adapt to evolving user demands,

TABLE I: Comparison of Existing Work on LEO Edge Optimization to This Work

Reference	Mechanism	Problem	Online / Offline	Optimization Objective	Constraints	Performance Guarantees
[10]	Heuristic	Convex, Linear	Offline	Latency Minimization	Instantaneous	No Theoretical Bound
[16]	Single-Timescale	Convex	Online	Energy Efficiency	Steady-state	Lyapunov Stability
[17]	Heuristic	Mixed-integer	Offline	Service Continuity	Resource limits	No Theoretical Bound
[23]	Heuristic	Linear	Offline	Server Placement Cost	Capacity	No Theoretical Bound
[24]	Controller Game	Non-convex	Offline	Load Balancing	Instantaneous	Nash Equilibrium
[28]	Heuristic	Linear	Offline	Link Stability	Orbital dynamics	No Theoretical Bound
[35]	Decoupled Control	Mixed-boolean	Online	Microservice Reliability	Time-averaged	Learning Regret Bound
[36]	Potential Game	Non-convex	Offline	Interference Mitigation	Channel capacity	Nash Equilibrium
[37]	Deep RL	Non-convex	Online	Adaptive MIMO	Nonstationary channels	Convergence Analysis
[38]	Two-Timescale Learning	Mixed-integer	Online	Task Offloading Cost	Hybrid resources	Learning Regret Bound
This Work	**Two-Timescale Optimization**	**Mixed-integer Nonlinear**	**Online**	**Total Cost Minimization (Global/Local Accuracy, Comm/Comp Costs)**	**Long-term, Instantaneous**	**Competitive Ratio**

thereby ensuring the continuous and optimal functionality of the system [17], [28]. Several studies have dedicated efforts to delve into the nuanced aspects of service migration, recognizing its pivotal role in maintaining a responsive and adaptable system [29], [30]. Jin et al. [31] proposed an online learning framework for provisioning edge inference services, which dynamically adapts to fluctuating workloads but focuses on terrestrial edge networks rather than satellite environments. Separately, Liu et al. [32] introduced a dynamic relocation mechanism to enhance service delivery efficiency in distributed systems, providing a foundational strategy for resource adaptation.

However, despite the strides made in understanding service migration, the existing literature in this domain often exhibits certain limitations. One notable gap lies in the lack of a comprehensive understanding of how service migration intricately interacts with other pivotal placement decisions within the system. The interplay between service migration and other decision-making processes, such as server placement, controller placement, and resource allocation, is not always sufficiently explored [33], [34]. Moreover, a critical aspect that the current body of literature may not effectively address pertains to the intricacies associated with multi-timescale optimization. Service migration, when considered in isolation, might not seamlessly align with the broader spectrum of decision-making processes operating at different timescales. The need for a holistic and integrated approach, considering the multi-faceted temporal dimensions of system optimization, remains an area where the existing literature falls short.

Two-Timescale Optimization for Edge:

Existing literature has mainly focused on single-timescale optimization, where decisions are made either in real-time or with long-term planning. These approaches often overlook the need for coordinating control decisions at different timescales to adapt to dynamic environments and achieve optimal performance. Thus, they may fail to effectively balance the trade-off between resource allocation and service quality, Chai et al. [35] use Lyapunov optimization methods to decouple the joint optimization problem of LEO, but there are also some existing research on two-timescale optimization, Shi et al. [36] constructed a novel two-timescale resource management framework for mobile edge computing. Moreover, the existing literature lacks comprehensive solutions that consider both online and offline optimization. Online optimization aims to dynamically adjust control decisions in response to real-time changes in user demands and satellite resources. On the other hand, offline optimization focuses on long-term planning, such as service migration and replica placement, considering factors like service performance, resource utilization, and cost efficiency [15].

The two-timescale division (time frames and slots) is justified by LEO orbital dynamics and service demand characteristics. Large-timescale frames align with satellite orbital periods (90–120 mins) and hourly demand patterns, while small-timescale slots address millisecond-level channel variations and bursty requests. This hierarchy is theoretically grounded in prior works: Chen et al. [37] for topology-aware resource allocation, Lin et al. [38] for adaptive nonstationary optimization, Shi et al. [36] for MEC-based two-timescale co-design, and Han et al. [39] for satellite-terrestrial hybrid networks. Such decoupling enables stable long-term planning and agile short-term adaptation. Our work significantly differs from the existing literature in multiple aspects. Firstly, we explicitly consider the two-timescale nature of the control decisions, encompassing both online and offline optimization. This allows us to capture the dynamic changes in user demands and satellite resources while optimizing long-term decisions such as service migration and replica placement. Secondly, we formulate the optimization problem as a finite-horizon, integer-variable optimization problem, taking into account the switching costs associated with accessing different satellites and managing service replicas. This formulation enables us to find near-optimal solutions that balance performance and resource utilization effectively.

Regarding the advantages of our approach over the two aforementioned papers [40], [41]: Both of these existing papers rely on the Lyapunov optimization framework [42] to design their dual-timescale optimization algorithms. They feature the following aspects: (i) Their problems need to be formulated in the format of optimizing a time-averaged objective subject to time-averaged constraints, while often pushing the length of the time horizon into infinity; (ii) Their algorithms need to be generally in a ***control theory*** style, i.e., constructing virtual queues and stabilizing such virtual queues through drift-plus-penalty functions; (iii) Their theoretical analysis focuses on upper-bounding the ***regret***, i.e., the difference between the objective value incurred by their proposed online approach and the objective value of the offline optimum. In contrast, our proposed approach is never related to Lyapunov optimization, and features the following aspects: (i) Our problem optimizes a cumulative objective over a time horizon of realistic finite length, subject to long-term cumulative constraints for the large timescale and the small timescale, respectively; (ii) Our algorithms are novel, with a unique switching-cost-aware ***online optimization*** structure for solving both timescales; (iii) Our theoretical analysis includes upper-bounding the ***competitive ratio***, i.e., the ratio of the objective value incurred by our proposed online approach over the objective value of the offline optimum. Due to all such stark discrepancies, while the algorithmic approaches in the two reference papers have their own advantages, it is unclear to us how they, with or without adaptions, can be applied to solving our problem; our own proposed dual-timescale approach is yet dedicatedly designed to solve our problem with provable performance guarantees. Our work also uniquely captures and addresses the LEO-specific challenges arising from orbital dynamics and intermittent connectivity, explicitly modeling time-varying satellite visibility and inter-satellite handover penalties and enabling adaptive cost minimization under LEO's unique spatiotemporal conditions.

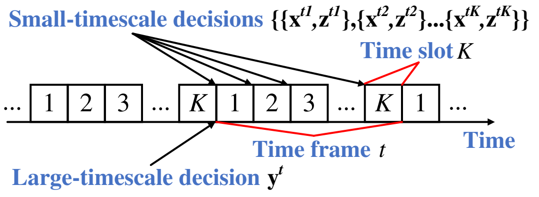

Small-timescale decisions $\{\{\mathbf{x}^{t1},\mathbf{z}^{t1}\},\{\mathbf{x}^{t2},\mathbf{z}^{t2}\}...\{\mathbf{x}^{tK},\mathbf{z}^{tK}\}\}$
Time slot K
... 1 2 3 ... K 1 2 3 ... K 1 ...
Time frame t
Time
Large-timescale decision $\mathbf{y}^t$

Fig. 2: Two-timescale decision making

III. Models and Formulation

A. System Models

In this section, we present our system model and formulate the total cost minimization problem. For the quick reference, we summarize all our major notations in Table II.

LEO Edge Constellation: We consider a LEO constellation of one orbit shell which consists of M orbit plains in total and N satellites evenly spreading in each orbit plain. We use $\mathcal{I} = \{1, 2, ..., MN\}$ to denote the set of all the satellites in the constellation. We also consider the set $\mathcal{J} = \{1, 2, ..., J\}$ of ground stations that are geographically distributed around the globe. To access the services hosted in the LEO edge constellation, a user connects to one of the ground stations via terrestrial networks (e.g., Internet), which further connects to the LEO satellite network. We do not explicitly consider the case where a user directly connects to the satellite network via a dedicated terminal device without ground stations, since this terminal device can be regarded as a "virtual" ground station and then our models and formulations still apply.

Control Decisions: We study the system over a series of time frames $\mathcal{T} = \{1, 2, ..., T\}$, where each time frame further consists of a series of time slots $\mathcal{K} = \{1, 2, ..., K\}$. Time frames correspond to service placement decisions, and time slots correspond to satellite selection and request dispatching decisions. To facilitate subsequent explanations, we define the series of time frames as the large time scale, and the series time slots as the small time scale. That is, we make control decisions as follows: $x_{ij}^{tk} \in \{1, 0\}$, $\forall i \in \mathcal{I}$, $\forall j \in \mathcal{J}$, $\forall k \in \mathcal{K}$, $\forall t \in \mathcal{T}$, denoting whether or not the ground station j selects and connects to the satellite i as the access satellite at the time slot k of the time frame t; $y_i^t \in \{1, 0\}$, $\forall i \in \mathcal{I}$, $\forall t \in \mathcal{T}$, denoting whether or not the satellite i hosts a service replica at the time frame t; $z_{im}^{tk} \geqslant 0$, $\forall i, m \in \mathcal{I}$, $\forall k \in \mathcal{K}$, $\forall t \in \mathcal{T}$, denoting the amount of user workload (e.g., requests) sent via the access satellite i to the service satellite m at the time slot k of the time frame t.

Two-Timescale: Fig. 2 illustrates the hierarchical two-timescale decision framework central to our LEO edge computing model, we divide the time frame t into multiple small time slots k. Each frame spans has the same fixed length and minutes to hours, aligning with the orbital periodicity of LEO satellites and global service demand trends. At this scale, decisions focus on **service placement** (e.g., determining which satellites host service replicas) and **service migration** (e.g., relocating replicas between satellites). These decisions are updated at the beginning of each frame. Each slot spans has the same fixed length and seconds to minutes, corresponding to rapid environmental variations such as bursty user requests,

TABLE II: Notations

Inputs	Meaning
$\mathcal{I}$	Set of all satellites
$\mathcal{J}$	Set of ground stations
$\mathcal{T}$	Set of time frames
$\mathcal{K}$	Set of time slots
z_{im}^{tk}	Amount of user workload sent via i to m at k of t
Q_i	Access capacity of satellite i
C_i	Service capacity of the satellite i
d_{ij}^{tk}	Transmission delay between i and j at k of t
λ_j^{tk}	User requests sent from j at k of t
e_j	Handover penalty for j to change access satellite
l_{im}^{tk}	Network delay between i and m at k of t
a_i^t	Cost of i hosting service replica at t
l_{im}^t	Average network delay for t
R	The total number of service replicas
Decisions	**Meaning**
x_{ij}^{tk}	Whether or not the ground station j selects i as the access satellite at k of t
y_i^t	Whether or not the satellite i hosts a service replica at t
z_{im}^{tk}	Amount of user workload sent via the access satellite i to the service satellite m at k of t

satellite-ground visibility windows, and instantaneous channel fading. At this scale, decisions involve **satellite selection** (e.g., choosing an optimal access satellite for ground stations) and **request dispatching** (e.g., distributing user workloads to service satellites). These decisions are updated at the beginning of each slot.

