Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 4899124, 19 pages
https://doi.org/10.1155/2023/4899124

Research Article

An Improved DTN Scheme for Large-Scale LEO Satellite Networks

Gaosai Liu,[1,2] **Xinglong Jiang**,[1,2,3] **Huawang Li**,[1,2,3] **Siyue Sun**,[1,2,3] **Zhenhua Zhang**,[1,2] **and Guang Liang**[1,2,3]

[1]*Innovation Academy for Microsatellites of CAS, Shanghai 201204, China*
[2]*University of Chinese Academy of Sciences, Beijing 100049, China*
[3]*Shanghai Engineering Center for Microsatellites, Shanghai 201204, China*

Correspondence should be addressed to Xinglong Jiang; jiangxinglong@microsate.com

Received 4 September 2022; Revised 31 October 2022; Accepted 24 November 2022; Published 21 February 2023

Academic Editor: Yanpeng Dai

A large-scale low Earth orbit (LEO) satellite network has the characteristics of a complex link environment, a large number of satellites, and the limited resources of a single satellite. Applying traditional routing algorithms has disadvantages such as high overhead, high end-to-end latency, and low message delivery rate. This paper proposes an improved delay tolerant (DTN) scheme for large-scale LEO satellite networks (LIDTN) to improve transmission efficiency and reduce the resource overhead and end-to-end latency of large-scale satellite networks. This scheme improves the network performance in three aspects: next hop selection, congestion control mechanism, and acknowledgment mechanism. For the next hop selection, we propose an equivalent distance and priori knowledge-based forwarding strategy (EPFS), which has the advantages of low overhead, loop avoidance, and fast convergence. For congestion control, we put forward an emergency function-based bundle drop algorithm (EBDA). For acknowledging, we propose the virtual acknowledgment algorithm (VAA) by combining the characteristics of many path hops and high link disruption rates in large-scale constellations. Finally, we simulate and verify the LIDTN scheme on the OneWeb constellation. The results show that the LIDTN scheme is suitable for large-scale constellations, the EPFS algorithm can reduce the network overhead during data transmission, EBDA can reduce the bundle drop rate, and VAA can reduce the end-to-end latency. LIDTN provides a new solution for large-scale constellation communication.

1. Introduction

Large-scale low Earth orbit (LEO) satellite constellations have uneven service distribution, frequent link switching, and a high probability of laser link interruption [1, 2], which brings more significant challenges to the design of routing algorithms. In recent years, routing algorithms for large-scale LEO constellations have gradually become an important issue in China and abroad [3–7]. Based on link characteristics, existing constellation routing algorithms are divided into delay tolerant (DTN) and nondelay tolerant (non-DTN) routing [8]. In particular, non-DTN routing protocols for single-layer satellite networks can be subdivided into virtual topology-based and node-based routing algorithms [9–12]. Virtual topology-based routing algorithms suffer from multiple time slice divisions in large-scale constellation scenarios, leading to high routing overhead and increased packet loss rates [13]. The virtual node-based routing algorithm will also result in high switching and computational resource overhead due to many periodic routing table updates. Non-DTN protocols oriented to multilayer satellite networks use high-orbiting satellites to manage low-orbiting, increasing construction costs and complexity [14–17]. Compared with non-DTN traditional constellations routing algorithms, DTN algorithms have significant advantages in resource consumption and have long been used in deep space-oriented uncertain networks [18–21], which have more significant potential for large-scale constellations network routing. In this paper, we will fully use the regularity and orbit predictability of large-

scale constellation networks, combined with the resource consumption advantage of DTN, to conduct algorithm research for large-scale single-layer LEO satellite networks.

In recent years, the application of DTN in the LEO satellite network has attracted significant attention. The predictable trajectory of nodes in the network is a characteristic of satellite networks, and nodes in traditional DTN application scenarios do not have this characteristic. Existing DTN routing techniques are classified as flooding-based, quota-based, and predictive forwarding-based [22–26]. Based on the above three DTN routing algorithms, the researchers took the reduction of packet copy number and end-to-end latency as the research objective. Literature [27, 28] are early studies on the application of DTN in satellite networks. They discussed the advantages and disadvantages of DTN applications in satellite networks and concluded that DTN has higher robustness than Performance Enhancing Proxies (PEPs). Literature [29] proposes the based on throughput constrained minimum delay backup path of contact graph routing (CGR) algorithm (TCMDB_CGR) for LEO constellations characteristics by optimizing the traditional CGR algorithm, which reduces the packet loss rate and average end-to-end latency when the link bandwidth is specific. Literature [30] investigates the congestion problem in DTN for satellite networks and proposes a fragmented congestion control algorithm (FCC) that reduces the packet loss rate. However, the number of iterations is related to the number of satellite cache bundles, and the increase in the number of bundles increases the computational overhead of the satellite. In literature [31], to reduce the replication overhead and forward packets efficiently, a quasideterministic network (DQN) is proposed. However, the application scenario of this algorithm is the network without intersatellite links (ISLs), and the Earth station is required to relay the transmission of data packets. In literature [32], in order to solve the problem of high latency and low network efficiency of DTN for satellite networks, a finite replication algorithm modified prophet with limited copies (MPWLC) based on service probability is proposed to reduce network overhead and average hop count.

In the above literature, the researchers consider the characteristics of satellite resource-constrained and unstable ISLs. The traditional DTN routing algorithm is optimized to make DTN more suitable for satellite networks by reducing packet replication, packet loss, and end-to-end latency. However, the application scenarios of DTN technology are mostly small-scale single-layer or multilayer constellations. There is no research on the application of large-scale single-layer LEO satellite networks. In addition, to maximize the amount of data received by the data processing center (DPC), the literature [33] designed the space-air-ground integrated network (SAGIN) model that combines LEO satellites, spacecraft, ground stations, and DPC devices, significantly increasing the amount of data received by the DPC. The storage, carrying, and forwarding mechanism in this reference is similar to DTN. In this paper, DTN technology is applied to large-scale LEO constellations for the first time, and a LIDTN scheme is proposed. The scheme reduces resource consumption and improves adaptability in large-scale LEO networks by optimizing the design of next hop selection, congestion control mechanisms, and acknowledgment mechanisms. The simulation results show that LIDTN performs better in reducing storage space and computation overhead, end-to-end latency, and bundle loss rate than traditional DTN technology. The main contributions of this article are as follows:

(1) We are combining the characteristics of storage, carrying, and forwarding of DTN and the predictable characteristics of satellite trajectory. An end-to-end intersatellite forwarding algorithm based on equivalent distance and priori knowledge is proposed for the first time. The algorithm can effectively reduce the routing overhead of large-scale constellations, and the storage and computing overhead of the next hop calculated by a single satellite is less affected by the constellation size. The satellite using this algorithm can select the next hop according to the equivalent distance and priori knowledge without storing the topology of the entire network. When the ISL of reaching the next hop fails, this algorithm can quickly select the new next hop according to the equivalent distance and priori knowledge without flooding

(2) In order to solve the congestion problem in the DTN carrying stage, a bundle dropping algorithm based on the emergency function was proposed according to the priority and time to live (TTL) value of the bundle. This algorithm can not only minimize the dropping rate of the bundle but also provide a priori knowledge for the subsequent bundle forwarding through the acknowledgment

(3) The virtual acknowledgment algorithm for large-scale constellations is proposed for the problem of the low efficiency of existing acknowledgment algorithms. This algorithm reduces the retransmission delay of the bundle and reduces the resource overhead of retransmission due to bundles and acknowledgment packets transmission incorrect or dropped. The simulation proves that the algorithm is better in the satellite network with more hops. Therefore, this algorithm is more suitable for a large-scale constellation. This algorithm is unnecessary for small-scale satellite networks, such as constellations with average hops of less than five

The rest of this paper is organized as below. Section II describes the system model and completes the construction of the constellation model, the ISL model, and the Earth station model. Section III completes the problem definition of bundle forwarding, congestion control mechanism, and acknowledgment mechanism. Section IV describes the specific implementation of the LIDTN scheme. The performance of the LIDTN scheme is simulated and analyzed in section V. Finally, the paper is summarized in section VI.

2. System Model

This section is aimed at establishing a large-scale satellite topology and ISL model. Furthermore, three pairs of Earth stations are selected for modeling based on the geographical distribution of actual Earth stations.

2.1. Constellation Model. OneWeb constellation has been one of the hotspots of large-scale constellation research in recent years. The OneWeb constellation is planned to carry ISLs in the future. Therefore, the first-generation OneWeb constellation is used as the modeling object in this paper, and the parameters related to the OneWeb constellation are shown in Table 1 [34, 35]. The topology of the constructed OneWeb constellation is shown in Figure 1. This paper defines the OneWeb constellation as an undirected graph $G = (V, E)$. V represents the satellite nodes and defines $V = \{v_{1,1}, v_{1,2}, \cdots, v_{n,m}, \cdots, v_{\mathcal{N},\mathcal{M}}\}$. n represents the orbit number, m represents the satellite number in orbit, and there are a total of $\mathcal{N}$ orbits in the constellation, each with $\mathcal{M}$ satellites. The set of satellites located in the crossing seam is defined as $\bar{v}$, $v_{1,m} \in \bar{v}$, and $v_{N,m} \in \bar{v}$. E represents the ISLs in the constellation. The Earth station is defined as ES_i, and i is the Earth station label. As shown in Figure 1, the Earth station is connected to the constellation through an Earth-satellite link (ESL). Earth stations connected to different satellites are connected by ISL relay [36].

2.2. ISL Establishment Rules. As shown in Figure 2, the satellites in the constellation can establish ISLs with the four surrounding satellites, which are the top and bottom satellites in the same orbit and the left and right satellites in adjacent orbits. In Figure 2, there are three types of intersatellite links: intraplane ISLs, fixed interplane ISLs, and variable interplane ISLs. During constellation operation, the satellites connected by the intraplane ISLs and the fixed interplane ISLs are fixed. The variable interplane ISLs connect the satellites on both sides of the crossing seam. The satellites connected by each variable interplane ISLs change dynamically with time. For the OneWeb constellation shown in Figure 1, the crossing seam does not affect the satellites in orbit 2 to 17. Any satellite in orbit 2 to orbit 17 can establish ISLs with the four satellites surrounding it. The orbit 1 and 18 satellites can only establish fixed ISLs with the three surrounding satellites due to the effect of the crossing seam. The satellites in orbit 1 and the satellites in orbit 18 have an unfixed ISL with each other. The ISLs of $v_{n,m}$ are represented as $e^x_{n,m}$, and x represents the direction of $v_{n,m}$ ISLs. $x = \{U, D, L, R, \mathrm{FL}\}$ represents the ISLs in the up, down, left, and right directions and feeder links, respectively. The feeder link connects to the feeder satellite and the Earth station. The feeder satellite represents the satellite that directly downloads data to the Earth station.

The satellites on both sides of the crossing seam establish ISLs, which can effectively reduce the transmission delay of the bundle [37]. This paper analyzes the time to establish ISLs between the satellites on both sides of the crossing seam in one period. Due to the symmetry of satellite orbit distribution, the time of establishing ISLs between satellites on both sides of the crossing seam is analyzed by taking the example of establishing ISLs between $v_{1,1}$ and each satellite in orbit 18. The orbital period of the OneWeb is $2\pi\sqrt{(R+H)^3/\mathrm{GM}} \approx 6555s$, where R is the Earth's radius, H is the altitude of OneWeb from the ground, and GM is the gravitational coefficient of the Earth. The shortest distance is used as the basis for establishing the ISLs between the satellites on both sides of the crossing seam. Since the satellites are in orbit 1 and 18 in opposite directions, $v_{1,1}$ has at least two opportunities to establish a link with each satellite in orbit 18 in one period. As shown in Figure 3, the start to established ISL time between $v_{1,1}$ and each of the satellites in orbit 18 is shown in one run cycle. On the connection line $(t_1, t_2, t_3, \cdots, t_n)$ in Figure 3, they represent the time to

Table 1: Constellation parameters.

