Computing-Aware Routing for LEO Satellite Networks: A Transmission and Computation Integration Approach

Jiaqi Cao, Shengli Zhang, Senior Member, IEEE, Qingxia Chen, Houtian Wang, Mingzhe Wang, and Naijin Liu

Abstract—The increasing demands of remote sensing (RS) have put a strain on computation and transmission resources. Traditional ground-offloading techniques, which involve sending large amounts of raw data to the ground, suffer from poor satellite-to-ground link quality. Additionally, current satellite-offloading methods, which involve using low earth orbit (LEO) satellites within the visible range of RS satellites for processing, fail to fully utilize the computing capability of the network due to limited computation resources of visible LEO satellites, particularly in hotspot areas. To address these issues, this article proposes a novel computing-aware routing scheme for efficient offloading via LEO satellite networks. This scheme integrates transmission and computation processes and optimizes overall delay. By modeling the LEO satellite network as a snapshot-free dynamic network model, cross-time pathfinding with low memory consumption is enabled. Instead of shielding the dynamics, the proposed model converts the topology dynamics into the association dynamics between satellites and virtual nodes, which represents self-loops, special edge and node weights. The proposed computing-aware routing scheme processes tasks during routing instead of offloading raw data to ground stations, reducing overall delay and preventing network congestion. The routing problem is formulated as a combination of multiple dynamic single source shortest path problems and a genetic algorithm-based method is proposed to approximate solutions in a reasonable amount of time. Simulation results indicate that the proposed computing-aware routing scheme decreases overall delay by 78.31% compared to offloading raw data to the ground for processing when computing capability is 100 Giga floating-point operations per second, which is a commonly supported capability by most LEO satellites.

Index Terms—LEO satellite network, remote sensing, computing-aware routing, dynamic network, genetic algorithm.

I. INTRODUCTION

REMOTE sensing (RS) plays an important role in various fields such as Earth science, space science, and exploration science [1]. However, completing these space missions requires a lot of computation and data transmission resources. Advances in image processing and target recognition techniques, particularly the use of machine learning [2], have increased the computational requirements [3], [4]. Furthermore, sensing technology improvements, such as hyperspectral image [5], also lead to a huge data volume of remote sensing images. Conventionally, these images are offloaded to ground servers for computation. The ground-offloading approach, where satellites act as bent pipes to route massive raw data to ground servers for processing, has its limitations. While the ground servers are powerful in computing capability, the overall delay of this approach is hard to meet the requirements of most RS applications that require real-time or near real-time processing [6]. It is because transmitting raw data can be a significant bottleneck: perturbed by the atmosphere frequently [7], the satellite-to-ground link (SGL) can be as low as 20 Mbps in state-of-the-art satellites [8]. To overcome these limitations, researchers have focused on onboard computing.

LEO satellites are a promising target for computation offloading. With the development of LEO satellite computing capabilities, high-performance onboard computing provided by a large number of LEO satellites could alleviate the challenges by processing data before transmitting them to ground servers [9]. This scheme can usually achieve impressive performance compared with ground offloading: LEO satellites are geographically closer to RS satellites, avoiding transmitting large amounts of raw data on SGLs; onboard processing significantly reduces the resulting data’s volume, which lowers the transmission delay. However, LEO satellite networks based offloading face major challenges.

Challenge 1: Uneven Available-Resource Distribution. The majority of human activity is concentrated on land, particularly in urban areas and ports, while a large portion of the Earth’s surface is covered by oceans and glaciers with minimal human

1The overall delay refers to the moment from a task is generated to the moment when the destination obtains the computation results of that task. For a computation task, both transmission and computation processes affect the overall delay.

Manuscript received 18 August 2022; revised 24 January 2023 and 24 April 2023; accepted 5 July 2023. Date of publication 20 July 2023; date of current version 19 December 2023. This work was supported in part by the Youth Fund of the National Natural Science Foundation of China under Grant 62201574, in part by the Joint Funds of the National Natural Science Foundation of China under Grant U22B2013, and in part by Shenzhen Key Research Project under Grants JSGG202208051956016002, JCYJ20220108100010602, and JCYJ2020018111609021. The review of this article was coordinated by Prof. Giovanni Giambene. (Corresponding author: Naijin Liu.)

Jiaqi Cao and Shengli Zhang are with the Shenzhen University, Shenzhen 518052, China (e-mail: jiaqicao@szu.edu.cn; zsl@szu.edu.cn).

Qingxia Chen, Houtian Wang, and Naijin Liu are with the Qian Xuesen Laboratory of Space Technology, China Academy of Space Technology, Beijing 100048, China (e-mail: qingxiaq@sqslab.cn; wanghoutian@sqslab.cn; liuqinjin@sqslab.cn).

Mingzhe Wang is with Tsinghua University, Beijing 100190, China (e-mail: wzmzhere@gmail.com).

Digital Object Identifier 10.1109/TVT.2023.3296893

0018-9545 © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.
activity. Therefore, available resources such as computing capability and spectrum resources are also unevenly distributed in the LEO satellite network [10], [11]. This can lead to a lack of computation resources for tasks generated by remote sensing satellites in certain areas. To address this issue, a routing strategy is needed to transmit these tasks from densely populated areas to remote LEO satellites with sufficient computation resources.

Challenge 2: Inapplicable Terrestrial Routing Strategy. Traditional terrestrial network routing strategies, which rely on static networks and shortest path algorithms, are not suitable for LEO satellite networks. The high-speed relative motion between adjacent-orbiting satellites and the limited communication distance result in intermittent connectivity, making it impossible to rely on contemporaneous paths between the source and destination [12]. Therefore, a routing strategy specifically designed for LEO satellite networks is needed to account for its unique physical characteristics.

Challenge 3: Unaccounted Computation Cost. Existing routing strategies for LEO satellite networks focus on data transmission only, but the transmission process and the computation process jointly determine the overall delay of a computation task. These strategies can only find the shortest path from a remote sensing satellite to a given LEO satellite used for computing, but cannot evaluate and select appropriate LEO satellites for computation automatically.

Our Solution: II. For efficient task offloading via LEO satellite networks, we propose a novel computing-aware routing scheme to minimize overall delay. The scheme, as illustrated in Fig. 1, involves three sub-processes: routing raw data of tasks from the source to the selected computing node, processing tasks and generating computation results at the computing node, and routing the computation results from the computing node to the destination. The proposed scheme can minimize overall delay because it optimizes the transmission and computation processes jointly. Additionally, to address the aforementioned challenges, the proposed scheme: (1) extends task offloading targets by introducing routing, allowing satellites beyond the visible range to be scheduled for computation; (2) models the network as a dynamic system, which can tolerate the fast change of available resources and network topology; (3) jointly optimizes the overall delay, taking into account computation delay in addition to transmission delay.

A. Main Contributions

1) A Snapshot-Free Dynamic Network Model: We propose a snapshot-free method for modeling LEO satellite networks to reduce memory consumption and enable cross-time pathfinding. The proposed model accounts for both resource dynamics and topology dynamics by using time-varying edge weights and node weights, respectively. Instead of shielding topology dynamics, the model converts them into association dynamics between satellites and virtual nodes (VNs), which can be represented by self-loops, special edge and node weights.

2) A Computing-Aware Routing Scheme: We propose a routing scheme that processes tasks during the routing process. The scheme optimizes the computation and transmission processes jointly, making it possible to offload tasks to the optimal satellite via the optimal path. By performing on-board computing, this scheme can achieve significant savings in bandwidth. Since the slow process of offloading raw data to the ground is avoided, this scheme reduces the overall delay.

3) A Genetic Algorithm Based Approximation Method: Since the LEO satellite network is highly dynamic and on-board resources need to be considered in the computing-aware routing, we formulate the routing problem based on the dynamic network model as a set of dynamic single source shortest path (DSSSP) problems. Due to the dynamics in graphs, conventional shortest path algorithms such as the static Dijkstra’s algorithm cannot be used to solve the problem; thus, we propose a genetic algorithm (GA)-based method to approximate the results in a reasonable time.

The rest of this article is organized as follows. Section II, summarizes the related studies. Section III investigates the capabilities of LEO satellites. Section IV presents the network model, traffic model, and delay model used throughout this article. Sections III, VI, and VII-A propose the modeling method, formulates the computing-aware routing problem based on the model, and presents a GA-based path finding algorithm for the model. Section VIII presents the simulation results and analyses. Section IX draws the conclusions.

II. RELATED WORK

The LEO satellite network is seen as a highly promising option for satellite mobile communication due to its advantages in latency, cost, and development cycle. As a result, there has been a significant amount of research in both academia and industry on LEO satellites and LEO satellite networks.

A. Routing Strategies for LEO Satellite Networks

Routing is a particularly challenging issue in LEO satellite networks due to the constantly changing relative positions of satellites, which can result in a lack of contemporaneous paths.
between the source and destination nodes [12]. Graph theory is an effective mathematical tool for modeling the network and providing a basis for routing design. As a result, graph-based routing strategies for LEO satellite networks have received significant attention of the research community. We illustrate typical graph-based network models in Fig. 2, whose characteristics are summarized in Table I.

Contact Graph Routing (CGR) was first proposed by NASA [20] for dynamic routing over the time-varying topology of satellite networks. CGR is based on the “contact plan,” a time-ordered list of scheduled, anticipated changes in the network topology [21], which can be converted to a Contact Graph (CG) (as shown in Fig. 2(e)). The nodes of a CG represent the episodes of contact during which data can be transferred, while the edges of a CG represent the periods of time during which data must be stored while awaiting subsequent contacts [27]. However, this method may not meet all mission requirements and may not fully utilize resources [22].

To address the dynamics in satellite networks, some existing works have proposed snapshot-based network modeling methods, such as the Virtual Topology (VT) model and the Virtual Node model, as well as corresponding routing strategies.

