International Journal of Computer Information Systems and Industrial Management Applications
ISSN ISSN 2150-7988 Volume 17 (2025) pp.310–329

Review

Shortest Path Orbit Prediction for Resilient LEO Satellite Networks to Navigating the Cosmos

V. Kishen Ajay Kumar [1], Sukerthi Sutraya [2], C. Sateesh Kumar Reddy [3], K. Roopa [4], N Balakrishna [5] and G. Sujatha [6]

[1] Department of Electronics and Communication Engineering, Institute of Aeronautical Engineering, Dundigal, Hyderabad 500 043, India; v.kishenajaykumar@iare.ac.in
[2] Department of Data Science, G Narayanamma Institute of Technology and Science for Women ; sukeerthisutraya@gmail.com
[3] Advanced CSE, School of Computing Informatics, Vignan's Foundation for Science, Technology and Research, Vadlamudi 522 213, Guntur. India; satishreddic@gmail.com
[4] Department MCA, Madanapalle Institute of Technology and Science, Kadiri Road Angallu, Madanapalle, Andhra Pradesh 517 325, India; balu1203@gmail.com
[5] Department of CSE (AI&ML), School of Computing, Mohan Babu University, Tirupati, Andhra Pradesh 517 102, India; balu1203@gmail.com
[6] Department of ECE, Sri Venkateswara College of Engineering (Autonomous), Karakambadi Road, Tirupati 517 507, India; sujatha.g@svcolleges.edu.in
* Correspondence author: V. Kishen Ajay Kumar

Received date: 12 September 2024; Accepted date: 26 February 2025; Published online: 27 March 2025

Abstract: TLow-Earth-Orbit (LEO) satellite networks have great potential to supplement terrestrial infrastructure by offering ultra-low latency and worldwide coverage. However, effective and resilient routing is a considerable challenge due to the frequent topology changes inherent in these networks. Existing methods, the use of terrestrial routing techniques such as OSPF, lead to persistent route convergence and extreme bandwidth consumption over inter-satellite links. To deal with these, this paper proposes OPSPF, a unique routing protocol for LEO satellite networks. OPSPF takes advantage of the constellation's predictability by periodically calculating routes for the creation of instantaneous routing tables, which allows it to handle frequent changes in topology efficiently. Furthermore, Additionally, OPSPF presents a dynamic routing system that is available on-demand to manage erratic topology changes caused by link failures and recovery. The evaluation showed that compared to OSPF, OPSPF eliminated route gathered overhead during periodic topology changes and acquired a 55% reduction in communication overhead and an 85% deduction in route convergence time during uneven topology changes. This work showed an optimistic solution to improve the efficiency and stability of routing in LEO satellite networks, paving the way for enhanced performance and flawless integration.

Keywords: orbit prediction; cosmos.Navigation; routing model; sortage path; Q-Learning framework

1. Introduction

In today's quickly developing world of technology and infrastructure, adaptability is essential for resilience. Imagine a network that can detect changes in its surroundings, anticipate movement, and swiftly modify its configuration to ensure continuous operation. This concept is forcing the integration of movement predictions into powerful systems. Network infrastructure resilience plays an important role in our everyday life, where digital connectivity is an essential element; with the constantly growing technological progress and the increasing mobility of devices and users, networks face a continuous challenge: changes in their configuration can appear due to movement.

Although the land-based network infrastructure has grown quickly, there are still some deployment issues that need to be resolved. First of all, densely populated areas have more infrastructure installed due to economic considerations. Nonetheless, more than 70% of Earth's surface is submerged under water, and even on land, the majority of people live in cities, and this reduces the economic viability of setting up network infrastructure in remote locations. The

Low Earth Orbit (LEO) satellite network is considered the perfect complement to terrestrial networks because of its worldwide coverage and incredibly low latency. Currently, 4000 reasonably priced, fast, low-Earth orbit satellites are scheduled for launch by SpaceX to handle internet and backhaul traffic [1]. Low-latency, dependable LEO satellite networks are becoming more and more important.

Satellite communication networks are essential for space-ground integrated information systems, delivering quick and reliable data transmission [2]. The rapid expansion of low-latency Low-Earth-Orbit (LEO) satellite networks is necessary for addressing the increasing demands of the Internet of Things. However, LEO satellites have their own set of drawbacks, too, including high speeds, condensed connection times, and restricted coverage, particularly when serving high-mobility users like speedy trains and aircraft. A diverse network with GEO, MEO, and LEO satellites [3] presents various services, but operating such a network and allocating resources becomes more challenging. Satellites can move quickly in relation to customers and one another, which can cause problems like frequency variations and weak connections. These problems can reduce the dependability of satellite transmissions and the capacity of on-demand services. Concerns about interference between and among satellites, as well as between satellites and ground stations, are further highlighted by the growing number of satellites and the finite amount of spectrum available. Transmission dependability is vital since satellites are vulnerable to several types of interference. Resilient satellite communication networks improve communication flexibility and offer creative solutions to these problems by utilizing several orbits, functions, and systems to allow users to switch between different satellite systems and frequencies with ease.

The routing algorithms employed in the Low Earth Orbit (LEO) satellite network can be separated into two main kinds: Virtual Topology-based Routing Algorithms (VTRA) [4,5] and Virtual Node-based Routing Algorithms (VNRA) [6,7]. In VTRA, the circling pattern of the satellite determines the division of network time into slots. Due to the stable network structure maintained by each slot, the route for data in each slot can be determined using conventional static routing techniques like the Open Shortest Path First (OSPF) technique. This method works dependably by making use of the structure of the satellite network, which is predictable and repeatable, but it does require prior knowledge of the satellite's orbit. But in Inter-Satellite Links (ISLs), this type of virtual topology-based and theoretical routing is prone to faults and disruptions. Virtual nodes with specified geographic locations are envisioned as part of the satellite network in VNRA. Every satellite is connected to a virtual node that is represented by its closest fixed coordinates. The proposed method strives to reduce the effect of satellite movements by isolating the logical structure of the network from the dynamic nature of satellite orbits by depending on these stable virtual nodes. Since the routing path in VNRA is defined between two virtual nodes rather than directly between the satellite pairs, it cannot be easy to maintain continuous end-to-end data transmission between satellites.

Recent advances in space technology study and evolution have sparked a wave of innovation in Low Earth Orbit (LEO) satellite constellations [8], with applications from positioning to communications and remote sensing. Numerous LEO satellites collaborate to present global or regional coverage services to ground stations [9,10]. An exhaustive coverage area not only supports essential military operations like strategic guidance and real-time surveillance for ground forces but also encourages civil applications such as satellite Internet in remote regions [11]. Moreover, it sets the basis for the expansion of 6G communication technologies [12,13]. As a result, a combination of models and algorithms [14] have appeared, with the aim of optimizing the setup and deployment of satellite constellations to maximize overall coverage effectiveness. However, as space systems are increasingly crucial in different sectors, the escalating security threats within the space domain have created satellites more vulnerable during their operational lifetimes, thus jeopardizing coverage performance. For illustration, biological attacks, including missiles and laser attacks, directly endanger the integrity of satellites. Additionally, cyber attacks like jamming [15] may interfere with and obstruct communication pathways, causing coverage services to deteriorate or stop working entirely. According to reports, since 1997, over a hundred satellites have been the target of attacks, including jamming and hijacking, which have had serious repercussions such as communication and navigational breakdowns [16]. Satellites are also seriously at risk from non-adversarial elements like space debris, which go beyond adversarial threats. For instance, the 2009 collision of two communication satellites resulted in approximately 2000 debris fragments, putting several satellites in orbit at risk. The space applications community is becoming increasingly concerned due to the mix of intentional, intelligent attacks and environmental risks.