Satellite Selection: In Fig. 2, at any time slot k of any time frame t, we denote the set of the satellites that fly over the ground station j as $\mathcal{I}_j^{tk}$. Correspondingly, we use $\mathcal{J}_i^{tk}$ to denote the set of the ground stations that are within the connection range of the satellite i at the time slot k of the time frame t. Thus, at k of t, the ground station j needs to choose one and only one satellite in $\mathcal{I}_j^{tk}$ as its access satellite, i.e., the first-hop satellite to connect to in order to access the satellite network. We use Q_i ($\forall i$) to denote the access capacity of the satellite i, i.e., the number of ground stations it can accept at most at any given time, which can be in terms of, for example, the number of beam signals equipped on the satellite. We also use d_{ij}^{tk} to represent the unit transmission cost between the ground station j and the satellite i at the time slot k of the time frame t. We further use e_j to denote the handover penalty for the ground station j to change the access satellite across time slots. Based on these notations, we can represent the user requests' total delay from ground stations to access satellites as $\sum_t\sum_k\sum_j\sum_{i\in\mathcal{I}_j^{tk}} d_{ij}^{tk}\lambda_j^{tk}x_{ij}^{tk}$, where λ_j^{tk} is described as below, and represent all the handover penalty of the system across time slots as

$$C_H^{tk}(\mathbf{x}^{tk},\mathbf{x}^{tk-1}) = \textstyle\sum_j\sum_{i\in\mathcal{I}_j^{tk}} e_j \max\{x_{ij}^{tk}-x_{ij}^{tk-1},0\}.$$

Request Dispatching: At any time slot k of any time frame t, we denote the number of user requests received at the ground station j as λ_j^{tk}. The user requests will firstly reach the access satellite of the ground station j and then be sent to those satellites with service replicas, i.e., service satellites, for processing. We denote by l_{im}^{tk} the network delay between satellites i and m at the time slot k of time frame t, and by C_i the service capacity of satellite i. Note that, depending on the request dispatching decisions, the requests from a single access satellite may be split and sent to multiple service satellites. We can then represent the user requests' total delay within the satellite network as $\sum_t\sum_k\sum_i\sum_m l_{im}^{tk}z_{im}^{tk}$.

Service Placement: We suppose there exist R service replicas of one service at any time in the system, where each service replica is placed at a different satellite. We consider only one service in this paper, and our work can be easily extended to multiple services. We use a_i^t to denote the cost of hosting the service replica at the satellite i at the time frame t. In fact, given any existing satellite-network routing algorithm, we can use l_{im}^t to denote the number of hops, or the average network delay for t, or the instant network delay as the service migration occurs during t, for migrating a service replica from the satellite i to the satellite m at the time frame t in our satellite network, and use w_{im}^t to denote the decision variable of whether or not to move the service replica from the satellite i to the satellite m at t. Then, we can define the service migration cost:

$$\begin{aligned} C_M^t(\mathbf{y}^t,\mathbf{y}^{t-1}) = \ & \min \textstyle\sum_i\sum_m l_{im}^t w_{im}^t \\ & s.t.\ \textstyle\sum_m w_{im}^t \leqslant y_i^{t-1}R, \forall i, && (1a)\\ & \textstyle\sum_i w_{im}^t = y_m^t, \forall m, && (1b)\\ & w_{im}^t \in \{1,0\}, \forall i, \forall m. \end{aligned}$$

We denote the optimal solution to C_M^t as $\widehat{\mathbf{w}}^t$. Note that even though we use a standard linear program solver to solve C_M^t in polynomial time, our optimal solutions are automatically integers because the coefficient matrix of the constraints in C_M^t is a "totally unimodular matrix" [43].

In the above, (1a) ensures that any service replica at $t-1$ can be replicated to up to R service replicas (or satellites) at t; (1b) ensures that any service replica at t can only be replicated from an existing service replica at $t-1$. That is, the service migration cost is the minimum cost needed to accomplish all the service replica movements within the satellite network from the time frame $t-1$ to the time frame t. Based on these, we represent the service operational cost and the service migration cost as $\sum_t\sum_i a_i^t y_i^t + \sum_t C_M^t(\mathbf{y}^t,\mathbf{y}^{t-1})$.

Regarding service migration costs, we acknowledge that satellites have greater computing and communication capabilities compared to Internet of Things (IoT) devices and small base stations. Yet, it is important to note that, compared to typical ground-based cloud or edge computing infrastructures, satellite resources are still restrictive due to size, weight, and power constraints, and thus the cost of service migration in satellite edge computing is not negligible and is still a serious concern. Frequent migrations can lead to significant performance overhead, including increased latency, energy consumption, and potential service disruptions, which are critical in a highly dynamic and resource-constrained environment like satellites. To further illustrate the impact of service migration costs, we provide two practical use cases as follows.

Global IoT Data Collection: In LEO edge computing, satellites are often used for global IoT data collection, such as agricultural sensor networks that require periodic data transmission to ground stations for processing. In this scenario, frequent service migration can lead to increased data transmission delays, impacting real-time decision making. For example, if agricultural sensors need to transmit data to a satellite, frequent migration of the processing task between satellites can cause delays in data analysis, which may hinder timely decisions on irrigation or pest control. This is supported by [10].

Emergency Rescue and Disaster Response: In emergency scenarios, LEO satellites are tasked with rapidly processing and analyzing large volumes of data to support rescue operations. For instance, during a natural disaster, satellites may be used to process real-time data from ground sensors to identify affected areas and coordinate rescue efforts. Frequent service migration in such high-stakes environments can lead to task interruptions and delays in data processing, severely impacting rescue efficiency. This is supported by [16].

B. Problem Formulation and Algorithmic Challenges

Total Cost Minimization:

Having the models as the above, we can then formulate the optimization problem of minimizing the long-term total cost of the system as follows:

$$\begin{aligned}
\mathbb{P}_0: \ \min & \sum_t\sum_k\sum_j\sum_{i\in\mathcal{I}_j^{tk}} d_{ij}^{tk}\lambda_j^{tk}x_{ij}^{tk} + \sum_t\sum_k C_H^{tk}(\mathbf{x}^{tk},\mathbf{x}^{tk-1}) \\
& + \sum_t\sum_k\sum_i\sum_m l_{im}^{tk}z_{im}^{tk} + \sum_t\sum_i a_i^t y_i^t + \sum_t C_M^t(\mathbf{y}^t,\mathbf{y}^{t-1}) \\
s.t. \ & \textstyle\sum_{i\in\mathcal{I}_j^{tk}} x_{ij}^{tk} = 1, \quad \forall j,\forall k,\forall t, & \text{(2a)} \\
& \textstyle\sum_{j\in\mathcal{J}_i^{tk}} x_{ij}^{tk} \leqslant Q_i, \quad \forall i,\forall k,\forall t, & \text{(2b)} \\
& \textstyle\sum_m z_{im}^{tk} \geqslant \sum_{j\in\mathcal{J}_i^{tk}} \lambda_j^{tk}x_{ij}^{tk}, \ \forall i,\forall k,\forall t, & \text{(2c)} \\
& \textstyle\sum_i z_{im}^{tk} \leqslant y_m^t C_m, \quad \forall m,\forall k,\forall t, & \text{(2d)} \\
& \textstyle\sum_i y_i^t = R, \quad \forall t, & \text{(2e)} \\
& x_{ij}^{tk}, y_i^t \in \{0,1\}, z_{im}^{tk} \geqslant 0, \ \forall i,\forall m,\forall j,\forall k,\forall t. & \text{(2f)}
\end{aligned}$$

Constraint (2a) ensures that every ground station selects one satellite only as the access satellite at every time slot. Constraint (2b) ensures that the access capacity of every satellite is respected at every time slot. Constraint (2c) ensures that the user requests received at every access satellite are fully dispatched at every time slot. Constraint (2d) ensures that the service capacity of every satellite is respected at every time slot. Constraint (2e) ensures that the total number of service replicas that exist in the satellite network is always as specified at every time frame. Constraint (2f) just enforces the domains of all the decision variables.

Algorithmic Challenges:

Solving our total cost minimization problem in an online manner confronts critical challenges.

First, the handover penalty $\sum_j\sum_{i\in\mathcal{I}_j^{tk}} e_j \max\{x_{ij}^{tk} - x_{ij}^{tk-1}, 0\}$ couples every time slot $k-1$ and its next time slot k, it is important to make the decision of x_{ij}^{tk-1} at $k-1$ for minimizing the long-term cost because any decision of x_{ij}^{tk-1} will potentially impact the handover penalty between $k-1$ and k, however, we can not know x_{ij}^{tk} at $k-1$, our x_{ij}^{tk-1} at $k-1$ can therefore hardly optimize $\sum_j\sum_{i\in\mathcal{I}_j^{tk}} e_j \max\{x_{ij}^{tk} - x_{ij}^{tk-1}, 0\}$. Attempting to solve each individual component of a series of single-round problems optimally or frequently switching the access satellite for each time slot can lead to an accumulation of handover penalties over time. This occurs because each switch may involve significant overhead in terms of communication interruptions, additional signal processing, and the re-establishment of connections, all of which contribute to reduced efficiency and increased latency.