Parameters	Value
Type of satellites	LEO
Altitude	1200 km
Number of satellites per plane	36
Number of planes	18
Eccentricity	0
Inclination	87.9 degrees
Phase	5 degrees

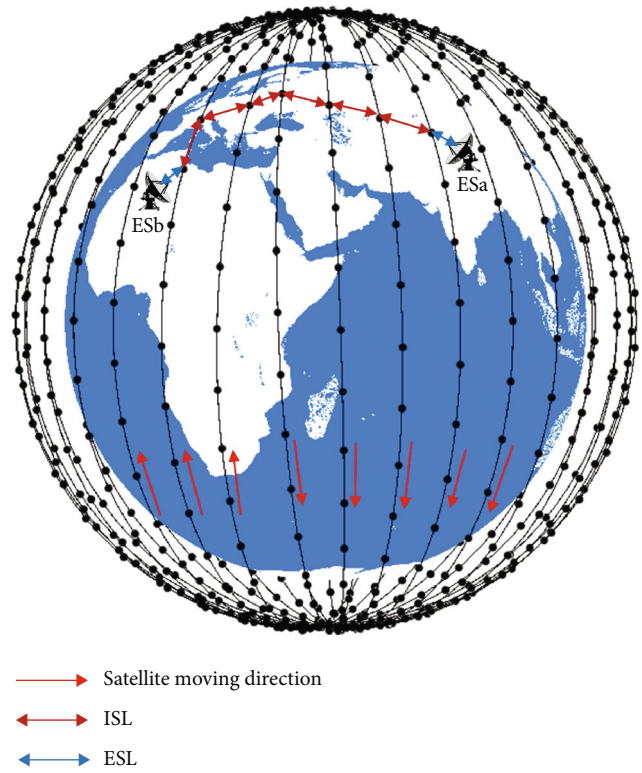

Figure 1: OneWeb constellation topology.

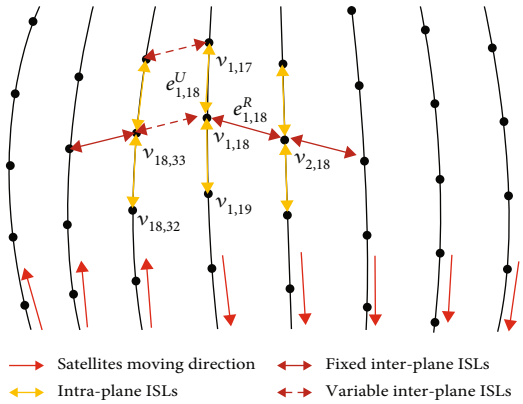

Figure 2: ISL schematic.

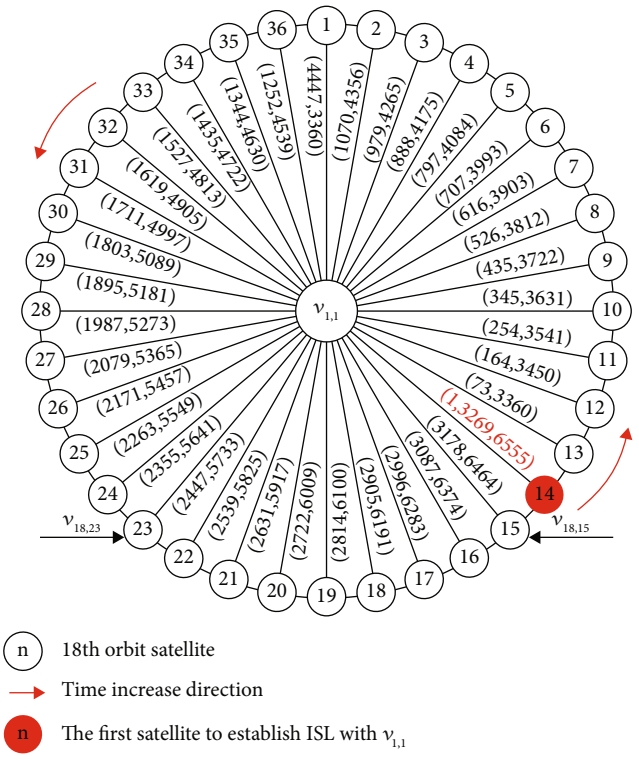

Figure 3: The time for $v_{1,1}$ to establish an ISL with the satellites in the 18th orbit.

establish the ISLs for the nth time. For example, the red fonts (1, 3269, and 6555) in the figure represent $v_{1,1}$ establishing an ISL with $v_{18,14}$ at 1 s, 3269 s, and 6555 s, respectively. $v_{1,1}$ disconnects the ISL with $v_{18,14}$ at 73 s and establishes a new ISL with $v_{18,13}$.

2.3. Earth Stations' Selection. In order to verify the effectiveness of the LIDTN scheme, three pairs are selected as modeling objects among the existing ground stations in this paper. Six Earth stations are distributed on six continents, for example, choose an Earth station in Asia, Africa, Europe, Oceania, South America, and North America, respectively. The Earth stations' names, labels, and locations are shown in Figure 4. We use ES_3 in Europe and ES_1 in Asia, which are in the northern hemisphere, as a pair of mutual transceiver Earth stations. ES_2 in South America and ES_6 in Oceania in the southern hemisphere serve as a pair of mutual transceiver Earth stations. ES_4 in North America, located in the northern hemisphere, and ES_5 in Africa, located in the southern hemisphere, act as a pair of mutual transceiver Earth stations. The selection of access satellites for Earth stations is based on the closest distance principle. According to the satellite run cycle, the number of satellites connected to the Earth stations at a given moment is fixed. The Earth stations automatically calculate the access satellites when sending data.

3. Problem Definition

The LIDTN scheme proposed in this paper has three problems to be solved.

(1) $v_{n,m}$ selects which of the surrounding satellites as the next hop. That is the next hop selection problem

(2) How to deal with congestion during forwarding

(3) How to improve the efficiency of the acknowledgment

The selection of the next hop directly affects link congestion and the transmission path of acknowledgments. Acknowledgments and congestion control provide a priori knowledge for subsequent next hop selection.

3.1. Next Hop Selection Problem. During its operation, the satellites obtain their positions in real-time through the global positioning system (GPS) and send position information to the surrounding four satellites through ISLs. The surrounding satellites (SS) are defined as those capable of establishing a direct ISL with the satellite. $SS_{n,m}$ represents the surrounding satellite set of $v_{n,m}$. $SS_{n,m} \subseteq \{v^U_{n,m}, v^D_{n,m}, v^L_{n,m}, v^R_{n,m}\}$, $v^U_{n,m}$, $v^D_{n,m}$, $v^L_{n,m}$, and $v^R_{n,m}$, respectively, represent the up, down, left, and right four adjacent satellites of $v_{n,m}$. The position information of $v_{n,m}$ at moment t is defined as $\mathscr{L}^t_{n,m} = (\mathrm{lon}^t_{n,m}, \mathrm{lat}^t_{n,m}, h_{n,m})$, in which $\mathrm{lon}^t_{n,m}$ and $\mathrm{lat}^t_{n,m}$ are the longitude and latitude corresponding to the subsatellite point of $v_{n,m}$ at time t, and $h_{n,m}$ is the altitude of $v_{n,m}$. The location of the ES_i is defined as $\mathscr{L}_i = (\mathrm{lon}_i, \mathrm{lat}_i)$. The distance $\mathscr{D}^i_{n,m}$ between ES_i and $v_{n,m}$ is shown in Equation (1) [38]. The distance $\mathscr{D}^{n',m'}_{n,m}$ between $v_{n,m}$ and $v_{n',m'}$ is shown in Equation (2), where $v_{n',m'} \in SS_{n,m}$.

$$\mathscr{D}^i_{n,m} = f_{v\leftrightarrow \mathrm{ES}}\left(\mathrm{lon}^t_{n,m}, \mathrm{lat}^t_{n,m}, h_{n,m}, \mathrm{lon}_i, \mathrm{lat}_i\right), \tag{1}$$

$$\mathscr{D}^{n',m'}_{n,m} = f_{v\leftrightarrow v}\left(lon^t_{n,m}, lat^t_{n,m}, h_{n,m}, lon^t_{n',m'}, lat^t_{n',m'}, h_{n',m'}\right). \tag{2}$$

The next hop selection is determined by the state of $v_{n,m}$. Defining the state of $v_{n,m}$ at moment t as $\mathbb{S}^t_{n,m}$. $\mathbb{S}^t_{n,m}$ is determined by the following inputs. $v_{n,m}$ obtains real-time

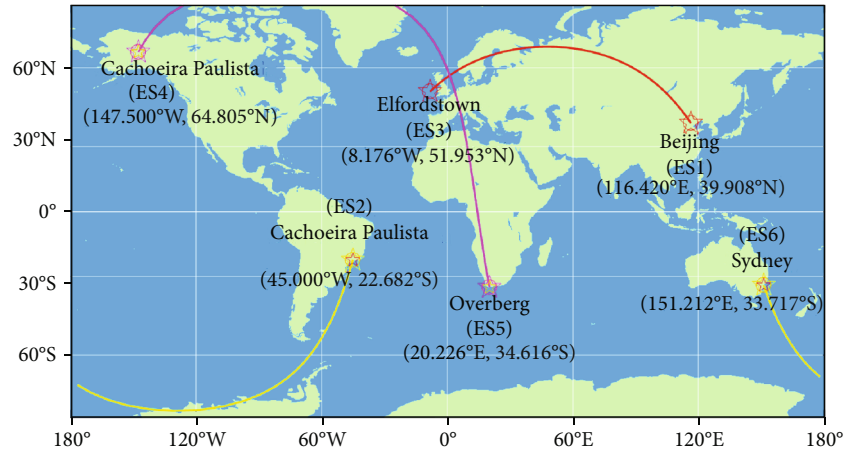

Figure 4: Earth station distribution.

location information of $SS_{n,m}$, congestion status (a priori knowledge) of $SS_{n,m}$ by acknowledgment factor, and queuing status of each port sent by $v_{n,m}$ to $SS_{n,m}$. The objective of $v_{n,m}$ at moment t is to find the optimal strategy $\pi_{n,m}^{t,*}$ to forward the bundle to one of the satellites in $SS_{n,m}$. $\pi_{n,m}^{t,*}$ is defined as shown in Equation (3). When the state is $\mathbb{S}_{n,m}^{t}$, the set of actions that $v_{n,m}$ can execute is $A_{n,m}^{t}$. We aim to find the optimal action $a_{n,m}^{t,*}$ to achieve strategy optimality.

$$\begin{aligned} &\pi_{n,m}^{t,*} \longleftarrow \left\{ a_{n,m}^{t,*} \middle| \mathbb{S}_{n,m}^{t} \right\} a_{n,m}^{t,*} \in A_{n,m}^{t}, \\ &A_{n,m}^{t} = \left\{ e_{n,m}^{t,U}, e_{n,m}^{t,D}, e_{n,m}^{t,L}, e_{n,m}^{t,R}, e_{n,m}^{t,FL} \right\}. \end{aligned} \tag{3}$$

Feeder satellites have up to five selections for the next hop, and other satellites have up to four selections for the next hop. The next hop selected by $v_{n,m}$ at moment t is defined as $N_{n,m}^{t}$. $N_{n,m}^{t}$ is the satellite corresponding to action $a_{n,m}^{t,*}$. For example, if $a_{n,m}^{t,*} = e_{n,m}^{t,D}$, then $N_{n,m}^{t} = v_{n,m}^{D}$. The calculation of $a_{n,m}^{t,*}$ is shown in

$$\begin{aligned} a_{n,m}^{t,*} &= \underset{a_{n,m}^{t}}{\operatorname{argmin}} \left(d_{n',m'}^{t,j} \right), \\ d_{n',m'}^{t,j} &= \mathscr{D}_{n',m'}^{j} + \frac{\mathscr{Q}_{n,m}^{n'm'} + \mathfrak{F}(B_0)}{v} c + \mathscr{F}_{\text{ack}}, \\ &v_{n',m'} \in SS_{n,m}, \end{aligned} \tag{4}$$

where $d_{n',m'}^{t,j}$ is the equivalent distance between $v_{n',m'}$ and ES_j at moment t, after the unification of the dimension of quantity. $\mathscr{D}_{n',m'}^{j}$ is the distance between $v_{n',m'}$ and ES_j. v is the data transmission rate, and c is the data propagation rate. $\mathscr{Q}_{n,m}^{n'm'}$ is the length of the port queue sent from $v_{n,m}$ to $v_{n',m'}$. B_0 represents the bundle currently being processed by the satellite. $\mathfrak{F}(B_0)$ represents the size of the memory space occupied by B_0. $\mathscr{F}_{\text{ack}}$ represents the acknowledgment factor.

3.2. Congestion Problem Definition. Satellite nodes cache bundles that have not received an acknowledgment. When the satellite cache space is insufficient, the bundles need to be dropped, defining $\mathscr{C}_{n,m}$ and $\mathscr{C}_{n,m}^{R}$ as the total cache capacity and the remaining cache capacity of $v_{n,m}$, respectively. When $\mathscr{C}_{n,m}^{R}$ satisfies in Equation (5), it is necessary to drop some of the bundles in the cache. ξ is the number of bundles in the cache.

$$\mathscr{C}_{n,m}^{R} < \mathfrak{F}(B_0) \mathscr{C}_{n,m}^{R} = \mathscr{C}_{n,m} - \sum_{i=1}^{\xi} \mathfrak{F}(B_i). \tag{5}$$

Which bundles are dropped when congestion occurs? How to mark the dropped bundles is the problem to be solved in this paper.