Creating a reliable and effective routing system is one of the issues LEO satellite networks must overcome. The architecture of the network fluctuates over time in a typical LEO satellite constellation due to constellation movement. Because of this, using terrestrial routing protocols like OSPF [17] directly may result in frequent route recalculations, which may use up all of the costly and scarce inter-satellite link bandwidth. Hello, timeouts are used in OSPF to

detect topological changes in the network and broadcast these changes to the whole network for route recalibration. Prior research has concentrated on developing customized routing protocols for low-latency networks, putting forth a range of snapshot-based techniques [18]. These methods split the satellite constellation's whole motion period up into many time segments. Every segment keeps its constellation topology fixed, and an instantaneous routing table snapshot is generated for it using the traditional Shortest Path First (SPF) algorithm. An LEO satellite must either continuously communicate with ground stations to upload the correct routing table snapshot at the appropriate time or store a lengthy sequence of routing table snapshots in its constrained and expensive memory to handle these periodic topology changes during the constellation's movement.

2. Related Work

LEO satellite constellations are becoming essential for many space applications,but their security and operational integrity are seriously threatened by increasing concerns about orbital debris and even hostile space threats. There is an urgent need for a distributed and autonomous protection architecture because traditional on-orbit repair techniques meet significant obstacles. Zhao et al. [19] offered a comprehensive strategy to guarantee the robustness and self-healing properties of satellite constellations in a single orbit. The system allowed for adaptation to different types of attacks, both hostile and non-adversarial, through the use of individual satellite decision-making, ensuring ongoing coverage services. To find equilibrium constellation deployment, researchers presented a two-stage resilient coverage planning-control issue, developed models to evaluate coverage performance, and suggested a coverage game. An agent-based algorithm was created to calculate this equilibrium using a multi-waypoint Model Predictive Control (MPC) technique to achieve autonomous self-healing control. A thorough case study employing a standard constellation of LEO satellites further supports the research and offers empirical confirmation for the suggested methodology.

In order to expand typical ground information processes to include onboard space information duties, the Low Earth Orbit Remote Sensing (LRS) satellite network is essential to the advancement of space information applications. Nevertheless, the network faces major challenges because of its dynamic satellite topology, sporadic ISLs (inter-satellite communication links), and the requirement that mobile terminals move between satellite access sites. There are obstacles in this unique networking environment that need to be addressed in terrestrial Internet networks. Yang et al. [20] suggested a resilient networking architecture designed specifically for LRS Satellite Networks (LRSSNs) in order to handle these complications, emphasizing inherent mobility issues, robust transmission, and dynamic routing. They used lifetime-aware dynamic routing and path-quality guided routing to improve routing robustness in the face of changing network topologies. In order to guarantee reliable data delivery in spite of sporadic ISL connections, they relied on hop-by-hop data transmission. To lessen the effect of dynamic satellite access point switching on communication continuity, data caching techniques were also used. They used a semi-physical simulation platform to measure the achievable performance metrics of this suggested architecture in order to determine its efficacy.

Mega-constellations in Low Earth Orbit that combine commercial and government space technologies have become a key component of sixth-generation (6G) networks, providing ubiquitous services and worldwide coverage for a wide range of military and civilian uses. Despite these benefits, the potential synergy between space and terrestrial systems has not yet been fully utilized by the convergence of LEO-based satellite networking infrastructures. Shih-Chun et al. [21] suggested incorporating serverless edge learning architectures into conventional serverless cloud platforms in order to service better the 6G network's proliferating LEO (p-LEO) satellite ecosystems. From a networking standpoint, a novel distributed training approach is shown that effectively supports multi-agent deep reinforcement learning for service-level agreements by dynamically orchestrating communication and computing features across heterogeneous physical units.

Mega-constellations in Low Earth Orbit, which integrates government and private space technology, are becoming an essential part of sixth-generation (6G) networks, offering global coverage and ubiquitous services for a variety of military and civilian applications. The convergence of LEO-based satellite networking infrastructures has yet to completely capitalize on the potential synergy between space and terrestrial systems despite these advantages. Lei et al. [22] proposed integrating serverless edge learning architectures into traditional serverless cloud platforms to better service the expanding LEO (p-LEO) satellite ecosystems connected to the 6G network. Through dynamically managing communication and processing features across heterogeneous physical units, a unique distributed training approach is demonstrated from a networking perspective that effectively enables multi-agent deep reinforcement learning for service-level agreements.

The constellation of LEO satellites is a perfect addition to the network infrastructure of terrestrial systems. However, LEO satellite networks pose a significant technical barrier to reliable and efficient routing, especially given their

cyclical topology changes. Traditional methods, including using terrestrial routing protocols like OSPF, result in persistent route convergence and significant bandwidth usage on inter-satellite links. Previous studies have suggested snapshot-based routing techniques, which either need regular communication with ground stations or the storage of a series of routing table snapshots in a constrained amount of satellite memory. OPSPF, a revolutionary routing protocol designed especially for LEO satellite networks, is introduced by Pan et al. [23] by using periodic route calculations for instantaneous routing table creation, OPSPF takes advantage of the regularity of the constellation to manage the predicted topology changes.

OPSPF also introduced an on-demand dynamic routing technique to handle erratic topology changes brought on by connection failures or recoveries. The evaluation results showed that OPSPF achieved a large reduction of 57% in communication overhead and 82% in route convergence time during irregular topology changes, while OSPF caused zero route convergence overhead during regular topology changes. Their novel protocol presented a viable way to improve the robustness and effectiveness of routing in low-orbit satellite networks, maximizing their functionality in the face of fluctuating network conditions.

Compared to a single unmanned aerial vehicle (UAV), flying ad hoc networks (FANETs) provide improved mission efficiency and coverage, making them essential in a variety of military and commercial applications. However, because of the changing topology of the network, it is difficult to build an energy-efficient routing protocol for FANETs that minimizes latency while maintaining a high packet delivery rate (PDR). Cui et al. [24] introduced adaptive Q-learning (TARRAQ), a topology-aware resilient routing technique that allowed for distributed, autonomous routing decisions and efficiently captured topology changes with low overhead. In order to derive closed-form solutions for the neighbors' change inter-arrival time (NCIT) and neighbors' change rate (NCR) distribution, they first used queuing theory to examine the dynamic behavior of UAV nodes. They defined the predicted sensing delay of network events and built a resilient sensing interval based on real-time NCR and NCIT metrics.

Moreover, a low-cost adaptive Q-learning technique was presented to enable UAVs to make dispersed, autonomous, and adaptive routing decisions. The sensing interval provided low-cost, timely updates to the action space. The findings of the simulation confirmed that the topology dynamic analysis model is accurate and showed that TARRAQ is better than previous methods. TARRAQ achieves significant improvements over Q-learning-based topology-aware routing, mobility prediction-based virtual routing, and greedy perimeter stateless routing based on energy-efficient Hello. It achieved lower overhead by 25.23%, 20.24%, and 13.73%, higher PDR by 9.41%, 14.77%, and 16.70%, and lower energy consumption by 5.12%, 15.65%, and 11.31%, respectively.

3. Efficient Routing Strategies in Network Protocols

3.1. Memory-Efficient Static Routing

Network administrators can manually configure routing tables on routers using the simple static routing method. The paths that data packets are expected to follow are predefined in these tables. Optimizing these tables to use the fewest system resources is the main goal of memory-efficient static routing.