- The VT-based network model [13], [14], [15] (as shown in Fig. 2(a)) is based on the predictability of satellite movements, representing a satellite network as a time-evolving and predictable network [28]. It considers a LEO satellite network as a discrete-time network and assumes a fixed topology in each time interval [15], known as a snapshot. The VT-based model updates snapshots based on topology changes or fixed time slots, and routes are defined at each snapshot using this method.

- The VN-based network model (as shown in Fig. 2(b)) consists of different logical locations, which are static and disjoint zones of Earth (i.e., latitude and longitude), associated with the nearest satellites. The assignment between the logical locations and satellites changes due to satellite movements. With this architecture, each change in the satellite assignment represents a new snapshot [18]. Each snapshot can be considered as a mesh network presenting a static state of the network topology.

However, these snapshot-based methods shield the network dynamics by splitting the network into multiple static snapshots. They only look for paths within a single snapshot and ignore the relationship between adjacent snapshots [16]. A snapshot becomes invalid if the duration of the generated route exceeds the snapshot’s valid time, which can result in routing failures, especially in highly dynamic LEO satellite networks. Additionally, these snapshot-based models consume large amounts of memory resources as the number of snapshots increases with time or network expansion [17], [18].

To overcome the limitations of snapshot-based routing strategies, the Time-Expanded Graph (TEG) (as shown in Fig. 2(c)) was proposed to establish connections between networks in adjacent time slots [23], [24], [25]. It duplicates the original

---

**Table I**

| Reference | Classification | Dynamics Representation | Topology Resource | Graph/Model Size | Main Shortcomings |
|-----------|----------------|-------------------------|-------------------|------------------|------------------|
| VT [13], [14], [15] | Snapshot model | ✓ | ✓ | O(|S|) O(|N|) | Isolated snapshots split the connectivity of the whole network [16] and consume massive memory [17], [18]. |
| CG [20], [21] | Contact Graph (CG) | ✓ | ✓ | N/A O(M) O(M) | No assurance for task demands and low resource utilization [22]. |
| TRG [23], [24], [25] | Non-snapshot model | ✓ | ✓ | N/A O(|N x |S|) O(|N x |S|) | High storage overhead and computational complexity [18]. |
| Our work [26], [22] | TAG/STAG model | ✓ | ✓ | N/A O(|S|) O(|S|^2) | Excessive model size for highly dynamic networks. |

- Indicates that the resource dynamic is represented as continuous changes. ✓ Indicates that the resource dynamic is represented as discrete changes.
- |S| is the satellite number and |N| is the time slot number, M is the quantity of contacts in the contact plan, N is the slot number. Generally, |N| < |S| and M > |S|.
network for each time slot and builds edges connecting each node and its copy at the next slot to represent data storage. TEGs are essentially an expansion of static graphs, and many standard flow maximization algorithms can be applied to them [17]. However, TEGs still incur high overhead in terms of storage [29] and algorithm complexity [17].

Another approach is the Time Aggregated Graph (TAG) (as shown in Fig. 2(d)), which aggregates time-dependent attributes over edges and nodes [26], [30]. It represents the time-variance of attributes by modeling them as time series. In order to consider the buffer size constraint of each relay node in TAG, researchers proposed the Storage Time Aggregated Graph (STAG) [17], [22], [31]. Although the TAG and STAG models can capture the possibility of edges and nodes being absent during certain instants of time [26], these methods still face the problem of edge explosion when modeling highly dynamic networks, such as giant LEO constellations. Specifically, when the periods (altitudes) of the adjacent orbits of the LEO constellation are different, any two satellites in adjacent orbits may establish an inter-satellite link (ISL) within a certain period of time. In this condition, when constructing TAGs or STAGs, any two satellites in adjacent orbits must be connected with an edge, resulting in a quadratic increase in the number of edges with respect to the number of satellites.

To overcome the limitations of these existing network modeling methods and routing strategies for LEO satellite networks, we propose a novel dynamic network model that can represent both resource dynamics and topology dynamics with low complexity, along with a corresponding computing-aware routing scheme.

### B. Computation and Transmission Joint Optimization

Previous research has investigated the joint optimization of computation and transmission for satellite networks. In these studies, satellites were not interconnected. For example, the work [32] and [33] investigated the joint computation assignment and resource allocation problem in multi-tier computing architectures composed of mobile devices, LEO satellites, etc. Authors in [34] and [35] proposed hybrid computation offloading architectures to solve the joint computation and resource allocation problem, where computation tasks could be offloaded to both ground servers and visible LEO satellites.

Other studies have discussed the integration of computation and transmission in terrestrial networks. For example, the work in [36] discussed the joint communication and computation resource allocation in a two-tier device-cloud network, where tasks could be processed locally, at the edge cloud, or both. Authors in [37] and [38] investigated the task offloading problem in fog-enabled cellular networks where radio, caching, and computing were jointly optimized. The work in [39] and [40] proposed joint communication and computation resource scheduling approaches for Unmanned Aerial Vehicle (UAV)-assisted local-edge/local-edge-cloud computing systems, where each UAV worked as an edge computing device to assist devices within its communicable range. Authors in [41] developed a cloud-fog-device computing architecture for Internet of Things, where the offloading ratio, transmission power, and local CPU computation were jointly optimized.

While these studies jointly optimize the allocation of multiple resources, they fail to achieve network-wide computation offloading. This is because the networks in these studies are tree networks, where tasks cannot be forwarded to other computing devices in the same network tier. To overcome this limitation, LEO satellites in this article are connected with ISLs, which form a mesh network. In this condition, computation tasks can be offloaded to any satellite via routing, which makes it possible to extend the offloading targets to the entire network. However, the network-wide computation offloading for LEO satellites introduces a new challenge: the optimization of transmission paths. To address this challenge, we propose a computing-aware routing scheme. It takes the transmission and computation stages into account, optimizing resources and transmission paths in a joint fashion. This allows for more efficient task offloading and minimizes overall delay.

### III. STATE-OF-THE-ART LEO SATELLITE CAPABILITIES

As the computational requirements and data volume of space missions increase, unprecedented interest and efforts have been devoted to enhancing the computing and transmission capabilities of LEO satellites.

#### A. Computation Capability of LEO Satellites

For next-generation science and defense missions, LEO satellites must provide advanced processing capability to support a variety of computationally intensive tasks [42]. The development of onboard computing systems has led to an increase in the computing capabilities of LEO satellites. Table II summarizes the computing capabilities of typical onboard computing systems.

| Product       | Processor                          | Computing Capability (GFLOPS) | Reference |
|---------------|------------------------------------|-------------------------------|-----------|
| Xiphos Q18    | Xilinx Zynq 7020                   | 160                           | [41]      |
| Xiphos Q8S    | Xilinx UltraScale+                 | 1800                          | [44]      |
| BAE RAD545    | RAD545                            | 3.7                           | [45]      |
| Innoflight CPC500 | Xilinx Kintex UltraScale+, NVIDIA TK1 | 1290                          | [46]      |
| MOOG G-Series Steppe Eagle | AMD G-Series compatible | 75                             | [47]      |
| MOOG V-Series Ryzen | AMD V-Series compatible | 1000                           | [47]      |
| Unihap iXS-100 | Microchip SmartFusion3, AMD G-Series SOC | 127                            | [48], [49] |
| Unihap IX10-100 | Microchip PolarFire, AMD V160hS (Ryzen) | 3600                          | [50], [49] |
| SpaceCube v2.0 | Xilinx Virtex 5                    | 200                           | [51]      |
| SpaceCube v3.0 | Xilinx Kintex UltraScale, Xilinx Zynq MPSoC | 590                            | [42], [52] |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
importance of increasing the computing capability of LEO satellites has been recognized and a great deal of research has been invested, indicating a promising future for onboard computing.

B. Data Rate of LEO Satellites

ISLs in free space typically have higher data rates. For instance, the data rate of optical ISLs can reach 5 Gbps [54]. Mynar’s laser terminal for LEO constellations can deliver 10 Gbps with a low SWaP unit over a wide range of constellation configurations [55] and operate within densely packed constellations with intra/inter-plane link distances up to 7,800 km.

In contrast, the data rate of the SGL is limited due to the perturbation induced by the atmosphere [7]. The downlink data rate for state-of-the-art satellite networks ranges from 20 Mbps to 1 Gbps [7]. For example, Hyperion Technologies’ CubeSat lasercom module enables a bidirectional space-to-ground communication link between a CubeSat and an optical ground station, with a downlink speed of 1 Gbps and an uplink data rate of 200 Kbps [8]. The limited SGL transmission capability further promotes the application of on-board computing.

IV. SYSTEM MODEL

A. Network Model

As shown in Fig. 3(a), the LEO satellites are uniformly distributed over orbits at an altitude of $h$ kilometers. The satellites on the same orbit are uniformly distributed. The set of satellites is denoted by $S = \{S_1, S_2, \ldots, S_{|S|}\}$ and the set of orbits is denoted by $O = \{O_1, O_2, \ldots, O_{|O|}\}$. The orbit inclination $i_0$ determines the latitude of coverage. The orbital period is $T_O = 2 \times \pi \times \sqrt{(R_e + h)^3/(G \times M_e)}$, where $R_e$ and $M_e$ represent the radius and mass of the earth respectively, and $G$ is the gravitational constant.

In Fig. 3(a), the Earth is divided into multiple static, disjoint zones according to longitude and latitude. These zones are stationary with respect to the ground and correspond to VNAs. A device in space, such as a satellite, associated with a VN implies that its sub-satellite point (on the ground) is located within the zone corresponding to the VN. However, the association between satellites and VNs changes over time. As a result, the topology dynamics of the LEO satellite network are converted into the association dynamics between satellites and VNs (as discussed in Section V-C).

