electronics

MDPI

Article

Performance Evaluation of Routing Algorithm in Satellite Self-Organizing Network on OMNeT++ Platform

Guoquan Wang, Jiaxin Zhang *, Yilong Zhang, Chang Liu and Zhaoyang Chang

Key Laboratory of Universal Wireless Communication, School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Ministry of Education, Beijing 100876, China; gqwang@bupt.edu.cn (G.W.); ylzhang@bupt.edu.cn (Y.Z.); 2019210087@bupt.edu.cn (C.L.); changczy@bupt.edu.cn (Z.C.)

* Correspondence: jxz@bupt.edu.cn

Abstract: Self-organizing networks of small satellites have gradually gained attention in recent years. However, self-organizing networks of small satellites have high topological change frequency, large transmission delay, and complex communication environments, which require appropriate networking and routing methods. Therefore, this paper, considering the characteristics of satellite networks, proposes the shortest queue length-cluster-based routing protocol (SQL-CBRP) and has built a satellite self-organizing network simulation platform based on OMNeT++. In this platform, functions such as the initial establishment of satellite self-organizing networks and cluster maintenance have been implemented. The platform was used to verify the latency and packet loss rate of SQL-CBRP and to compare it with Dijkstra and Greedy Perimeter Stateless Routing (GPSR). The results show that under high load conditions, the delay of SQL-CBRP is reduced by up to 4.1%, and the packet loss rate is reduced by up to 7.1% compared to GPSR. When the communication load is imbalanced among clusters, the delay of SQL-CBRP is reduced by up to 12.7%, and the packet loss rate is reduced by up to 16.7% compared to GPSR. Therefore, SQL-CBRP performs better in networks with high loads and imbalance loads.

Keywords: satellite self-organizing; routing algorithm; OMNeT++

check for updates

Citation: Wang, G.; Zhang, J.; Zhang, Y.; Liu, C.; Chang, Z. Performance Evaluation of Routing Algorithm in Satellite Self-Organizing Network on OMNeT++ Platform. *Electronics* **2024**, *13*, 3963. https://doi.org/10.3390/electronics13193963

Academic Editor: Paul Mitchell

Received: 23 August 2024
Revised: 29 September 2024
Accepted: 1 October 2024
Published: 9 October 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the popularization and development of the Internet, the Internet of Things, and mobile terminals, people's demand for accessing the network anytime and anywhere is becoming increasingly urgent. Satellite networks, due to their wide coverage, large communication capacity, and reliable and secure characteristics, play an increasingly important role in communication and have attracted extensive attention from academia and industry [1]. Traditional "all-in-one" satellites are expensive to build, and their launch technology is complex, making them unsuitable for mass production and launch. Compared to "all-in-one" satellites, small satellites have the advantages of low cost, simple launch and use, and high flexibility [2]. However, one small satellite has a weaker performance in power and computing capabilities. To address this issue, the concept of wireless self-organizing networking can be combined with satellite communication to form a satellite self-organizing network by combining multiple small satellites. In this model, when a task arrives, satellites within a certain area can construct a network and complete the task together like a large "virtual satellite". In this way, multiple small satellites can integrate resources, improving overall communication and computing capabilities to complete tasks successfully [3].

Currently, satellite self-organizing networks can be categorized into three types based on the spatial distribution of satellites: self-organized single constellation, self-organized satellite formation, and self-organized hybrid mega-constellation [4,5].

Self-organized single constellations are formed by temporarily integrating satellites within one constellation according to the scale of the mission and the coverage area required,

to jointly complete tasks through self-organized networking. In this scenario, satellites are isomorphic, with the same functionality and payload.

Self-organized satellite formation integrates satellites within a formation to create a network. There are two types of satellites in a satellite formation based on the function and payload: the leader satellite and the follower satellite. The follower satellites move around the leader satellite and have inter-satellite links with the leader satellite [6]. The leader satellite has stronger computing and communication capability and a more complete protocol stack, hence it often serves as the control node and network node for the formation. Data generated by the follower satellites are all forwarded through the leader satellite.

Self-organized hybrid mega-constellations are constructed by satellites with different functions distributed at different orbital altitudes. Similar to a single constellation self-organizing network, this type of network selects multiple satellites to form a self-organizing network to jointly complete tasks based on mission requirements. Due to the dense distribution of satellites in a heterogeneous mega-constellation and their placement at various altitudes, a single satellite may establish more than just four inter-satellite links. For example, Yan et al. proposed the Grid+ type inter-satellite link connection strategy, which allows a satellite to establish more than four inter-satellite links [7].

Despite the advantage small satellites have, the frequent topological changes, high transmission delays, and complex communication environment in networks of small satellites pose challenges to satellite self-organizing networks, making the routing schemes applied in traditional terrestrial networks no longer suitable for satellite self-organizing networks. In detail, The high-speed movement of satellites leads to frequent link switching and topology updates. If the routing strategies of terrestrial networks are applied to satellite self-organizing network directly, it would result in significant signaling overhead when re-building a routing table, occupying limited communication resources and affecting the performance of the network, especially in large-scale networks [5]. The frequent topology changes also lead to a high packet loss rate because of the outdated routing table. Additionally, the complex and harsh space environment, including multipath fading, the ionosphere, atmosphere, and solar activity, also severely impact inter-satellite links, potentially leading to the destruction of the links and even the nodes themselves [8]. Additionally, owing to the long-distance transmission between satellites, the delay of transmission between satellites is high [9].

Therefore, this paper proposes a shortest queue length-cluster-based routing protocol (SQL-CBRP) for the satellite self-organizing network scenario. This algorithm is based on a cluster-based network structure, dividing routing into inter-cluster and intra-cluster routing. Intra-cluster routing is forwarded by the cluster head, while inter-cluster routing selects the node with the shortest queue length among neighboring cluster heads as the next hop. This effectively avoids packet loss due to queue overflow (i.e., a new packet arrives when the queue is full) and reduces queuing delay. Furthermore, to prevent network collapse caused by the destruction of the cluster head, a sub-head is established. When the cluster head is destroyed, the sub-head can take over the original cluster head's role in data forwarding and cluster management. Additionally, this paper has constructed a satellite self-organizing network simulation platform based on OMNeT++ and has validated the performances of SQL-CBRP on this platform, comparing it with Dijkstra's algorithm and the GPSR (Greedy Perimeter Stateless Routing) algorithm.

The main works of this paper are as follows:

1. A novel routing strategy for satellite self-organizing network, shortest queue length-cluster-based routing protocol is proposed, which considers nodes' queue length when making routing decisions;
2. A satellite self-organizing network simulation platform is built based on OMNeT++. In this platform, some functions are designed and implemented, including initial network forming, cluster maintenance, data generation and routing, etc.;
3. The delay and packet loss performance of the proposed algorithm are tested and compared with GPSR and Dijkstra.

2. Related Works

The routing technologies currently applied to wireless self-organizing networks can generally be divided into three types: topology-based routing algorithms, location-based routing algorithms, and cluster-based routing algorithms. These three types of algorithms and their typical algorithms are shown in Table 1.