3.1.1. Overview of LEO Satellite Networks

The launch of Sputnik, the first artificial satellite in history, by the Soviet Union in 1957 marks the beginning of the narrative of low-Earth orbit satellites. In the present day, SpaceX's Starlink, OneWeb, and Amazon's Project Kuiper are the main players in the constellation of LEO satellites. Key advantages are provided by LEO satellites, which orbit at altitudes ranging from 160 to 2,000 kilometres. Because of are closeness to Earth, they have lower latency, which means that data travels more quickly. By deploying many LEO satellites, more ground may be covered with greater efficiency.

A constellation of low-Earth orbit (LEO) satellites is a group of satellites that orbit closer to the planet than other kinds of satellites, such as geostationary satellites. This proximity makes better signal strength and reduced latency possible for communication needs. Usually, the constellation is made up of several orbital planes, each housing a number of satellites. The architecture's ability to offer worldwide coverage makes communication services possible for a range of uses, including internet access, remote sensing, and telephones. The satellites in a typical LEO constellation travel in their orbital planes. This movement is depicted in Figure 1, which shows various orbits where satellites travel in either a northward or southward direction. For example, satellites in orbit 1 may be traveling northward, whereas those in orbit 6 may be traveling southward. Because of this configuration, as the satellites orbit the Earth, worldwide coverage is ensured.

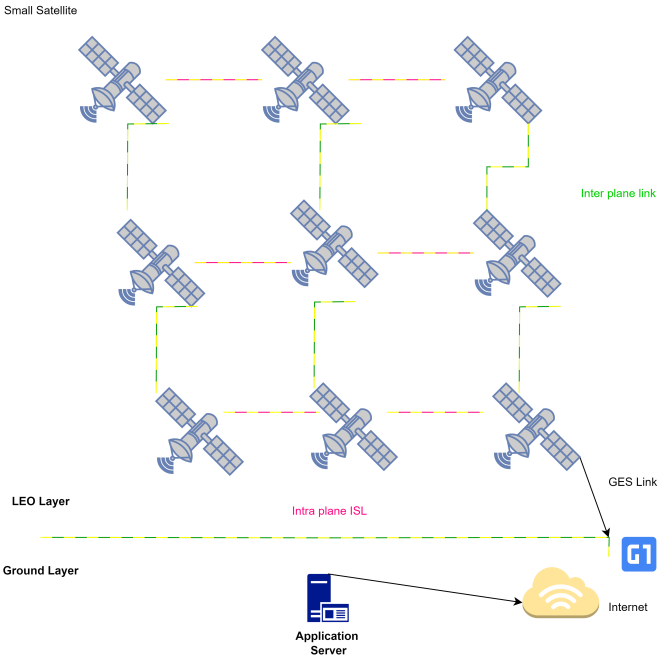

Figure 1. The constellation's architecture for low-Earth orbit satellites.

Interface of Satellite:

All of the satellites in the constellation have 4+1 network interfaces, which are used for various purposes inside the network:

- Ground Interface: It points in the direction of the surface of the Earth. This interface is essential for data transmission to and from terrestrial users, facilitating communication between the satellite and ground stations.
- Intra-Plane Inter-Satellite Links (ISLs): two interfaces are set aside specifically for establishing connections with nearby satellites in the same orbital plane. These intra-plane ISLs (e.g., from S52 to S53) facilitate inter-satellite communication and allow for effective coordination and data transfer.
- Inter-Plane Inter-Satellite Links (ISLs): Links to satellites in adjacent orbital planes are made via the other two interfaces. For example, these inter-plane ISLs from S53 to S63 are crucial to preserving network connectivity throughout the constellation. They allow satellites in several orbital planes to exchange data, providing seamless communication coverage.

Connectivity Challenges:

Although the intra-plane ISLs stay connected during the satellite's mobility inside its orbital plane, problems occur when the satellite approaches the polar regions. Because of antenna tracking restrictions, once a satellite reaches a high-latitude location, the inter-plane ISL towards a satellite in a higher-latitude orbit may become disconnected. The Earth's curvature brings about this restriction and the satellite's orientation, which in some cases can prevent direct line-of-sight communication between satellites.

A LEO satellite constellation's architecture is carefully planned to provide effective communication and worldwide coverage. We are able to comprehend the intricacies of these sophisticated networks by studying the motion of satellites inside orbital planes, the use of intra- and inter-plane ISLs, and the difficulties presented by high-latitude areas. This review provides a starting point for investigating routing protocol design specific to LEO satellite constellations.

3.1.2. Periodic Routing Table Calculation

One important component of computer network routing protocols is the computation of periodic routing tables. Routers employ routing tables to figure out the best way to forward data packets to their destination. Routing tables are recalculated on a regular basis to guarantee that routers are capable of making effective routing decisions and have the most recent topology information.

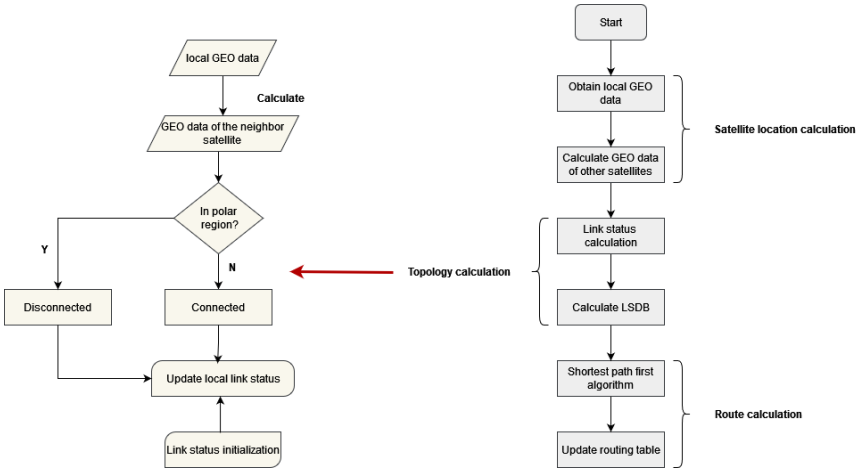

Figure 2. The process of the static routing technique that uses less memory.

The suggested method of routing in a constellation of low-Earth orbit satellites seeks to reduce the amount of storage required on each satellite by computing routing tables on a regular basis rather than maintaining a large number of snapshots. Because of these satellites' low memory, this is essential.The plan is to profit from LEO satellite movements' predictable nature. Because the orbits of these satellites are well-known and consistent, we can use the location of one local satellite to determine the positions of every other satellite in the constellation. Thanks to this estimation, every satellite may determine its position with respect to other satellites. Furthermore, we are able to ascertain the instantaneous network architecture of the entire constellation by distinguishing between the satellites that are and are not in the polar regions. The routing table is then created using this information as a foundation. As illustrated in Figure 2, the procedure begins with determining the satellite positions based on the location of the local satellite. The location of each satellite is estimated in this step. The system compares each satellite's position with the border of the polar zone after obtaining this global view of all satellite positions. Lastly, the system creates the routing table by using the Shortest Path First (SPF) algorithm. The optimal routes between each satellite are shown in this table according to the present network structure. This method makes use of the regularity of satellite motions to enable memory-efficient routing. Frequent recalculation of routing tables—for example, once per second—allows the system to adjust to the dynamic topological changes inside the constellation. In the LEO constellation, this ensures effective communication between satellites.