In Fig. 3(b), a part of the VN network shown in Fig. 3(a) is depicted. The edges between each pair of VNs represent the communication links between the associated satellites. The inter-satellite and satellite-ground connection strategies are as follows:

- **Inter-satellite connections**: each satellite has four inter-satellite links with its neighboring satellites, two of which are intra-plane and two are inter-plane. The maximum inter-satellite communication distance is 5000 km [56].
- **Satellite-ground connections**: a satellite can communicate with a ground station only when the elevation angle between them is greater than the minimum elevation angle. For simplicity, in this article, it is assumed that a satellite can communicate with ground stations in a zone when its sub-satellite point is located within that zone.

In this VN network, the transmission resources are related to edges, whereas the computation resources are related to VNs. The available resources decrease as they are occupied and increase as they are released, forming resource dynamics (discussed in Section V-B).

It is crucial to emphasize that, while we do generate VNs in this article, our proposed dynamic network model is not a simple extension of conventional VN-based network models. As a result, our model is not constrained by usage conditions typically associated with traditional VN-based methods, such as high regularity requirements. The key differences between them lie in the areas of dynamic abstraction and application scope. Specifically, our proposed model uses graph structures to capture association changes, in contrast to traditional approaches that represent these changes with isolated static snapshots. Consequently, our model is time-continuous rather than time-discrete. Moreover, our model features lower regularity requirements. This flexibility enables our dynamic network model to accommodate large, heterogeneous LEO constellations.

B. Traffic Model

In order to simplify the analysis, a traffic model is adopted in this article without considering different types of applications. The arrival of tasks is assumed to follow a Poisson stochastic process with parameter $\lambda$, as these processes have desirable theoretical properties [57]. The $k$th computation task arrives at VN $u$ at instant $t_{u,k}$, denoted as $T_{u,k}$. It can be divided into multiple independent subtasks, denoted as $T_{u,k} = \{τ_{u,k}^1, τ_{u,k}^2, \ldots, τ_{u,k}^l, \ldots, τ_{u,k}^{n_{u,k}}\}$ where $n_{u,k}$ is the total number of subtasks of $T_{u,k}$, $τ_{u,k}$ is the $l$th subtask of $T_{u,k}$, $l \leq n_{u,k}$ and $l, n_{u,k} \in \mathbb{Z}^+$. Subtasks are the smallest unit of transmission and computation, and subtasks belonging to the same task are routed to the same destination.

To describe subtask $τ_{u,k}^l$, seven items are used, denoted as $\Lambda_{τ_{u,k}^l} = (C_{τ_{u,k}^l}, N_{τ_{u,k}^l}, S_{τ_{u,k}^l}, ϕ_{τ_{u,k}^l}, α_{τ_{u,k}^l}, β_{τ_{u,k}^l}, t_{τ_{u,k}^l})$. $C_{τ_{u,k}^l}$, $N_{τ_{u,k}^l}$, $S_{τ_{u,k}^l}$ and $ϕ_{τ_{u,k}^l}$ are the computational requirement in GFLOPS,
the data volume in gigabytes (GB), the amount of memory needed to complete the computation in GB and the required delay threshold to process subtask $\tau_{i,k}^t$ in seconds, respectively. $\alpha_{u,k}^t$ and $\beta_{u,k}^t$ represent the longitude and latitude of the destination of $\tau_{i,k}^t$, respectively.

Furthermore, $n_{u,k}, c_{u,k}^t, S_{u,k}^t$ and $G_{u,k}^t$ are log-normally distributed, i.e., $\ln(n_{u,k}) \sim N(\mu_n, \sigma_n^2)$, $\ln(c_{u,k}^t) \sim N(\mu_C, \sigma_C^2)$, $\ln(S_{u,k}^t) \sim N(\mu_S, \sigma_S^2)$, and $\ln(G_{u,k}^t) \sim N(\mu_G, \sigma_G^2)$. The delay threshold of subtask $\tau_{i,k}^t$ is randomly chosen from $\vartheta_1, \vartheta_2$. Additionally, the longitude $\alpha_{u,k}^t$ and latitude $\beta_{u,k}^t$ of the destination of $\tau_{i,k}^t$ are randomly generated and subject to a uniform distribution.

C. Delay Model

The overall delay in this article is composed of several components: the transmission delay, the propagation delay, the computation delay, and the waiting delay.

In dynamic networks, the transmission delay $T_{trans}$ of subtask $\tau_i$ on edge $e(i,j)$ starting from instant $t$ is determined by the equation: $N_{i}^t = \int_{t}^{t + T_{trans}} R_{i,j}^t(t) dt$, where $N_{i}$ is the data volume of subtask $\tau_i$. $R_{i,j}^t(t)$ is the available transmission rate of edge $e = (i,j)$ at instant $t$ and always smaller than the maximum transmission rate $R_{max}$. This equation states that the total data volume of the subtask must be equal to the integral of the available transmission rate over the time duration of the transmission.

Similarly, the computation delay $T_{comp}$ of subtask $\tau_i$ at node $i$ starting from instant $t$ is determined by the equation: $C_{i}^t = \int_{t}^{t + T_{comp}} C_{i}^t(t) dt$, where $C_{i}$ is the computational requirement of subtask $\tau_i$. $C_{i}^t(t)$ is the amount of available computing capability that node $i$ can provide at instant $t$ and always smaller than the maximum computing capability $C_{max}$.

This equation states that the integral of the available computing capability over the computation duration must be equal to the total computational requirement of the subtask.

Ignoring the minor distance changes during transmissions, the propagation delay $T_{prop}$ on edge $e(i,j)$ starting from instant $t$ is mathematically defined as $T_{prop}(t) = D_{i,j}(t)/c$, where $D_{i,j}(t)$ represents the distance between node $i$ and node $j$ at instant $t$, and $c$ is the speed of light.

The waiting delay is the duration from the arrival of a subtask to the moment when the subtask starts being transmitted or processed. In the proposed model, the waiting delay before transmission and computation are included in the corresponding transmission and computation delay, respectively.

V. SNAPSHOT-FREE DYNAMIC NETWORK MODELING

A. Definition of Dynamic Network Model

The proposed dynamic network model generates zones and VNs in the same way as existing VN network models, but it is a continuous-time model and can represent both topology dynamics and resource dynamics.

The snapshot-free dynamic network model (SFDNM) could be defined as $G_{SFDNM}(t) = (V, E(t), W_E(t), W_V(t))$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of VNs, $E(t) = \{e_1, e_2, \ldots, e_m(t)\}$ and $W_E(t)$ is the set of edges. An edge could be represented as $e = (i,j)$ if $i$ is the head and $j$ is the tail of $e$. $W_E(t) = \{\omega_{e_1}(t), \omega_{e_2}(t), \ldots, \omega_{e_m(t)}(t)\}$ is the weight set of edges and $W_V(t) = \{\omega_{v_1}(t), \omega_{v_2}(t), \ldots, \omega_{v_n}(t)\}$ is the weight set of node (i.e., VNs). Since both transmission and computation resources have impact on computing-aware routing problem stated in Section I, the proposed model have both edge weights and node weights. These weights are task-related, which will be elaborated in Section V-B2.

B. Time-Varying Resources Modeling

1) Impact Factors of Edges and VNs: The routing process in LEO satellites is affected by various factors such as intermittent communications, ISL bandwidths, and on-board resource availability. These factors can be grouped into two categories: those that affect the transmission process (such as intermittent communications and ISL bandwidths) and those that affect the computation process (such as available on-board resources). We study the impact factors and present their definitions below.

The impact factor set of edges $\Lambda_E = \{\mathbb{D}, \mathbb{B}\}$: The intermittent communication between satellites is determined by the variation of distance caused by the relative motion between satellites. Here the inter-satellite distance matrix is represented as $\mathbb{D} = \{D_{i,j}(t), i,j \in V, t \in [0,T]\}$, where $D_{i,j}(t)$ is the distance between satellites associated with VN $i$ and VN $j$ at instant $t; T$ is the duration of simulation. $\mathbb{B} = \{B_{i,j}(t), i,j \in V, t \in [0,T]\}$ presents the available spectrum bandwidth matrix of inter-satellite links, where $B_{i,j}(t)$ is the available spectrum bandwidth of the communication link between VN $i$ and VN $j$ at instant $t$.

For transmission links, $R_{i,j}^t(t) = \sigma \times B_{i,j}(t)$ indicates that the data transmission rate of edge $e = (i,j)$ at instant $t$ is proportional to the spectrum bandwidth $B_{i,j}(t)$.