2.1. Topology-Based Routing Algorithms

Topology-based routing algorithms make decisions based on the network's topological structure (such as the number of hops, connectivity). This type of routing strategy is divided into proactive and reactive approaches. Proactive routing algorithms require nodes to maintain a global routing table locally and generate path selection schemes based on this routing table, such as optimized link state routing (OLSR) and its improved algorithms [10]. Reactive routing algorithms, on the other hand, calculate routes on demand based on real-time node communication requests, such as ad hoc on-demand distance vector routing (AODV), and its improved algorithms [11]. Yang tested the performances of AODV and OLSR in wireless self-organizing networks. The results showed that the maximum latency of AODV is about 4 s, and the packet delivery ratio is approximate 85%; the maximum latency of OLSR is nearly 2 s, and the packet delivery ratio fluctuates between 20% and 80% [12].

Table 1. Three types of self-organizing routing algorithms.

Self-Organizing Routing Algorithm	Characteristics	Delay	Packet Delivery Rate
Topology-based routing algorithm	Route calculation is based on the network's topological structure and is divided into active and passive methods.	OLSR: up to 2 s [12] AODV: up to 4 s [12]	OLSR: 20–80% [12] AODV: about 85% [12]
Location-based routing algorithm	Routing is calculated based on the geographical location of the nodes.	GPSR and its improved algorithm: up to 0.45 s [13]	GPSR and its improved algorithm: up to 95% [13]
Cluster-based routing algorithm	Divide the network into layers and categorize routing into inter-cluster and intra-cluster.	CBRP: up to 2.2 s [14]	CRBP: higher than 90% [14]

2.2. Location-Based Routing Algorithm

The basic idea of location-based routing algorithms is to make routing decisions based on the geographical location information of nodes. Greedy Perimeter Stateless Routing (GPSR) is a typical location-based routing algorithm. In networks using GPSR, nodes do not need to maintain the topological information of the network. Instead, they only need to maintain the location information of neighboring nodes and the destination node to select the neighbor closest to the destination node as the next hop. Dong improved GPSR based on Double Deep Q-Network (D-DQN) and tested the performance of the new algorithm and GPSR in mobile self-organizing networks. The results show that the delay of GPSR and its improved routing algorithm is up to 0.45 s, and the packet delivery rate can reach 95% at maximum [13].

2.3. Cluster-Based Routing Algorithm

The cluster-based routing algorithm divides routing into intra-cluster routing and inter-cluster routing. Members within a cluster only maintain routing information within their cluster, while inter-cluster routing is decided by the cluster head or gateway through on-demand routing. In large-scale networks, this type of algorithm can effectively reduce the number of nodes involved in routing calculations and shrink the size of the routing table, thereby reducing storage overhead. Therefore, in large-scale networks, it has better performance than other algorithms [14]. For example, Cluster-Based Routing Protocol

(CBRP) confines flooding to cluster heads and gateway nodes, thereby reducing the overhead compared to flooding across all nodes in the entire network [15]. Zhang proposed the Cluster-Based On-demand Multi-path Routing Protocol (COMRP) which searches for multiple paths that meet QoS requirements through the cluster structure. The results showed that the delay of COMRP is lower than 1.5 s, while the delay of CBRP is up to 2.2 s [14]. Xu et al. proposed Efficient Clustering V2V Routing Based on PSO (Particle Swarm Optimization) in VANETs (Vehicular Ad Hoc Networks), whose packet delivery rate is higher than 90% in VANETs [16].

Although the above algorithms have solved the end-to-end routing problem in self-organizing networks through various methods, they do not consider the overhead of the nodes, which may lead to high queuing delay and high loss rate of packet. Therefore, this paper proposes a new routing algorithm, the shortest queue length-cluster-based routing protocol (SQL-CBRP), which calculates routes based on the queue length of neighbor nodes.

3. Shortest Queue Length-Cluster-Based Routing Protocol

In the traditional satellite scenario, one satellite can establish up to four inter-satellite links. But some researchers have proposed encountering inter-satellite links (eISLs), allowing satellites to communicate without predetermined inter-satellite links when they are visible to each other [17]. In this way, one satellite can establish more than four ISLs. Based on the concept of eISLs, the scenario considered in this article is as shown in Figure 1. The satellite nodes are divided into multiple cluster structures, with members within a cluster directly connected to the cluster head. The cluster heads are interconnected to form a satellite self-organizing network. Users connect to the satellite and communicate via the satellite network.

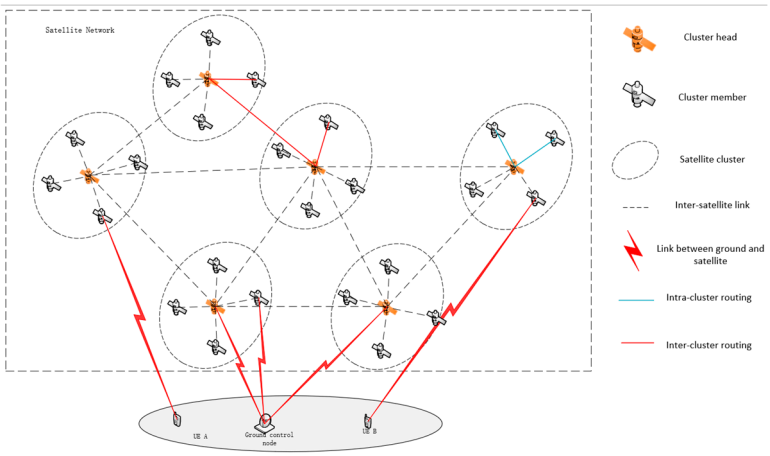


Figure 1. Satellite self-organizing network scenario.

Based on this scenario, a new routing algorithm called Shortest Queue Length-Cluster-Based Routing Protocol (SQL-CBRP) is proposed. In this algorithm, different nodes perform different functions. The specific functions are as follows:

- Cluster head: The cluster heads are generally located at the center of the cluster and are responsible for managing the cluster structure, maintaining network topology information, routing computation, and forwarding data packets;
- Cluster member: The cluster members orbit around the cluster head, directly connected to the cluster head. When a node has data to forward, it sends the data to the cluster head;
- Ground control node: The ground control node can communicate with all satellites in the network. When receiving a request to join the network from a satellite node, the ground control node assigns it to the nearest cluster head or creates a new cluster head, and allocates time slots for it;

- User: Users on the ground send data to a certain satellite. The data can be forwarded to the destination through the satellite network, and finally arrive at another user.

3.1. Initial Cluster Creation Process

This paper designs a satellite self-organizing network clustering process based on the high-rate wireless self-organizing network clustering method from reference [18]. A ground control node is introduced, and satellites send network access requests and their information (mainly position information) to the ground node. The ground node completes the division of cluster structures and the assignment of cluster heads and then sends the cluster affiliation information of each node back respectively.