(1) Which bundles are dropped from $v_{n,m}$? Define the emergency function of B_i as [30]:

$$\begin{aligned} \mathscr{A}(B_i) &= \begin{cases} \text{PRI}_i & \text{PRI}_i \neq \text{PRI}_j, \\ 1/\text{TTL}_i & \text{PRI}_i = \text{PRI}_j, \end{cases} \\ \text{i.e.} \mathscr{A}(B_i) &= [\text{PRI}_i, 1/\text{TTL}_i], \end{aligned} \tag{6}$$

where TTL_i and PRI_i represent the time to live and the priority of B_i, respectively. TTL_j and PRI_j represent the time to live and the priority of B_j, respectively. If two bundles have the same priority, that is, $\text{PRI}_i = \text{PRI}_j$, then $\mathscr{A}(B_i) = 1/\text{TTL}_i$. If the priorities of the two bundles are not equal, that is, $\text{PRI}_i \neq \text{PRI}_j$, then $\mathscr{A}(B_i) = \text{PRI}_i$. If $\mathscr{A}(B_i) > \mathscr{A}(B_j)$, the bundle B_j is discarded first. The initial values of priority are equal. Definitions are as follows:

$$\begin{aligned} &B_k \in \tilde{B} B_k \text{ s.t.} \mathscr{A}(B_k) < \mathscr{A}(B_0), \\ &k = 1, 2 \cdots \xi, \end{aligned} \tag{7}$$

where B_0 represents the bundle currently being processed by

the satellite. B_k represents a bundle with an emergency function value less than $\mathscr{A}(B_0)$. $\tilde{B}$ represents the bundle set whose emergency function value of the cached bundles in the satellite is less than $\mathscr{A}(B_0)$. The relationship between $\mathscr{C}^R_{n,m}$, B_0, and $\tilde{B}$ is shown in Equation (8), where $\mathfrak{F}(\mathrm{Tag}_{\tilde{B}})$ is the cache capacity occupied by the $\tilde{B}$ tag after dropping the set $\tilde{B}$. When $\mathscr{C}^R_{n,m} + \mathfrak{F}(\tilde{B}) = \mathfrak{F}(B_0) + \mathfrak{F}(\mathrm{Tag}_{\tilde{B}})$ is satisfied, retain B_0 and drop all bundles in set $\tilde{B}$. When $\mathscr{C}^R_{n,m} + \mathfrak{F}(\tilde{B}) < \mathfrak{F}(B_0) + \mathfrak{F}(\mathrm{Tag}_{\tilde{B}})$ is satisfied, drop B_0. When $\mathscr{C}^R_{n,m} + \mathfrak{F}(\tilde{B}) > \mathfrak{F}(B_0) + \mathfrak{F}(\mathrm{Tag}_{\tilde{B}})$ is satisfied, B_0 is retained, and all or part of the bundles in set $\tilde{B}$ is dropped.

$$\begin{cases} \mathscr{C}^R_{n,m} + \mathfrak{F}(\tilde{B}) > \mathfrak{F}(B_0) + \mathfrak{F}(\mathrm{Tag}_{\tilde{B}}), \\ \mathscr{C}^R_{n,m} + \mathfrak{F}(\tilde{B}) < \mathfrak{F}(B_0) + \mathfrak{F}(\mathrm{Tag}_{\tilde{B}}), \\ \mathscr{C}^R_{n,m} + \mathfrak{F}(\tilde{B}) = \mathfrak{F}(B_0) + \mathfrak{F}(\mathrm{Tag}_{\tilde{B}}). \end{cases} \tag{8}$$

How to mark the dropped bundles? This paper uses labels instead of each dropped bundle; the label composition is shown in Equation (9), where S^{ID}_k represents the source node ID for sending B_k. SN_k represents the sequence number of B_k. $\mathfrak{F}(B_k)$ represents the size of the cache capacity occupied by B_k. $\mathrm{Tag}^{[n]}_k$ denotes the nth term of Tag_k.

$$\begin{aligned} &\mathrm{Tag}_k = \left[S^{\mathrm{ID}}_k, \mathrm{SN}_k, \mathfrak{F}(B_k)\right] \quad B_k \in \tilde{\mathrm{B}}, \mathrm{SN}_k \in N^*, \\ &\mathrm{Tag}^{[1]}_k = S^{\mathrm{ID}}_k \quad \mathrm{Tag}^{[2]}_k = \mathrm{SN}_k \quad \mathrm{Tag}^{[3]}_k = \mathfrak{F}(B_k). \end{aligned} \tag{9}$$

To reduce the drop rate of the same bundle in the path, Tag_k is sent to the next hop after dropping B_k. The next hop receives the label and updates $\mathscr{A}(B_k)$ as shown in

$$\mathscr{A}(B_k) = [\mathrm{PRI}_k + 1, 1/\mathrm{TTL}_k]. \tag{10}$$

3.3. Acknowledgment Problem Definition. In large-scale LEO networks, acknowledgment is necessary to ensure correct data transmission [39]. The destination node acknowledges the received bundle and replies with an acknowledgment packet (ACK) after decoding that the received bundle is correct. The bundle received is an incorrect reply to a negative acknowledgment packet (NACK). The theoretical time of the satellite from sending the bundle to receiving the ACK is $t^{ack}_{n,m}$. $t^{ack}_{n,m}$ is defined as

$$t^{\mathrm{ack}}_{n,m} = \sum_{[n,m]}^{R^*} \left(2\frac{\mathscr{D}^{n',m'}_{n,m}}{c} + \frac{\mathscr{Q}^{n',m'}_{n,m} + \mathfrak{F}(B_0) + \mathscr{P}_{\mathrm{ack}}}{v} + t^{\rho}_{n,m} \right), \tag{11}$$

where R^* is the transmission path of the bundle. $\mathscr{P}_{\mathrm{ack}}$ is the ACK packet size, and $t^{\rho}_{n,m}$ is the $v_{n,m}$ processing time. $[n, m]$ represents the label of the satellite, and $[n', m']$ is the satellite label of the next hop of $v_{n,m}$. The value of $[n, m]$ starts from $v_{n,m}$ and then takes the satellite label in the path R^* until $[n, m]$ is the last hop satellite label of the destination node. For example, the first time $[n, m]$ is the label of $v_{n,m}$. The second time the label of the next hop of $v_{n,m}$ is set to $[n, m]$ and iterates until $[n', m']$ is the label of the last node in the path R^*.

There are two main reasons why the source node does not receive an ACK/NACK at the expected time. (1) The bundle is lost during transmission from the source node to the destination node. This situation is defined as $\vec{E}$. The set of satellites in the path that correctly receive the bundle is denoted as $\vec{\mathrm{v}}$. (2) The ACK/NACK is lost during transmission from the destination node to the source node. This situation is defined as $\overleftarrow{E}$. The set of satellites in the path that correctly receive the ACK/NACK is denoted as $\overleftarrow{\mathrm{v}}$.As shown in Equation (12), $\vec{v}^*$ represents the satellite in the path that receives the bundle closest to the destination node in case $\vec{E}$. $\mathscr{D}^j_{n,m}$ denotes the distance between $v_{n,m}$ and the destination node ES_j. $v_{n,m}$ is any satellite in set $\vec{v}$. $\vec{v}^*$ is the satellite corresponding to the minimum value of $\mathscr{D}^j_{n,m}$. $\overleftarrow{v}^*$ represents the satellite in the path that is closest to source node to receive the ACK/NACK in case $\overleftarrow{E}$. $\mathscr{D}^i_{n,m}$ denotes the distance between $v_{n,m}$ and the source node ES_i. $v_{n,m}$ is any satellite in set $\overleftarrow{\mathrm{v}}$. $\overleftarrow{v}^*$ is the satellite corresponding to the minimum value of $\mathscr{D}^i_{n,m}$.

$$\begin{aligned} &\vec{E} : \vec{v}^* = \underset{v}{\mathrm{argmin}} \mathscr{D}^j_{n,m}\, v_{n,m} \in \vec{\mathrm{v}}, \\ &\overleftarrow{E} : \overleftarrow{v}^* = \underset{v}{\mathrm{argmin}} \mathscr{D}^i_{n,m}\, v_{n,m} \in \overleftarrow{\mathrm{v}}. \end{aligned} \tag{12}$$

4. LIDTN Scheme Implementation

The LIDTN scheme proposed in this paper for large-scale LEO constellations mainly includes three algorithms. The first algorithm is equivalent distance and priori knowledge-based forwarding strategy (EPFS). This algorithm is primarily to reduce the routing overhead. The second algorithm is an emergency function-based bundle drop algorithm (EBDA). This algorithm is mainly to solve the congestion problem during bundle transmission. The third algorithm is the virtual acknowledgment algorithm (VAA). This algorithm mainly reduces the delay and overhead of the retransmission bundles.

4.1. Equivalent Distance and Priori Knowledge-Based Intersatellite Forwarding Strategies. As shown in Equation (4), $v_{n,m}$ calculates the optimal action $a^{t,*}_{n,m}$ at time t and then transmits the bundle to $N^t_{n,m}$. The forwarding of $v_{n,m}$ to the bundle is influenced by the information flow shown in Figure 5. In the figure, $\mathscr{Q}^U_{n,m}$, $\mathscr{Q}^D_{n,m}$, $\mathscr{Q}^L_{n,m}$, and $\mathscr{Q}^R_{n,m}$ are the queue lengths of the up, down, left, and right four sending ports, respectively.

When the bundle reaches $v_{n,m}$, the following steps are performed to forward the bundle to the next hop.

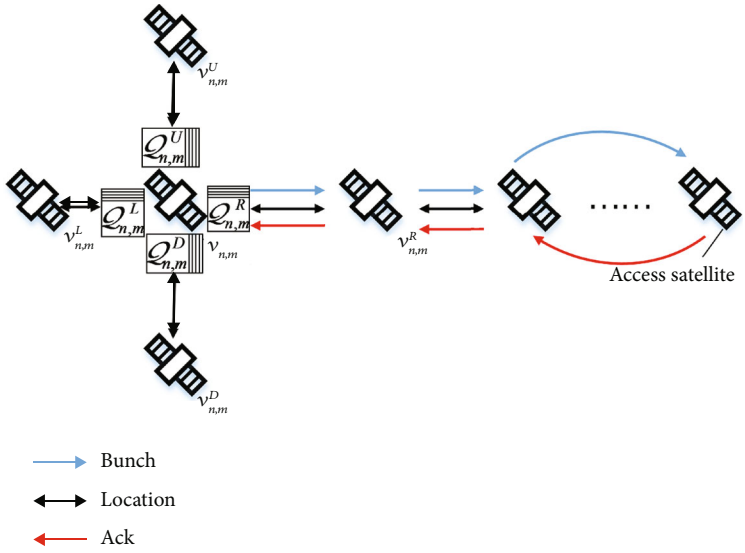

Figure 5: Schematic diagram of intersatellite information flow.

(1) Based on the received $SS_{n,m}$ position information, the distance between $v_{n',m'}$ and ES_j is calculated by Equation (1), $v_{n',m'} \in SS_{n,m}$. If $v_{n,m} \notin \bar{v}$, $v_{n,m}$ is not in the crossing seam, $SS_{n,m}$ of $v_{n,m}$ is fixed. If $v_{n,m} \in \bar{v}$, $v_{n,m}$ is in the crossing seam, three satellites in set S $S_{n,m}$ are fixed, and the other satellite changes in a pattern similar to Figure 3

(2) The length of the port queue sent to the satellites surrounding of $v_{n,m}$ is calculated and unifies the dimension of quantity of the queue length through $(\mathcal{Q}_{n,m}^{n',m'} + \mathfrak{T}(B_0)/v)c$ in Equation (4)

(3) The acknowledgment factor $\mathcal{F}_{\text{ack}}$ is not calculated if $v_{n,m}$ has not received an acknowledgment packet from $v_{n',m'}$ before. Otherwise, $\mathcal{F}_{\text{ack}}$ is calculated. The value of $\mathcal{F}_{\text{ack}}$ is calculated through a distance between $v_{n,m}$ and $v_{n',m'}$, and $\text{Tag}_k^{[3]}$ in the acknowledgment packet sent back to $v_{n,m}$ by $v_{n',m'}$. $v_{n',m'}$ is the next hop of $v_{n,m}$.The expression of $\mathcal{F}_{\text{ack}}$ is

$$\mathcal{F}_{\text{ack}} = (1-\lambda)\mathcal{F}_{\text{ack}} + \lambda\left[\frac{\sum_{i=1}^{\xi}\text{Tag}_i^{[3]}}{v}c + \mathcal{D}_{n,m}^{n',m'}\right], \quad 0 \le \lambda \le 1 \tag{13}$$

where λ denotes the smoothing factor.