The impact factor set of VNs $\Lambda_V = \{\mathbb{C}, \mathbb{S}, \mathbb{E}\}$: Here the available computing capability matrix is denoted as $\mathbb{C} = \{C_{i}(t), i \in V, t \in [0,T]\}$, the available memory matrix is defined as $\mathbb{S} = \{S_{i}(t), i \in V, t \in [0,T]\}$, and the available energy matrix is represented as $\mathbb{E} = \{E_{i}(t), i \in V, t \in [0,T]\}$. The impact of VNs is determined by the computation process and is defined as its

2) Edge Weights and Node Weights: In computing-aware routing, the routing process involves both data transmission and task computation. The weight of an edge is determined by the transmission process and is represented by the sum of transmission and propagation delays\(^2\). The weight of a VN is determined by the computation process and is defined as its

\(^2\)Propagation delays should be accounted for in LEO satellite networks, because the inter-satellite distance ranges from ten to thousands of kilometers. The propagation delays at milliseconds levels are much larger than those in terrestrial mobile networks [58].
processing delay. The equations for calculating edge and node weights are as follows:

The set of edge weights: \( W_E(t) = \{ \omega_{u,k,l}(t) | e \in E(t), u \in V, t \in [0, T] \} \) is the set of edge weights. \( \omega_{u,k,l}(t) \) is the weight of edge \( e = (i, j) \) corresponding to subtask \( \tau_{u,k} \) and instant \( t \). It is the sum of the transmission delay \( T_{trans}^{u,k,l}(t) \) and propagation delay \( T_{prop}^{u,k,l}(t) \) of subtask \( \tau_{u,k} \) on edge \( e = (i, j) \) starting from instant \( t \), which can be mathematically defined as follows.

\[
\omega_{u,k,l}(t) = \left\{ \begin{array}{ll}
T_{trans}^{u,k,l}(t) + T_{prop}^{u,k,l}(t), & \zeta \leq \mathcal{S}_{ij}(t), \\
\infty, & \text{otherwise},
\end{array} \right.
\]

In equation (1), \( \zeta = T_{trans}^{u,k,l}(t) + T_{prop}^{u,k,l}(t) \) denotes the visible duration between \( i \) and \( j \) at \( t \) reflecting the intermittent communication of LEO satellite networks. It can be concluded that \( \omega_{u,k,l}(t) \) equals 0 when \( \zeta > \mathcal{S}_{ij}(t) \), which indicates that task \( k \) cannot be transmitted through edge \( e = (i, j) \) successfully at \( t \) because the visible duration is shorter than the time required by the transmission process.

The set of node weights: \( W_V(t) = \{ \omega_{u,k,l}(t) | i, u \in V, t \in [0, T] \} \) is the set of node weights. \( \omega_{u,k,l}(t) \) is the weight of VN \( i \) corresponding to subtask \( \tau_{u,k} \) and instant \( t \). It is the processing delay \( T_{proc}^{u,k,l}(t) \) of \( \tau_{u,k} \) at VN \( i \) start from instant \( t \), which can be mathematically defined as follows.

\[
\omega_{u,k,l}(t) = \left\{ \begin{array}{ll}
T_{proc}^{u,k,l}(t), & S_{ij}^{u,k,l}(t) \geq S_{ij}^{u,k,l} \& E_{ij}^{u,k,l}(t) \geq f(C_{ij}^{u,k,l}), \\
\infty, & \text{otherwise},
\end{array} \right.
\]

In equation (2), \( S_{ij}^{u,k,l}(t) \) and \( E_{ij}^{u,k,l}(t) \) are the amount of available memory and energy that VN \( i \) can provide at instant \( t \), respectively. \( S_{ij}^{u,k,l} \) is the amount of memory required to complete subtask \( \tau_{u,k} \). \( C_{ij}^{u,k,l} \) is the computational requirement of subtask \( \tau_{u,k} \). \( f(\cdot) \) maps the amount of computation to the amount of energy consumption. It can be concluded that \( \omega_{u,k,l}(t) \) equals 0 when \( S_{ij}^{u,k,l}(t) < S_{ij}^{u,k,l} \) or \( E_{ij}^{u,k,l}(t) < f(C_{ij}^{u,k,l}) \), which means that the computational requirement of subtask \( \tau_{u,k} \) cannot be completed by VN \( i \) at instant \( t \) if the VN’s available memory or energy is insufficient.

The proposed dynamic network model has two main differences from conventional static network models. Firstly, both edges and nodes in the dynamic model are weighted, as the processing delay of subtasks is related to the available-on-board resources of the selected computing VN. This requires the nodes in the dynamic model to be weighted to assist in computing VN selection. Secondly, both edge weights and node weights are time-varying, as LEO satellites are moving at high speed, resulting in changes to inter-satellite distances, available ISL data rates, and on-board resources in real-time. This necessitates modeling the LEO satellite network as a dynamic network with time-varying weights.

C. Dynamic Topology Modeling

The topology dynamics of the proposed model are translated into the dynamics of the association between satellites and VN, which can be represented by graph structures as shown in Fig. 4. Four basic cases and an example of a combined case are presented in the figure. Zone 1 and Zone 2 are two adjacent zones, associated with VNs \( i \) and \( j \), respectively. Unless otherwise stated, it is assumed that the available resources of each VN are between zero and the maximum value.

In basic case 1, there is no satellite located in Zone 1 at instant \( t \), making communication between VNs \( i \) and \( j \) impossible, with edge attributes \( D_{ij}(t) = \infty \) and \( B_{ij}(t) = 0 \). Additionally, the available computing capability, memory resources, and energy storage of VN \( i \) are all equal to 0.

In basic case 2, there is one satellite located in both Zone 1 and Zone 2 at instant \( t \), allowing for communication between VNs \( i \) and \( j \) with edge attributes \( D_{ij}(t) \in (0, \infty) \) and \( B_{ij}(t) \in (0, B_{max}) \).

In basic case 3, there are two satellites located in Zone 1 and no satellite in Zone 2 at instant \( t \), forming a link from VN \( i \) to itself with edge attributes \( D_{ij}(t) = 0 \) and \( B_{ij}(t) = \infty \), and a zero transmission and propagation delay for transmitting tasks from VN \( i \) to VN \( j \).

Based on the above four basic cases, the association between satellites and VNs at any instant and the switching of the association can be accurately represented graphically, since any other case in the dynamic network model of LEO satellites is a combination of the basic cases. A combined case is shown in Fig. 4. In this example, at instant \( t \), there are 2 satellites associated with VN \( i \) and 1 satellite associated with VN \( j \) at instant \( t \). This is a combination of basic cases 1 and 3, where there is a loop from VN \( i \) to itself due to the presence of two satellites in Zone 1, and both satellites in Zone 1 can communicate with the satellite in Zone 2, resulting in two edges with different attribute values between VN \( i \) and \( j \).

The topology of the proposed model is dependent on the number and location of satellites in each predefined disjoint zone, rather than on the satellite moving direction or orbital altitude. As a result, the proposed dynamic network model can be applied to both polar and inclined orbit constellations, and is compatible with constellations having different orbital altitudes, making it more dynamic than models of constellations having the same orbital altitude under the same settings.

It is important to note that inter-satellite communication policies\(^3\) constrain edge generation and selection in the proposed model, so a data structure has been designed to assist in the specification and selection of edges. The data structure records the edges between two adjacent VNs and can be expressed as \( [VN_{start}, VN_{end}, e_1, e_2, ..., e_i] = \{S_i_{start}, S_i_{end}, t_i_{start}, t_i_{end}\} \). The set of edges \( e_1, e_2, ..., e_i \) is between VN\(_{start}\) and VN\(_{end}\), where

\(^3\)Inter-satellite communication policies include constraints such as whether satellites moving in opposite directions can communicate with each other, whether the satellite’s antennas are steerable, and the maximum number of connections that can be established by each satellite.
$e_i$ represents the ISL between satellites $S_{i,\text{start}}$ and $S_{i,\text{end}}$, and $t_{i,\text{start}}$ and $t_{i,\text{end}}$ are the start and end instances of the validation period of $e_i$, respectively. This allows for the specification of multiple edges between the same VN pair.

Notably, the significance of employing VN in our article is twofold. First, by determining the appropriate zone size, considering VNs as nodes can reduce both node and edge numbers of the network model in comparison to treating satellites as nodes. This results in a more efficient network modeling approach, particularly well-suited for managing large-scale heterogeneous LEO satellite networks. Second, considering VNs as nodes can reduce both node and edge numbers of the network model in comparison to treating satellites as nodes.

This approach considerably streamlines the evaluation process for satellite-ground communication, as it eliminates the need for elevation angle calculations typically performed in non-VN-based methods.

### VI. Problem Formulation

In this section, a typical DSSSP problem is introduced first, and then the computing-aware routing problem is formulated and converted into a set of multiple DSSSP problems.

#### A. Dynamic Single Source Shortest Path Problem

Many problems, such as the shortest path problem, can be solved by searching for the path with the minimum cost from the source node to the destination node. The cost model varies in different problems, for example, it can be time or the number of hops. However, when the weight of each edge changes with the evolution of time, the problem becomes a DSSSP problem [59], [60], [61].

DSSSP problems cannot be solved by traditional dynamic programming methods such as the static Dijkstra’s algorithm and have attracted wide interests among researchers.

**Problem $P_0$ [62]:** Let $G = (V, E(t), w(t))$ be a simple directed graph, where $V = \{V_1, V_2, \ldots, V_n\} \ (n = |V|)$ and $E(t) = \{e_1, e_2, \ldots, e_{m(t)}\} \ (m(t) = |E(t)|)$ are the sets of vertices and edges, respectively. Let $e = (u, v) \in E(t)$; then $u$ is the head of $e$ denoted as $e_h$, and $v$ is the tail of $e$ denoted as $e_t$. The edge weight function $w(e, t)$ maps $e \in E(t)$, $t \in [0, T]$ to non-negative real numbers. It gives the weights of corresponding edges at instant $t$. In other words, the length of the path depends on time $t$: assuming that there is a path $P_{u,v} = \{(u_1, v_1), (u_2, v_2), \ldots, (u_p, v_p)\}$ ($u_1 = u$, $v_p = v$, $u_p = t_{p-1}$, $p \in Z^+ + 1$) and the start time is $t_1$, then the length of path $P_{u,v} = L_{P_{u,v}} = \psi(P_{u,v}, t_1) = w(e_1, t_1) + w(e_2, t_2) + \cdots + w(e_p, t_p), \text{ where } t_p = t_{p-1} + w(e_{p-1}, t_{p-1}) \ (p \in Z^+ + 1), \psi(\cdot)$ maps the path and its start time to the path length. Then the DSSSP problem is defined as finding the shortest path $\pi_{u,v,t} = \phi(u, v, t)$ ($\phi(\cdot)$ maps the source node, the destination node and the start time to the shortest path) and its length $L_{\pi_{u,v,t}}$ from a specific source node $u$ to each $v \in V$ at time $t$.