The details of the clustering process executed by the ground control node are as follows:

1. The ground control node receives a network access request from a node;
2. The ground node determines whether there is a cluster head that is close enough to the node. If there is, the node is assigned to the nearest cluster head; if not, the node becomes a new cluster head;
3. After the assignment is completed, the ground node updates the local cluster structure information and ends the node's cluster allocation process.

3.2. Topology Information Maintenance Mechanism

Despite the routing decision process mainly happening in cluster heads, the source node also needs to know the cluster where the destination node is located to establish a path from the source node's cluster to the destination node's cluster. Therefore, each cluster head needs to maintain the global cluster structure information of the network. However, due to the mobility of the satellite self-organizing network, the cluster structure may change over time. Therefore, it is necessary for the cluster heads to periodically flood their connection table which contains the topology information of the cluster such as members and neighbors. The period is consistent with the cluster maintenance cycle.

The details of the topology information maintenance mechanism are as follows:

1. After the periodic cluster maintenance is completed, if the local cluster structure has been changed, the cluster head updates the connection table and sends it to neighboring cluster heads;
2. After receiving the broadcast connection table, the neighboring cluster head first checks if it is a new connection table. If it is, create a new connection table locally and update the local connection table list. If not, proceed to the next step;
3. Determine whether the connection table is the latest table of the corresponding cluster. If it is, update the local connection table list. Otherwise, discard this table;
4. Repeat the above three steps in a cycle until the connection table floods over the entire network.

3.3. Specific Routing Process of SQL-CBRP

As mentioned before, satellite self-organizing network routing faces issues such as high dynamics, high latency, and high signaling overhead. Therefore, SQL-CBRP adopts the concept of cluster-based routing, dividing the network into multiple cluster structures and separating routing into inter-cluster and intra-cluster routing. This approach can reduce the signaling overhead for network management and to some extent mitigate the impact of topology changes on routing, thereby reducing packet loss. Additionally, unlike traditional cluster-based routing protocols, SQL-CBRP does not use on-demand routing to establish paths passively. This is because in satellite networks with frequent topology changes, on-demand routing could lead to the frequent transmission of route discovery packets, which not only result in significant signaling overhead but also introduce latency in the process of searching for the routes. Accordingly, SQL-CBRP adopts a greedy algorithm to select the neighbor node with the shortest queue as the next hop, which not only eliminates the need for frequent route discovery flooding but also reduces queuing delay and packet loss due to queue overflow.

The specific routing decision process of SQL-CBRP is as follows: Every time a cluster head node u forwards a data packet, the next hop cluster head node v sends a message back to node u, informing it of its current queue length. When node u needs to forward data the next time, it compares the queue lengths sent back by all neighbor nodes last time according to the locally maintained neighbor queue length table and selects the neighbor with the shortest queue length to forward, as shown in Algorithm 1. The process of this algorithm is as follows:

1. When a cluster head node CH has a data packet to forward, it determines whether the destination node is within its cluster. If it is, the packet is sent directly to the destination node. If not, it consults the connection table;
2. If the local connection table indicates that there is a neighbor node closer to the destination node, then proceed to the next step. Otherwise, it is determined that the routing has failed;
3. Among the neighbor nodes closer to the destination node, compare their queue lengths and select the node with the shortest queue length as the next hop.

Algorithm 1 Routing Algorithm Based on Shortest Queue Length

Input: destination node *Dest*, neighbor cluster heads'set *N*

Output: *Nexthop*;

```
Calculate the distance between this node and the destination node DistanceLocal
if Dest is in this cluster then
    Nexthop <- Dest
else
    for each neighbor n ∈N do
        Calculate distance between neighbor n and destination node
    end for
    if there are neighbors closer to destination than this node then
        Nexthop <- id of the neighbor with shortest queue length
    else
     Nexthop <- −1
    end if
end if
return Nexthop
```

After the next hop node receives the data packet, it sends back its current queue length to node CH, and based on this, CH updates the queue length table of its neighbors. The updates of local queue length information are frequent as the cluster heads forward packets continuously, so a cluster head can determine next hop based on the queue length information updated recently, which can lead to a relevant reliable routing decision.

Although the information of nodes' queue lengths adds overhead, it does not need to be flooded over the entire network. Its transmission is limited between cluster heads, and it only needs to be sent to the last hop cluster head instead of being broadcasted. Therefore, the increased overhead is not severe. Compared to routing protocols that require global flooding of signaling, this algorithm actually reduces the overall network overhead.

4. Satellite Self-Organizing Network Simulation Platform Based on OMNeT++

In this part, a satellite self-organizing network simulation platform based on OMNeT++ is built to evaluate the performance of the satellite routing algorithm. OMNeT++ is an

open-source, component-based discrete event simulation tool widely used for network simulation. It has the following features:

1. Component-based simulation models: OMNeT++ models are based on components, which means that models can be constructed from multiple modular components, facilitating the building and expansion of models.
2. Flexible simulation process design: OMNeT++ allows users to freely design the simulation process, including the implementation of network protocols and the transmission of packets. Therefore, users can design the network structure and node behavior according to the specific conditions of their research network scenarios.
3. Graphical User Interface (GUI): OMNeT++ provides a GUI that enables users to visually observe the simulation process. By observing, users can confirm whether the designed and implemented functions operate correctly, thus verifying and modifying the design process.

Given these advantages of OMNeT++, this paper utilizes it to build a satellite self-organizing network simulation platform. In this platform, initial network formation clustering, periodic cluster maintenance, data generation and routing, and other functions are designed and implemented for the satellite self-organizing network. In the following sections, the simulation platform is introduced from three domains: network domain, module domain, and function domain.

4.1. Network Domain

In OMNeT++, the distribution of the network is set in the *.ned and *.ini files, where the number, mobility, and distribution of nodes can be set. In this paper, the distribution of satellite nodes in the network simulates a leader–follower satellite formation flying scenario, with one leader node and four follower nodes in a formation. The follower nodes follow the leader node in orbiting flight, as shown in Figure 2. A total of 20 formations are set up, with a total of 80 satellites. The distance between the master and slave nodes is 100 km, and the distance between the master nodes is between 600 km and 700 km [19]. During the simulation process, the topology between the master satellites remains relatively stable.