3.1.3. Orbit Prediction through Mathematics

Determining the positions of satellites inside a constellation requires the use of the satellite location calculation technique, as shown in Figure 3. This approach is very helpful in situations where we have Geo data from a nearby satellite $T(a, b)$, which is frequently obtained via GPS or other similar systems. The local satellite's Geo data, $T(a, b)$, is what we start with. The next step in the process is to compute the Geo data, represented as $T(a, b)$, for satellites in the same orbit. 'j' stands for the changing index for satellites in the same orbit as the local satellite, and this is done within the inner loop. Starting from $T(a + 1, b)$, the algorithm proceeds to the outer loop where it computes the Geo data of satellites in the adjacent orbit. Within the constellation of satellites, the index 'a' denotes the various orbits. Making sure that we compute the Geo data for every satellite in the constellation, the outer loop travels through all of the orbits.

Sign and Range Limitations:

- Latitude (b) is specified to the range of [-90, 90] degrees.
- Longitude (a) is limited within [-180, 180] degrees.

These limitations are essential to ensure that the derived results remain within the correct geographical ranges.

The order in which the calculations were made:

Ensuring correct positioning in satellite location calculations requires careful handling of indications and ranges. A satellite's position north or south of the equator is determined by its latitude (b), where positive values indicate places northward and negative values indicate positions southward. In contrast, a satellite's east-west location with respect to the prime meridian is defined by its longitude (a), where positive values denote eastward positions and negative values, westward ones. By following these guidelines, we protect the integrity of satellite positioning data, which is essential for accurate scientific research coordination, dependable GPS navigation, and efficient commu-

nication services. Satellite systems and their operations, which mostly rely on precise spatial information, may be affected by location data mistakes resulting from improper management of signs and ranges. Therefore, paying close attention to these elements is essential to maintaining the performance and dependability of satellite navigation and communication systems globally. Finding a satellite's position inside a constellation can be done methodically with the help of the satellite location computation procedure. It makes sure that the satellite orbits and their geographic locations are taken into account within reasonable bounds. We can compute and maintain satellite positions with accuracy and dependability by following these principles, which are necessary for many applications like GPS navigation, communication services, and scientific research.

Variables

M - number of orbits

N - number of satellites in each orbit

T(a,b) - Local satellite, where a is the orbit ID, and b is the satellite ID in that orbit

T(a,b).cj - longitude of T(a,b)

T(a,b).cw - latitude of T(a,b)

c0 - latitude difference between 2 neighbor satellites in the same orbit

c1 - latitude difference between 2 neighbor satellites in neighbor orbits

c2 - longitude difference between 2 neighbor orbits

d - direction indicator, move towards the north,d=1; move towards south,d=-1

flag - initialized to 0

Algorithm

```
for(i=a;;i=(i+1)%M)
if(flag==1)break;
if(i!=a)
T(i,b).cj=(T((i+M-1)%M,b).cj+c2+180)%360-180;//T(i,b).cj is in [-180,180]
T(i,b).cw=S((i+M-1)%M,b).cw+c1*d;
if(T(i,b).cw>90)T(i,j).cw=180-T(i,j).cw;
if(T(i,b).cw<-90)S(i,j).cw=-180-T(i,j).cw;
}
for (j=b;;j=(j+1)%N)
if(i==a&&j==b){
if(flag==0){
flag=1
continue;
}
else break;
}
T(i,j).cj=T(i,(j+N-1)%N).cj;
T(i,j).cw=T(i,(j+N-1)%N).cw+c0*d;
if(T(i,j).cw>90)T(i,j).cw=180-T(i,j).cw:
if(T(i,j).cw<-90)T(i,j).cw=-180-T(i,j).cw;
}
}
```

Figure 3. Pseudocode to calculate the Geo data.

3.2. On-Demand Dynamic Routing

Depending on the state of the network at the time and the destination of each packet, routing decisions are made dynamically and packet-by-packet using a technique known as on-demand dynamic routing. This is as opposed to conventional static routing, in which routes are preset and stay that way unless they are manually altered.Routers in an on-demand dynamic routing environment use real-time data, such as network congestion, link quality, and available

bandwidth, to determine the optimal path for each packet to take in order to reach its destination. This makes it possible to use network resources more effectively and to adjust to changes in the network topology more effectively.

3.2.1. Analyzing Design Space

In the context of On-Demand Dynamic Routing, Design Space Analysis entails looking at and assessing different routing algorithm configurations and design options in a network. Understanding the trade-offs between various strategies and choosing the best design for a particular situation depend on this analysis.

Optimized Static and OSPF routing (OPSPF) is a creative combination of two routing techniques designed to improve network resilience and efficiency. The problem it attempts to solve is the shortcomings of conventional techniques: memory-efficient static routing works well in most cases, but it breaks down in the rare event of a sudden link failure. However, while standard OSPF dynamic routing manages these unforeseen events quite well, it causes a network-wide traffic jam during regular topology changes, which is especially problematic in satellite networks. OPSPF is outstanding because it can balance responsiveness and efficiency by using OSPF for irregular events with static routing for regular changes. In the process of designing OPSPF, two important factors are highlighted. The way that OSPF reacts to changes asynchronously and static routing, which runs on a timer, initiates route computations is different. This problem requires a careful solution, requiring route calculation triggers that can handle both unexpected and regular events. Second, attention needs to be paid to the possible conflict that could result from OSPF and static routing creating distinct routing tables. In order to address this issue, OSPF sets up a hierarchy. Traffic first follows the static routing table, but OSPF updates the static module when there are sporadic changes, guaranteeing that the most accurate and up-to-date routes are taken. Essentially, OPSPF works by letting the static routing module manage recurring topology changes by doing periodic route computations. The OSPF module simultaneously monitors the network closely, prepared to identify and handle any unforeseen interruptions. When OSPF notices something strange, it gets in touch with the static routing module quickly so that the network gets the most recent routing data as soon as possible. With this cooperative effort, an optimal network is produced where the responsiveness of OSPF dynamically adjusts to unforeseen events, but the efficiency of static routing is maintained for recurring adjustments. In the end, OSPF dynamic routing and static routing combine to form a complex but user-friendly system that builds a robust and effective network architecture.

3.2.2. Dynamic Routing Integration with the Static Framework

Different paradigms in networking are represented by dynamic routing and static frameworks. While dynamic routing protocols enable routers to interact with one another to modify routing tables based on network conditions dynamically, static frameworks rely on predetermined routes that are configured and set manually. Although it may initially appear counterintuitive, there are situations in which integrating dynamic routing into a static architecture can be helpful.

Six modules that make up the architecture of OPSPF's hybrid route calculation and link detection, as shown in Figure 4: local interface flag, global interface flag, LSDB & routing table, satellite geographic data collecting, static routing, and dynamic routing. Every one of these modules is essential to the functioning of the system.

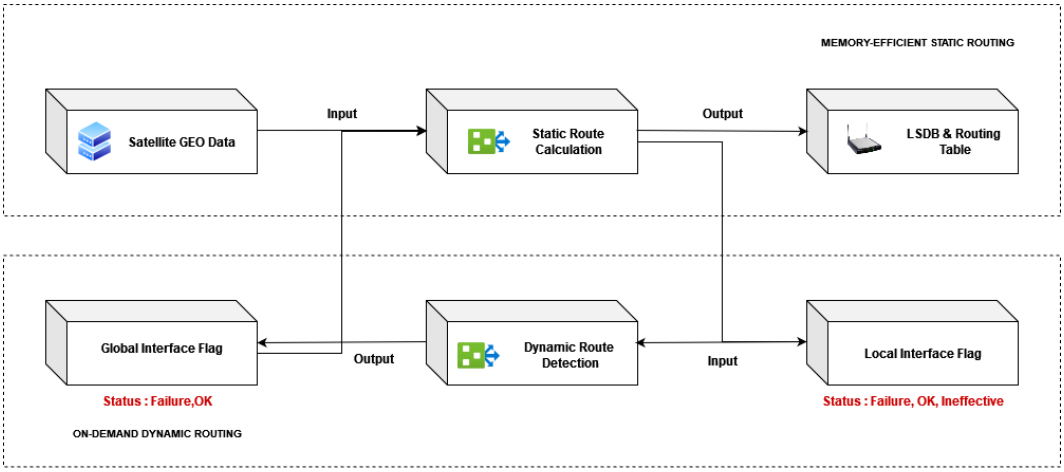

Figure 4. The hybrid architecture of link detection and route calculation in OPSPF.