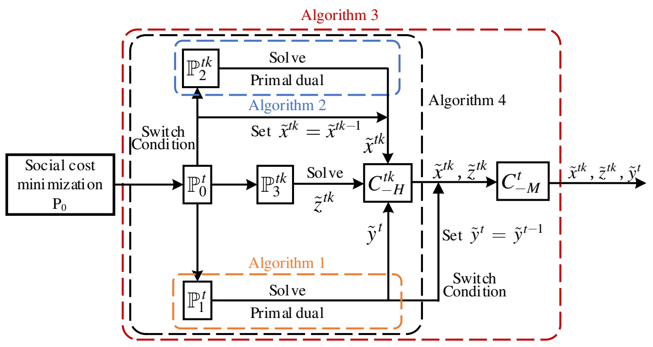


Fig. 3: Algorithm design

Second, the total cost minimization problem is NP-hard. It is difficult to achieve in an offline situation, and it will be far more difficult in an online setting. Even without replacement cost, our problem is made up of a succession of single-round problems, each of which can be reduced from the NP-hard weighted set cover problem. To address our NP-hard, we desire polynomial-time online approximation algorithms.

Third, the problem involves multiple constraints related to ground station and satellite capacity limitations, user request dispatching requirements, and other system-specific constraints. These constraints can be nonlinear, coupled, or have complex structures. Effectively handling these constraints to ensure compliance with system requirements is a significant challenge [44]. This is not easy, especially considering the existence of the previous two challenges.

Our proposed algorithms provide a parameterized-constant competitive ratio, ensuring that the gap between the approximate solution and the theoretical optimal solution is bounded and stable under the given assumptions. This is validated by both theoretical analysis and experimental results.

IV. Algorithm Design

We design a polynomial-time approximation algorithm to solve the aforementioned problems while simultaneously solving the total cost minimization problem and determining satellite selection, satellite migration, and mission scheduling problems. We rigorously prove the approximation ratio, the truthfulness, and the individual rationality as the performance guarantees for our algorithm. We then determine content placements strategically in an online manner in the next section. Our entire approach can be structured as in Fig. 3.

A. Primal-Dual Algorithm

We choose to split the single-shot problem $\mathbb{P}_0^t$ into multiple subproblems for analysis. We design Algorithm 1 to simultaneously construct integral feasible solutions to the primal problem $\mathbb{P}_1^t$ and feasible solutions to the dual problem $\mathbb{D}_1^t$. we define the following problems:

$$\mathbb{P}_1^t: \ \min f^t(\mathbf{y}^t) = \textstyle\sum_i a_i^t y_i^t \qquad (3)$$

Algorithm 1: Primal-Dual Algorithm for $\mathbb{P}_1^t$, $\forall t$

```
1  Input $a_i, C_i, L_k, R$
2  Initialize $\delta_k$
3  for $k \in \mathcal{K}$ do
4      $i^+ = argmin_{i\in\mathcal{I}}(\delta_k + a_i)$
5      $i^* = i^+$
6      $\delta_k = \delta_k(1 + \frac{1}{L_k}) + \frac{a_{i^*}}{L_k \xi}$
7      $R = -min(\delta_k + a_{i^*})$
8      $y_i^* = 1$
9  end
10 Output $\hat{\mathbf{y}}^t$
```

$$s.t.\ \textstyle\sum_i y_i^t C_i \geqslant \sum_j \lambda_j^{tk}, \forall k, \quad (3a)$$
$$\textstyle\sum_i y_i^t = R, \quad (3b)$$
$$y_i^t \in \{0,1\}, \forall i. \quad (3c)$$

By relaxing the variables y_i into real domains and introducing dual variables δ_k, R for (4a) and (4b), respectively, to solve the problem $\mathbb{P}_1$ in an online manner, we write the Lagrange dual problem $\mathbb{D}_1$ of the problem $\mathbb{P}_1$ as follows:

$$\mathbb{D}_1^t:\ \max\ f^t(\mathbf{y}^t) = -\textstyle\sum_k L_k\delta_k - R \quad (4)$$
$$s.t.\ a_i + \delta_k + R \geqslant 0, \forall k, \quad (4a)$$
$$var.\ \delta_k \leqslant 0. \quad (4b)$$

We set $\sum_j \lambda_j^{tk} = L_k$ and then design a primal-dual-based online algorithm, we design Algorithm 1 to simultaneously construct integral feasible solutions to the primal problem (3) and feasible solutions to the dual problem (4).

Algorithm 1 is devised to concurrently develop integral feasible solutions for the primary problem (3) and viable solutions for its counterpart dual problem (4). The conceptual framework of the primal-dual approach involves progressively increasing the dual variables until each dual constraint is precisely met (i.e., a constraint of the nature $ax \leqslant b$ reaches a state of tightness when $ax = b$). At this juncture, the corresponding primal variable can be adjusted to a non-zero figure, ensuring the primal and dual solutions maintain the complementarity required by the Karush-Kuhn-Tucker (KKT) optimality conditions [45]. The algorithm is applied iteratively at each time step t; therefore, the notation t is excluded for brevity in the algorithm's description.

Algorithm 1 is structured as a primal-dual method that iteratively seeks feasible solutions for a set of constraints represented by K. It begins by initializing dual variables, δ_k, which are crucial for the dual aspect of the problem. For each constraint k, the algorithm selects an action i that minimizes the sum of the current value of δ_k and a cost factor a_i, reflecting the action's relative expense or penalty. The selected action's index is denoted by i^*. Subsequently, scaling up δ_k in proportion to the tightness of constraint k and the cost of the chosen action i^*. This process essentially evaluates the dual problem's constraints, gradually increasing their values until they are tight, meaning the inequality turns into an equality. When a constraint's requirements are deemed adequately met—usually signaled by δ_k reaching a certain threshold—the corresponding primal variable y_i^* is set to 1, marking the selection of the action for constraint k in this iteration. After iterating through all constraints in K, the algorithm concludes by outputting the feasible solution $\hat{\mathbf{y}}^t$ from the primal problem $\mathbb{P}_1$, signifying the decisions or actions determined to be feasible at time t. This output reflects a balance between meeting immediate system constraints and working towards long-term objectives, accounting for the dynamic interplay between the primal and dual aspects of the problem. The update of δ_k is carefully designed for achieving low additive loss in approximation ratio, as in Line 6, where $\xi = \max_{k\in\mathcal{K}}\{L_k\}$. Lines 7 and 8 update the dual variable R and the primal variables y_i^*, respectively.

Algorithm 2: Primal-Dual Algorithm for $\mathbb{P}_2^{tk}$, $\forall k, \forall t$

```
1  Input $d_{ij}, \lambda_j, L_k, y_i$
2  Initialize $\delta_k$
3  for $j \in \mathcal{J}$ do
4      $i^+ = argmin_{i\in\mathcal{I}}(d_{ij}\lambda_j + n_i)$
5      $i^* = i^+$
6      $n_{i^*} = n_{i^*}(1 + \frac{1}{Q_i}) + \frac{d_{i^*j}\lambda_j}{Q_i P}$
7      $\mu_j = -(n_{i^*} + d_{i^*j}\lambda_j)$
8      $x_{i^*j} = 1$
9  end
10 Output $\hat{\mathbf{x}}^{tk}$
```

Similar to problem $\mathbb{P}_1$, where we formulate the dual problem $\mathbb{D}_1$ to enable an online solution approach, we also require a feasible solution for $\mathbb{P}_2$.

$$\mathbb{P}_2^{tk}:\ \min\ g^{tk}(\mathbf{x}^{tk}) = \textstyle\sum_j \sum_{i\in\mathcal{I}_j^{tk}} d_{ij}^{tk}\lambda_j^{tk}x_{ij}^{tk} \quad (5)$$
$$s.t.\ \textstyle\sum_{i\in\mathcal{I}_j^{tk}} x_{ij}^{tk} = 1, \quad \forall j, \quad (5a)$$
$$\textstyle\sum_{j\in\mathcal{J}_i^{tk}} x_{ij}^{tk} \leqslant Q_i, \forall i, \quad (5b)$$
$$x_{ij}^{tk} \in \{0,1\}, \qquad \forall i, \forall j. \quad (5c)$$

Given that $\mathbb{P}_2$ introduces an additional set of constraints and objectives, it necessitates the construction of a tailored dual problem, denoted as $\mathbb{D}_2$. By relaxing the variables x_{ij} into real domains and introducing dual variables n_i, μ_j for (6a) and (6b), respectively, we write the Lagrange dual problem as

$$\mathbb{D}_2^t:\ \max\ f^t(\mathbf{y}^t) = -\textstyle\sum_i D_i n_i - \sum_j \mu_j \quad (6)$$
$$s.t.\ d_{ij}\lambda_j + \mu_j + n_i \geqslant 0, \forall i, j, \quad (6a)$$
$$var.\ n_i \geqslant 0, \mu_j \in R, . \quad (6b)$$

Following the framework established by Algorithm 1, we devise Algorithm 2, a similar primal-dual-based online algorithm, aimed at constructing integral feasible solutions for the primal problem (5) as well as viable solutions for the dual problem (6).

The algorithm concludes by outputing the feasible solution $\hat{\mathbf{x}}^{tk}$ from the primal problem $\mathbb{P}_2$, signifying the decisions or actions determined to be feasible at k of t. The update of n_i is carefully designed for achieving low additive loss in approximation ratio, as in Line 6, where $P = \max_{i\in\mathcal{I}_j^{tk}}\{D_i\}$. Lines 7 and 8 update the dual variable μ_j and the primal variables x_{ij}^{tk}, respectively.

For **Algorithm 1**, the time complexity is $O(K \cdot |\mathcal{I}|)$, where K is the number of time slots and $|\mathcal{I}|$ is the total number of satellites. **Algorithm 2** exhibits $O(|\mathcal{J}| \cdot \overline{|\mathcal{I}_j^{tk}|})$ complexity, where $|\mathcal{J}|$ is the number of ground stations and and $\overline{|\mathcal{I}_j^{tk}|}$ is the average visible satellites per station per time frame, computed as $\frac{1}{K}\sum_{k=1}^{K} |\mathcal{I}_j^{tk}|$ to reflect typical orbital visibility patterns.