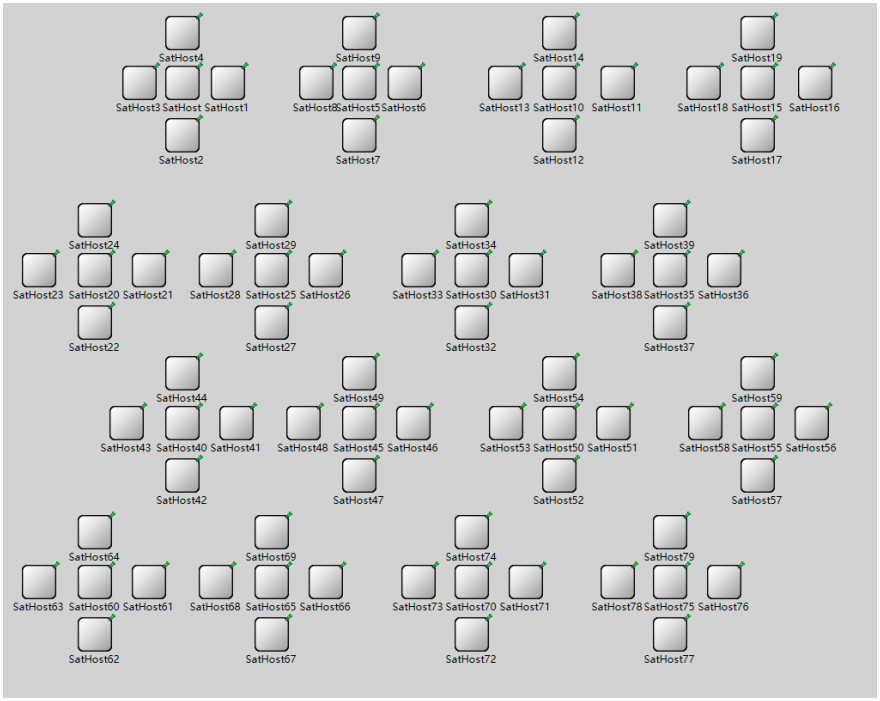


Figure 2. Node distribution in simulation.

4.2. Module Domain

In OMNeT++, a module is a functional entity that can be programmed to perform certain functions. Modules can communicate through artificially defined interfaces. Additionally, modules can be nested to form a hierarchical structure. In the satellite self-organizing network system implemented in OMNeT++, the module structure of the ground nodes and satellite nodes is consistent, both inheriting from the NodeBase module of the INET Framework. Based on the NodeBase module, network layer modules and Mac layer modules are added, as shown in Figure 3. The downward interface of the network layer module and the upward interface of the Mac layer module are bidirectionally connected, and the downward interface of the Mac is bidirectionally connected with the external interface of the node module. Although the structures of the two types of nodes are consistent, the functional design of their network layers and Mac layers is not the same, as shown in Figure 4.

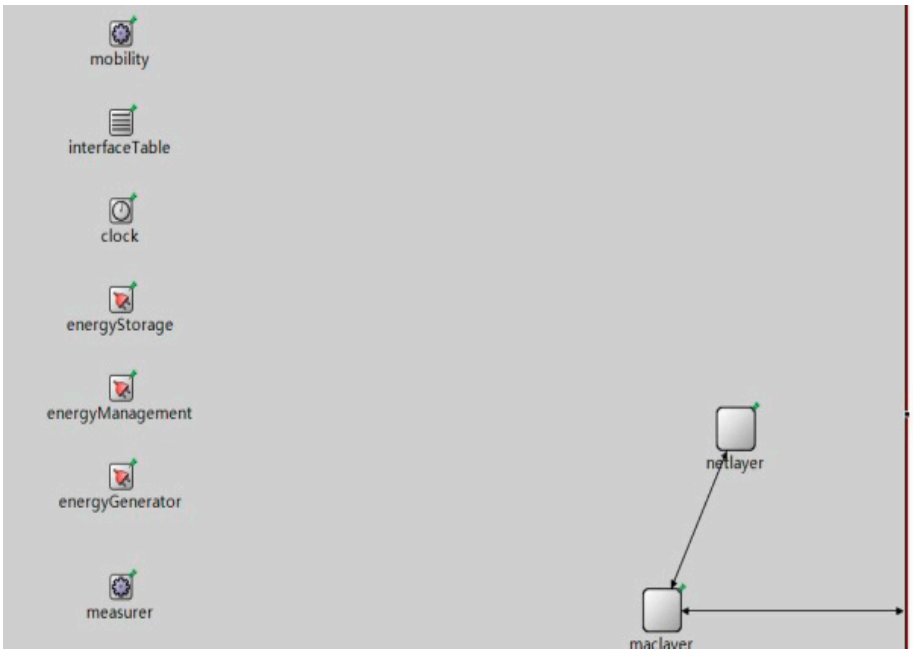


Figure 3. Structure of satellite and ground node.

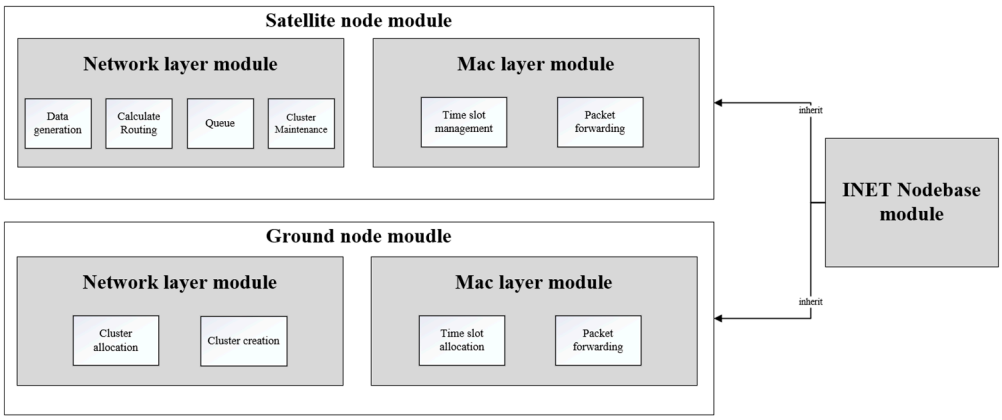


Figure 4. Functions of network layer and Mac layer of satellite and ground node.

The functions of satellite nodes' network layer module include cluster maintenance, data generation, and routing. The Mac layer module of the satellite nodes is responsible for sending messages passed down from the network layer to other nodes, which includes end-to-end transmission and broadcasting functions.

The network layer module of the ground nodes is responsible for cluster allocation to each node when the network is initially established, and the Mac layer module is responsible for allocating time slots for each node.

The specific function is illustrated in the following part.

4.3. Function Domain

The function domain mainly has four parts: initial network formation, cluster maintenance, time slots allocation, and data generation and forwarding.

4.3.1. Initial Network Formation

Based on the high-speed wireless self-organizing network clustering method from reference [18], this paper implements a satellite self-organizing network clustering process, which introduces a ground control node for clustering. Satellites send network access requests and their information (mainly location information) to the ground node, which completes the division of the cluster structure and the allocation of cluster heads and then sends back the cluster allocation information for each node separately. The initial network formation process is deployed at the network layer module of the ground node.

The specific network formation process is as follows:

1. The joining satellite node sends a network access request message to the ground node;
2. Based on the location information of the joining node and current network cluster structure, the ground node performs an initial cluster allocation for the new joining node. After the allocation is completed, the ground node sends a message to the joining node carrying the assigned node ID, cluster information, whether it is a cluster head, time slots, and other information;
3. The joining node takes corresponding actions based on its assigned status. If it is assigned as a member node, it sends a message to the cluster head node, informing its own ID information and time slot. If it is assigned as a cluster head, it broadcasts a "hello" message to its neighbors to announce its presence, while also listening for messages from neighboring cluster heads and new joining member nodes in its cluster to establish a local connection table.