It is worth noting that if $v$ is not accessible from $u$, then $\pi_{u,v,t} = \emptyset$ and $L_{\pi_{u,v,t}} = \infty$. The problem $P_0$ is a non-convex optimization problem and it has been proven to be NP-hard [63].

- $t$ is the instant when data is transmitted to the head of $e$ (i.e., $e_h$). It is also called “the start time of $w(e, t)$” or “the time of $w(e, t)$” for short.
Therefore, it is computationally prohibitive to find the optimal solution for the optimization problem $\mathcal{P}0$.

**B. Computing-Aware Routing Problem**

In this article, subtasks are the smallest unit of transmission and computation, and are the unit of routing. Since subtasks cannot be further partitioned, only one computing node is on each path in the proposed computing-aware routing scheme.

1) **Computing-Aware Routing Problem in $G_{SFDNM}(t)$:** The multipath-single-computing-node routing strategy is adopted, meaning that multiple independent subtasks that make up a task can be routed on different paths simultaneously.

The ultimate goal of this article is to find the optimal path for each subtask in the dynamic network $G_{SFDNM}(t)$ established in Section V that could minimize the overall delay of each subtask, which could be formulated as follows.

**Problem $\mathcal{P}1$:** Let $G_{SFDNM}(t) = (V, E(t), W_{E}(t), W_{V}(t))$ be a directed graph, where $V = \{v_1, v_2, \ldots, v_n\}$ ($n = |V|$) is the node set, $E(t) = \{e_1(t), e_2(t), \ldots, e_m(t)\}$ ($m(t) = |E(t)|$) is the edge set. Assuming that $\mathcal{P}_{u,k}$ is the $k$th ($k \in Z^+$) computation task arrived at node $u$ and its destination node is $v$. For subtask $\mathcal{T}_{u,k}^t \in \mathcal{P}_{u,k}$, $v \in V$, $t \in [0, T]$ is the set of edge weights and $W_{V}(t) = \{\omega_{v,k,l}(t) \mid v \in E(t), u \in V, t \in [0, T]\}$ is the set of node weights. Assuming that subtask $\mathcal{T}_{u,k}^t$ is transmitted on path $P_{u,v}$, which is $\{\{u_1, v_1\}, \{u_2, v_2\}, \ldots, \{u_p, v_p\}\}$ ($u_1 = u$, $v_p = v$, $u_p = v_{p-1}$, $p \in Z^+$, $p \geq q$) and processed by $u_q$ (the selected computing node) on path $P_{u,v}$. If the start time of $P_{u,v}$ is $t_1$, the length of path $P_{u,v}$ is defined as $L_{P_{u,v}} = \Psi(P_{u,v}, u_q, t_1) = \omega_{u,v,k,l}(t_1) + \omega_{u,v,k,l}(t_2) + \ldots + \omega_{u,v,k,l}(t_{q-1}) + \omega_{u,v,k,l}(t_q) + \omega_{u,v,k,l}(t_{q+1}) + \omega_{u,v,k,l}(t_{p-1})$, where $t_q$ is the instant that $\mathcal{T}_{u,k}^t$ arrives at node $u_q$ (i.e., the head of edge $e_q = (u_q, v_q)$, $\Psi(\cdot)$ maps the path, the computing node and the start time to the shortest path. The relations of $\{t_1, t_2, \ldots, t_{q-1}, t_q, t_{q+1}, \ldots, t_p\}$ are stated as follows:

$$t_2 = t_1 + \omega_{u,v,k,l}(t_1),$$

$$t_3 = t_2 + \omega_{u,v,k,l}(t_2),$$

$$\ldots,$$

$$t_{q-1} = t_{q-2} + \omega_{u,v,k,l}(t_{q-2}),$$

$$t_q = t_{q-1} + \omega_{u,v,k,l}(t_{q-1}),$$

$$t_{q+1} = t_q + \omega_{u,v,k,l}(t_q),$$

$$\ldots,$$

$$t_p = t_{p-1} + \omega_{u,v,k,l}(t_{p-1}).$$

The computing-aware routing problem in the dynamic network $G_{SFDNM}(t)$ (i.e., $\mathcal{P}1$) is defined as finding the optimal path, denoted as $\pi_{u,v,t}$, from the source node $u$ to the destination node $v$ at time $t$ that minimizes the overall delay of each subtask. Additionally, a computing node $u_q$ on the path must be selected that minimizes the path length $L_{P_{u,v}}$, where $\Pi = \pi_{u,v,t}, u_q = \Phi(u, v, t)$ and $\Phi(\cdot)$ maps the source node, the destination node and the start time to the shortest path. Mathematically, the problem $\mathcal{P}1$ is formulated as

$$\begin{align*}
(\mathcal{P}1) : \min_{\Pi} \{\Psi(\Pi, t)\}, \quad & \Pi = \{\pi_{u,v,t}, u_q\}, \\
\text{s.t.} \quad & u, v, u_q \in V, \\
& t \geq 0.
\end{align*}$$

It could be concluded from the definitions stated above that there are some differences between $\mathcal{P}0$ and $\mathcal{P}1$. In $\mathcal{P}1$, both the edges and nodes are weighted, and there is a computing node on the path to execute the computation of the corresponding subtask.

The shortest path of $\mathcal{P}1$ contains multiple transmission edges and a computing node. Problem $\mathcal{P}1$ can be converted to $\mathcal{P}0$ when the computation is executed at the source node $u$ or the destination node $v$. Since $\mathcal{P}0$ is NP-hard, problem $\mathcal{P}1$ is NP-hard as well.

The computing-aware routing process in $G_{SFDNM}(t)$ can be divided into three stages: (1) finding the shortest path from the source node $u$ to the computing node $u_q$, (2) transmitting the raw data of subtask $\mathcal{T}_{u,k}^t$, (3) processing the computation of $\mathcal{T}_{u,k}^t$ on a specific computing node $u_q$ (3) finding the shortest path from $u_q$ to the destination node $v$ to transmit the computation result of subtask $\mathcal{T}_{u,k}^t$. It is worth noting that stage (1) and stage (3) are both typical DSSP problem (i.e., $\mathcal{P}0$) whose optimal results are $\pi_{u,u_q,t} = \Phi(u, u_q, t)$ and $\pi_{u_q,v,t} = \Phi(u_q, v, t)$, respectively. Assuming that the path length of $\pi_{u,u_q,t}$ and $\pi_{u_q,v,t}$ are $L_{\pi_{u,u_q,t}} = \psi(\pi_{u,u_q,t}, t)$ and $L_{\pi_{u_q,v,t}} = \psi(\pi_{u_q,v,t}, t)$. Then the problem $\mathcal{P}1$ can be rewritten as the following problem $\mathcal{P}2$.

**Problem $\mathcal{P}2$:** Let $G_{SFDNM}(t) = (V, E(t), W_{E}(t), W_{V}(t))$ be a directed graph. (The definition of $G_{SFDNM}(t)$ here is the same as that in (1)). For any subtask $\mathcal{T}_{u,k}^t \in \mathcal{P}_{u,k}$, $v \in V$, $t \in [0, T]$ is the node set, $\pi_{u,v,t}$ is the instant that $\mathcal{T}_{u,k}^t$ arrives at node $u_q$ (i.e., the head of edge $e_q = (u_q, v_q)$, $\Psi(\cdot)$ maps the path, $\pi_{u,u_q,t}$ and $\pi_{u_q,v,t}$ are $L_{\pi_{u,u_q,t}} = \psi(\pi_{u,u_q,t}, t)$ and $L_{\pi_{u_q,v,t}} = \psi(\pi_{u_q,v,t}, t)$. Then the problem $\mathcal{P}1$ can be rewritten as the following problem $\mathcal{P}2$.

The computing-aware routing problem in the dynamic network $G_{SFDNM}(t)$ (i.e., $\mathcal{P}1$) is defined as finding the optimal path, denoted as $\pi_{u,v,t}$, from the source node $u$ to the destination node $v$ at time $t$ that minimizes the overall delay of each subtask. Additionally, a computing node $u_q$ on the path must be selected that minimizes the path length $L_{P_{u,v}}$, where $\Pi = \pi_{u,v,t}, u_q = \Phi(u, v, t)$ and $\Phi(\cdot)$ maps the source node, the destination node and the start time to the shortest path. Mathematically, the problem $\mathcal{P}1$ is formulated as

$$(\mathcal{P}1) : \min_{\Pi} \{\Psi(\Pi, t)\}, \quad \Pi = \{\pi_{u,v,t}, u_q\},$$

s.t. $u, v, u_q \in V,$

$t \geq 0.$

For a specific subtask, $u, v, t$ are known. The mapping of $\psi(\cdot)$ is given in the definition of $\mathcal{P}0$. The mapping of $\phi(\cdot)$ can be obtained by solving the DSSP problem defined in $\mathcal{P}0$. In addition, $t_q$ and $t + g'$ can be calculated by (3d) and (3e).

Given the above discussion, a computing-aware routing process can be separated into the transmission process (i.e., stage (1) and stage (3)) and the computation process (i.e., stage (2)).
In this way, the problem $P1$ can be converted to problem $P2$ which divides the optimization procedure of $P1$ into finding the shortest transmission path with a specific computing node and finding the optimal computing node with the corresponding shortest transmission path.

VII. COMPUTING-AWARE ROUTING BASED ON SNAPSHOT-FREE DYNAMIC NETWORK MODEL

Section V provides an accurate model of the LEO satellite network studied in this article. In this section, we discuss how to solve the computing-aware routing problem (presented in Section VI-B1) in the established dynamic network model.