B. Two-Timescale Optimization Online Algorithm

We split the objective function $\mathbb{P}_0^t$ into two components $C^t = C^t_{-M} + C^t_M$ in time frame scale, where C^t_M is the service placement cost and

$$\begin{aligned} C^t_{-M}(\mathbf{x}^t, \mathbf{z}^t, \mathbf{y}^t) = \textstyle\sum_k \left(C^{tk}_{-H}(\mathbf{x}^{tk}, \mathbf{z}^{tk}) + C^{tk}_H(\mathbf{x}^{tk}, \mathbf{x}^{tk-1}) \right) \\ + \textstyle\sum_i a_i^t y_i^t \end{aligned}$$

is other costs that need to be analyzed at the time slot scale, where

$$C^{tk}_{-H}(\mathbf{x}^{tk}, \mathbf{z}^{tk}) = \textstyle\sum_j \sum_{i \in \mathcal{I}_j^{tk}} d_{ij}^{tk} \lambda_j^{tk} x_{ij}^{tk} + \sum_i \sum_m l_{im}^{tk} z_{im}^{tk}$$

corresponds to request dispatching decisions. Now, through C^{tk}_{-H} and the corresponding constraints, we define the following problem:

$$\begin{aligned} \mathbb{P}_3^{tk}: \ & \min\ h^{tk}(\mathbf{z}^{tk}) = \textstyle\sum_i \sum_m l_{im}^{tk} z_{im}^{tk} \\ & s.t.\ \textstyle\sum_m z_{im}^{tk} \geqslant \sum_{j \in \mathcal{J}_i^{tk}} \lambda_j^{tk} x_{ij}^{tk},\ \forall i, \\ & \quad \textstyle\sum_i z_{im}^{tk} \leqslant y_m^t C_m,\ \forall m, \\ & \quad z_{im}^{tk} \geqslant 0,\ \forall i, \forall m. \end{aligned}$$

We denote the optimal solution to $\mathbb{P}_3^{tk}$ as $\hat{\mathbf{z}}^{tk}$. Note that even though we use a standard linear program solver to solve $\mathbb{P}_3^{tk}$ in polynomial time, our optimal solutions are automatically integers because the coefficient matrix of the constraints in $\mathbb{P}_3^{tk}$ is a "totally unimodular matrix" [43], the other decision variables x^{tk}, y^t and w^t are all feasible.

$$\begin{aligned} \mathbb{P}_4^{tk}: \ & \min\ C^{tk}_{-H}(\mathbf{x}^{tk}, \mathbf{z}^{tk}) \\ & s.t.\ \text{(2a), (2b), (2c)}, \\ & \quad \textstyle\sum_i z_{im}^{tk} \leqslant y_m^t C_m, \quad \forall m, \\ & \quad x_{ij}^{tk} \in \{0,1\}, z_{im}^{tk} \geqslant 0, \forall i, \forall m, \forall j. \end{aligned}$$

We also define the following problem that we do not need to solve but use in our performance analysis.

Next, we designed Algorithm 3 and Algorithm 4, two online algorithms that balance the different parts of the original problem $\mathbb{P}_0^t$ split at different time scales. Firstly, we implement Algorithm 4 at a smaller time scale, time slot k , then utilize the output of Algorithm 4 at a larger time scale, time frame t, to implement Algorithm 3. This hierarchical approach ensures that the real-time adjustments made by Algorithm 4 are effectively integrated and scaled up through Algorithm 3 to address the broader aspects and constraints of the original problem over longer periods. This design not only enhances the adaptability and efficiency of our solution in dynamically changing environments but also ensures a consistent and cohesive strategy that aligns short-term decisions with long-term objectives.

Algorithm 4 operates within each individual time frame t and makes decisions at smaller time scales, each represented by k. It begins by initializing variables and getting the first solution from Algorithm 2. Then it iteratively updates this solution within the time frame as follows: The algorithm first solves a sub-problem $\mathbb{P}_3^{tk}$ using the output from Algorithms 1 and 2. In each subsequent time slot k, it checks if the current solutions are still feasible within the defined thresholds. If not feasible, it reverts to the last feasible solutions and solves $\mathbb{P}_3^{tk}$ again to update the solutions. The algorithm progresses through the time frame until it finalizes the solution for that time frame.

Algorithm 3: Online Algorithm on Large timescale

```
1  Initialize $\hat{t} = 1$, $\tilde{\mathbf{y}}^0$, $0 < \beta_1 \leqslant 1$;
2  Get $\hat{\mathbf{y}}^1$ by **Algorithm 1**, and set $\tilde{\mathbf{y}}^1 = \hat{\mathbf{y}}^1$;
3  Given $\tilde{\mathbf{y}}^1$, get $\{\tilde{\mathbf{x}}^{1k}, \tilde{\mathbf{z}}^{1k}, \forall k\}$ by **Algorithm 4**;
4  for $t = 2, 3, ..., T$ do
5      if $C_M^{\hat{t}}(\tilde{\mathbf{y}}^{\hat{t}}, \tilde{\mathbf{y}}^{\hat{t}-1}) \leqslant \beta_1 \sum_{\tau=\hat{t}}^{t-1} C_{-M}^{\tau}(\tilde{\mathbf{x}}^t, \tilde{\mathbf{z}}^t, \tilde{\mathbf{y}}^t)$, or $\exists k$,
         $\mathbb{P}_{tk}^3$ infeasible given $\tilde{\mathbf{y}}^{t-1}$ then
6          Get $\hat{\mathbf{y}}^t$ by **Algorithm 1**, and set $\tilde{\mathbf{y}}^t = \hat{\mathbf{y}}^t$;
7          Given $\tilde{\mathbf{y}}^t$, get $\{\tilde{\mathbf{x}}^{tk}, \tilde{\mathbf{z}}^{tk}, \forall k\}$ by **Algorithm 4**;
8          if $\tilde{\mathbf{y}}^t \neq \tilde{\mathbf{y}}^{t-1}$ then
9              Set $\hat{t} = t$;
10         end
11     end
12     if $\hat{t} < t$ then
13         Set $\tilde{\mathbf{y}}^t = \tilde{\mathbf{y}}^{t-1}$;
14         Given $\tilde{\mathbf{y}}^t$, get $\{\tilde{\mathbf{x}}^{tk}, \tilde{\mathbf{z}}^{tk}, \forall k\}$ by **Algorithm 4**;
15     end
16 end
```

Algorithm 4: Online Algorithm for Each Time Frame t

```
1  Initialize $\hat{k} = 1$, $\tilde{\mathbf{x}}^{t0} = \tilde{\mathbf{x}}^{t-1K}$, $0 < \beta_2 \leqslant 1$;
2  Get $\hat{\mathbf{x}}^{t1}$ by **Algorithm 2**, and set $\tilde{\mathbf{x}}^{t1} = \hat{\mathbf{x}}^{t1}$;
3  Given $\tilde{\mathbf{x}}^{t1}$, get $\tilde{\mathbf{z}}^{t1}$ by solving $\mathbb{P}_3^{t1}$;
4  for $k = 2, 3, ..., K$ do
5      if $C_H^{t\hat{k}}(\tilde{\mathbf{x}}^{t\hat{k}}, \tilde{\mathbf{x}}^{t\hat{k}-1}) \leqslant \beta_2 \sum_{\tau=\hat{k}}^{k-1} C_{-H}^{t\tau}(\tilde{\mathbf{x}}^{tk}, \tilde{\mathbf{z}}^{tk})$, or $\mathbb{P}_{tk}^3$
         infeasible given $\tilde{\mathbf{x}}^{tk-1}$ then
6          Get $\hat{\mathbf{x}}^{tk}$ by **Algorithm 2**, and set $\tilde{\mathbf{x}}^{tk} = \hat{\mathbf{x}}^{tk}$;
7          Given $\tilde{\mathbf{x}}^{tk}$, get $\tilde{\mathbf{z}}^{tk}$ by solving $\mathbb{P}_3^{tk}$;
8          if $\tilde{\mathbf{x}}^{tk} \neq \tilde{\mathbf{x}}^{tk-1}$ then
9              Set $\hat{k} = k$;
10         end
11     end
12     if $\hat{k} < k$ then
13         Set $\tilde{\mathbf{x}}^{tk} = \tilde{\mathbf{x}}^{tk-1}$;
14         Given $\tilde{\mathbf{x}}^{tk}$, get $\tilde{\mathbf{z}}^{tk}$ by solving $\mathbb{P}_3^{tk}$;
15     end
16 end
```

We denote the optimal solution to $\mathbb{P}_3^{tk}$ as $\tilde{\mathbf{z}}^{tk}$. Based on this, we first design Algorithm 4, an online algorithm which balances C^{tk}_{-H} and C^{tk}_H dynamically in the small time scale. Our main strategy is to delay rotating the locations of content caches, i.e., to prevent frequent changes to access satellites, until either the current cache locations cause $\mathbb{P}_1^{tk}$ or $\mathbb{P}_2^{tk}$ to become unfeasible for request dispatching, or until the cumulative non-replacement cost times a pre-specified constant (i.e., β_2) exceeds the latest replacement cost. There is also a special case where when our ground station is out of the communication range of the access satellite, it is necessary to forcibly switch to another access satellite. This situation is illustrated by Line 5 of the algorithm, where we denote the time slot of changing content caching locations as $\hat{k}$. If ground station needs to change access satellite, we solve $\mathbb{P}_2^{tk}$ to get the new locations (Line 6) and given such new locations, Otherwise, if it turns out that we do not change the access satellite (Line 12), and we keep the access mission at current locations (Line 13) and given such locations, invoking get all the other control decisions [45], [46].

Then we take $\widetilde{\mathbf{x}}^{tk}$ and $\widetilde{\mathbf{z}}^{tk}$ by Algorithm 4 as inputs for Algorithm 3, an online algorithm which balances C^t_{-M} and C^t_M dynamically in the large time scale t. This key idea is to delay changing the service satellite group and avoid unnecessary migration costs for service replicas.

To put it in one sentence, our algorithmic framework works as follows to address the complex coupling of the optimization variables $\mathbf{x}^{tk}$, $\forall k$, $\forall t$ and $\mathbf{z}^{tk}$, $\forall k$, $\forall t$, which are for each time slot k in each time frame t, and also $\mathbf{y}^t$, $\forall t$, which are for each time frame t: As time goes to the beginning of the time frame t, we determine the value of $\mathbf{y}^t$, denoted as $\widetilde{\mathbf{y}}^t$; then, given such $\widetilde{\mathbf{y}}^t$, for each time slot k sequentially within the current time frame t, we determine the values of $\mathbf{x}^{tk}$ and $\mathbf{z}^{tk}$, denoted as $\widetilde{\mathbf{x}}^{tk}$ and $\widetilde{\mathbf{z}}^{tk}$, respectively. The key here is that our Algorithm 3 determines $\widetilde{\mathbf{y}}^t$ through balancing its switching cost (i.e., $C^t_M(\cdot,\cdot)$) and non-switching cost; and Algorithm 4 determines $\widetilde{\mathbf{x}}^{tk}$ through balancing its switching cost (i.e., $C^{tk}_H(\cdot,\cdot)$) and non-switching cost, and determines $\widetilde{\mathbf{z}}^{tk}$ given $\widetilde{\mathbf{x}}^{tk}$. These "balancing" operations are controlled by the parameters β_1 and β_2, respectively, and such operations require "initial" or "tentative" decisions for $\mathbf{y}^t$ and $\mathbf{x}^{tk}$ without considering any switching cost, denoted as $\widehat{\mathbf{y}}^t$ and $\widehat{\mathbf{x}}^{tk}$, respectively, which are then provided by our Algorithms 1 and 2, respectively. This is also how our four algorithms work jointly together in an online manner.