4.3.2. Cluster Maintenance

After the initial cluster structure is established, due to the movement of the satellites, the relative positions of the nodes may change. These changes can affect the structure of the clusters, which may lead to network instability. Therefore, it is necessary to establish a cluster maintenance mechanism to maintain the relative stability of the clusters.

The content of cluster maintenance mainly involves monitoring and responding to the behavior of nodes, such as leaving clusters, joining clusters, and damage. The satellite nodes in the network are divided into three types: cluster head nodes, member nodes, and isolated nodes. Different nodes perform different cluster maintenance methods to work together to complete the cluster maintenance. This process is deployed at network layer modules of satellite nodes.

The specific cluster maintenance operations performed by nodes of different types are as follows:

1. Cluster head node. The cluster head node maintenance process is shown in Figure 5a. The cluster maintenance of the cluster head is periodically triggered. By using the "ScheduleAt()" method in OMNeT++, the network layer module of the cluster head can send a message object to itself after a certain period. This method can be used to set a timer to periodically trigger cluster maintenance. After the cluster head triggers the cluster maintenance operation, it first broadcasts an "isConnected" message to the member nodes to confirm whether the member nodes are still within their connection range. If any node has left the cluster, the cluster head updates the local connection table and floods this new connection table. Meanwhile, the member with the highest connection degree (i.e., it has the most neighbors in the cluster) is elected as the

sub-head, and a message is sent to this member to notify it of becoming the sub-head. When the cluster head is damaged due to certain circumstances, the sub-head can become the new cluster head to manage other member nodes. This can effectively avoid the direct demise of the cluster after the cluster head is damaged, thereby reducing the frequency of cluster structure changes and enhancing the stability of the network;

2. Member nodes. The cluster maintenance process of member nodes is shown in Figure 5b. A timer is also set within the member nodes. If a member node receives an "isConnected" message from the cluster head before the timer is up, it resets the timer and sends its own position information and node ID in a "Reply" message to the cluster head for updating the local connection table and for the next sub-head election. If the timer is up and the member node still has not received a message from the cluster head, it is considered that the node has left the communication range of the cluster head due to its movement. Therefore, the node changes its status to "isolate" and switches to performing the cluster maintenance methods of an isolated node;
3. Isolate nodes. The process for isolated nodes to join a cluster is shown in Figure 5c. An isolated node listens for broadcast messages from cluster heads. If it receives a broadcast message from a cluster head, it sends a network access request to the cluster head. Then, a timer starts counting. If the node receives a confirmation message from the cluster head before the timer is up, the node changes its state to "member". If the timer is up and the node still has not received a confirmation message from the cluster head, it continues to listen for cluster head broadcasts.

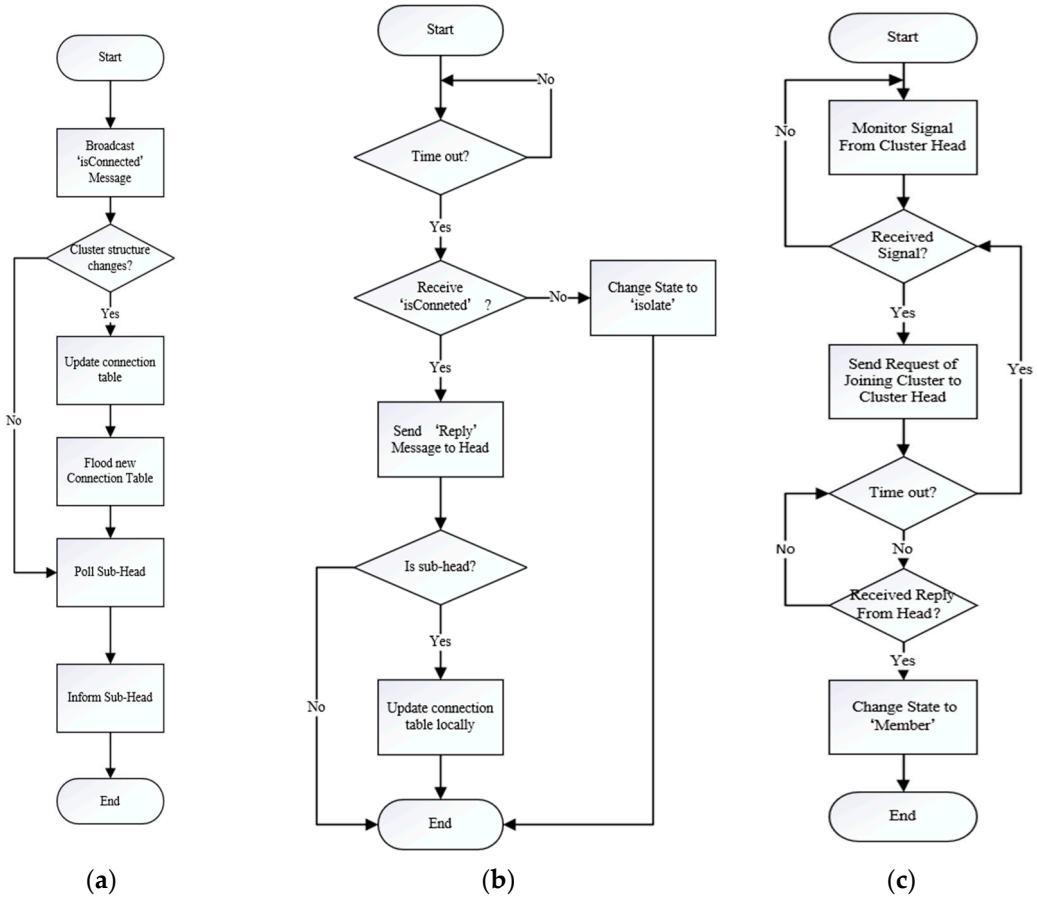


Figure 5. Cluster maintenance process of different nodes. (**a**) Cluster maintenance process of cluster head. (**b**) Cluster maintenance process of member. (**c**) Cluster maintenance process of isolate node.

4.3.3. Time Slot Allocation

In order to avoid congestion and packet loss caused by multiple nodes sending messages simultaneously during message transmission, it is necessary to allocate a time slot to each node. Nodes send messages within their allocated time slots to prevent sending messages simultaneously. Since the network has a layered structure and different clusters are relatively independent, time slots can be independently allocated within each cluster to achieve the reuse of time slot resources and improve resource utilization. In the scenario considered in this paper, the number of nodes in a cluster is limited to under 9, so 1 s can be divided into ten time slots, each 100 ms, and they can be allocated to the member nodes within the cluster. Time slot allocation can be divided into time slot allocation for nodes initially joining the network and time slot allocation for isolated nodes joining the network.