The computing-aware routing problem is a DSSSP problem in essence. It cannot be solved by static shortest path algorithms. Neither can dynamic Dijkstra’s algorithms solve this problem. They must update the shortest path tree for every change [59], [64], [65], suffering from the excessive computational complexity caused by the time-continuous and time-varying edge weights of the dynamic network.

A. Algorithm Design

We propose a genetic algorithm based algorithm to solve this problem, as it has similar performance to dynamic Dijkstra’s algorithms in shortest path finding in dynamic graphs [66]. Algorithm 1 outputs the set of the optimal routing path and the computing node $\Pi = \{\pi_l(u,v), u_q\}$ and the corresponding path length $L^\ast$.

Before running Algorithm 1, sets of VNs and edges should be generated based on the dynamic network model construct method introduced in Section II. Additionally, the impact factor set of edges $\Lambda_E$, the impact factor set of VNs $\Lambda_V$ and subtask $\tau_{u,k}$’s parameter set $\Lambda_{\tau_{u,k}}$ are required as input data. Algorithm 1 outputs the set of the optimal routing path and the computing node $\Pi = \{\pi_l(u,v), u_q\}$, the corresponding minimum overall delay $L^\ast$, the updated impact factor set of edges $\Lambda_E$ and the updated impact factor set of VNs $\Lambda_V$.

Except the initialization (line 1), Algorithm 1 contains three parts. The first part is the graph generation (line 2 to line 6). The algorithm prepares the key data structure $G_{SFDNM}$ and assigns weight to it according to physical constraints. Inside the algorithm, the sets of edge weights and node weights of the dynamic network model $G_{SFDNM}(t)$ are calculated based on $\Lambda_E$, $\Lambda_V$ and $\Lambda_{\tau_{u,k}}$ with Equ. (1) and (2) (line 2 and line 3). Because the data volume of $\tau_{u,k}$ can be reduced to a few bits after being computed, this article only considers the propagation delay and ignores the transmission delay in the computation result transmission process. Therefore, a new edge weight set is calculated and a new dynamic network model $G'_{SFDNM}(t)$ is generated (line 4 to line 6).

The second part is the optimal path and computing node finding. The main loop (line 7) iterates through all nodes. In each iteration, $u_q$ is selected for computing the subtask, and the overall delay under this scenario is evaluated as $L_{\text{temp}}$ which has three components (i.e., $d_1$, $d_2$ and $d_3$). $d_1$ is caused by the transmission of raw data from the source node $u$ to the computing node $u_q$. The time at which the subtask is created is $t_{u,k}$. The shortest path $\pi_1$ and the instance set $T_1$ (recording the instances when $t_{u,k}$ first arrived at each node on $\pi_1$) are first solved with GA (line 8), and the delay $d_1$ is calculated thereafter (line 9). $d_2$ is caused by the processing of the subtask. Since the subtask is arrived at $t_{u,k} + d_1$ which affects the computation delay, the delay is computed as $w_{u,k}^{t_{u,k}}(t_{u,k} + d_1)$ (see line 10). $d_3$ is caused by the computation result transmission from the computing VN $u_q$ back to the destination VN $v$. Similar to the calculation of the first part of the delay, the shortest path is first solved with GA (line 11) and the delay is calculated next (line 12). In this step, the transmission delay of each edge is ignored because the computation results are usually only a few bits; therefore, $G_{SFDNM}(t)$ rather than $G_{SFDNM}(t)$ is adopted as the input of GA. After calculating the delay under the assumption that node $u_q$ is selected for computing, the global state is updated to find the best node. Here the optimal path $P^\ast$, the optimal computing node $u_q^\ast$ and the minimum overall delay $L^\ast$ are updated in line 15 and 16, respectively. Furthermore, for updating $\Lambda_E$ and $\Lambda_V$ in the next part, the optimal path from $u$ to $u_q$' the instance set

\begin{algorithm}
\caption{GA-Based Computing-Aware Routing.}
\begin{algorithmic}
\State Data: The VN set and edge set of the dynamic network model $V, E(t)$
\State Data: Impact factor sets of edges and VNs $\Lambda_E = \{D, R\}$, $\Lambda_V = \{C, S, E\}$
\State Data: Subtask $\tau_{u,k}$’s parameter set $\Lambda_{\tau_{u,k}} = (C_{u,k}, N_{u,k}, S_{u,k}, \alpha_{u,k}, \beta_{u,k}, t_{u,k})$
\State Result: The optimal path and computing node set $\Pi = \{P^\ast, u_q^\ast\}$ and the minimum overall delay $L^\ast$
\State Result: The updated $\Lambda_E$ and $\Lambda_V$
\begin{algorithmic}
\State Initialize $P^0 = \emptyset$, $u_q^0 = \emptyset$, $L^0 = \infty$, the instant set $T^0 = \emptyset$
\State Calculate $W_{E}(t)$, $W_{V}(t)$ based on $\Lambda_E$, $\Lambda_V$ and $\Lambda_{\tau_{u,k}}$
\State with Equ. (1) and (2):
\State Set $G_{SFDNM}(t) = (V, E(t), W_{E}(t), W_{V}(t))$
\State Set $\Lambda_{\tau_{u,k}} = (C_{u,k}, N_{u,k}, S_{u,k}, \alpha_{u,k}, \beta_{u,k}, t_{u,k})$
\State Calculate $W_{E}(t)$ based on $\Lambda_E$ and $\Lambda_{\tau_{u,k}}$ with Equ. (1);
\State Set $G_{SFDNM}(t) = (V, E(t), W_{E}(t), W_{V}(t))$
\For{$u_q \in V$}
\State \begin{align*}
&\{\pi_1, T_1\} \leftarrow GA(G_{SFDNM}(t), u, u_q, t_{u,k})\
&d_1 \leftarrow L(\pi_1, t_{u,k})\
&d_2 \leftarrow w_{u,k}^{t_{u,k}}(t_{u,k} + d_1)\
&\{\pi_2, T_2\} \leftarrow GA(G_{SFDNM}(t), u_q, v, t_{u,k} + d_1 + d_2)\
&d_3 \leftarrow L(\pi_2, t_{u,k} + d_1 + d_2)\
&L_{\text{temp}} \leftarrow d_1 + d_2 + d_3\
\end{align*}
\If{$L_{\text{temp}} < L^\ast$}
\State $\Pi = \{P^\ast, u_q^\ast\} \leftarrow \{\pi_1 \cup \pi_2, u_q\}$
\State $L^\ast \leftarrow L_{\text{temp}}$
\State $\pi_1^\ast \leftarrow \pi_1$
\State $T^\ast \leftarrow T_1$
\State $T_{\text{process}} \leftarrow d_2$
\EndFor
\State end
\end{algorithmic}
\end{algorithmic}
\end{algorithm}

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Fig. 5. Centralized framework for the computing-aware routing approach. Based on the reserved channels of the satellite network, the source satellite transmits the routing requirements to the global control center to obtain its optimized routing decision.

The third part is impact factor sets update. After the second part, the route for current subtask is resolved as \( \Pi \leftarrow \{ P^*, u^*_q \} \). However, the offloading of the subtasks occupies computation and transmission resources of the network. To reflect these changes, line 23 to line 25 subtracts the resources occupied by \( \Pi \) from the impact factor values of the time when \( \tau^\Pi_{u,q} \) arrives at \( u_q \) and each edge on \( \pi \).

It is worth noting that the proposed computing-aware scheme and the ground-offloading scheme are not mutually exclusive, but complementary. If computation resources on satellites are not sufficient, tasks can be offloaded to ground servers for computation. By adding ground servers as nodes of the dynamic network model, these ground servers will be selected for computation if the delay of on-board computing is larger than the delay of ground offloading.

B. System Framework Design

We design a centralized framework which helps to manage the global task state as required by Algorithm 1. As depicted in Fig. 5, this framework utilizes a control center, specifically a LEO satellite with advanced computing capabilities, to collect and manage routing processes. The control center is responsible for gathering task information, which has lower data volume and synchronization requirements than instantaneous network state information. This allows for reduced overhead. Additionally, the control center stores and updates instantaneous network state information based on previous routing decisions to ensure alignment throughout the network.

This framework, as illustrated in Fig. 5, includes: 1) generation of tasks at the source LEO satellite, 2) local extraction of task attribute sets \( \Lambda_{\tau^\Pi_{u,k}} \), 3) transmission of task attribute sets to the control center via the reserved channels,\(^5\) 4) calculation of routing paths using the proposed algorithm with task attribute sets and the dynamic network model, 5) transmission of routing decisions back to the source satellite.

In the control center, there is a visibility table that contains information on the topology dynamics and a resource table that contains information on the resource dynamics. Using these two tables and inter-satellite communication policies, the dynamic network model can be developed. Additionally, task attribute sets collected from the entire network are stored in the control center. The proposed computing-aware routing algorithm is used to calculate the routing paths for tasks to be processed within the generated dynamic network model. In this manner, the routing decision for each task can be determined in the control center.

The framework has minimal routing overhead because of the limited transmission: only task attribute sets and routing decisions are transmitted, which incurs significantly lowered overhead compared to transmitting real-time available resources of each satellite.

VIII. Simulation Results and Analyses

In this section, we evaluate the routing scheme proposed in Section VII-A and analyze the simulation results. We aim to answer the following research questions:

- **RQ1**: How much computing capability is required for LEO satellites to make the proposed scheme reduce delay?
- **RQ2**: How does the transmission capability affect the performance of the proposed scheme?
- **RQ3**: What kind of tasks can be accelerated by the proposed scheme?