(4) The action $a_{n,m}^{t,*}$ of $v_{n,m}$ is calculated by Equation (4), and $a_{n,m}^{t,*}$ is performed to forward the bundle to the next hop satellite

(5) $v_{n,m}$ forwards the bundle to $v_{n',m'}$. $v_{n',m'}$ performs the above four steps. The above steps are repeated until the bundle is at the destination node

The bundle queuing method follows the traditional first-in-first-out (FIFO) strategy, but the priority of the retransmitted bundle is higher than that of the first transmitted bundle.

The bundle is forwarded according to the above process, and there is a probability of loops occurring. The two cases where loops occur are shown in Figure 6. Since we use the equivalent distance forwarding bundle, Figure 6(b) has a lower probability of loop occurrence than Figure 6(a). To solve the loop problem shown in Figure 6, we propose the concept of a lock flag in the EPFS algorithm. The lock flags are set on the bundle sending ports of the satellite. When the satellite receives the bundle, the satellite will lock the port corresponding to the received bundle, and the satellite will not send the received bundle at this port. For example, in Figure 6(a), after v_y receives B_i sent by v_x on the left, it locks the port on the left of v_y to this bundle. That is, v_y cannot send the bundle through this port. This method can only avoid (Figure 6(a)) loops. For loop in Figure 6(b), the EPFS algorithm upgrades the lock flags. If the satellite receives the same bundle from a different port, the bundle is transmitted back, and the satellite that receives the returned bundle locks the port that receives the returned bundle. As shown in Figure 6(b), the up port of v_b receives B_i transmitted by v_e. Since there is already B_i in v_b, B_i is transmitted back to v_e through the up port. v_e detects that the sent B_i, by computation, returns from the same port, and then, v_e locks the down port for B_i. The flow of the EPFS is shown in Algorithm 1.

$v_{n,m}$ forwards the bundle to $v_{n',m'}$ for the first time. $v_{n,m}$ cannot obtain the congestion status of $v_{n',m'}$ by the acknowledgment factor. The bundle has the probability of being dropped by $v_{n',m'}$. For this case, we will solve it in EBDA.

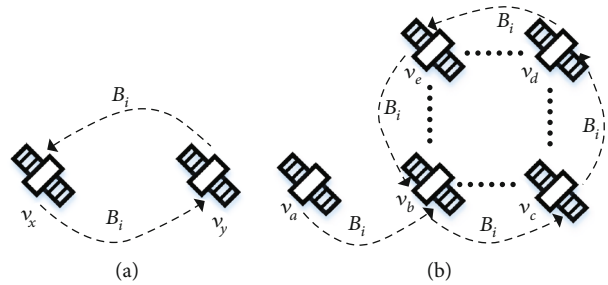

Figure 6: Schematic diagram of the forwarding loop. (a) The first type of loop. (b) The second type of loop.

Received bundle
Input: Bundle B_0, Port to receive B_0: P
Output: Lock flag: LF
1: **if**B_0 received for the first time **then**
2: LF=TURE; P is locked, and B_0 cannot be forwarded through P
3: **else**
4: **if**B_0 is received again from the port that forwarded the B_0**then**
5: Locks the port that receives B_0 again;
6: **else**
7: Forward B_0 from the port where the B_0 was received;
8: **end if**
9: **end if**
Forward bundle
Input: Time t, $v_{n,m}$ port queue $\{\mathcal{Q}^U_{n,m}\ \mathcal{Q}^D_{n,m}\ \mathcal{Q}^L_{n,m}\ \mathcal{Q}^R_{n,m}\}$, The position of $SS_{n,m}$, The position of the ES_j
Output: Action $a^{t,*}_{n,m}$
1: **if**$v_{n,m} \in \bar{\mathbf{v}}$**then**
2: The ISLs between $v_{n,m}$ and the satellites moving in the opposite direction are not fixed.
Calculate the satellite that is closest to $v_{n,m}$ in the opposite direction by equation (2)
3: **else**
4: The ISLs of $v_{n,m}$ can only be established with four satellites: up, down, left, and right;
5: **end if**
6: **for**$v_{n',m'} = [v^U_{n,m}, v^D_{n,m}, v^L_{n,m}, v^R_{n,m}]$**do**
7: **if** The port forwarded by $v_{n,m}$ to $v_{n',m'}$ is not locked **then**
8: Calculate the distance between $v_{n',m'}$ and ES_j by equation (1);
9: The normalized distance resulting from queuing is calculated by equation (4);
10: **if**$v_{n,m}$ has received the ACK from $v_{n',m'}$**then**
11: Calculate $\mathcal{F}_{ack}$ by equation (13);
12: **else**
13: $\mathcal{F}_{ack} = 0$;
14: **end if**
15: Calculate $d^{t,j}_{n',m'}$ by equation (4);
16: **end if**
17: **end for**
18: Calculate $a^{t,*}_{n,m}$ by equation (4);

Algorithm 1: Equivalent distance and priori knowledge-based intersatellite forwarding strategies (EPFS).

4.2. Emergency Function-Based Bundle Drop Algorithm. EBDA primarily solves the problem of congestion in $v_{n',m'}$ after $v_{n,m}$ executes action $a^{t,*}_{n,m}$. The EBDA proposed in this paper has the following advantages. The first is the use of the emergency function $\mathcal{A}(B_i)$, which reduces the successive drops of B_i in the path and reduces the drops of B_i with small TTL_i values. The second is the use of label Tag_k, which replaces the larger memory space of B_k with a smaller memory capacity. $\mathrm{Tag}^{[3]}_k$ in $v_{n',m'}$ provides a priori knowledge for $a^{t,*}_{n,m}$ of $v_{n,m}$.

According to the emergency function of Equation (6), all the bundles cached in the satellite have the probability of being dropped. There are two types of dropped bundles in this paper. The first type is to drop the current bundle B_0 (not yet forwarded). The second type is to drop part or all of the $\{B_1, B_2, \cdots, B_\xi\}$. $\{B_1, B_2, \cdots, B_\xi\}$ is stored in the cache

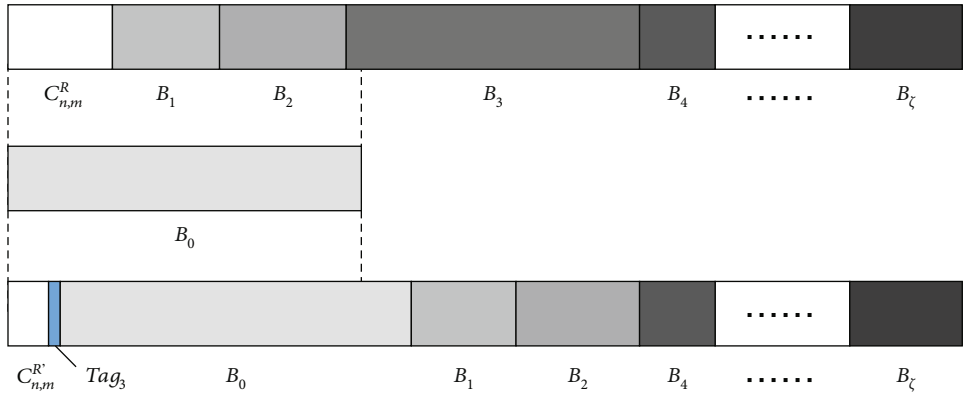

Figure 7: The bundles stored in the $v_{n,m}$ at a certain moment.

before B_0. ($\{B_1, B_2, \cdots, B_\xi\}$ is the bundles that have been forwarded and have not yet received the acknowledgment).

Those satisfying $\mathscr{A}(B_0) < \min(\mathscr{A}(B_1), \mathscr{A}(B_2), \cdots, \mathscr{A}(B_\xi))$ are the first type of bundle dropping. After dropping B_0, Tag_0 is used instead of B_0 and updates $\mathscr{A}(B_0)$. $\mathscr{A}(B_0)$ is updated according to Equation (14). Since B_0 has not been forwarded yet, dropping B_0 directly will cause B_0 transmission to be interrupted. Therefore, execute action $a^{t,*}_{n,m}$ and then drop B_0.

$$\begin{aligned} \text{Tag}_0 &\xleftarrow{\text{replaced}} B_0, \\ \text{Tag}_0 &= \left[S^{\text{ID}}_0, \text{SN}_0, \text{size}(B_0)\right], \\ \mathscr{A}(B_0) &= [\text{PRI}_0, \text{TTL}_0], \\ \text{PRI}_0 &= \text{PRI}_0 + 1. \end{aligned} \tag{14}$$

The second bundle drop type needs to drop part or all of $\{B_1, B_2, \cdots, B_\xi\}$. In the traditional method [40, 41], the bundles are dropped in first-in-first-out (FIFO) order until satisfied:

$$\mathscr{C}^R_{n,m} + \mathfrak{F}(\tilde{B}) \geq \mathfrak{F}(B_0) + \mathfrak{F}(\text{Tag}_{\tilde{\mathcal{B}}}), \tag{15}$$

where $\tilde{B}$ represents the set of dropped bundles. In the literature [30], an iterative approach is used to reduce the bundle drop. The EBDA proposed in this paper can further reduce bundle dropout. In addition, using tags increases the bundle's traceability and can provide priori knowledge for the subsequent bundle forwarding.

The bundles cached in the $v_{n,m}$ at a certain moment are shown in Figure 7. B_0 is the bundle newly received by $v_{n,m}$. $v_{n,m}$ remaining cache space $\mathscr{C}^R_{n,m} < \mathfrak{F}(B_0)$. Assume that $\mathscr{A}(B_0) > \mathscr{A}(B_1) > \cdots > \mathscr{A}(B_{\zeta-1}) > \mathscr{A}(B_\zeta)$ in Figure 7. If the bundles are dropped according to the emergency function in Equation (6), B_1, B_2, and B_3 are dropped. Because $\mathscr{C}^R_{n,m} + \mathfrak{F}(B_3) > \mathfrak{F}(B_0) + \mathfrak{F}(\text{Tag}_3)$, dropping B_3 alone can satisfy the cache requirements of B_0. To further reduce the bundle drop, the bundles are grouped according to the value of A. The bundles cached in the $v_{n,m}$ are grouped as follows:

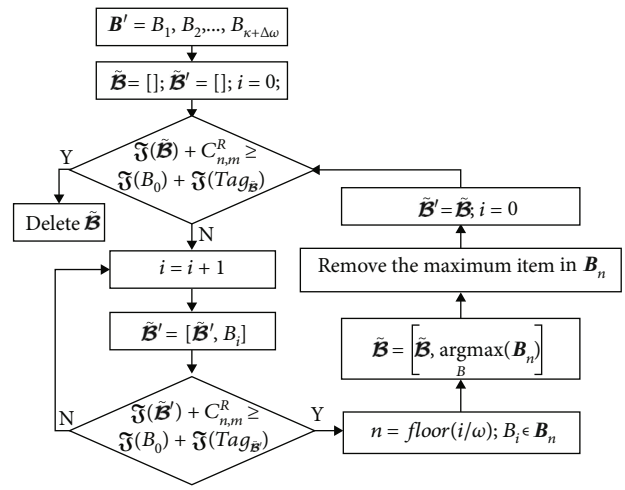

Figure 8: Calculate set $\tilde{B}$.

(1) Arrange the values of the emergency function in the order from smallest to largest. Select the first κ bundles that satisfy Equation (16).

$$\begin{aligned} \mathscr{C}^R_{n,m} + \sum_{i=1}^{\kappa}(\mathfrak{F}(B_i) - \mathfrak{F}(\text{Tag}_i)) &\geq \mathfrak{F}(B_0), \\ \mathscr{C}^R_{n,m} + \sum_{i=1}^{\kappa-1}(\mathfrak{F}(B_i) - \mathfrak{F}(\text{Tag}_i)) &\leq \mathfrak{F}(B_0), \end{aligned} \tag{16}$$

(2) The first κ bundles are grouped. Every ω bundles are defined as a group according to the size of the value of the emergency function. The κth bundle is in the last group. If the last group has less than ω bundles, the last group will be supplemented to ω bundles according to the value of the emergency function

The set $\tilde{B}$ of dropped bundles is calculated according to the flowchart of the algorithm shown in Figure 8. In the figure, B' represents the set of bundles $B_1, B_2, \cdots, B_{\kappa+\Delta\omega}$, where

Input: Bundle: B_0
Output: Dropped bundles set: $\tilde{\mathscr{B}}$
1: **while** $\mathscr{C}^R_{n,m}+\mathfrak{T}(\tilde{\mathscr{B}}) < \mathfrak{T}(B_0) + \mathfrak{T}(Tag_{\tilde{\mathscr{B}}})$ **do**
2: $\mathscr{A} = [\mathscr{A}(B_1), \mathscr{A}(B_2), \cdots, \mathscr{A}(B_\zeta)]$; The emergency function of the bundles in $v_{n,m}$ is deposited into $\mathscr{A}$;
3: $\tilde{\mathscr{B}} = [\tilde{\mathscr{B}}, \underset{B}{argmin}(\mathscr{A})]$; Store the bundle corresponding to the smallest value in $\mathscr{A}$ into set $\tilde{B}$;
4: Deletes the smallest item in $\mathscr{A}$;
5: **end while**
6: Follow Figure 8 to find set $\tilde{\mathscr{B}}$ that needs to be finally dropped.
7: Replace the dropped bundles with tags;

Algorithm 2: Emergency function-based bundle drop algorithm (EBDA).