The specific process of time slot allocation for initial network access is as follows:

1. When the ground node receives the satellite node's network access request, after assigning the cluster affiliation in the network layer, the Mac layer allocates an idle time slot to the node and sends the time slot information to the node through a message;
2. After the node receives the message containing the time slot information, it informs the cluster head of its time slot information through a message;
3. The cluster head updates the utilization of time slot resources in the cluster.

The time slot allocation process for isolated nodes joining the network is as follows:

1. The isolated node sends a network access request to the cluster head;
2. After receiving the access request, the cluster head checks if there are any remaining idle time slots in the cluster. If there are idle time slots, proceed to the next step;
3. The cluster head allocates an idle time slot to the new joining node and notifies the node through a message.

The time slot allocation process is deployed at the Mac layer modules of satellite nodes and the ground node. Additionally, the Mac layer modules of the satellite nodes also have the function of managing time slots, where the cluster heads manage the time slots of the members within the cluster, and the member nodes send data based on their time slots.

4.3.4. Data Generation and Forwarding

Each member node has a certain probability of generating a data packet every second. The node sends this data packet to the cluster head within its time slot. If the cluster head's queue is full when it receives the data packet, the packet is discarded. If the data queue is not full when receiving a data packet, the data are added to the queue for processing. Once all the data packets ahead in the queue have been forwarded, the cluster head parses the destination node of the data packet, calls the routing algorithm function to calculate the next hop, and forwards it. In the previous text, the specific process of the SQL-CBRP algorithm has been introduced. To implement this algorithm on the OMNeT++ platform, nodes need to be able to obtain the location information of other nodes. Since the nodes inherit from the INET Framework's NodeBase module, which includes a mobility module, the location information of each node can be obtained from it. In addition, the algorithm requires maintaining a neighbor table locally at each node. Therefore, in this platform, a neighbor class that includes the neighbor's ID and its queue length is created, and a neighbor list composed of neighbor objects is created in the network layer module of each satellite node. Each time a neighbor sends back the queue length information, the queue length of the corresponding neighbor object is updated for routing decisions. When a cluster head receives a data packet forwarded from another cluster head, it executes the same queuing and forwarding process, until the packet arrives at its destination node.

The packet queue and routing calculation functions are deployed at network layer modules, and forwarding is implemented by Mac layer modules.

5. Simulation Analysis

The simulation of this paper is based on the satellite self-organizing network platform introduced before, mainly focusing on the performance of packet delay and packet loss rate when making routing decisions through SQL-CBRP, and comparing it with the Dijkstra algorithm based on the minimum number of hops and the GPSR algorithm based on the shortest distance.

5.1. Simulation Parameters

The simulation parameters are shown in Table 2. The bandwidth setting for satellite nodes is adapted from the reference [20]. This paper uses 15 Mbps as the message service transmission rate. According to ZTE's white paper '5G Video Uplink Transmission Applications and Challenges' published in 2021, the transmission rate for high-definition video is generally between 8 and 15 Mbps. This paper uses 8 Mbps as the video service transmission rate. In [21], it is mentioned that the satellite data uplink rate in emergency scenarios is 13.8 Mbps. In this paper, 15 Mbps is adopted as the transmission rate for message services. Additionally, since the standard rate used for telephony voice communication in G.711 is 64 kbps [22], this paper also uses 64 kbps for the voice service transmission rate. Since all data must be forwarded by the cluster heads, the cluster heads are prone to be overwhelmed. To avoid a queue overflow at the cluster head, the number of member nodes in a cluster should not be too large. Therefore, without loss of generality, in this paper the maximum number of cluster member nodes is set to 4.

Table 2. Simulation parameters.

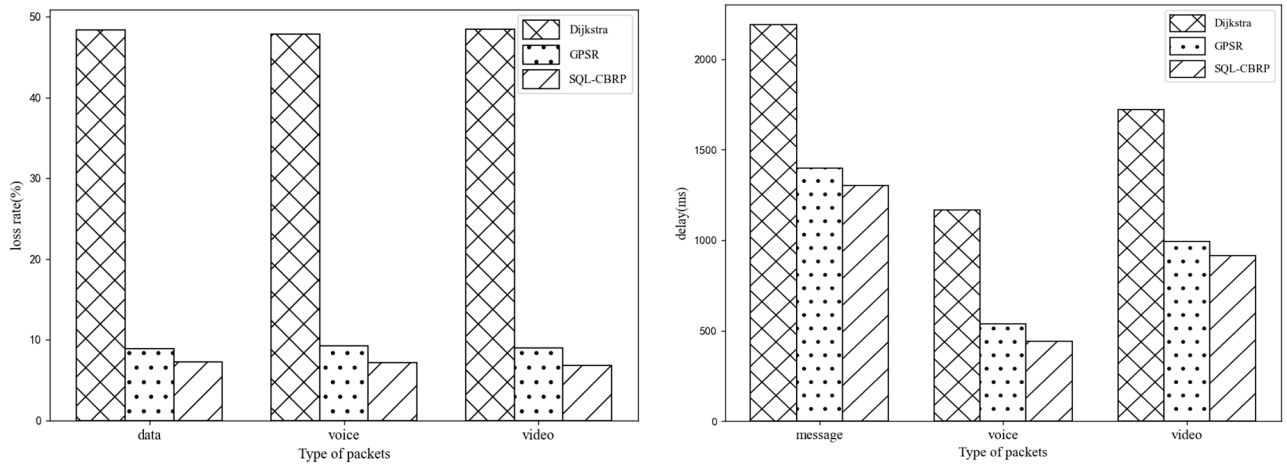
Parameter Name	Parameter Value
Number of satellite nodes	80
Distance between leaders	600–700 km
Distance between leader and follower	100 km
Nodes' speed	7.6 km/s
Nodes' longest queue length	9
Rate of packet generation	1 packet/s
Bandwidth of cluster head	30 Mbps [20]
Simulation duration	1500 s
Message service's data rate	15 Mbps [21]
Voice service's data rate	64 kbps [22]
Video service's data rate	8 Mbps

5.2. Simulation Results

5.2.1. Performance of Different Types of Packets

The packet loss rates for the three types of services are shown in Figure 6a. It can be observed that the packet loss rate of the Dijkstra algorithm is significantly higher than the other two algorithms. For different services, the algorithm proposed in this paper performs better than the traditional GPSR in terms of packet loss rate. Specifically, the average loss rate of SQL-CBRP is about 21.5% lower than GPSR relatively. This is because the SQL-CBRP algorithm selects the neighbor node with the shortest queue as the next hop, which can effectively prevent packets from being forwarded to a node whose queue is already full, thereby reducing the probability of packet loss.

The average delay results for the three types of services are shown in Figure 6b. It can be seen from the figure that for the three different service scenarios, the routing algorithm proposed in this study has shown some optimization in terms of delay compared to the GPSR algorithm. Specifically, the reduction in average delay is about 90 ms, which is about 11.0% lower than GPSR relatively. The emergence of this result is because SQL-CBRP takes into account both the queue length and the geographical location, which can reduce the queuing delay and propagation delay simultaneously.