Static Routing Module Periodically calculating static routes, this module serves as the system's backbone. This computation produces the final routing table that is utilized for data plane packet forwarding, as well as the vital Link State Database (LSDB). It also refreshes each network interface's local interface flag, which provides crucial status data.

- Three different statuses are carried by the local interface flag: "Failure," "OK," and "Ineffective."
- "Failure" indicates that an interface failure has occurred, but the satellite is outside the polar area.
- "OK" signifies that the satellite has left the polar region and that it is still possible to use the interface.
- "Ineffective" indicates that the satellite is in the polar area regardless of whether the interface is in operation.

Dynamic Routing Module Interchange

An essential input for the functions of the dynamic routing module is the local interface flag. Only after the local interface flag has an "OK" status can link detection, which is made possible by Hello packets, begin. Thus, in the event that a satellite moves into the polar zone, connection detection procedures between adjacent satellites do not take place, even in the event that an interface fails. On the other hand, any such malfunctions can be identified later on when the satellite leaves the polar region and the local flag changes to "Failure."

Dynamic Route Flooding

This procedure, which is triggered by various events like link failure, link recovery, and link state updates, is essential for updating routing data throughout the satellite network. The parts that follow go into further detail on these occurrences and their effects. The local interface flag is updated to the global interface flag by the dynamic routing module based on route flooding information with just two statuses—"Failure" and "OK"—the latter is used as one of the inputs by the static routing module.

Geographic Data

The geographic data, which provides the exact latitude and longitude coordinates of each local satellite, is an extra input for the static routing module. The local interface flag is updated as satellites pass through their orbits with the use of this data, which also ensures precise route computations.

Static Routing supplication

It is possible to invoke the static routing module synchronously, particularly if the invocation cycle is brief. On the other hand, the dynamic routing module might call the static routing module asynchronously in situations when there are erratic topology changes. The goal of this approach is to guarantee effective network change adaption while reducing route convergence time.

The architecture effectively isolates regular topology changes, preventing any negative effects on the dynamic routing module. Rather, only the static routing module has the authority to handle such modifications. Outside of the polar region, any abnormal changes are quickly detected by the dynamic routing module. This capacity improves the overall robustness of the system by having a major impact on the final routing decisions.

The OPSPF architecture is painstakingly constructed to achieve the best possible performance in satellite communication. By effectively managing topology changes and precisely initiating dynamic routing as necessary, network dependability is improved, and disruptions are reduced. Route convergence time (Tc) in the worst-case situation is computed as Tc = Td + Tp + Ts, where Td is the maximum Keepalive timer for Hello detection, Tp is the longest packet travel time between two satellites, and Ts is the shortest time the static routing algorithm takes to execute.

3.2.3. Asynchronous Event Handling

Ensuring an accurate and efficient routing protocol in OPSPF requires careful handling of asynchronous events, including connection failure, link recovery, and link status updates. Several methods are used to detect these occurrences. For example, link failure can be identified by keeping an eye out for Hello Protocol timeouts, a lack of neighboring Link-State Advertisements (LSAs), or a failure to receive keep-alive signals. When the router detects a link failure, it invalidates the related network and the impacted connection. Following this, LSAs are sent out to notify other routers of the change, and the routing database is updated to reflect the modified network topology. Proactively addressing network dynamics and preserving the best possible routing paths for data transmission means that the routing system will quickly adjust. To ensure the robustness and reliability of the OPSPF protocol, similar vigilante and responsive methods are used to detect and manage link recovery and link state adjustments.

In OPSPF, we need to address and handle three asynchronous events: link failure, link recovery, and link state update.

1. Link failure detection and handling: Maintaining a robust and effective routing protocol in OPSPF (Open Shortest Path First) requires the identification and management of connection failures. The topology of the network is disturbed by a link failure, which may affect the routes that data takes to get to its destination. When an interface breakdown occurs, a systematic process in the operational workflow of a satellite network interface within the context of link failure management finds the breakdown, fixes it, and converges the network routing. Every satellite interface diligently sends out Hello packets on a regular basis while simultaneously lowering its Keepalive Counter during each transmission. The Keepalive Counter ensures that active connections are continuously acknowledged by immediately resetting to their maximum value upon receiving a Hello packet from an adjacent interface.Nevertheless, if a neighbor interface's Keepalive Counter falls to 0, indicating an extended lack of communication, the network will identify it as a "Failure." This sets off a series of critical reactions. Initially, the neighbor interface that is impacted is quickly updated to indicate its state of failure. Next, in order to recalculate the most effective paths amid the topological changes, the network runs the Shortest Path First (SPF) method, such as Dijkstra's algorithm [25].This new data is quickly added to the local Link State Database (LSDB) so that the router always has a current representation of the network topology. The router wraps the modifications by simultaneously encapsulating this updated routing intelligence into Link State Update (LSU) packets. Subsequently, the LSU packets are distributed throughout the network, inundating the satellite constellation with the latest routing information.This coordinated process ensures that the satellite network dynamically adjusts its routing paths following a link failure to maintain the best possible flow of data transfer. The reliable and efficient operation of modern satellite communications systems is ensured by this strong link failure management strategy, which is embodied in the systematic handling of satellite network interfaces. Figure 5 describes the procedure of link failure detection and handling.

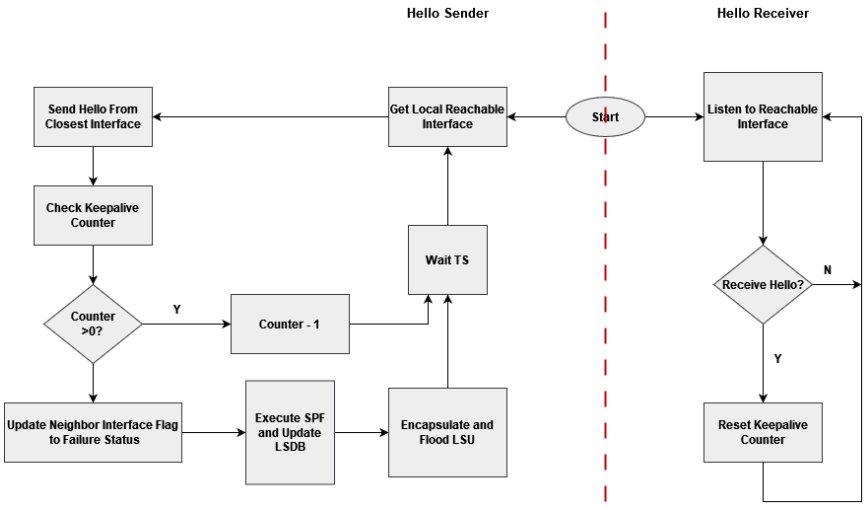

Figure 5. Graphical representation of Link failure detection and handling.