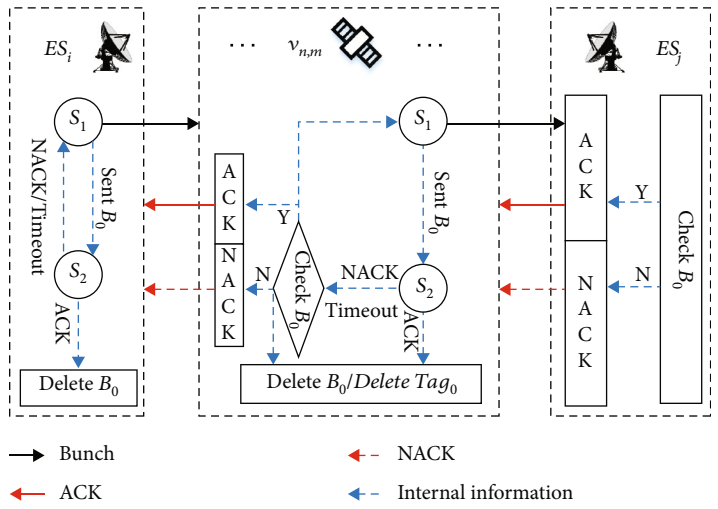

Figure 9: Schematic diagram of the data flow of VAA.

the emergency function value of $B_1, B_2, \cdots, B_\kappa$ is less than $\mathscr{A}(B_0)$, and the emergency function value of $B_{\kappa+1}, B_{\kappa+2}, \cdots, B_{\kappa+\Delta\omega}$ is greater than $\mathscr{A}(B_0)$. $\tilde{B}'$ represents the temporary set variable. $\tilde{B}' = [\tilde{B}', B_i]$ denotes the storage of the bundle B_i into the set $\tilde{B}'$. B_n is the set of bundles of the nth group. Each group included ω bundles. Set B' contains $(\kappa + \Delta\omega)/\omega$ groups. floor(i/ω) represents the integer part of i/ω. $\mathfrak{T}(B_n)$ represents the total cache capacity occupied by all bundles in collection B_n. $\underset{B}{\text{argmax}}(B_n)$ represents the bundle with the largest cache capacity in set B_n. $\tilde{B} = [\tilde{B}, \arg\max_B(B_n)]$ denotes storing the bundle $\underset{B}{\text{argmax}}(B_n)$ into the set $\tilde{B}$. Figure 8 contains both internal and external iteration calculations. The internal iteration calculates the temporary set variable $\tilde{B}'$, and the external iteration calculates the set variable $\tilde{B}$. The condition for each end of the internal iteration is that there is sufficient capacity to store B_0 after the satellite drops the bundles in set $\tilde{B}'$. However, some bundles in set $\tilde{B}'$ may not need to be dropped and still can meet the storage requirements of the bundle B_0. Therefore, the set $\tilde{B}$ is calculated by external iteration to reduce the number of dropped bundles on the premise of meeting the storage requirements of B_0.

The pseudocode of the EBDA algorithm is shown in Algorithm 2. The labels in this algorithm will be used again in the VAA.

4.3. Virtual Acknowledgement Algorithm. The VAA proposed in this paper is applied to the following scenarios:

(i) The bundle is lost or occurs erroneously during transmission

(ii) The ACK/NACK is lost or occurs erroneously during transmission

VAA has the following advantages.

(i) Reduce the cache capacity occupied by the bundle in the satellite

(ii) Reduce end-to-end retransmission latency and acknowledgment delay

(iii) Directly provide priori knowledge for calculating the next hop

VAA in source nodes $\boldsymbol{ES_i}$
Input: State S_1, S_2; ACK; NACK; time t
Send bundle B_0; Set the retransmission timer $\mathcal{T}_i^0$ according to equation (22);
if Receive ACK of B_0 **then**
  Delete B_0 from the cache;
  Delete the retransmission timer $\mathcal{T}_i^0$;
end if
if Receive NACK of B_0 OR $\mathcal{T}_i^0 = 0$ **then**
  Retransmission B_0;
  Reset the retransmission timer $\mathcal{T}_i^0$;
end if
VAA in relay nodes $v_{n,m}$
Input: State S_1, S_2; ACK; NACK; time t
Forward the bundle B_0; Set the retransmission timer $\mathcal{T}_{n,m}^0$ according to equation (22);
if Received ACK of B_0 **then**
  Replace B_0 with Tag_0'; Start timer $\bar{\mathcal{T}}_{n,m}^0$;
  Delete the retransmission timer $\mathcal{T}_{n,m}^0$;
  Calculate $\mathfrak{F}(Tag_{n,m}^{[3]})$ by (17); ACK⟵$\mathfrak{F}(Tag_{n,m}^{[3]})$;
  Forward ACK;
end if
if $\bar{\mathcal{T}}_{n,m}^0 \le 0$ or Received B_1 //note: B_1 and B_0 were sent from the same source node. **then**
  Delete Tag_0';
end if
if Received NACK of B_0 OR $\mathcal{T}_{n,m}^0$=0 **then**
  Calculate $\mathfrak{F}(Tag_{n,m}^{[3]})$ by equation (17);
  Check whether B_0 decoded in the cache is correct;
  if B_0 decoded correctly **then**
    ACK⟵$\mathfrak{F}(Tag_{n,m}^{[3]})$; //note: Put $\mathfrak{F}(Tag_{n,m}^{[3]})$ into the ACK package.
    Retransmission B_0; Forward ACK; Reset the retransmission timer $\mathcal{T}_{n,m}^0$;
  else
    NACK⟵$\mathfrak{F}(Tag_{n,m}^{[3]})$; //note: Put $\mathfrak{F}(Tag_{n,m}^{[3]})$ into the NACK package.
    Delete B_0; Forward NACK to source node ES_i;
  end if
end if

Algorithm 3: Virtual acknowledgement algorithm (VAA).

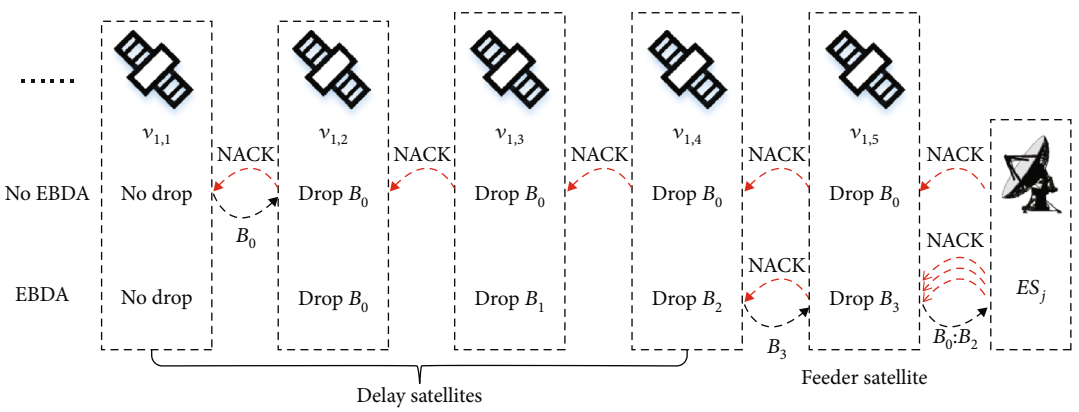

Figure 10: Acknowledgement and retransmission schematics.

VAA has different processing methods at destination nodes, relay satellites, and source nodes. The data flow of VAA in the constellation is shown schematically in Figure 9. In the figure, ES_i represents the source node, $v_{n,m}$ represents the relay satellite, and ES_j represents the destination node. S_1 represents the state of sending B_0, and S_2

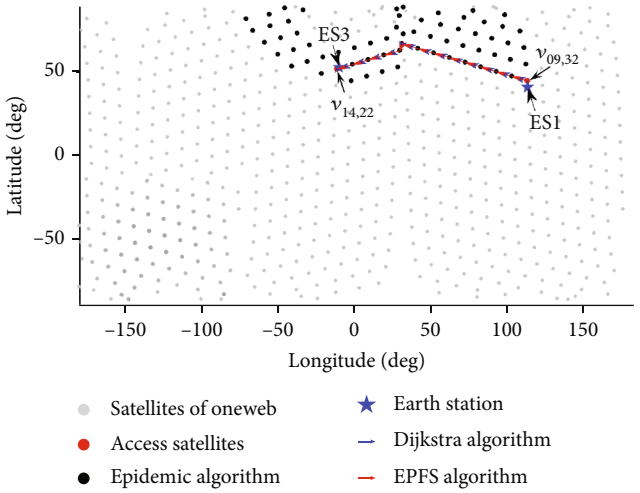

Figure 11: $\mathfrak{R}_1$ schematic.

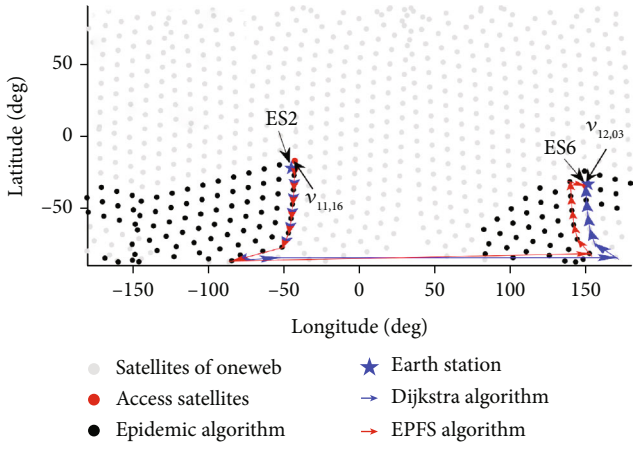

Figure 12: $\mathfrak{R}_2$ schematic.

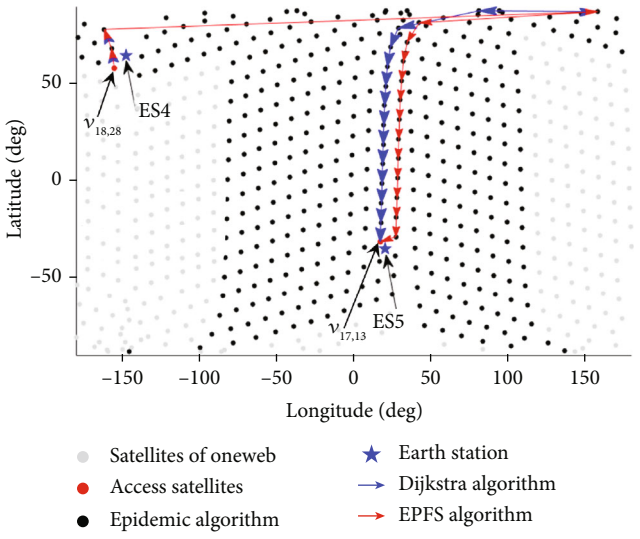

Figure 13: $\mathfrak{R}_3$ schematic.

represents the state of waiting for an acknowledgment after sending B_0.

The mechanism of operation of VAA in ES_i is as follows. After ES_i forwards B_0 in state S_1, it opens the timer, waits for the ACK of B_0, and stores B_0 in the cache. Then, ES_i from S_1 switches to state S_2. The timer waiting for ACK in the node is defined as $\mathcal{T}^n_{\mathrm{ID}}$. n represents the sequence number of the bundle, and ID represents the node's ID. For example, the timer waiting for the bundle B_0 in $v_{n,m}$ is represented as $\mathcal{T}^0_{n,m}$. The timer waiting for the bundle B_n in ES_i is represented as $\mathcal{T}^n_i$.Therefore, the timer waiting for B_0 in ES_i is represented as $\mathcal{T}^0_i$. When $\mathcal{T}^0_i > 0$ receives the ACK of B_0, ES_i deletes B_0 in the cache. When $\mathcal{T}^0_i > 0$ receives a NACK or $\mathcal{T}^0_i \le 0$, ES_i switchs to state S_1, retransmits B_0, and resets $\mathcal{T}^0_i$.