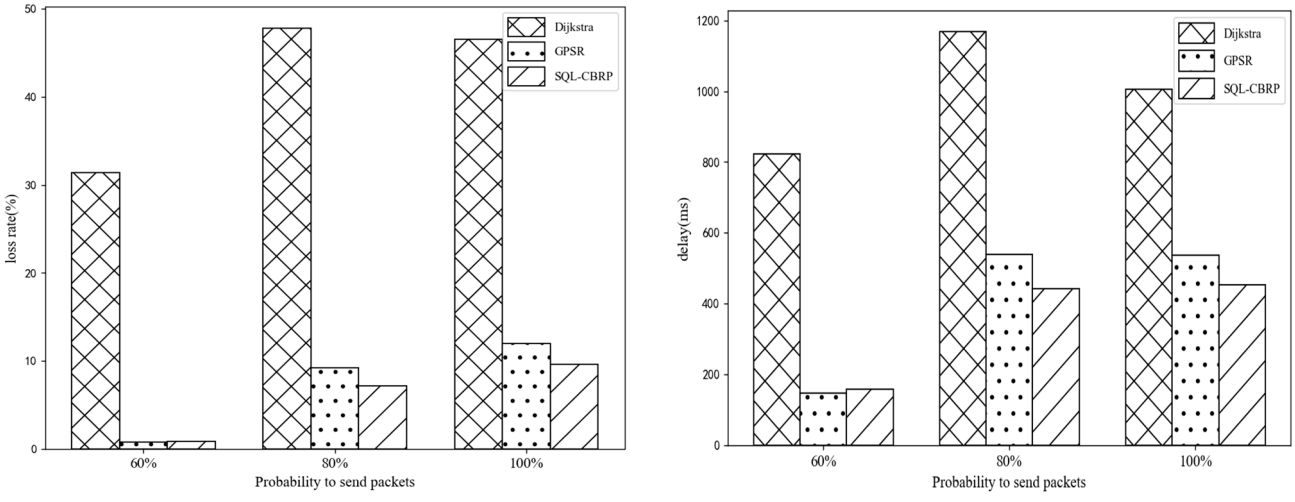
(**a**) (**b**)

Figure 6. Packet loss rate and delay of voice, message, and video service in Dijkstra, GPSR, and SQL-CBRP. (**a**) Packet loss rate of different types of packets; (**b**) Delay of different types of packets.

5.2.2. Performance of Different Packet-Sending Probability

The SQL-CBRP proposed in this paper makes routing decisions based on the queue length of neighboring nodes. Therefore, the number of packets in the network is likely to affect the performance of the algorithm. To test the impact of the number of network packets, this section sets different packet-sending probabilities and tests the performances of the Dijkstra algorithm, GPSR algorithm, and SQL-CBRP algorithm under different sending probabilities.

According to Figure 7, when the packet-sending probability is 60%, the delay and packet loss rate of SQL-CBRP are slightly higher than those of GPSR, but the difference is negligible. When the packet-sending probability is 80% and 100%, the delay and packet loss rate of SQL-CBRP are reduced to some extent compared to GPSR. Specifically, the delay is reduced by up to 18.0% relative to GPSR, and the packet loss rate is reduced by up to 22.1% relative to GPSR.



(**a**) (**b**)

Figure 7. Packet loss rate and delay of Dijkstra, GPSR, and SQL-CBRP when the probability of sending packet is 60%, 80%, and 100%. (**a**) Packet loss rate of different packet-sending probability; (**b**) Delay of different packet-sending probability.

5.2.3. Performance under Load Imbalance Conditions

To test the performance of SQL-CBRP under load imbalance conditions, the packet-sending probability of nodes within several clusters is set to 90%, while the packet-sending probability of the remaining nodes within clusters is set to 60%. The latency and packet loss rate of the three algorithms are tested under these conditions.

Figure 8 illustrates the comparison of delay and packet loss rate for the three algorithms when the number of high-load clusters is 4, 8, and 12. It is evident that under the condition of load imbalance, the SQL-CBRP algorithm has a lower delay and packet loss rate compared to Dijkstra and GPSR. Moreover, compared to GPSR, the delay is reduced by up to 24.5%, and the packet loss rate is reduced by up to 32.7% relatively. This is because SQL-CBRP selects nodes with shorter queuing queues (i.e., nodes with lower load) as the next hop. Therefore, in a network with load imbalance, SQL-CBRP can play a role in load balancing to a certain extent, thereby directing packets to areas with lower load, which in turn reduces the network's latency and packet loss rate.

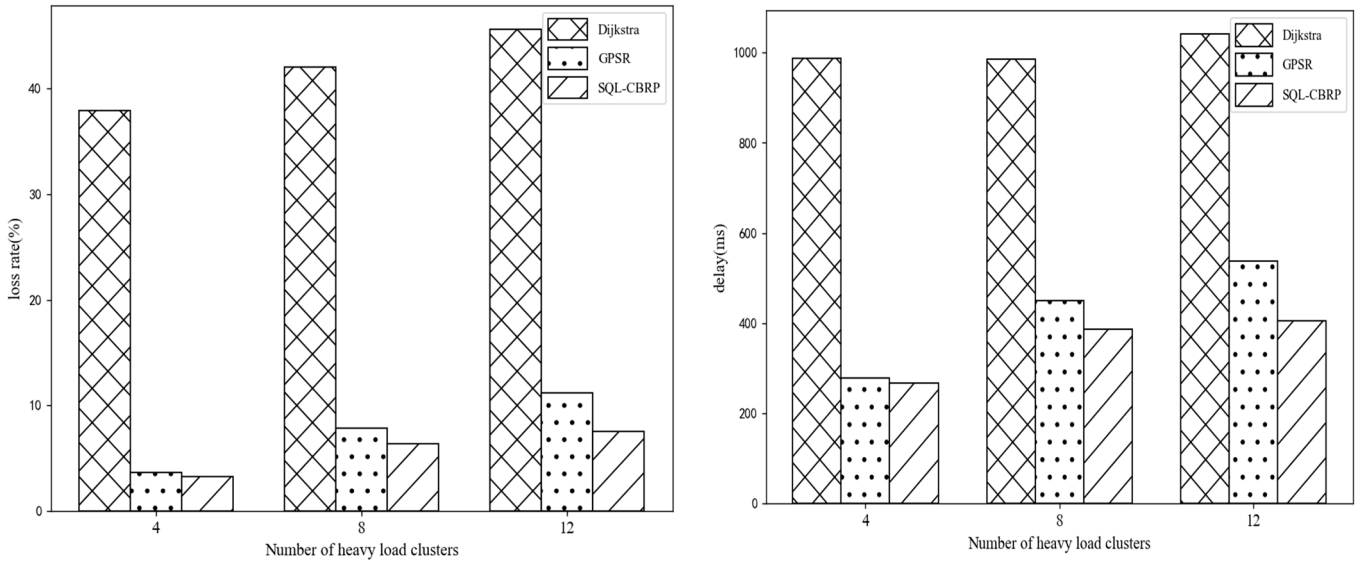


Figure 8. Packet loss rate and delay of Dijkstra, GPSR, and SQL-CBRP when the number of heavy load clusters is 4, 8, and 12. (**a**) Packet loss rate of different numbers of heavy load clusters; (**b**) Delay of different numbers of heavy load clusters.