In each time slot, **Algorithm 4** performs a polynomial number of operations to optimize satellite selection and request dispatching. The complexity is $O(K \cdot |\mathcal{I}| \cdot |\mathcal{J}|)$. At the beginning of each time frame, **Algorithm 3** solves a linear program to optimize service placement driven by linear programming for R service replicas. The complexity is $O(R \cdot |\mathcal{J}|)$. Combined with empirical results showing stable real-time performance under varying network sizes, the polynomial-time complexity at both scales confirms the framework's practicality in dynamic LEO environments. The hierarchical design ensures responsiveness to immediate changes while maintaining long-term resource stability.

V. Performance Analysis

We recap existing notations and also introduce some new notations as follows.

- $\{\{\widetilde{\mathbf{x}}^{tk}, \widetilde{\mathbf{z}}^{tk}, \forall k\}, \widetilde{\mathbf{y}}^t, \forall t\}$, denoting the solutions produced by our proposed algorithms;
- $\{\{\bar{\mathbf{x}}^{tk}, \bar{\mathbf{z}}^{tk}, \forall k\}, \bar{\mathbf{y}}^t, \forall t\}$, denoting the offline optimal solutions to the problem $\mathbb{P}_0$;
- $\check{\mathbf{y}}^t$, denoting the optimal solution to the problem $\mathbb{P}^t_1$ for the time frame t;
- $\check{\mathbf{x}}^{tk}$, denoting the optimal solution to the problem $\mathbb{P}^{tk}_2$ for the time slot k of the time frame t;
- $\{\mathbf{x}^{*tk}, \mathbf{z}^{*tk}, \forall k\}$, denoting the optimal solutions to the problem $\mathbb{P}^{tk}_4$ at the time frame t, given $\widetilde{\mathbf{y}}^t$.

Also, note that $\widetilde{\mathbf{z}}^{tk}$ is the optimal solution to the problem $\mathbb{P}^{tk}_3$ for the time slot k of the time frame t, given $\widetilde{\mathbf{y}}^t$ and $\widetilde{\mathbf{x}}^{tk}$.

Theorem 1. Approximation Ratios. By Algorithms 1 and 2, we have $f^t(\widehat{\mathbf{y}}^t) \leqslant \theta_1 f^t(\check{\mathbf{y}}^t)$, $\forall t$ and $g^{tk}(\widehat{\mathbf{x}}^{tk}) \leqslant \theta_2 g^{tk}(\check{\mathbf{x}}^{tk})$, $\forall k$, $\forall t$, respectively, where $\theta_1 = \frac{\xi}{\xi-1}$ and $\theta_2 = \frac{P}{P-1}$ are constants.

Proof. Let ΔP_1 and ΔD_1 denote the increment of the objective function P^t_1 and D^t_1, respectively, $\Delta P_1 = a_{i*}$, $\Delta D_1 = -(\Delta R + L_{k*}\Delta\delta_{k*})$, where $\Delta\delta_{k*} = \frac{\delta_{k*}}{L_k} + \frac{a_{i*}}{L_k\xi}$ and $\Delta R = -(\delta_{k*} + a_{i*})$ stands for the increment in δ_{k*} and R. Thus, we have $-(\Delta R + L_{k*}\Delta\delta_{k*}) = -L_k(\frac{\delta_{k*}}{L_k} + \frac{a_{i*}}{L_k\xi}) + (\delta_{k*} + a_{i*}) = -\delta_{k*} - \frac{a_{i*}}{\xi} + -\delta_{k*} + a_{i*} = \frac{\xi-1}{\xi}a_{i*} = \frac{\xi-1}{\xi}\Delta P_1$. Due to $P^K_1 = \sum_k(P^k_1 - P^{k-1}_1) = \frac{\xi}{\xi-1}\sum_k(D^k_1 - D^{k-1}_1) = \frac{\xi}{\xi-1}(D^K_1 - D^0_1) \leqslant \frac{\xi}{\xi-1}D^K_1$, according to the nature of the original duality, when two Linear programming are dual to each other, any objective value of the Linear programming seeking the maximum value will not be greater than any objective value of the Linear programming seeking the minimum value, so $P^K_1 \leqslant \frac{\xi}{\xi-1}D^K_1 \leqslant \frac{\xi}{\xi-1}P^{k*}_1$, P^{k*}_1 refers to the optimal objective function value of the primal problem P_1, we obtain $\theta_1 = \frac{\xi}{\xi-1}$.

Let ΔP_2 and ΔD_2 denote the increment of the objective function P^{tk}_2 and D^{tk}_2, respectively, $\Delta P_2 = d_{i*j}\lambda_j$, $\Delta D_2 = -(\mu_{j*} + Q_{i*}\Delta n_{i*})$, where $\Delta n_{i*} = \frac{n_{i*}}{Q_i} + \frac{d_{i*j}\lambda_j}{Q_iP}$ stands for the increment in n_{i*}. We have $\Delta D_2 = -(\mu_{j*} + Q_{i*}n_{i*}) = -\mu_{j*} - Q_{i*}(\frac{n_{i*}}{Q_i} + \frac{d_{i*j}\lambda_j}{Q_iP}) = n_{i*} + d_{i*j}\lambda_j - Q_{i*}(\frac{n_{i*}}{Q_i} + \frac{d_{i*j}\lambda_j}{Q_iP} = (1 - \frac{1}{P})d_{i*j}\lambda_j = \frac{P-1}{P}\Delta P_2$. Due to $P^J_2 = \sum_j(P^j_2 - P^{j-1}_2) = \frac{P}{P-1}\sum_j(D^j_2 - D^{j-1}_2) = \frac{P}{P-1}(D^J_2 - D^0_2) \leqslant \frac{P}{P-1}D^J_2 \leqslant \frac{P}{P-1}P^*_2$, P^{k*}_2 refers to the optimal objective function value of the primal problem P_2, we obtain $\theta_2 = \frac{P}{P-1}$ □

Theorem 2. Competitive Ratio. The The social cost achieved via Algorithm 3 is at most α times the offline optimal social cost. β_1 and β_2 are the parameters introduced in Algorithm 3 and Algorithm 4.

Proof. First, due to Algorithm 3 and Algorithm 4, we have

$$\begin{aligned}&\textstyle\sum_t C^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t)\\ =&\textstyle\sum_t \Big(C^t_{-M}(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t) + C^t_M(\widetilde{\mathbf{y}}^t, \widetilde{\mathbf{y}}^{t-1})\Big)\\ \leqslant&\textstyle(1+\beta_1)\sum_t C^t_{-M}(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t). \end{aligned} \tag{11a}$$

We explain $(10a)$. Consider the set of the time frames recorded by the variable $\hat{t}$ when executing Algorithm 3. Let us denote those time frames as $\{\hat{t}_1, \hat{t}_2, ...\}$. Consider any $p \geqslant 1$. For the consecutive time frames $\{\hat{t}_p, \hat{t}_p + 1, ..., \hat{t}_{p+1} - 1\}$, we either have

$$C^{\hat{t}_p}_M(\widetilde{\mathbf{y}}^{\hat{t}_p}, \widetilde{\mathbf{y}}^{\hat{t}_{p-1}}) \leqslant \beta_1 \cdot \textstyle\sum_{t=\hat{t}_p}^{\hat{t}_{p+1}-1} C^t_{-M}(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t)$$

Second, note that, analogously, due to Algorithm 4, for each k, we have

$$\begin{aligned}&\textstyle\sum_{k=1}^K \Big(C^{tk}_{-H}(\widetilde{\mathbf{x}}^{tk}, \widetilde{\mathbf{z}}^{tk}) + C^{tk}_H(\widetilde{\mathbf{x}}^{tk}, \widetilde{\mathbf{x}}^{tk-1})\Big)\\ \leqslant&\textstyle(1+\beta_2)\sum_k C^{tk}_{-H}(\widetilde{\mathbf{x}}^{tk}, \widetilde{\mathbf{z}}^{tk}). \end{aligned} \tag{12a}$$

Third, based on $(10a)$ and (12a), now we can denote $\alpha = (1+\beta_1)(1+\beta_2)$, from the previous text, we know that β_1 and β_2 are both positive real numbers, and then have

$$\textstyle\sum_t C^t(\widetilde{\mathbf{x}}^t, \widetilde{\mathbf{z}}^t, \widetilde{\mathbf{y}}^t)$$

$$\begin{aligned}&\leqslant(1+\beta_1)\textstyle\sum_t C^t_{-M}(\tilde{\mathbf{x}}^t,\tilde{\mathbf{z}}^t,\tilde{\mathbf{y}}^t)\\&\leqslant(1+\beta_1)\textstyle\sum_t\left((1+\beta_2)\sum_k C^{tk}_{-H}(\tilde{\mathbf{x}}^t,\tilde{\mathbf{z}}^t)+\sum_i a^t_i\tilde{y}^t_i\right)\\&\leqslant(1+\beta_1)(1+\beta_2)\textstyle\sum_t\left(\sum_k C^{tk}_{-H}(\tilde{\mathbf{x}}^t,\tilde{\mathbf{z}}^t)+\sum_i a^t_i\tilde{y}^t_i\right)\\&\leqslant(1+\beta_1)(1+\beta_2)\textstyle\sum_t\left(\sum_k C^{tk}_{-H}(\bar{\mathbf{x}}^t,\bar{\mathbf{z}}^t)+\sum_i a^t_i\bar{y}^t_i\right)\\&\leqslant(1+\beta_1)(1+\beta_2)\textstyle\sum_t C^t_{-M}(\bar{\mathbf{x}}^t,\bar{\mathbf{z}}^t,\bar{\mathbf{y}}^t)\\&\leqslant\alpha\textstyle\sum_t C^t(\bar{\mathbf{x}}^t,\bar{\mathbf{z}}^t,\bar{\mathbf{y}}^t).\end{aligned}$$

□

Theorem 3. NP-Hard. The total cost minimization problem $\mathbb{P}_0$ is NP-hard.