2. Link recovery detection and handling: When it comes to satellite network interfaces, rapid detection and efficient handling of link recovery are critical to the seamless functioning of the network and the transfer of data. The systematic process for link recovery detection and subsequent actions aims to restore connectivity and optimize routing patterns. Satellite interfaces maintain active communication channels by exchanging Hello packets continuously during network operation. This persistent communication persists even in the face of link failures, offering a basis for identifying the restoration of a previously damaged link. When Hello packets are received from a neighbor interface that has been recovered, the Keepalive Counter is immediately reset to its maximum value. This action indicates that connectivity has been restored and that the interface is functional. As a result, the interface keeps an eye on the Keepalive Counters and interprets any increase from zero to a positive value as a sign that the neighbor link has been re-established. When a recovered neighbor interface is detected, the interface modifies the link's state from "Failure" to "Active" or "Operational." The restoration of functional connectivity and preparedness for data transmission is shown in this status change. After the network verifies that the link has recovered, the Shortest Path First (SPF) procedure is started. In order to guarantee effective data transfer throughout the network, the SPF algorithm recalculates

the best routing routes in light of the restored link. The router can retain an accurate representation of the network topology thanks to the smooth integration of this updated routing information into the local Link State Database (LSDB). The router creates connection State Update (LSU) packets to broadcast the repaired connection information throughout the network. The satellite network is inundated with these LSU packets, which include the most recent routing data. Because these LSU packets are broadcast, every router on the network gets the updates it needs to have a synchronized and efficient routing setup. The systematic process of identifying and managing link recovery ensures that connectivity will be restored quickly, data will flow without interruption, and the best possible routing patterns will be kept within the satellite network architecture. This methodical technique embodies the efficiency and dependability required by contemporary satellite communications systems. Figure 6 outlines the process as mentioned above.

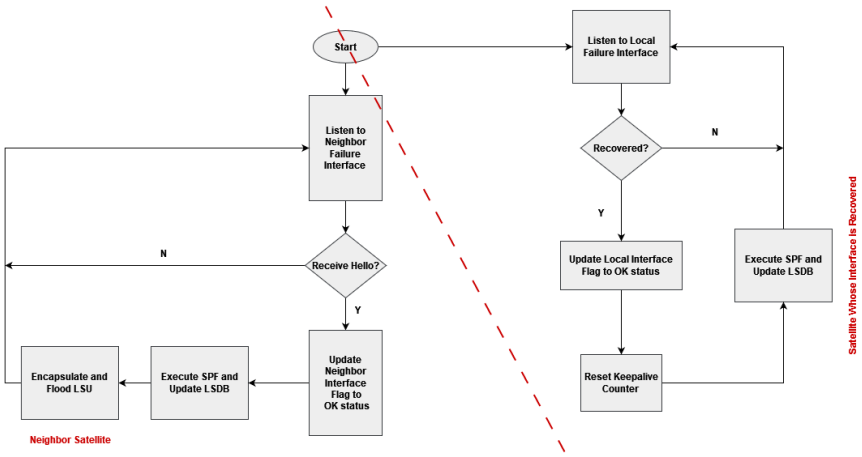

Figure 6. Graphical representation of Link recovery detection and handling.

3. Link state update handling: Link State Update (LSU) management is a critical procedure in satellite network interfaces that ensures the network's distribution of current routing data. LSUs are the channels through which routers inform all other routers in the network about changes in the network topology, such as link failures, recoveries, or newly found links. The LSU handling process is designed to ensure that these updates are distributed throughout the satellite network in an efficient and trustworthy. Routers quickly provide LSUs with relevant data when modifications take place. This data includes information about the modified links, their associated expenses, and the affected network segments. The most recent network status is carefully reflected in these LSUs, ensuring accuracy and precision in routing decisions. All routers in the satellite constellation receive the revised routing information when the LSUs are subsequently flooded into the network. Regardless of where a router is located inside the network, this flooding technique ensures that it gets the updates it needs. Routers make educated decisions about the best path selection when LSUs move throughout the network by comparing the information they receive with the routing tables they already have. Routers update their local Link State Databases (LSDBs) in accordance with the LSUs they receive. This phase makes sure that every router has a synchronized, current view of the topology of the network. Routers may efficiently determine the shortest paths to different destinations, maximizing the efficiency of data transmission by maintaining consistent LSDBs throughout the network. Network convergence, in which every router in the satellite constellation converges to a uniform and updated representation of the network topology, is also guaranteed by the flooding of LSUs. Seamless communication between network nodes, packet loss reduction, and network stability all depend on this convergence. In order to preserve network dependability, flexibility, and efficiency, Link State Updates are handled carefully in satellite network interfaces. Satellite networks are able to reliably guarantee optimal data transmission channels and strong network performance by quickly disseminating updated routing information through LSUs and maintaining network-wide synchronization. The above procedure is shown in Figure 7.

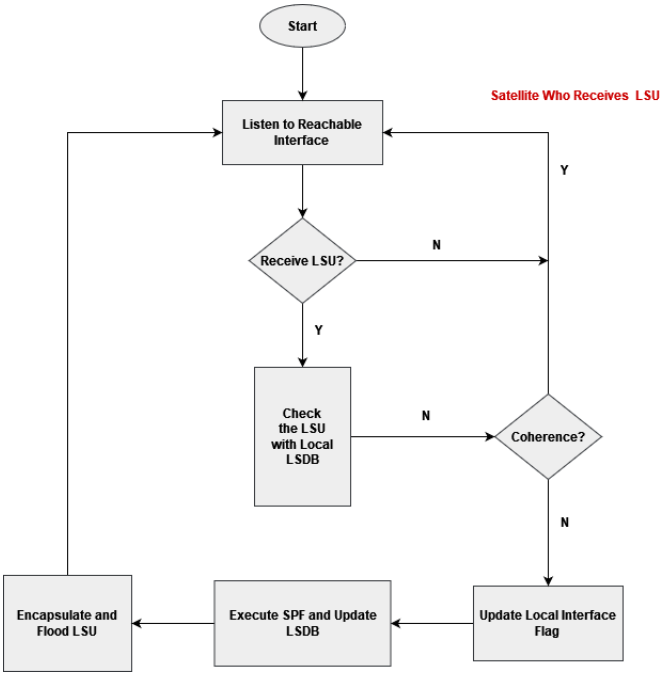

Figure 7. Graphical representation of Link state update handling.

3.2.4. The Case Study Approach

In the vastness of space, satellite-to-satellite communication is essential to maintaining smooth operations. These vital links, however, can be broken by unforeseen circumstances such as network interface malfunctions. In this case study, we investigate how, inside a constellation of satellites, our adaptive routing protocol handles such obstacles. The way in which our routing system responds to an unexpected failure of a router interface is shown in Figure 8. Now, let's look at satellite E, which is situated outside of the polar area. Satellite A, its nearby satellite, will detect a network failure at its interface E(0) using a Hello timeout technique. As soon as satellite A notices a problem, it encapsulates Link State Update (LSU) packets and sends them over the network. Until every satellite in the constellation updates its routing table to include the new, corrected paths, this process is ongoing. As a result, our system does not react right away if satellite E is in the polar zone when the interface failure occurs. Rather, it bides its time until satellite E departs the arc. Following that, normal operations for failure detection and link state updating are resumed. The time it takes for all of the routes in a constellation to converge is greatly shortened by this dynamic routing on-demand method, which also lowers communication overhead during frequent topology changes in the constellation. Only when abnormal changes are noticed outside the polar zone do LSU packets flood the entire network. This guarantees that unforeseen events are effectively managed and adapted to by our routing system without generating needless disruptions.