According to the analysis above, it is evident that since the SQL-CBRP algorithm selects the neighbor with the shortest queue as the next hop, it reduces queuing delay and effectively avoids packet loss due to queue overflow. Therefore, compared to the Dijkstra algorithm and the GPSR algorithm, the SQL-CBRP algorithm has lower packet loss rates and delays for services with different data rates. Specifically, when the network has high loads or the load distribution is imbalanced, the advantages of SQL-CBRP become more pronounced.

6. Conclusions

This paper proposes the SQL-CBRP, a new routing strategy for satellite self-organizing networks, and tests the performance of this routing strategy through a satellite self-organizing network simulation platform based on OMNeT++. SQL-CBRP combines the cluster-based routing and greedy algorithm, selecting the next hop by the queue lengths of neighbor nodes, which can reduce delay, packet loss rate, and the signaling overhead in the whole network. The results show that for the three types of services with specified transmission rates, the packet loss rate and delay performance of SQL-CBRP are generally better than Dijkstra and GPSR. Furthermore, this paper also tests the performance of the

SQL-CBRP under different packet-sending probabilities. The results indicate that under conditions of high packet-sending probability and imbalance load, the performance of the SQL-CBRP algorithm is superior to the other two algorithms, confirming the superiority of SQL-CBRP in terms of performance under higher and imbalanced network loads.

Author Contributions: Conceptualization, G.W. and J.Z.; methodology, G.W. and J.Z.; software, G.W.; validation, G.W. and C.L.; formal analysis, G.W.; investigation, G.W. and J.Z.; resources, J.Z.; data curation, G.W. and Y.Z.; writing—original draft preparation, G.W.; writing—review and editing, J.Z. and C.L.; visualization, Z.C. and Y.Z.; supervision, J.Z.; project administration, J.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by "the Fundamental Research Funds for the Central Universities", grant number 2023ZCJH09.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhang, J.; Chang, C.; Zhang, Y.; Li, R.; Wang, K.; Zhang, X. Survey on Routing Technology of Mega Constellation. *Space-Integr.-Giround Inf. Netw.* **2024**, *5*, 2–13.
2. Liu, J.; Zhang, X.; Zhang, R.; Huang, T.; Yu, F.R. Reliable and low-overhead clustering in LEO small satellite networks. *IEEE Internet Things J.* **2021**, *9*, 14844–14856. [CrossRef]
3. Radhakrishnan, R.; Edmonson, W.W.; Afghah, F.; Rodriguez-Osorio, R.M.; Pinto, F.; Burleigh, S.C. Survey of inter-satellite communication for small satellite systems: Physical layer to network layer view. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 2442–2473. [CrossRef]
4. Hui, L.; Xue, Z.; Ma, W. A TDMA-based Combined MAC Protocol forthe Small Satellite Formation System. *Space Electron. Technol.* **2016**, *13*, 23–28.
5. Zhang, J.; Wang, K.; Li, R.; Chang, Z.; Zhang, X.; Wang, W. MaCro: Mega-Constellations Routing Systems with Multi-Edge Cross-Domain Features. *IEEE Wirel. Commun.* **2023**, *30*, 69–76. [CrossRef]
6. Liu, G.-P.; Zhang, S. A Survey on Formation Control of Small Satellites. *Proc. IEEE* **2018**, *106*, 440–457. [CrossRef]
7. Yan, F.; Nie, H.; Xia, W.; Shen, L. Inter-satellite link connection strategy for large-scale low earth orbit satellite networks. *J. Commun.* **2024**, *45*, 100–109.
8. Wei, W.; Fu, L.; Wang, K.; Lu, X.; Zhou, Z. Survey of routing technologies for the satellite Internet. *J. Xidian Univ.* **2024**, accepted.
9. Cao, X.; Li, Y.; Xiong, X.; Wang, J. Dynamic Routings in Satellite Networks: An Overview. *Sensors* **2022**, *22*, 4552. [CrossRef] [PubMed]
10. Singh, J.; Singh, G.; Gupta, D.; Muhammad, G.; Nauman, A. OCI-OLSR: An Optimized Control Interval-Optimized Link State Routing-Based Efficient Routing Mechanism for Ad-Hoc Networks. *Processes* **2023**, *11*, 1419. [CrossRef]
11. Yang, B. Research on the Improvement of AODV Routing Protocol in ad Hoc Networks of UAVs. Master's Thesis, Xian University of Technology, Xian, China, 2023.
12. Yang, M. Research and Optimization of Routing Link Selection Method in Wireless ad Hoc Network. Master's Thesis, Harbin University of Science and Technology, Harbin, China, 2023.
13. Dong, L. Research on Mobile ad Hoc Network Routing Technology. Master's Thesis, Beijing University of Post and Telecommunication, Beijing, China, 2023.
14. Zhang, Y. Research on the Networking Technology of the Large-Scale Small Satellites Cluster. Master's Thesis, University of Chinese Academy of Sciences, Beijing, China, 2021.
15. Xu, J. Research on Centralized Adaptive Hybrid Routing Mechanism for ad Hoc Network. Master's Thesis, Beijing University of Post and Telecommunication, Beijing, China, 2015.
16. Bao, X.; Li, H.; Zhao, G.; Chang, L.; Zhou, J.; Li, Y. Efficient clustering V2V routing based on PSO in VANETs. *Measurement* **2020**, *152*, 107306. [CrossRef]
17. Wang, K. Research on Highly Reliable Routing Control Systems and Methods for Satellite Network. Master's Thesis, Beijing University of Post and Telecommunication, Beijing, China, 2024.
18. Guo, G. Research and Implementation on a Network Layer Protocol of High-Rate Wireless ad Hoc Network. Master's Thesis, Beijing University of Post and Telecommunication, Beijing, China, 2019.
19. Cheng, C.; Ren, X.; Xu, S. Starlink and its operational analysis. *Command Control Simul.* **2024**, *46*, 154–160.
20. Wang, C.; XU, P.; Zhang, S.-b.; Wang, L.-q.; Wang, W.-d. Research and implementation of dynamic routing protocol for LEO satellites based on Linux system. *J. Beijing Univ. Posts Telecommun.* **2020**, *43*, 94.

21. Xiao, K.; Liu, X.; Wei, H.; Fei, Z. Application of the on the move phased array satellite communication antenna in emergency communication. *Chang. Inf. Commun.* **2023**, *36*, 187–190.
22. Feng, B. Research on Preprocessing Algorithm of G.722 Wideband Speech Signal. Master's Thesis, Dalian University of Technology, Dalian, China, 2009.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.