Proof. To prove the NP-hardness of our problem $\mathbb{P}_0$, we can reduce an existing NP-hard problem "p-median" to $\mathbb{P}_0$. In the following, we recap our problem $\mathbb{P}_0$, formulate the p-median problem, and exhibit the reduction process. We have already analyzed the gap between the obtained approximate solution and the theoretical optimum, and we also explain that as below. We have already listed the formulation of $\mathbb{P}_0$ in Section III.B.

1. Formulation of the p-median problem:

$$\begin{aligned}\min\ &\sum_i\sum_j c_{ij}x_{ij}\\ s.t.\ &\textstyle\sum_i x_{ij}=1,\forall j, && (14a)\\ &\textstyle\sum_i y_i=p, && (14b)\\ &x_{ij}\leqslant y_i,\forall i,\forall j, && (14c)\\ &x_{ij},y_i\in\{0,1\},\forall i,\forall j.\end{aligned}$$

2. Reduction: We set $T=1$ and $K=1$ while neglecting the switching cost terms, i.e., we consider a time horizon of a single time frame which contains a single time slot. We also set $\mathcal{I}^{tk}_j=\mathcal{I}$, $\forall j$ and $\mathcal{J}^{tk}_i=\mathcal{J}$, $\forall i$. That is, $\mathbb{P}_0$ becomes

$$\begin{aligned}\min\ &\sum_i\sum_j d_{ij}\lambda_j x_{ij}+\sum_i\sum_m l_{im}z_{im}+\sum_i a_i y_i\\ s.t.\ &\textstyle\sum_i x_{ij}=1,\forall j, && (15a)\\ &\textstyle\sum_j x_{ij}\leqslant Q_i,\forall i, && (15b)\\ &\textstyle\sum_m z_{im}\geqslant\sum_j\lambda_j x_{ij},\forall i, && (15c)\\ &\textstyle\sum_i z_{im}\leqslant y_m C_m,\forall m, && (15d)\\ &\textstyle\sum_i y_i=R, && (15e)\\ &x_{ij},y_i\in\{0,1\},z_{im}\geqslant 0,\ \forall i,\forall m,\forall j.\end{aligned}$$

In this $\mathbb{P}_0$, we further set

$$\begin{aligned}&d_{ij}=c_{ij},\forall i,\forall j,\\&\lambda_j=1,\forall j,\\&a_i=0,\forall i,\\&Q_i=|\mathcal{J}|,\forall i,\\&C_m=|\mathcal{J}|,\forall m,\\&R=p,\\&l_{im}=\textstyle\sum_i\sum_j c_{ij}+1\ \text{ (for } i\neq m)\ \text{ and }\ 0\ \text{ (for } i=m),\forall m.\end{aligned}$$

$\mathbb{P}_0$'s objective becoming p-median's objective: Note that the objective function of $\mathbb{P}_0$ is now the same as the objective function of p-median, except that the former contains the extra term $\sum_{i,m}l_{im}z_{im}$. Yet, for every m, the coefficient $l_{i,m}$ is set to 0 if $i=m$ and set to a large constant (i.e., $\sum_i\sum_j c_{ij}+1$, or any larger constant) if $i\neq m$. This indeed means that, in the optimum of $\mathbb{P}_0$, for every m, z_{im} is 0 for $i\neq m$ and could take any non-negative value for $i=m$; this further means that, in this case, the optimal objective value of $\mathbb{P}_0$ is exactly the same as that of p-median.

$\mathbb{P}_0$'s constraints becoming p-median's constraints: Note that (15a) is actually (14a), and (15e) is actually (14b). With $Q_i=|\mathcal{J}|$, (15b) always holds and can be ignored. Next, we show how (15c) and (15d) jointly lead to (14c). Following the above analysis of $\sum_{i,m}l_{im}z_{im}$, the left-hand side of (15c) becomes z_{ii} and the right-hand side of it becomes $\sum_j x_{ij}$, i.e., (15c) becomes $z_{ii}\geqslant\sum_j x_{ij}$; similarly, (15d) becomes $z_{ii}\leqslant y_i|\mathcal{J}|$. Connecting them, we have $\sum_j x_{ij}\leqslant y_i|\mathcal{J}|$, which is equivalent to $x_{ij}\leqslant y_i$, i.e., (14c). That is, in this case, the constraints of $\mathbb{P}_0$ are exactly the same as those of p-median.

□

Gap: The gap between $\mathbb{P}_0$'s objective value incurred by the approximate solution from our proposed algorithms and $\mathbb{P}_0$'s own optimal objective value is analyzed and mathematically proved in Theorem 2. In Theorem 2, we have proved a parameterized-constant competitive ratio to characterize the multiplicative gap, defined as the ratio of the optimization objective's value incurred by the online decisions over that incurred by the offline optimal decisions. While the approximation ratio is often used in the offline setting, the competitive ratio is the online version of the approximation ratio in the online setting (where "online" vs. "offline" will be discussed in our response to the next comment). The competitive ratio holds for all inputs, and is thus stable in this sense. A constant competitive ratio implies that the multiplicative gap does not grow as the length of the time horizon (e.g., the total number of the time frames) grows, even though it may still depend on other parameters. In our case, the competitive ratio depends on the control parameters β_1 and β_2 introduced in our Algorithms 3 and 4.

VI. Experimental Evaluations

In this section, we present the experimental evaluations conducted to assess the performance and effectiveness of the proposed system.

A. Experimental Settings

Edge System: We simulate a Walker-Delta constellation with an orbital height of 1000km and an inclination of 53 degrees through Satellite Tool Kit (STK) platform [47], [48], along with the deployment of 20 ground stations in China, each satellite has a communication coverage with a 30° cone half angle and different satellites have overlapped coverage. Coverage at a location varies over time as satellites move in and out of view [49]. We set the total length of the time range to 10 consecutive time frames, each with 12 time slots, and obtain content requests for each time slot [50].

Content Requests: We export the position and connection relationship data of the ground station and satellite simulated in STK to form all $\mathcal{I}^{tk}_j$ and $\mathcal{J}^{tk}_i$ at the current moment. We set the cost of each hosting service replica as within the range of $[0.2,1]$. We use the geographical distance to estimate the network delay between the two edges and the delay between ground stations and satellites. We note that we can associate proper non-negative weights to these different types of costs to indicate how the importance of each type of cost can impact the results.

Algorithms for Comparison: We implement the following algorithms for comparison: (1) Proposed refers to our proposed online algorithms; (2) Offline Optimum, this method considers the long-term issue as an offline issue, with all inputs identical to the online algorithm but known in advance; (3) Greedy seeks the current optimal solution in each corresponding time slot without considering the global optimal solution, and does not postpone ground station changing access satellite across time slots; (4) Random selects access satellites and service satellites considering any cost-related optimization. (5) Single-Timescale, solving on a single time scale as a comparative algorithm for our dual time scales. (6) An advanced Markov decision process Approach for User Allocation in Edge Computing called MDP [51], the core idea of this method is to describe the interaction between the server and the edge nodes in the form of evolutionary games.

B. Experimental Results

Fig. 4 depicts the impact of satellite size on set $\mathcal{I}_j^{tk}$, i.e. the number of satellites flying over one of the ground stations at k of t. It can be observed that smaller constellations fail to ensure a satellite presence in every time slot for connectivity. This is attributed to the sparse distribution of satellites determined by the constellation size [6]. To avoid situations where ground stations cannot choose to access satellites, it is advisable to opt for larger-scale constellations whenever possible, that is, a large constellation ensures that at least one satellite in each time slot can be connected to a ground station.

Fig. 5 depicts the total cost under different constellation sizes. Based on Fig.4, we can find that when (M, N) is $(16, 12)$ and $(12, 8)$, ground stations may sometimes stop working due to missing access to satellites, therefore the total total cost under small-scale constellations are much lower than those under large-scale constellations. On the contrary, as long as the ground station can be ensured to be within the communication coverage range of at least one satellite in each time slot, the total cost gap between different large-scale constellations will not be very significant. The additional portion arises from the migration costs incurred in the service transfer within the more extensive constellation, we can see that our proposed method performs better than Single-Timescale and MDP.

Fig. 6 depicts the total cost of different algorithms per time slot.Due to the transition process of the satellite i from becoming an access satellite to performing access handover (transferring the access role to the next satellite), a certain number of time slots are required. During this period, the connections established by the ground stations are stable. However, when the handover occurs, the cost increases significantly due to the associated handover penalty and the migration cost of tasks from the partially serviced satellite. Consequently, there is a noticeable elevation in cost. The simulation in Systems Tool Kit maintains a fixed starting time and initial topology. Therefore, when running different algorithms, the handover occurs at the same time point. Our approach outperforms Random and Greedy by reducing $5\% \sim 50\%$ total cost and is also close to the offline optimum. Fig. 5 and Fig. 6 showing that our approach achieves near-optimal performance with low computational overhead, and validates the scalability of our approach in terms of its running time under different network sizes. These results collectively demonstrate that our algorithms converge quickly, scale well with increasing network sizes, and achieve near-optimal performance with low computational overhead.

Fig. 7 depicts the influence of changes in the service cost weight per satellite on the total cost across different numbers of service satellites. In this scenario, we ensure that the service capacity of satellites can meet all task requirements. As the total number of specified service satellites in the entire system increases, the incurred task migration delay and hosting costs also rise, thereby impacting the overall total cost. This dynamic may arise due to the escalation in the complexity of internal communication and resource scheduling within the system as the number of service satellites grows. With the augmentation of service satellites, there is an increase in the frequency of task migration and hosting operations, consequently leading to elevated resource consumption and communication overhead. In this context, the rise in total cost reflects a relative decline in system performance under the burden of increased service workload

Fig. 8 illustrates the impact of the control parameter β_2 as in Algorithm 2 on the total cost, in this case, we give β_1 in Algorithm 1 as 0.25. When the weight of the handover penalty is provided, increasing β_2 makes it easier to meet the control requirement in Algorithm 2. Although a larger value of β_2 may delay the handover of satellite access as much as possible, it not only faces increased costs due to frequent downloading of copies from other satellites, but also leads to excessive time slot delay, which can cause the ground station to deviate from the coverage range of the currently connected satellites, and then force satellite handover to generate handover penalties. Similarly, when the parameter β_2 Given the timing, a higher handover penalty weight may delay the necessary content download, leading to an increase in the cumulative non replacement cost, thus increasing the total cost. Similarly, Fig. 9 illustrates the impact of the control parameter β_1 as in Algorithm 1 on the total cost. We compared the two graphs and found that the change in β_2 has a greater impact on total costs than β_1. Algorithm 1 operates at the time frame scale, and Algorithm 2 performs computations at the time slot scale. Since a time frame encompasses multiple time slots, the impact of β_2 in Algorithm 2 on total costs is slightly larger than the impact of β_1 in Algorithm 1. This difference in impact arises from the fact that the decisions made in each time slot accumulate over the entire time frame, and β_2 affects the total costs within each time slot, leading to a more significant overall impact.