The mechanism of operation of VAA in $v_{n,m}$ is as follows. $v_{n,m}$ switches to state S_2 after state S_1 finishes forwarding, caching, and turning on timer $\mathcal{T}^0_{n,m}$ for B_0. When $\mathcal{T}^0_{n,m} > 0$ receives the ACK of B_0, $v_{n,m}$ deletes B_0 or Tag_0 from the cache. The sum of the third term of all tags of $v_{n,m}$ is calculated by Equation (17), and $\mathfrak{F}(\mathrm{Tag}^{[3]}_{n,m})$ is included in the ACK for forwarding. $v_{n,m}$ decodes B_0 in the cache when NACK is received at $\mathcal{T}^0_{n,m} > 0$ or $\mathcal{T}^0_{n,m} \le 0$. If B_0 is correct, $v_{n,m}$ switches to state S_1, retransmits B_0, resets $T^0_{n,m}$, and forwards the ACK to the source node. If B_0 is incorrect, remove B_0 from the cache and forward the NACK to the source node.

$$\mathfrak{F}\left(\mathrm{Tag}^{[3]}_{n,m}\right) = \sum_{i=1}^{\zeta} \mathrm{Tag}^{[3]}_i. \tag{17}$$

The above process deletes B_i from the cache when $v_{n,m}$ receives the ACK of B_i. However, when ACK is lost during transmission, it triggers timeout retransmission of B_i for satellites on the path that did not receive ACK. It will cause the satellites that have received the ACK to cache and forward B_i again, resulting in resource waste. In order to solve this problem, after the relay satellite deletes the bundle corresponding to the ACK, the deleted bundle is temporarily replaced by the first two items in the bundle tag. The first two terms of the bundle tag are defined as $\mathrm{Tag}'_i = [S^{\mathrm{ID}}_i, \mathrm{SN}_i]$. Through the mapping relationship between Tag'_i and the bundle, when $v_{n,m}$ receives a bundle that has already replied to the ACK, it will send the ACK again but will no longer forward the bundle. Tag'_i can be deleted in two ways. The first is that $v_{n,m}$ again receives a bundle sent by the source node of the bundle corresponding to Tag'_i. The second is that $v_{n,m}$ waits for time $\bar{T}^i_{n,m}$ after forwarding the ACK and then deletes Tag'_i. The calculation formula of $\bar{T}^i_{n,m}$ is

$$\bar{T}^i_{n,m} = T^i_{\mathrm{src}} - T^i_{n,m}, \tag{18}$$

where T^i_{src} represents the timer waiting for the ACK of B_i in the source node.

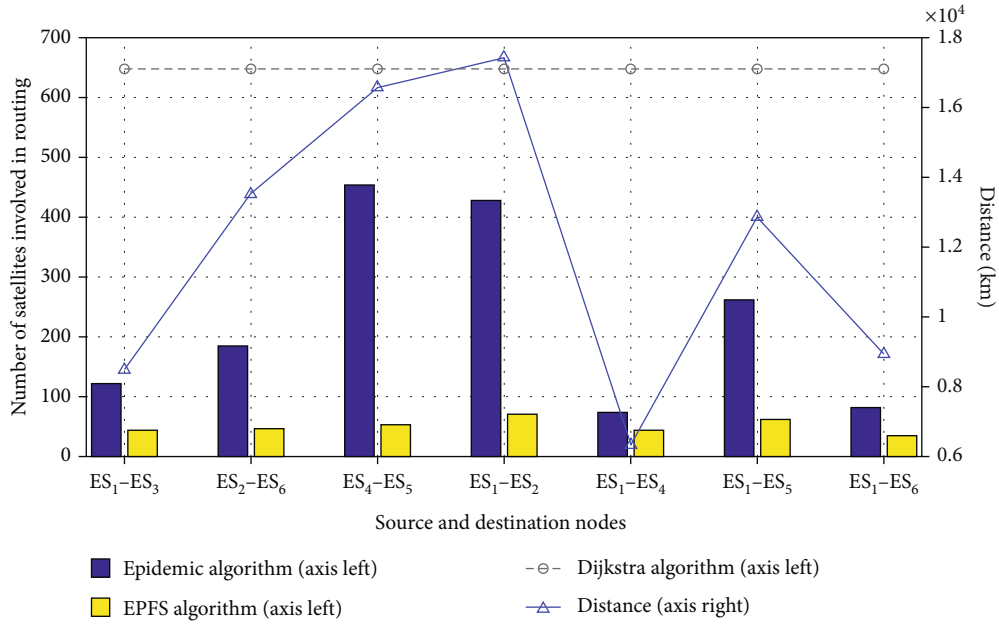

Figure 14: The number of satellites to be involved using different algorithms on different paths.

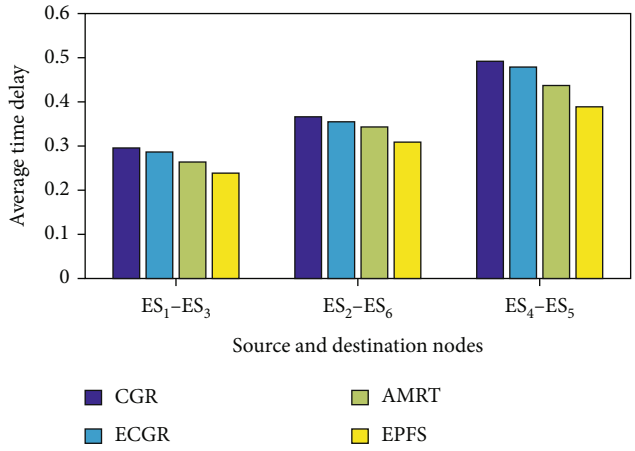

Figure 15: Average time delay of different algorithms.

In the VAA algorithm, the processing time of ACK is ignored. When the satellite receives the ACK, it is forwarded directly without considering the cache of ACK and queuing modeling of ACK.

The mechanism of operation of VAA in ES_j is as follows. ES_j receives B_i and decodes B_i. If B_i is decoded correctly, reply with an ACK. If B_i is decoded incorrectly, reply with a NACK. The pseudocode of VAA in ES_i and $v_{n,m}$ is shown in Algorithm 3.

4.4. Scheme Analysis. In the EPFS algorithm, the satellite selects the next hop by calculating the equivalent distance and a priori knowledge. The proposed concept of a port lock flag can avoid loops in the path. EPFS algorithm does not need a routing table when forwarding bundles. After the satellite receives the bundle, $a_{n,m}^{t,*}$ is calculated, and the bundle is forwarded to the next hop. The satellite that receives the bundle later uses the same calculation method until the bundle is transmitted to the destination node. Because the equivalent distance between the satellite and the destination node is greater than that of the next hop of the satellite, theoretically, the bundles can be transmitted to the destination node 100%.

Using the emergency function in the EBDA algorithm can reduce the successive drops of the same bundle in the path. Thus, the VAA based on EBDA can reduce the retransmission time due to bundle decoding errors or timeouts. As shown in Figure 10, if B_0 is only wrong when $v_{1,5}$ is transmitted to ES_j, only B_0 received by ES_j is incorrect. The first case in Figure 10 does not use EBDA; the satellites after $v_{1,1}$ have a high probability of dropping B_0 because of the lack of cache capacity. When B_0 is continuously dropped, and the retransmission delay is t_0^r

$$t_0^r = t_{1,5}^j + \sum_{i=1}^{4} t_{1,i}^{1,i+1}, \tag{19}$$

where $t_{1,i}^{1,i+1}$ is the NACK transmission delay between $v_{1,i}$ and $v_{1,i+1}$. $t_{1,5}^j$ is the NACK transmission delay between ES_j and $v_{1,5}$. The second case in Figure 10 uses EBDA. Because using the emergency function in the EBDA algorithm can increase the priority of the dropped bundle, reducing the probability of the same bundle being continuously dropped. For example, after B_0 is dropped in $v_{1,2}$, the priority of B_0 increases, and B_0 is no longer dropped in subsequent satellites. Therefore, $v_{1,5}$ can retransmit B_0 after receiving NACK. The B_0 retransmission time delay is $t_0^{re} = t_{1,5}^j$. Denote B_0, B_1, B_2, and B_3 as $B_{0:3}$. If $B_{0:3}$ makes mistakes in the transmission process from $v_{1,5}$ to ES_j, $B_{0:3}$ received by ES_j are all incorrect.

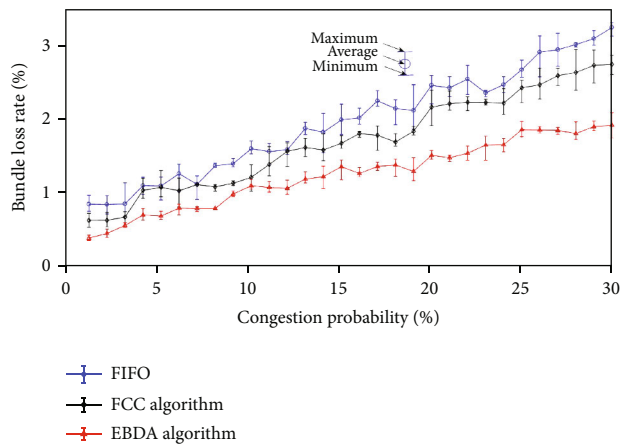

Figure 16: The average drop rate of $\mathfrak{R}_1$, $\mathfrak{R}_2$, and $\mathfrak{R}_3$ paths in different congestion probability.

Table 2: Satellites with congestion occurring in one of the simulations at 15% probability of congestion.

Route	Congestion satellites			
$\mathfrak{R}_1$	$v_{04,32}$	$v_{01,32}$	$v_{16,22}$	
$\mathfrak{R}_2$	$v_{11,09}$		$v_{11,08}$	
$\mathfrak{R}_3$	$v_{18,26}$	$v_{18,23}$	$v_{18,21}$	$v_{18,18}$

And B_0 is dropped in all satellites after $v_{1,1}$ because of insufficient cache capacity. The time of retransmission $B_{0:3}$ without EBDA is $t^r_{[0:3]}$:

$$t^r_{[0:3]} = 4t^j_{1,5} + \sum_{i=1}^{4} t^{1,i+1}_{1,i}. \tag{20}$$

Because the EBDA algorithm is used, the priority corresponding to the bundle will increase after the bundle is dropped, and the emergency function value will be increased. So the probability of the same bundle being continuously dropped by the satellite in the path is reduced. As shown in Figure 10, suppose the bundles dropped by $v_{1,2}$, $v_{1,3}$, $v_{1,4}$, and $v_{1,5}$ are B_0, B_1, B_2, and B_3, respectively. The time to retransmit B_0:B_3 using EBDA is $t^{re}_{[0:3]}$:

$$t^{re}_{[0:3]} = 4t^j_{1,5} + t^{1,5}_{1,4}. \tag{21}$$

Because $t^{re}_{[0:3]} < t^r_{[0:3]}$ and $t^{re}_0 < t^r_0$, using the EBDA + VAA algorithm can theoretically reduce the retransmission delay of destination node decoding errors.

EBDA + VAA also can reduce timeout retransmission time. A timeout timer $\mathcal{T}^0_{n,m}$ is set to wait for the ACK after $v_{n,m}$ forwards B_0. The calculation of $\mathcal{T}^0_{n,m}$ is shown in Equation (22) [42]. RTT denotes round trip time, SRTT denotes the RTT after smoothing, and RTTV denotes the absolute value of SRTT − RTT after smoothing. The relay satellite transmits B_0 to destination node ES_j. The distance from the relay satellite to ES_j is decreasing, and the RTT value is decreasing. Therefore, in Equation (12), $\vec{v}^*_{n,m}$ retransmits the bundle in case $\vec{E}$, and $\overleftarrow{v}^*_{n,m}$ retransmits the acknowledgment packet in case of $\overleftarrow{E}$, theoretically realizing optimality in reducing the retransmission delay.

$$\begin{aligned} \mathcal{T}^0_{n,m} &= \text{SRTT} + 4\text{RTTV}, \\ \text{SRTT} &= (1-\alpha)\text{SRTT} + \alpha\text{RTT}, \\ \text{RTTV} &= (1\text{-}\beta)\text{RTTV} + \beta(|\text{RTT-SRTT}|). \end{aligned} \tag{22}$$

The source node starts the retransmission timer $\mathcal{T}^0_i$ after sending B_0. After receiving B_0, the relay satellite $v_{n,m}$ caches, forwards, and starts the retransmission timer $\mathcal{T}^0_{n,m}$. From Equation (22), $\mathcal{T}^0_{n,m} < \mathcal{T}^0_i$ can be derived. As shown in Figure 10, the retransmission B_0 with No EBDA needs to wait for time $\mathcal{T}^0_{1,1}$. The retransmission B_0 with EBDA needs to wait for time $\mathcal{T}^0_{1,5}$. Since $\mathcal{T}^0_{1,5} < \mathcal{T}^0_{1,1}$, EBDA can reduce the retransmission time of B_0. If there are timeouts for retransmission of both $B_{0:3}$, the time required to complete the retransmission of $B_{0:3}$ in the No EBDA case is $T^r_{[0:3]}$:

$$T^r_{[0:3]} = \max\left(\mathcal{T}^0_{1,1}, \mathcal{T}^1_{1,5}, \mathcal{T}^2_{1,5}, \mathcal{T}^3_{1,5}\right). \tag{23}$$

The time required to retransmit $B_{0:3}$ using EBDA is $T^{re}_{[0:3]}$:

$$T^{re}_{[0:3]} = \max\left(\mathcal{T}^0_{1,5}, \mathcal{T}^1_{1,5}, \mathcal{T}^2_{1,5}, \mathcal{T}^3_{1,4}\right). \tag{24}$$

From Equation (22), we can derive $T^{re}_{[0:3]} < T^r_{[0:3]}$. Thus, EBDA + VAA can reduce the timeout retransmission time.