\(^5\)These reserved channels are only used to transmit a small amount of specific data (e.g. task attribute sets and routing decisions); thus, these data can be successfully transmitted as long as there is ISL between satellites. In other words, the routing path of these data is only relevant to the topology of the satellite network, which is calculated locally by the source satellite. Furthermore, the corresponding delay approximates to the propagation delay.
The following assumptions are made in the simulations:

- **Sufficient energy and storage resources are assumed for modern LEO satellites.** As shown in Table II, existing onboard computing systems can provide thousands of GFLOPS of computing capability. The impact of energy consumption will be discussed and evaluated in future work.

- **Zero computation delay is assumed for the ground-offloading scheme, as ground servers usually have large amounts of computation resources.** We round the computation delay towards zero for a fair comparison.

- **The evaluation is platform-agnostic, because a variety of configurations can be found for LEO satellites.** The focus is on how factors affect performance in general, rather than evaluating specific configurations exhaustively.

- **The comparison is at the scheme level, with the GA algorithm as the pathfinding algorithm for both schemes, and subtask as the smallest unit of transmission and computation in both schemes.** This allows for a fair comparison.

The simulation parameters are presented in Table III.

### Table III

| Parameter | Value |
|-----------|-------|
| Radius of the earth $R_e$ | 6.371 km |
| Mass of the earth $M_e$ | $5.976 \times 10^{24}$ kg |
| Earth rotation angular velocity $\omega_e$ | $7.29211510 \times 10^{-5}$ s$^{-1}$ |
| Gravitational $G$ | $6.67428 \times 10^{-11}$ m$^3$kg$^{-1}$s$^{-2}$ |
| Kepler constant $K$ | $3.9860 \times 10^{14}$ m$^3$kg$^{-1}$s$^{-2}$ |
| Velocity of light $c$ | 299,792,458 m/s |
| Number of orbits | 10 |
| Satellites per orbit | [10,12,10,12,12,10,12,10,12] |
| Orbit altitude $h$ | [200,300,400,500,600,200,300,400,500,600] km |
| Orbit inclination $i_o$ | 90° |
| Max data rate of ISL $R_{ISL}^{max}$ | 5 Gbps |
| Channel number per ISL | 1 |
| Max data rate of SGL $R_{SGL}^{max}$ | 0.2 Gbps |
| Max computing capability of satellites $C_{max}$ | 100 GFLOPS |
| The number of tasks arrived at each VN per second $\lambda$ | 1/60 |
| Distribution parameters of subtask number per task $(\mu_{\lambda}, \sigma_{\lambda})$ | (3.1) |
| Distribution parameters of subtask's computational requirement $(\mu_C, \sigma_C)$ | (50.2) GFLOP |
| Distribution parameters of sub-task's data volume $(\mu_N, \sigma_N)$ | (0.1, 0.02) Gbps |

The maximum transmission rate reflects the transmission capacity of satellites. In actual transmissions, the transmission rate is proportional to the bandwidth, which decreases as resources are occupied and increases as resources are released. Thus, the available resources are time-varying. Similarly, the available computation resources are time-varying. Accordingly, the weights of the proposed model are time-varying.

A. *Reduced Overall Delay (RQ1)*

Figs. 6 and 7 shows how the computing capability of LEO satellites affects the performance of computing-aware routing schemes. The simulation covers computing capability from 100 GFLOPS to 400 GFLOPS, and the remaining parameters follow Table III. For each computing capability, the delay of the computing-aware routing scheme is calculated, and the result is normalized with the corresponding delay of the benchmark routing scheme.

The proposed computing-aware routing scheme can reduce overall delay, as shown in Fig. 6. Even with relatively low computing capabilities, delays can be shortened (11 GFLOPS when $R_{ISL}^{max} = 0.2$ Gbps, 60 GFLOPS when $R_{SGL}^{max} = 5.0$ Gbps). For example, when picking a moderate computing capability of 100 GFLOPS, a 3.61x speedup is achieved when the SGL transmission rate is 0.2 Gbps. Similarly, a 0.34x speedup can also be observed when the SGL transmission rate is 5 Gbps. In short, the proposed scheme is both effective and feasible as existing onboard computing systems can provide the necessary computing capabilities. As the two curves in Fig. 6 have similar trends, we take the curve following $R_{SGL}^{max} = 1/25 \times R_{ISL}^{max} = 0.2$ Gbps as an example for the following analysis.

As shown in Fig. 6, the proposed computing-aware routing scheme can reduce overall delay compared to the ground-offloading scheme, particularly when computing capability is greater than 11 GFLOPS. However, when computing capability is less than 11 GFLOPS, the ground-offloading scheme performs better. In this case, tasks can be offloaded to ground servers for computation. The proposed computing-aware scheme and the ground-offloading scheme are not mutually exclusive but complementary. By adding ground servers as nodes in the dynamic network model, these ground servers will be selected for computation if the delay of on-board computing is larger than the delay of ground-offloading.

The curve in Fig. 6 also shows that as the computing capability increases, the reduction in overall delay also increases, but it begins to flatten out as the computing capability continues to increase. This is because the computation delay of the proposed scheme approaches 0 after a certain point, while the transmission delay remains constant. This results in the overall delay of the proposed scheme converging to a constant value and the curve flattening out. This diminishing return...
Fig. 7. Delay decomposition of computing-aware routing scheme v.s. ground-offloading scheme. The overall delay of computing-aware routing mainly consists of the transmission delay on ISL (#1), and the processing delay (#2). Similarly, the overall delay of ground-offloading mainly consists of the transmission delay on ISL (#3), and the transmission delay on SGL (#4).

Fig. 8. Delay of the computing-aware routing scheme under different SGL transmission rates, normalized to the ground-offloading scheme.

Fig. 9. Delay of computing-aware routing under different ISL transmission rates, normalized to the ground-offloading delay.

B. Impact From Transmission Capability (RQ2)

Figs. 8 and 9 demonstrate the impact of SGL and ISL transmission rates on the performance of the proposed computing-aware routing scheme. The simulation ranges from 0.2 Gbps to 10 Gbps for SGL transmission rates and from 0.25 Gbps to 20 Gbps for ISL transmission rates, using the parameters listed in Table III.

In Fig. 8, the leftmost points represent the current practical scenarios with SGL transmission rates of 0.2 Gbps, while the rightmost points represent ideal scenarios with SGL transmission rates of 10 Gbps. The results show that satellites with 100 GFLOPS and 1000 GFLOPS computing capability can save 84.43% and 93.86% of overall delay, respectively, with the most common SGL transmission rate setting of 0.2 Gbps. This is because the ground-offloading scheme transmits a much larger amount of data over the SGL than the proposed computing-aware scheme. As the SGL transmission rate increases, the ratio of...
delay reduction also increases, as the ground-offloading scheme transmits more data over the SGL than the proposed computing-aware scheme. However, even with a higher SGL transmission rate of 10 Gbps, the proposed scheme still offers a significant improvement of 11.07%.

Fig. 9 illustrates the performance at various ISL transmission rates. The ratio of the proposed scheme to the ground-offloading scheme can be approximated using the equation \( \frac{T_{\text{comp}}}{T_{\text{trans,SGL}}} + \frac{x}{y} \times \Delta_{\text{ISL}} \), where \( T_{\text{comp}} \) is the computation delay of the proposed scheme, \( T_{\text{trans,SGL}} \) is the transmission delay on SGL of the ground-offloading scheme, and \( \Delta_{\text{ISL}} \) is the average delay on each ISL. \( x \) and \( y \) are the average hop number of transmitting raw data on ISL for the proposed scheme and the benchmark scheme, respectively. The conclusion is that when \( T_{\text{comp}}/T_{\text{trans,SGL}} \) is greater than \( x/y \), the ratio is greater than \( x/y \), and vice versa. As the ISL transmission capability increases, the delay on each ISL will decrease and converge to zero, resulting in all three curves converging to \( T_{\text{comp}}/T_{\text{trans,SGL}} \). This conclusion allows us to determine whether to perform onboard computing or the ground-offloading scheme by estimating \( T_{\text{comp}}/T_{\text{trans,SGL}} \).

The proposed scheme outperforms the ground-offloading scheme for all the realistic ISL/SGL configurations used in simulations. For the worst case of ISL (\( C_{\text{max}} = 20 \) GFLOPS, \( R_{\text{ISL}}^{\text{max}} = 20 \) Gbps), the proposed scheme still reduces 37.12% of the baseline delay. For the worst case of SGL (\( C_{\text{max}} = 100 \) GFLOPS, \( R_{\text{SGL}}^{\text{max}} = 10 \) Gbps), the proposed scheme still reduces 11.07% of the baseline delay.

C. Impact From Task Properties (RQ3)

Figs. 10, 11, and 12 demonstrate the impact of the data volume and computational requirement of subtasks on the performance of the proposed computing-aware routing scheme. The simulation ranges from 1 MB to 1 GB for data volumes and from 50 Giga floating-point operations (GFLO) to 800 GFLO for computational requirements, using the parameters listed in Table III. In these figures, the delay of the proposed computing-aware routing scheme is calculated and normalized against the corresponding delay of the ground-offloading scheme.

Fig. 10 shows that the proposed computing-aware routing scheme can significantly reduce overall delay over a wide range of data volumes (greater than 4 MB) with current network settings. As the data volume of each subtask increases, the superiority of the computing-aware routing scheme becomes more pronounced. This is because the computing-aware routing scheme reduces the amount of data to be transferred by replacing transmissions of large-scale raw data with final results. This reduced demand also opens up opportunities for future data-intensive applications; for example, for applications transferring 1 GB of data, the task execution efficiency can be boosted by 17.66x with computing-aware routing.
Fig. 11 illustrates the effect of computational requirements on the performance of the proposed computing-aware routing scheme. The simulation ranges from 50 GFLO to 800 GFLO for computational requirements, and the remaining parameters follow Table III.