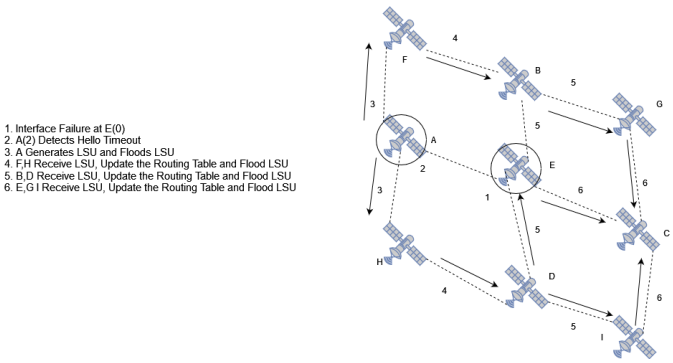

Figure 8. An analysis of the route convergence following a failure of the router interface.

Our adaptive routing system is a valuable tool for maintaining the dependability and effectiveness of satellite constellations. In the event of unforeseen circumstances, such as router interface failures, it reacts quickly to maintain network connectivity. Its efficacy in the dynamic world of space-based communications is demonstrated by the way

the protocol minimizes disruptions and maintains connectivity through adaptive adaptation and controlled floods of LSU packets.

3.3. Analysis of Topology Dynamics Predicted on Queuing Theory

The routing protocol's dynamic topology adaptation is primarily demonstrated by its prompt detection of network fragmentation and its ability to modify routing choices in response to topology changes, which are influenced by factors such as node density, mobility model, and transmission range. Therefore, it is essential to investigate the features of mobility behavior, such as the rate and the inter-arrival time distribution of neighbor's arrival, departure, and change, as the main task of implementing a topology-aware routing protocol for FANETs.Because of the topology changes, communication links between UAVs will be formed and broken frequently. The process by which other nodes enter node Sc's communication range Ω_c, maintain a link with Sc, and exit Ω_c is considered a queuing service process, as Figure 9 illustrates. Every node is separated into two groups: the nodes inside Ω_c and the nodes outside of it, represented by the collections $M1$ and $M2$, respectively. The nodes in $M1$ will join $M2$ as soon as their links established with Sc are broken because UAV moves according to the 3D RWP mobility model; in other words, they will be considered the customer source once the service is over.

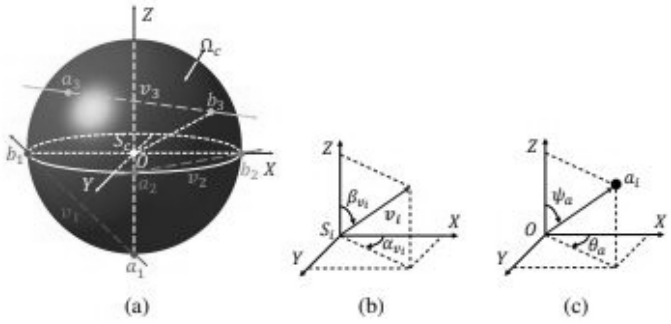

Figure 9. (Ω_c) of the UAV S_c, depicting their entry and exit points. In subfigure (a), nodes with relative velocities v_1, v_2, and v_3 enter Ω_c at locations a_1, a_2, and a_3 and exit from points b_1, b_2, and b_3, following distinct trajectories influenced by their velocities. Subfigure (b) highlights the spatial positioning of these entry points by considering their elevation and azimuth angles, offering a three-dimensional perspective on the nodes' movement. Finally, subfigure (c) focuses on the dynamic nature of these interactions, illustrating how the relative velocities of the nodes impact their duration within the communication range and their eventual exit points.

Therefore, the procedure mentioned above is basically a cyclic queuing system. As soon as the nodes enter Ω_c, they can immediately form links with Sc. This implies that customers can be served instantly and without having to wait, so the number of servers and system capacity can be considered infinite. Given the assumption that the CQS is not empty, meaning that at least one UAV is present in $M1$, then the CQS's service duration is equal to LD. Let's specifically look at the scenario where there are two nodes in 3-D space, Sc and So, and So is an ordinary node that has been served by or will be served by the central node Sc. $v_c = v_{ck}$ and $v_o = v_o \sin\beta\cos\alpha\mathbf{v_o}i + v_o \sin\beta\sin\alpha\mathbf{v_o}j + v_o\cos\beta\mathbf{v_o}k$ are the respective definitions of Sc and So. Their relative velocities, v and v, can be obtained by

$$v = v_o - v_c = v_o \sin\beta\cos\alpha\mathbf{v_o}i + v_o \sin\beta\sin\alpha\mathbf{v_o}j + (v_o\cos\beta\mathbf{v_o} - v_c)k \tag{1}$$

$$v = |v| = \sqrt{v_o^2 + v_c^2 - 2v_c v_o\cos\beta} \tag{2}$$

As can be seen in Figure 9(b), the angle β_{vk} is the angle between v_k and the Z-axis, and the angle α_{vk} is the angle between the X-axis and the horizontal component of v_k. Based on the relative motion of Sc and So, the analysis procedure will be greatly simplified; that is, Sc is considered a fixed node when So enters Ω_c with relative velocity v, the direction of which can be determined as $\alpha_v = \alpha_{vo}$.

$$v = \arccos\left(\frac{v_o\cos\beta v_o - v_c}{\sqrt{v_o^2 + v_c^2 - 2v_c v_o\cos\beta}}\right) \tag{3}$$

Assuming that So enters Ω_c from $a3$ and exits Ω_c from $b3$ without varying the velocity, as illustrated in Figure 9(a), we obtain $oa3 = R(\cos\theta_{a3}\sin\psi_{a3}\mathbf{i} + \sin\theta_{a3}\sin\psi_{a3}\mathbf{j} + \cos\psi_{a3}\mathbf{k})$, and $a3b = |a3b|(\cos\alpha_v\sin\beta_v\mathbf{j} + \cos\beta_v\mathbf{k})$, where θ_{a3} and ψ_{a3} .

We have $d_{\text{link}} = |a3b| = 2R|\sin\psi_{a3}\sin\beta_v\cos(\theta_{a3} - \alpha_v) + \cos\psi_{a3}\cos\beta_v|$ since $|ob| = R = |oa3 + a3b|$.

The entire LD can be computed using.

$$T_{\text{wl}} = \frac{2R}{v}|\sin\psi_{a3}\sin\beta_v\cos(\theta_{a3} - \alpha_v) + \cos\psi_{a3}\cos\beta_v| \tag{4}$$

Q-learning Framework

It should be noted that routing involves making numerous single-hop decisions in order to choose the best path for transferring packets from the source to the destination. The fundamental idea of Q-learning is precisely this: finding the optimal action-selection strategy for a finite Markov decision process even in the absence of prior knowledge about how actions affect the environment. It involves the probability of choosing the best relay from finite neighbors despite the lack of information about the entire network. As a result, the Markov decision process can be used to model the routing process, and the Q-learning algorithm can solve it in the following way. The entire network can be viewed as the environment by designating the packet as an agent, and the state's space is made up of all UAVs that forward packets. The available neighbors are represented in the action space since choosing the subsequent hop is viewed as an action. Therefore, UAVs' choice regarding packet forwarding is equal to the state transition. When an agent acts from state st in Q-learning, they will receive the reward R(st, at), and the Q-table is updated by

$$Q_{t+1}(s_t, a_t) \leftarrow (1-\alpha)Q_t(s_t, a_t) + \alpha\left[R(s_t, a_t) + \gamma \max_{a' \in A_{t+1}} Q_t(s_{t+1}, a') - Q_t(s_t, a_t)\right] \tag{5}$$