Fig. 10 illustrates the correlation between the total number of service satellites R in the system and the service capacity C_i weight per individual service satellite. This is crucial as our system must ensure that, in each time slot, it respects the service capacity to avoid situations where service satellites cannot meet the demand for task assignments. For instance,

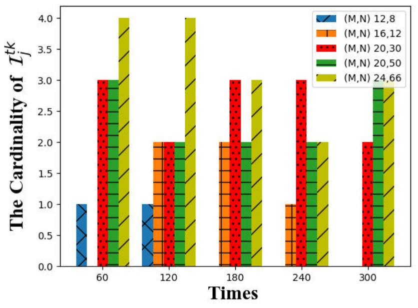


Fig. 4: The impact of constellation

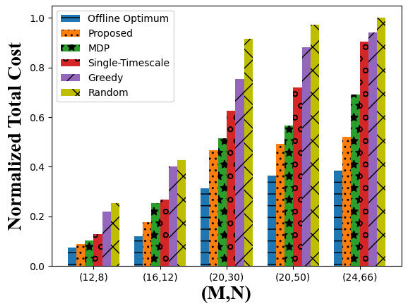


Fig. 5: Total cost comparison

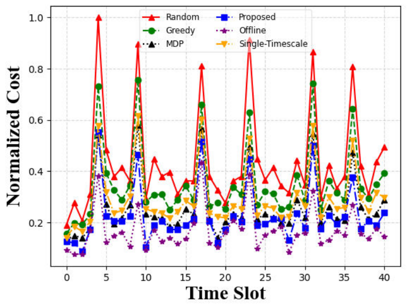


Fig. 6: Cost per time slot

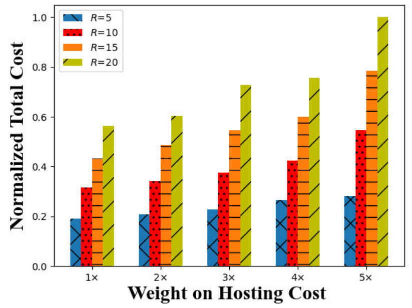


Fig. 7: Impact of hosting service a_i

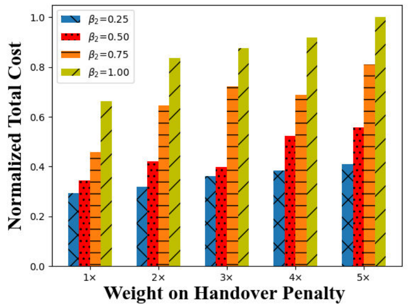


Fig. 8: Impact of parameter β_2

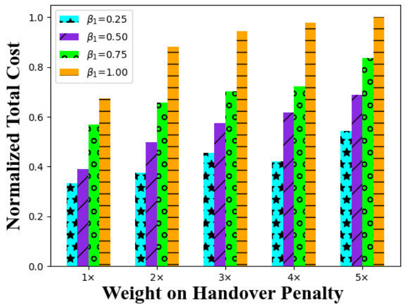


Fig. 9: Impact of parameter β_1

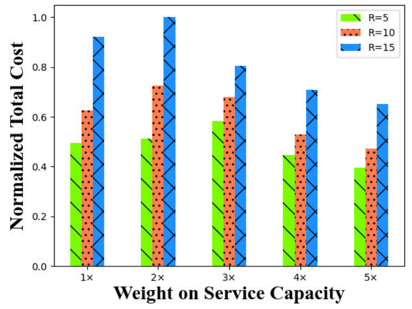


Fig. 10: Impact of capacity limitation

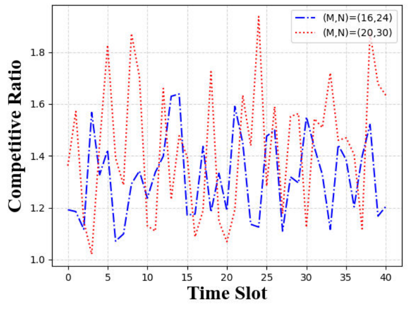


Fig. 11: Competitive ratio

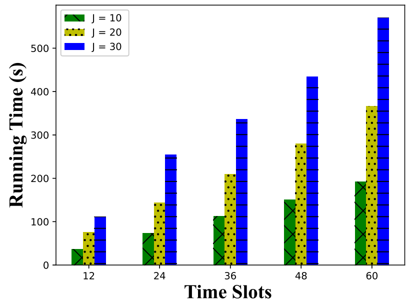


Fig. 12: Algorithms running time

if the specified total number of service satellites is too low and their service capacity is insufficient, some task demands may remain unallocated to service satellites in the current time slot. Consequently, we observe relatively low total costs in such scenarios because those task demands neither consume resources nor are handled by the system. On the other hand, when the system capacity is large enough to handle all task demands, with an increase in the weight of C_i, total costs decrease. This is because a single service satellite can accommodate as many task demands as possible, thereby alleviating the migration costs associated with the dispersed transmission of some task demands.

Fig. 11 indicates that our approach effectively achieves a remarkably low approximation ratio. Specifically, for any individual time slot, the approximation ratio is less than 2, considering constellations with sizes (16, 24) and (20, 30). Additionally, the constellation with parameters (16, 24), characterized by a smaller scale, exhibits an even smaller approximation ratio, approximately below 1.7. Furthermore, the stability of the approximation ratio in larger constellations is not as robust as in smaller constellations.

Fig. 12 respectively show the time spent on execution time of our proposed online algorithms. Although collecting various satellite data from STK took a lot of time, fortunately, data acquisition and our algorithm are two independent processes that do not affect the performance of our algorithm. Because the size of the constellation has almost no impact on the algorithm we propose, while ensuring that ground stations have access to satellites at each time slot, we choose to compare the running time of the algorithm with different numbers of ground stations. As the length of the entire time horizon in terms of the total number of time slots increases, the total execution time of our approach grows moderately.

VII. Conclusion

In this paper, we address the dynamic resource management challenges in LEO satellite edge computing by proposing a novel two-timescale optimization framework. We first formulate the social cost minimization problem as a non-convex mixed-integer program spanning the time horizon, considering joint optimization of satellite access selection, user request dispatching, and service replica placement. To solve this problem efficiently, we design polynomial-time online algorithms including a hierarchical framework (Algorithms 3-4) that coordinates long-term service migration decisions with short-term resource allocation, supported by primal-dual based subproblem solvers (Algorithms 1-2) for service placement and access selection. The exceptional performance of our technique in practice has been validated by conducting comprehensive evaluations, and by carefully demonstrating many theoretical features and guarantees.

References

[1] D. Bhattacherjee, S. Kassing, M. Licciardello, and A. Singla, "In-orbit computing: An outlandish thought experiment?" in *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*, 2020, pp. 197–204.

[2] Y. Guo and S. Wang, "Challenges and opportunities in space service computing," in *2021 IEEE International Conference on Services Computing (SCC)*. IEEE, 2021, pp. 44–51.

[3] Y. Li, H. Li, L. Liu, W. Liu, J. Liu, J. Wu, Q. Wu, J. Liu, and Z. Lai, "" Internet in Space" for Terrestrial Users via Cyber-Physical Convergence," in *Proceedings of the 20th ACM Workshop on Hot Topics in Networks*, 2021, pp. 163–170.

[4] S. Kassing, D. Bhattacherjee, A. B. Águas, J. E. Saethre, and A. Singla, "Exploring the" Internet from space" with Hypatia," in *Proceedings of the ACM Internet Measurement conference*, 2020, pp. 214–229.

[5] T. Pfandzelter and D. Bermbach, "Edge (of the earth) replication: Optimizing content delivery in large leo satellite communication networks," in *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*. IEEE, 2021, pp. 565–575.

[6] J. Wang, L. Li, and M. Zhou, "Topological dynamics characterization for leo satellite networks," *Computer Networks*, vol. 51, no. 1, pp. 43–53, 2007.

[7] C. Li, Y. Zhang, R. Xie, X. Hao, and T. Huang, "Integrating edge computing into low earth orbit satellite networks: Architecture and prototype," *IEEE Access*, vol. 9, pp. 39 126–39 137, 2021.

[8] T. Pfandzelter and D. Bermbach, "Qos-aware resource placement for leo satellite edge computing," in *2022 IEEE 6th International Conference on Fog and Edge Computing (ICFEC)*. IEEE, 2022, pp. 66–72.

[9] D. Vasisht, J. Shenoy, and R. Chandra, "L2d2: Low latency distributed downlink for leo satellites," in *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, 2021, pp. 151–164.

[10] Q. Li, S. Wang, X. Ma, Q. Sun, H. Wang, S. Cao, and F. Yang, "Service coverage for satellite edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 695–705, 2021.

[11] Z. Hou, X. Yi, Y. Zhang, Y. Kuang, and Y. Zhao, "Satellite-ground link planning for leo satellite navigation augmentation networks," *IEEE Access*, vol. 7, pp. 98 715–98 724, 2019.

[12] Y. Yuan, L. Jiao, K. Zhu, and L. Zhang, "Scheduling online EV charging demand response via V2V auctions and local generation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11 436–11 452, 2021.

[13] T. Van Chien, E. Lagunas, T. H. Ta, S. Chatzinotas, and B. Ottersten, "User scheduling and power allocation for precoded multi-beam high throughput satellite systems with individual quality of service constraints," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 1, pp. 907–923, 2022.

[14] C.-Q. Dai, M. Zhang, C. Li, J. Zhao, and Q. Chen, "Qoe-aware intelligent satellite constellation design in satellite internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4855–4867, 2020.

[15] N. Zhang, S. Zhang, P. Yang, O. Alhussein, W. Zhuang, and X. S. Shen, "Software defined space-air-ground integrated vehicular networks: Challenges and solutions," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 101–109, 2017.

[16] Z. Song, Y. Hao, Y. Liu, and X. Sun, "Energy-efficient multiaccess edge computing for terrestrial-satellite internet of things," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 14 202–14 218, 2021.