5. LIDTN Scheme Simulation Results

In this section, we perform simulation experiments on the LIDTN scheme. The OneWeb constellation topology used in the simulation is shown in Figure 1. The location and numbering of the Earth stations are shown in Figure 4. The paths $ES_1 \leftrightarrow ES_3$, $ES_2 \leftrightarrow ES_6$, and $ES_4 \leftrightarrow ES_5$ in Figure 4 are defined as $\mathfrak{R}_1$, $\mathfrak{R}_2$, and $\mathfrak{R}_3$, respectively.

5.1. Performance of the EPFS Algorithm. This paper analyzes the EPFS algorithm by taking the 3000th second of the constellation run period. As shown in Figures 11–13, the path between the source and destination is calculated using the EPFS algorithm, Dijkstra shortest path algorithm, and Epidemic algorithm. In Figure 11, the route calculated by the EPFS algorithm and the path computed by the Dijkstra shortest path algorithm overlap entirely. That is, the path calculated by the EPFS algorithm is the shortest. The paths using the EPFS algorithm in Figures 12 and 13 partially overlap with the paths calculated by the Dijkstra algorithm, indicating that the paths calculated by the EPFS algorithm are not optimal every time. Simulations are performed for the whole constellation cycle, and the results show that the

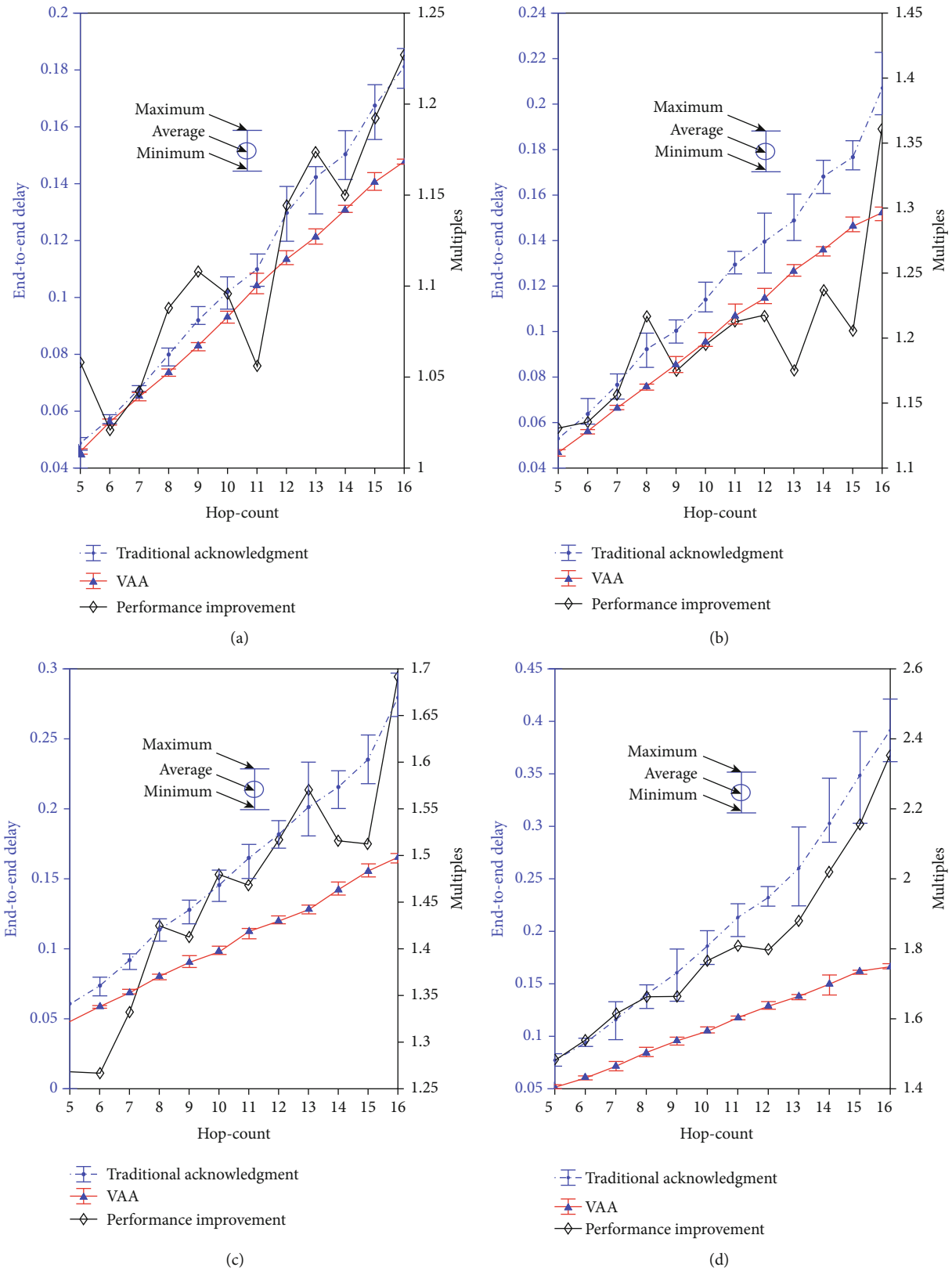

Figure 17: End-to-end delay variation with hop count under different ISL loss rates. (a) The maximum packet loss rate of 6% of the ISLs. (b) The maximum bundle loss rate of 10% of the ISLs. (c) The maximum bundle loss rate of 20% of the ISLs. (d) The maximum bundle loss rate of 30% of the ISLs.

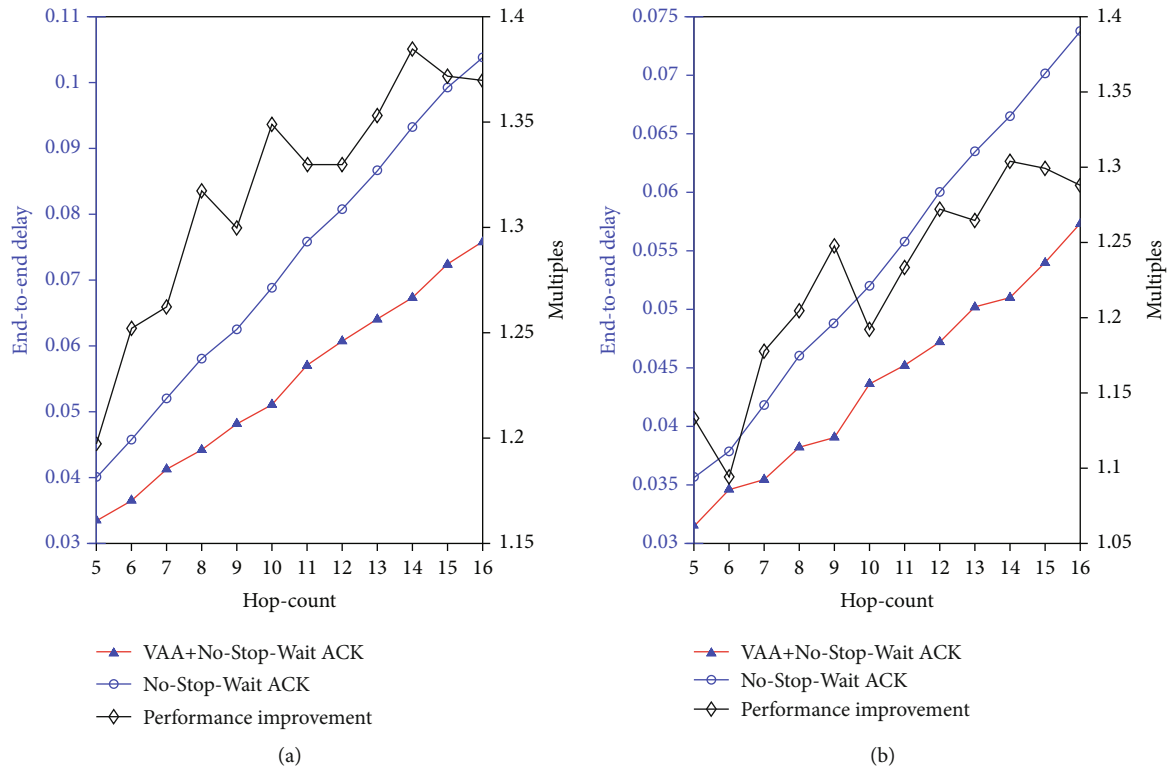

Figure 18: Comparison of end-to-end latency of VAA + No-Stop-Wait ACK and No-Stop-Wait ACK. (a) A group contains three bundles. (b) A group contains five bundles.

bundle can be sent from the source node to the destination node using EPFS.

Figure 14 shows the number of satellites that need to be involved for one routing based on the Dijkstra algorithm, Epidemic algorithm, and EPFS algorithm. This figure shows that the Dijkstra routing algorithm requires each satellite to be informed of the status of 648 satellites in the constellation. The EPFS algorithm, on the other hand, enables the transmission of the bundle with the least amount of knowledge of the constellation satellite state. As the distance between the source and destination nodes increases, the number of participating satellites required by the EPFS algorithm varies less. The Epidemic algorithm has a large variation in the number of satellites needed to participate as the distance increases.

To verify the delay of the EPFS algorithm, the transmission rate is set to 0.8 Mpbs-1.2 Mpbs, the size of each bundle is set to 1 KByte-2 Kbyte, and the link failure rate is set to 1%-5%. The Contact Graph Routing (CGR) [19] algorithm is forwarded only as a single copy. Figure 15 shows the experimental results of CGR, Enhanced Contact Graph Routing (ECGR) [43], Adaptive Message Replication Technique (AMRT) [44], and EPFS algorithms. The experimental results show that the EPFS algorithm is more advanced than other algorithms. The experimental results of the ECGR algorithm and CGR algorithm are not much different. The AMRT algorithm is superior to ECGR and CGR, because the AMRT algorithm considers historical congestion when selecting the next hop. The EPFS algorithm proposed by this paper is superior to the other three algorithms in terms of delay. This is because the EPFS algorithm equivalents the queue length to distance and obtains the link state to the next hop in real-time according to priori knowledge. When the link fails, the EPFS algorithm can avoid it in time.

5.2. Performance of the EBDA. In this experiment, we set the size of each bundle to be 1-20 M. Five bundles with similar emergency functions are used as a group. The probability of link congestion ranges from 2% to 30%. As shown in Figure 16, the relationship between the average congestion probability and the bundle drop rate for the three algorithms, FIFO, FCC, and EBDA, in the $\mathfrak{R}_1$, $\mathfrak{R}_2$, and $\mathfrak{R}_3$ paths is shown. The red curve shows the variation of the bundle drop rate with congestion probability for the EBDA algorithm in the three paths. The blue curve represents the traditional FIFO bundle dropping algorithm. The black curve represents the FCC algorithm. From Figure 16, the EBDA algorithm has the lowest drop rate of the bundle for the same congestion probability. As the congestion probability increases, the pronounced advantage of the EBDA algorithm (Table 2) shows the 15% probability of congestion, $\mathfrak{R}_1$, $\mathfrak{R}_2$, and $\mathfrak{R}_3$ paths, with the occurrence of congested satellites.

5.3. Performance of the VAA. In this simulation, the VAA algorithm is compared with the traditional acknowledgment algorithm and the No-Stop-Wait ACK proposed in literature [39]. The traditional acknowledgment algorithm uses the acknowledgment method in TCP/IP. As shown in Figure 17, the VAA algorithm is compared with the end-to-end latency of the traditional acknowledgment method

with the maximum bundle loss rate of 6% (Figure 17(a)), 10% (Figure 17(b)), 20% (Figure 17(c)), and 30% (Figure 17(d)) of the ISLs, respectively. The retransmission timing is calculated iteratively according to Equation (22), with α and β set to 0.5. The red curve in the figure represents the VAA algorithm. The blue curve represents the traditional acknowledgment algorithm. The black curve represents the end-to-end latency multiplier reduction of the VAA algorithm over the traditional acknowledgment algorithm. From Figure 17, the VAA algorithm has more significant advantages with increased link bundle loss rate and path hop count. When the maximum bundle loss rate of the ISLs is 10%, the VAA algorithm reduces the end-to-end latency by about 1.4 times compared to the traditional acknowledgment algorithm. When the maximum ISL bundle loss rate is 30%, the VAA algorithm reduces the end-to-end latency by about 2.4 times compared to the traditional acknowledgment algorithm.