From the figure, it can be concluded that the computing-aware routing scheme has linear scalability concerning the computational requirements of each subtask. Because the onboard computation resources are limited, the proposed computing-aware routing scheme is not intuitively suitable for tasks with extreme computation demands. However, for most tasks (400 GFLO or less), the proposed computing-aware routing scheme still performs well. In cases where the computing-aware routing scheme is not the best approach, its performance degrades gracefully in a linear manner as the computational requirements increase.

Fig. 12 provides a comprehensive view of the delay of the proposed computing-aware routing scheme, taking into account both the data volume and computational requirements of subtasks. It offers a strategy for compensating for the limitations of the computing-aware routing scheme for tasks with high computation costs. Specifically, the decision on whether to use the computing-aware routing scheme or fall back to the conventional ground-offloading scheme for a specific task can be pre-determined by the control center.

Once the satellite network has been deployed, the parameters of onboard resources (e.g., computing capability) and the data rate of ISLs and SGLs are known in advance. Thus, the ground station can predict the overall performance of the computing-aware routing scheme for a specific task by considering its computational requirements and data volume. The evaluation results (as shown in Fig. 12) can be uploaded to the satellite network in advance, and all newly generated tasks can then follow the predetermined threshold accordingly.

The proposed scheme is applicable to most types of tasks. Unless the task has minimal data volume (less than 4 MB) or extremely high computational requirements (more than 400 GFLO), the proposed scheme can be used to improve offloading performance.

IX. CONCLUSION

This article addresses the challenges in LEO satellite networks, including the highly dynamic network topology, limited onboard resources, and intensive computational demands. To tackle these challenges, a computing-aware routing scheme is proposed. The scheme models the LEO satellite network as a snapshot-free dynamic network model with time-varying weights, formulates the computing-aware routing problem as a combination of multiple DSSSP problem, and uses a GA-based method to approximate the results in reasonable time. Simulation results show the effectiveness of the proposed approach, with the ability to reduce overall delay in a wide range of network settings. Future work includes further optimization of the proposed scheme to reduce complexity and studying the task splitting mechanism in computing-aware routing.

REFERENCES

[1] A. J. Plaza and C.-J. Chang, High Performance Computing in Remote Sensing, Boca Raton, FL, USA: CRC, 2007.
[2] L. Zhang, L. Zhang, and B. Du, “Deep learning for remote sensing data: A technical tutorial on the state of the art,” IEEE Geosci. Remote Sens. Mag., vol. 4, no. 2, pp. 22–40, Jun. 2016.
[3] A. Plaza, Q. Du, Y.-L. Chang, and R. L. King, “High performance computing for hyperspectral remote sensing,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 4, no. 3, pp. 528–544, Sep. 2011.
[4] C.-J. Chang, H. Ren, and S.-S. Chiang, “Real-time processing algorithms for target detection and classification in hyperspectral imagery,” IEEE Trans. Geosci. Remote Sens., vol. 39, no. 4, pp. 760–768, Apr. 2001.
[5] P. Ghamsiri et al., “Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art,” IEEE Geosci. Remote Sens. Mag., vol. 5, no. 4, pp. 37–78, Dec. 2017.
[6] A. J. Plaza, “Special issue on architectures and techniques for real-time processing of remotely sensed images,” J. Real-Time Image Process., vol. 4, no. 3, pp. 191–193, 2009.
[7] A. Alonso, M. Reyes, and Z. Sodnik, “Performance of satellite-to-ground communications link between artemis and the optical ground station,” in Proc. Opt. Atmospheric Propag. Adaptive Syst. VII, 2004, pp. 372–383.
[8] B. Yost et al., “State-of-the-art small spacecraft technology,” 2021.
[9] T. M. Lovelly and A. D. George, “Comparative analysis of present and future space-grade processors with device metrics,” J. Aerosp. Inf. Syst., vol. 14, no. 3, pp. 1–14, 2017.
[10] X. Pan, R. Liu, and X. Lv, “Low-complexity compression method for hyperspectral images based on distributed source coding,” IEEE Geosci. Remote Sens. Lett., vol. 9, no. 2, pp. 224–227, Mar. 2012.
[11] P. Liu, H. Chen, S. Wei, L. Li, and Z. Zhu, “Hybrid traffic-detour based load balancing for onboarding routing in LEO satellite networks,” China Commun., vol. 15, no. 6, pp. 28–41, 2018.
[12] M. Madni, S. Irramanesh, and R. Raad, “DTN and Non-DTN routing protocols for inter-cubesat communications: A comprehensive survey,” Electronics, vol. 9, no. 3, 2020, art. no. 482.
[13] S. El Alaoui, “Routing optimization in interplanetary networks,” 2015.
[14] F. Tang, H. Zhang, and L. T. Yang, “Multipath cooperative routing with efficient acknowledgement for LEO satellite networks,” IEEE Trans. Mobile Comput., vol. 18, no. 1, pp. 179–192, Jan. 2019.
[15] M. Werner, “A dynamic routing concept for ATM-based satellite personal communication networks,” IEEE J. Sel. Areas Commun., vol. 15, no. 8, pp. 1636–1648, Oct. 1997.
[16] T. Zhang, J. Li, H. Li, S. Zhang, P. Wang, and H. Shen, “Application of time-varying graph theory over the space information networks,” IEEE Netw., vol. 34, no. 2, pp. 179–185, Mar./Apr. 2020.
[17] H. Li, T. Zhang, Y. Zhang, K. Wang, and J. Li, “A maximum flow algorithm based on storage time aggregated graph for delay-tolerant networks,” Ad Hoc Netw., vol. 59, pp. 63–70, 2017.
[18] J. A. Ruíz de Azúa, A. Calveras, and A. Camps, “Internet of Satellites (IoSat): Analysis of network models and routing protocol requirements,” IEEE Access, vol. 6, pp. 20390–20411, 2018.
[19] E. Elici, I.F. Akylzul, and M.D. Bender, “A distributed routing algorithm for datagram traffic in LEO satellite networks,” IEEE/ACM Trans. Netw., vol. 9, no. 2, pp. 137–147, Apr. 2001.
[20] S. C. Burleigh, “Contact graph routing,” NASA Tech. Briefs, Tech. Rep. NPO-45488, 2011.
[21] G. Araniti et al., “Contact graph routing in DTN space networks: Overview, enhancements and performance,” IEEE Commun. Mag., vol. 53, no. 3, pp. 38–46, Mar. 2015.
[22] T. Zhang, H. Li, S. Zhang, J. Li, and H. Shen, “Stag-based QoS support routing strategy for multiple missions over the satellite networks,” IEEE Trans. Commun., vol. 67, no. 10, pp. 6912–6924, Oct. 2019.
[23] E. Köhler, K. Langkau, and M. Skutella, “Time-expanded graphs for flow-dependent transit times,” in Proc. Eur. Symp. Algorithms, 2002, pp. 241–251.
[24] C. Jiang and X. Zhu, “Reinforcement learning based capacity management in multi-layer satellite networks,” IEEE Trans. Wireless Commun., vol. 19, no. 7, pp. 4685–4699, Jul. 2020.
[25] D. Zhou, M. Sheng, B. Li, J. Li, and Z. Han, “Distributionally robust planning for data delivery in distributed satellite cluster network,” IEEE Trans. Wireless Commun., vol. 18, no. 7, pp. 3642–3657, Jul. 2019.
[26] B. George and S. Shekhar, “Time-aggregated graphs for modeling spatio-temporal networks,” in Journal on Data Semantics XI. Berlin, Germany: Springer, 2008, pp. 191–212.
[37] N. Waqar, S. A. Hassan, A. Mahmood, K. Dev, D.-T. Do, and M. Gid-
Shengli Zhang (Senior Member, IEEE) received the B.Eng. degree in electronic engineering and the M.Eng. degree in communication and information engineering from the University of Science and Technology of China, Hefei, China, in 2002 and 2005, respectively, and the Ph.D. degree from the Department of Information Engineering, Chinese University of Hong Kong, Hong Kong, in 2008. He joined the Communication Engineering Department, Shenzhen University, Shenzhen, China, where he is currently a Full Professor. From 2014 to 2015, he was a Visiting Associate Professor with Stanford University, Stanford, CA, USA. He is the pioneer of Physical-Layer Network Coding. He has authored or coauthored several IEEE top journal papers and ACM top conference papers, including IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE TRANSACTIONS ON COMMUNICATIONS, and ACM mobihoc. His research interests include blockchain, physical-layer network coding, and wireless networks. He was the Editor of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE WIRELESS COMMUNICATIONS LETTERS, and IET Communications. He was a TPC Member in several IEEE conferences.

Qingxia Chen received the Ph.D. degree in information and communication engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2018. She is currently an Associate Researcher with the Qian Xuesen Laboratory of Space Technology, China Academy of Space Technology, Beijing. Her research interests include future network architecture design, space intelligent information networking, and dispersed computing.

Houtian Wang received the Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, China. He is currently an associate Researcher with the Qian Xuesen Space Technology Laboratory, China Academy of Space Technology, Beijing. His research interests include spatial intelligent information networking, mobile cloud computing, and software defined networking.

Mingzhe Wang received the B.S. degree in software engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2018 and the Ph.D. degree in software engineering from Tsinghua University, Beijing, in 2023. His research interests revolve around optimization problems, programming languages, and software testing. He is dedicated to advancing knowledge and innovation in both academia and industry.

Naijin Liu received the Ph.D. degree from the China University of Science and Technology, New Taipei, Taiwan. He is currently the deputy Director of Qian Xuesen Space Technology Laboratory, China Academy of Space Technology, Beijing, China. He is also a Professor with the School of Mechanical Engineering, Xi’an Jiaotong University, Xi’an, China. His research interests include satellite communication and space intelligent information networking. He is a council Member of the Chinese Society of Astronautics.