[17] 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 *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[18] H. Al-Hraishawi, H. Chougrani, S. Kisseleff, E. Lagunas, and S. Chatzinotas, "A survey on nongeostationary satellite systems: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 1, pp. 101–132, 2022.

[19] T. Chen, J. Liu, Q. Ye, Q. Tang, W. Zhang, T. Huang, and Y. Liu, "Efficient uplink transmission in ultra-dense leo satellite networks with multiband antennas," *IEEE Communications Letters*, vol. 26, no. 6, pp. 1373–1377, 2022.

[20] X. Chen, W. Ni, T. Chen, I. B. Collings, X. Wang, R. P. Liu, and G. B. Giannakis, "Multi-timescale online optimization of network function virtualization for service chaining," *IEEE Transactions on Mobile Computing*, vol. 18, no. 12, pp. 2899–2912, 2018.

[21] M.-M. Zhao, Q. Wu, M.-J. Zhao, and R. Zhang, "Intelligent reflecting surface enhanced wireless networks: Two-timescale beamforming optimization," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 2–17, 2020.

[22] Y. Zhang, Y. Tang, and W. Wang, "Service deployment and service request optimization scheduling in MEC enabled LEO networks," in *2021 International Conference on Computer Communications and Networks (ICCCN)*. IEEE, 2021, pp. 1–6.

[23] Z. Yan, T. de Cola, K. Zhao, W. Li, S. Du, and H. Yang, "Exploiting edge computing in internet of space things networks: dynamic and static server placement," in *2021 IEEE 94th vehicular technology conference (VTC2021-Fall)*. IEEE, 2021, pp. 1–6.

[24] X. Li, F. Tang, L. Fu, J. Yu, L. Chen, J. Liu, Y. Zhu, and L. T. Yang, "Optimized controller provisioning in software-defined LEO satellite networks," *IEEE Transactions on Mobile Computing*, vol. 22, no. 8, pp. 4850–4864, 2022.

[25] L. Chen, F. Tang, and X. Li, "Mobility-and load-adaptive controller placement and assignment in LEO satellite networks," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[26] Z. Lai, H. Li, Q. Zhang, Q. Wu, and J. Wu, "Cooperatively constructing cost-effective content distribution networks upon emerging low earth orbit satellites and clouds," in *2021 IEEE 29th International Conference on Network Protocols (ICNP)*. IEEE, 2021, pp. 1–12.

[27] Y. Lyu, Z. Liu, R. Fan, C. Zhan, H. Hu, and J. An, "Optimal computation offloading in collaborative LEO-IoT enabled MEC: A multiagent deep reinforcement learning approach," *IEEE Transactions on Green Communications and Networking*, vol. 7, no. 2, pp. 996–1011, 2022.

[28] Q. Chen, G. Giambene, L. Yang, C. Fan, and X. Chen, "Analysis of inter-satellite link paths for leo mega-constellation networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 3, pp. 2743–2755, 2021.

[29] Q. He, G. Cui, X. Zhang, F. Chen, S. Deng, H. Jin, Y. Li, and Y. Yang, "A game-theoretical approach for user allocation in edge computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 3, pp. 515–529, 2019.

[30] X. Lin, A. Liu, C. Han, X. Liang, K. Pan, and Z. Gao, "Leo satellite and UAVs assisted mobile edge computing for tactical Ad-Hoc network: A game theory approach," *IEEE Internet of Things Journal*, vol. 10, no. 23, pp. 20 560–20 573, 2023.

[31] Y. Jin, L. Jiao, Z. Qian, S. Zhang, N. Chen, S. Lu, and X. Wang, "Provisioning edge inference as a service via online learning," in *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2020, pp. 1–9.

[32] S. Liu, C. Zheng, Y. Huang, and T. Q. Quek, "Distributed reinforcement learning for privacy-preserving dynamic edge caching," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 3, pp. 749–760, 2022.

[33] X. Chen, Y. Bi, X. Chen, H. Zhao, N. Cheng, F. Li, and W. Cheng, "Dynamic service migration and request routing for microservice in multicell mobile-edge computing," *IEEE Internet of Things Journal*, vol. 9, no. 15, pp. 13 126–13 143, 2022.

[34] J. Xu, X. Ma, A. Zhou, Q. Duan, and S. Wang, "Path selection for seamless service migration in vehicular edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 9040–9049, 2020.

[35] R. Chai, S. Zhang, and W. Jiang, "Long-term energy consumption optimization-based task offloading algorithm for satellite-IoT systems," in *2023 IEEE 34th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2023, pp. 1–6.

[36] Y. Shi, C. Yi, R. Wang, Q. Wu, B. Chen, and J. Cai, "Service migration or task rerouting: A two-timescale online resource optimization for mec,"

IEEE Transactions on Wireless Communications, vol. 23, no. 2, pp. 1503–1519, 2023.

[37] J. Chen, Y. Xu, Q. Wu, Y. Zhang, X. Chen, and N. Qi, "Interference-aware online distributed channel selection for multicluster fanet: A potential game approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3792–3804, 2019.

[38] X. Lin, A. Liu, C. Han, X. Liang, Y. Sun, G. Ding, and H. Zhou, "Intelligent adaptive MIMO transmission for nonstationary communication environment: A deep reinforcement learning approach," *IEEE Transactions on Communications*, 2025.

[39] D. Han, Q. Ye, H. Peng, W. Wu, H. Wu, W. Liao, and X. Shen, "Two-timescale learning-based task offloading for remote iot in integrated satellite–terrestrial networks," *IEEE Internet of Things Journal*, vol. 10, no. 12, pp. 10 131–10 145, 2023.

[40] Y. Shi, Y. Yang, C. Yi, B. Chen, and J. Cai, "Towards online reliability-enhanced microservice deployment with layer sharing in edge computing," *IEEE Internet of Things Journal*, vol. 11, no. 13, pp. 23 370–23 383, 2024.

[41] Y. Yang, Y. Shi, C. Yi, J. Cai, J. Kang, D. Niyato, and X. Shen, "Dynamic human digital twin deployment at the edge for task execution: A two-timescale accuracy-aware online optimization," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 12 262–12 279, 2024.

[42] M. Neely, *Stochastic network optimization with application to communication and queueing systems*. Morgan & Claypool Publishers, 2010.

[43] B. H. Korte, J. Vygen, B. Korte, and J. Vygen, *Combinatorial optimization*. Springer, 2011, vol. 1.

[44] F. Wang, L. Jiao, K. Zhu, X. Lin, and L. Li, "Toward sustainable ai: Federated learning demand response in cloud-edge systems via auctions," in *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*. IEEE, 2023, pp. 1–10.

[45] Y. Yuan, L. Jiao, K. Zhu, X. Lin, and L. Zhang, "AI in 5G: The case of online distributed transfer learning over edge networks," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 810–819.

[46] Y. Song, L. Jiao, R. Yang, T. Wo, and J. Xu, "Incentivizing online edge caching via auction-based subsidization," in *2022 19th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2022, pp. 253–261.

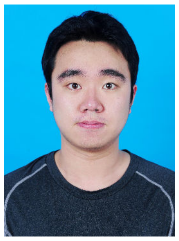
[47] J. Jiang, S. Yan, and M. Peng, "Regional leo satellite constellation design based on user requirements," in *2018 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE, 2018, pp. 855–860.

[48] Y. Zhang, C. Chen, L. Liu, D. Lan, H. Jiang, and S. Wan, "Aerial edge computing on orbit: A task offloading and allocation scheme," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 1, pp. 275–285, 2022.

[49] M. Zuo, G. Dai, L. Peng, and M. Wang, "An envelope curve-based theory for the satellite coverage problems," *Aerospace Science and Technology*, vol. 100, p. 105750, 2020.

[50] D. Xia, X. Zheng, P. Duan, C. Wang, L. Liu, and H. Ma, "Ground-station based software-defined leo satellite networks," in *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2019, pp. 687–694.

[51] T. Huang, Z. Fang, Q. Tang, R. Xie, T. Chen, and F. R. Yu, "Dual-timescales optimization of task scheduling and resource slicing in satellite-terrestrial edge computing networks," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 14 111–14 126, 2024.



Siru Chen received the bachelor's degree in Communication Engineering from Beijing University of Posts and Telecommunications, China, in 2020. He is currently working towards the Ph.D. degree in School of Artificial Intelligence in Beijing University of Posts and Telecommunications. His research interests are in the areas of satellite edge computing.

Lei Jiao received the Ph.D. degree in computer science from the University of Göttingen, Germany. He is currently with the University of Oregon, USA, and was previously a member of technical staff at Nokia Bell Labs, lreland. He researches Al infrastructures, cloud/edge networks, energy systems, cybersecurity, and multimedia. He has published 80+ papers mainly in leading journals such as IEEE Journal on Selected Areas in Communications, IEEE/ACM Transactions on Networking, IEEE Transactions on Mobile Computing, and IEEE Transactions on Parallel and Distributed Systems and conferences such as INFOCOM, MOBIHOC, ICDCS, SECON, and ICNP. He is a U.S. National Science Foundation CAREER awardee, and is also a recipient of the Ripple Faculty Fellowship, the Alcatel-Lucent Bell Labs UK and lreland Recognition Award, and the Best Paper Awards Of IEEE CNS 2019 and IEEE LANMAN 2013. He has been on the program committees as a track chair for ICDCS and as a member for many conferences such as INFOCOM, MOBIHOC, ICDCS, WWW, and IWQoS, and has also served as the program chair of multiple workshops with INFOCOM and ICDCs.

Konglin Zhu received the master's degree in computer Science from the University of California, Los Angeles, CA, USA, and the Ph.D. degree from the University of Göttingen, Germany, in 2009 and 2014, respectively. He is now a Professor with the Beijing University of Posts and Telecommunications, Beijing, China. His research interests include Internet of Vehicles, Edge Computing and Distributed Learning.

Lin Zhang received the B.S. and the Ph.D. degrees in 1996 and 2001, both from the Beijing University of Posts and Telecommunications, Beijing, China. From 2000 to 2004, he was a Postdoctoral Researcher with Information and Communications University, Daejeon, South Korea, and Nanyang Technological University, Singapore, respectively. He joined Beijing University of Posts and Telecommunications in 2004, where he has been a Professor since 2011. He is also the director of Beijing Big Data Center. His current research interests include mobile cloud computing and Internet of Things.