In the No-Stop-Wait ACK strategy in literature [39], packets are sent without waiting for an acknowledgment to continue sending packets. After a certain number of packages are sent, the destination satellite acknowledges them uniformly. This strategy reduces the time consumption caused by each acknowledgment and reduces the end-to-end latency. Figure 18 compares the No-Stop-Wait ACK strategy and VAA + No-Stop-Wait ACK optimized with the VAA algorithm. Figure 18(a) shows a group of three bundles, and the destination node replies after sending three consecutive bundles. Figure 18(b) shows a group of five bundles. The destination node responds with an acknowledgment after receiving five consecutive bundles. From Figure 18, as the number of hops on the paths increases, the use of VAA optimization can reduce the end-to-end latency when transmission errors occur in each group of bundles. The more the number of hops, the better the effect of the VAA algorithm.

6. Conclusion

This paper proposes a new LIDTN scheme suitable for large-scale constellation data transmission. First, the next hop to which the bundles will be forwarded is determined based on the equivalent distance and priori knowledge. Second, a bundle drop algorithm based on the emergency function is proposed to solve the congestion problem in the bundle forwarding process. Finally, a virtual acknowledgment algorithm is proposed to reduce the retransmission time and retransmission overhead. The EPFS algorithm provides data input for the EBDA and VAA algorithms, and the EBDA and VAA algorithms provide a priori knowledge for the EPFS algorithm. The three algorithms of the LIDTN scheme complement each other and adaptively solve the problems of bundles forwarding, congestion, and acknowledgment. Simulation results show that the LIDTN scheme is effective. In space-based networks with high link failure rates, the EPFS, EBDA, and VAA algorithms offer better performance than existing methods.

In future research, we will further optimize the LIDTN scheme so that the LIDTN scheme can be applied to large-scale multilayer polar/inclined orbit constellations.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (grant number U21A20443), the Youth Innovation Promotion Association CAS (grant number 2020294), and the Shanghai Industrial Collaborative Innovation Project (grant number 2021-CYTX2-KJ03).

References

[1] J. Li, H. Lu, K. Xue, and Y. Zhang, "Temporal Netgrid model-based dynamic routing in large-scale small satellite networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 6, pp. 6009–6021, 2019.

[2] S. Ji, M. Sheng, D. Zhou, W. Bai, Q. Cao, and J. Li, "Flexible and distributed mobility management for integrated terrestrial-satellite networks: challenges, architectures, and approaches," *IEEE Network*, vol. 35, no. 4, pp. 73–81, 2021.

[3] F. R. Zhang, J. F. Han, and P. Ruan, "Point-ahead analysis and pre-pointing link stability study of intersatellite laser communication," *Optica Applicata*, vol. 50, no. 1, pp. 111–126, 2020.

[4] J. Liu, R. Z. Luo, T. Huang, and C. W. Meng, "A load balancing routing strategy for LEO satellite network," *IEEE Access*, vol. 8, pp. 155136–155144, 2020.

[5] T. Zhang, F. Bai, T. Dong, J. Yin, Z. Liu, and Y. Su, "Switching-aware dynamic control path planning for software defined large-scale LEO satellite networks with GEO controllers," in *International Conference on Wireless and Satellite Systems*, pp. 153–167, Springer, Cham, 2021.

[6] N. Wang, L. Liu, D. Chen, H. Liu, and S. Hao, "A low time complexity segment routing approach for multi-commodity traffic flow in mega LEO constellation," *Acta Electronica Sinica*, vol. 49, no. 11, pp. 2124–2132, 2021.

[7] Z. Jia, M. Sheng, J. Li, D. Zhou, and Z. Han, "VNF-based service provision in software defined LEO satellite networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 9, pp. 6139–6153, 2021.

[8] M. A. A. Madni, S. Iranmanesh, and R. Raad, "DTN and non-DTN routing protocols for inter-CubeSat communications: a comprehensive survey," *Electronics*, vol. 9, no. 3, p. 482, 2020.

[9] Q. Chen, X. Q. Chen, L. Yang, S. Wu, and X. F. Tao, "A distributed congestion avoidance routing algorithm in mega-constellation network with multi-gateway," *Acta Astronautica*, vol. 162, pp. 376–387, 2019.

[10] E. Ekici, I. F. Akyildiz, and M. D. Bender, "A distributed routing algorithm for datagram traffic in LEO satellite networks," *IEEE/ACM Transactions on Networking*, vol. 9, no. 2, pp. 137–147, 2001.

[11] D. Fischer, D. Basin, and T. Engel, "Topology dynamics and routing for predictable mobile networks," in *2008 IEEE International Conference on Network Protocols*, p. 207, Orlando, FL, USA, Oct. 2008.

[12] Y. Lu, F. C. Sun, and Y. J. Zhao, "Virtual topology for LEO satellite networks based on earth-fixed footprint mode," *IEEE Communications Letters*, vol. 17, no. 2, pp. 357–360, 2013.

[13] G. K. Zhao, M. C. Yang, Q. Guo, and G. Wang, "An improved earliest-delivery routing algorithm in double-layered satellite delay tolerant networks," in *2018 International Symposium on Networks, Computers and Communications (ISNCC)*, Rome, Italy, Jun. 2018.

[14] F. Long, F. Sun, and Z. Yang, "A novel routing algorithm based on multi-objective optimization for satellite networks," *Journal of Networks*, vol. 6, no. 2, pp. 238–246, 2011.

[15] C. Chen and E. Ekici, "A routing protocol for hierarchical LEO/MEO satellite IP networks," *Wireless Networks*, vol. 11, no. 4, pp. 507–521, 2005.

[16] Y. Lu, Y. J. Zhao, F. C. Sun, and H. B. Li, "A survivable routing protocol for two-layered LEO/MEO satellite networks," *Wireless Networks*, vol. 20, no. 5, pp. 871–887, 2014.

[17] H. Nishiyama, Y. Tada, N. Kato, N. Yoshimura, M. Toyoshima, and N. Kadowaki, "Toward optimized traffic distribution for efficient network capacity utilization in two-layered satellite networks," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 3, pp. 1303–1313, 2013.

[18] S. E. Alaoui and B. Ramamurthy, "N-look ahead routing and scheduling (N-LARS) for DTN space networks," in *2018 IEEE International Conference on Communications (ICC)*, Kansas City, MO, USA, May 2018.

[19] I. Komnios, S. Diamantopoulos, and V. Tsaoussidis, "Evaluation of dynamic DTN routing protocols in space environment," in *2009 International Workshop on Satellite and Space Communications*, pp. 191–195, Siena, Italy, Sep. 2009.

[20] H. Liang, Y. Yang, and Z. Y. Wang, "Routing optimization algorithm of spatial DTN network based on multiattribute decision," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 4830701, 11 pages, 2021.

[21] A. P. Silva, S. Burleigh, C. M. Hirata, and K. Obraczka, "Congestion control in disruption-tolerant networks: a comparative study for interplanetary and terrestrial networking applications," *Ad Hoc Networks*, vol. 44, pp. 1–18, 2016.

[22] J. Shen, S. Moh, I. Chung, and X. M. Sun, "Buffer scheme optimization of epidemic routing in delay tolerant networks," *Journal of Communications and Networks*, vol. 16, no. 6, pp. 656–666, 2014.

[23] S. Iranmanesh, R. Raad, and K. W. Chin, "A novel destination-based routing protocol (DBRP) in DTNs," in *2012 International Symposium on Communications and Information Technologies (ISCIT)*, pp. 325–330, Gold Coast, QLD, Australia, Oct. 2012.

[24] F. Dang, X. L. Yang, and K. P. Long, "Spray and forward: efficient routing based on the Markov location prediction model for DTNs," *Science China-Information Sciences*, vol. 55, no. 2, pp. 433–440, 2012.

[25] H. Guo, X. W. Wang, M. Huang, and D. D. Jiang, "Adaptive epidemic routing algorithm based on multi queue in DTN," *Journal of Chinese Computer Systems*, vol. 33, no. 4, pp. 829–832, 2012.

[26] J. Zhang, G. Wang, C. Liu, F. Z. Zhao, and X. Zhang, "Delay tolerant network and the algorithms of DTN routing," in *2018 3rd International Conference on Communication, Image and Signal Processing (CCISP)*, vol. 1169, Sanya, China, Nov. 2018.

[27] C. Caini, P. Cornice, R. Firrincieli, and D. Lacamera, "A DTN approach to satellite communications," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 5, pp. 820–827, 2008.

[28] C. Caini, P. Cornice, R. Firrincieli, D. Lacamera, and S. Tamagnini, "A DTN approach to satellite communications," in *2007 International Workshop on Satellite and Space Communications*, pp. 173–177, Salzburg, Austria, Sep. 2007.

[29] H. Zhu, T. Zhang, Q. Wang, and Y. Gu, "Improvement of contact graph routing algorithm in LEO satellite DTN network," in *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, vol. 346, pp. 484–492, Springer, Cham, 2021.

[30] Z. Zhu, M. Chen, and C. Yan, "An efficient fragmentation congestion control algorithm in satellite DTN," in *2018 IEEE/CIC International Conference on Communications in China (ICCC)*, pp. 620–624, Beijing, China, 2018.

[31] R. Diana, E. Lochin, L. Franck, C. Baudoin, E. Dubois, and P. Gelard, "DTN routing for quasi-deterministic networks with application to LEO constellations," *International Journal of Satellite Communications and Networking*, vol. 35, no. 2, pp. 91–108, 2017.

[32] Z. Huang, Q. Zhang, X. Xin et al., "DTN routing algorithm based on service probability and limited copy for satellite networks," in *2017 16th International Conference on Optical Communications and Networks (ICOCN)*, Wuzhen, China, Aug. 2017.

[33] Z. Jia, M. Sheng, J. Li, and Z. Han, "Toward data collection and transmission in 6G space–air–ground integrated networks: cooperative HAP and LEO satellite schemes," *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 10516–10528, 2022.

[34] D. D. A. Handschuh and E. Bourgeois, "Optimization of constellation jettisoning regards to short term collision risks," *Acta Astronautica*, vol. 145, pp. 284–292, 2018.

[35] S. Ji, D. Zhou, M. Sheng, and J. Li, "Mega satellite constellation system optimization: from a network control structure perspective," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 913–927, 2022.

[36] Q. Chen, G. Giambene, L. Yang, C. G. Fan, and X. Q. 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.

[37] A. Lindgren and K. S. Phanse, "An approach to cross counter-rotating seams in LEO satellite network," in *32nd AIAA International Communications Satellite Systems Conference*, San Diego, CA, Aug. 2014.

[38] G. Dai, X. Chen, M. Zuo, L. Peng, M. Wang, and Z. Song, "The influence of orbital element error on satellite coverage calculation," *International Journal of Aerospace Engineering*, vol. 2018, Article ID 7547128, 13 pages, 2018.

[39] F. L. Tang, H. T. Zhang, and L. T. Yang, "Multipath cooperative routing with efficient acknowledgement for LEO satellite networks," *IEEE Transactions on Mobile Computing*, vol. 18, no. 1, pp. 179–192, 2019.

[40] J. Iwatani and T. Yakoh, "A detection method of active queue management in communication paths," in *2010 IEEE International Conference on Industrial Technology*, Via del Mar, Chile, Mar. 2010.

[41] A. Lindgren and K. S. Phanse, "Evaluation of queueing policies and forwarding strategies for routing in intermittently connected networks," in *2006 1st International Conference on Communication Systems Software & Middleware*, pp. 1–10, Via del Mar, Chile, Mar. 2006.

[42] A. Loukili, A. L. Wijesinha, R. K. Karne, and A. K. Tsetse, "TCP's retransmission timer and the minimum RTO," in *2012 21st International Conference on Computer Communications and Networks (ICCCN)*, Munich, Germany, July. 2012.

[43] J. Segui, E. Jennings, and S. Burleigh, "Enhancing contact graph routing for delay tolerant space networking," in *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, Houston, TX, USA, December 2011.

[44] S. Iranmanesh and M. Saadati, "A novel congestion control technique in delay tolerant networks," *International Journal of Interdisciplinary Telecommunications and Networking*, vol. 10, no. 1, pp. 20–35, 2018.