In this case, the Q-value at time $t+1$ is Q_{t+1}, the Q-value at state s_t is $Q_t(s_t, a_t)$, the learning rate is $\alpha \in (0, 1)$, and the discount factor is $\gamma \in (0, 1)$. The state of the agent will change from s_t to s_{t+1} after the action a_t is completed. The maximum Q-value of all possible actions $a' \in A_{t+1}$, where A_{t+1} is the action space in state s_{t+1}, is $\max_{a' \in A_{t+1}} Q_t(s_{t+1}, a')$. The information contained in Hello messages can be used to define and calculate the reward $R(s_t, a_t)$, which aims at various QoS metrics. By routinely exchanging status updates with neighbors, all UAVs will permanently maintain the neighbor table and Q-table. While the latter is used to identify the optimal course of action, the former is used to update the action space and compute the reward. According to Figure 10, an agent will act from the action space when the traffic packet is generated or received at the UAV node (such as UAV S5). In this case, S5 will choose a relay from neighbors S2, S4, and S6. After that, the reward can be determined using QoS requirements. The Q-table will, therefore, be updated continuously via (5) as the agent picks up knowledge from its surroundings. The Q-learning-based scheme is thought to be the best choice in our motivation scenario, even though other advanced RL techniques can be used, such as Deep Q Network (DQN) [26] and Deep Deterministic Policy Gradient (DDPG) [27], to solve the routing problem. Because a global view of the network is not feasible, each UAV performs the aforementioned process in a distributed manner, meaning that each node only stores and retrieves a specific row of the Q-table. TARRAQ is no longer hindered by the slow convergence that results from an unmanageably large Q-table. Even in a dense network, each state's action space will be extremely small because of its small number of neighbors.

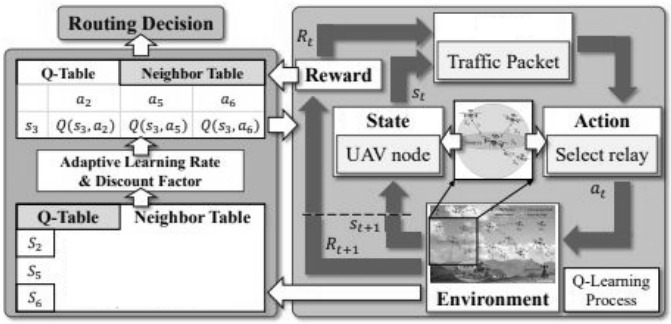

Figure 10. Q-learning Structure for Routing FANETs.

4. Result and Discussion

Our goal is to assess the memory-efficient static and on-demand dynamic routing in the proposed OPSPF. To do this, we construct a LEO-48 satellite network using an emulator environment for networks. There are eight satellites in each of the six polar orbits that make up the LEO-48 satellite constellation. An orbit around the earth takes a satellite about 6,900 seconds. For regular topology changes, the boundary of the polar region is set at ±70◦. Table I contains a list of the satellite constellation's precise parameter settings. In order to simulate the satellite network, the satellite trajectory for a given period is first generated using the above parameter setting in STK 9.2.1, a satellite constellation modeling environment. This data is then fed into the Exata.To create end-to-end packet flows in Exata, we pair up CBRs (constant bit rate). Every end host has its packet size set to 512 bytes and its packet transmission rate set to 100 bps. A total of 4,897 lines of C code make up the OPSPF implementation (3853 for the foundation and static routing and 1,044 for the dynamic routing). A desktop computer equipped with a quad-core Intel Core i7-6700 processor and 16GB DDR4 memory runs the emulator.

Simulation

Traffic bursts during route convergence:

- **OPSPF versus OSPF in a single topology shift:**The quantity of packets generated during route convergence, which is brought about by a single change in constellation topology, is displayed in Figure 11. Either irregular network failure/recovery or regular satellite motion across the polar region (±70◦) can cause the topology change. When the Hello packet identifies a topology change in OSPF, the updated topological data is broadcast throughout the network until every router has the same topological view. LSU packets contain the updated topological information, which LSAck packets will also acknowledge. One (regular or irregular) topology change causes OSPF to generate 235 LSU/LSAck packets during route convergence, as shown in Figure 11. On the other hand, OPSPF experiences no communication overhead during route convergence when topology changes on a regular basis because each router independently calculates the updated routing table instead of the network as a whole converging via distributed negotiation. To achieve global route convergence in the event of an irregular topology change, OPSPF has transmitted 101 negotiation packets. We currently use a very basic route convergence model that is devoid of the LSU acknowledgement and retransmission mechanism in the event of a timeout. Although this implementation is less dependable than OSPF, it sends fewer negotiation packets when the topology changes irregularly. Observe that even though the future reliable OPSPF and OSPF will experience comparable communication overhead during an irregular topology change, OPSPF still performs better than OSPF because these types of irregular topology change cases are far less common in real-world deployments.
- **OSPF vs. OPSPF when there are frequent topology changes:** Figure 12 depicts the number of packets generated during route convergence—which is caused by regular changes in the constellation topology—during a single satellite motion time interval—roughly 6900 seconds. The packet number is counted every 2 seconds. The packets counted are LSU, LSAck, and Hello. Figure 12 for OPSPF shows a consistent traffic burst rate of about 300 (Hello) packets every two seconds. Since there are 75 connected ISLs (or 150 connected router interfaces) on average in our experimental setup and the hello packet sending rate is 1 packet per second, the total number of hello packets in two seconds is 70 * 2 * 2 = 280.
- **OSPF vs. OPSPF when there are erratic topology changes:** The quantity of packets generated during route convergence, which is brought on by erratic changes in the constellation topology, is depicted in Figure 13. We carefully choose a time interval of 200s between two consecutive regular topology changes in order to hide the route convergence brought on by the changes in topology. Every five seconds, an irregular topology change is scheduled to happen, with one network failure and one network recovery in between. Both OSPF and OPSPF will transmit negotiation packets for route convergence in Figure 13. In contrast to OSPF, our less dependable implementation of OPSPF has less communication overhead because it lacks the mechanism for LSU acknowledgment and retransmission following a timeout.

Route convergence time:

- **OPSPF versus OSPF in a single topology shift:** The route convergence time resulting from one (regular or irregular) change in the constellation topology is displayed in Figure 14. The time interval between the first and last detected LSU, or LSAck, is used to calculate the route convergence time. The OSPF,

OPSPF (regular), and OPSPF (irregular) route convergence times in Fig. 12 are 543, 0, and 100 ms, respectively. The tiny route calculation time is actually the route convergence time for OPSPF (regular). Because OSPF reliably recognizes LSUs and retransmits after a timeout, its route convergence time is longer than that of OPSPF (irregular).

- **OSPF vs. OPSPF when there are frequent topology changes:** Figure 15 depicts the route convergence time resulting from regular constellation topology changes during a single satellite motion period (roughly 6900s). OPSPF appears to perform better than OSPF because of its zero route convergence time. This kind of frequent topology shift is also fairly typical in a constellation of LEO satellites.
- **OSPF vs. OPSPF when there are erratic topology changes:** Figure 16 depicts the route convergence time as a result of irregular constellation topology changes in the 200s. The topology is changed every five seconds, and there is one network failure and one network recovery. When comparing OSPF's route convergence time to the unstable OPSPF implementation, OSPF performs better.
- **Effect of the periodic route calculation interval:** The effect of the route calculation interval on the accuracy of the satellite's routing table is examined in Figure 17. Due to the independent calculation of the routing table in each satellite under OPSPF, a large route calculation interval will result in a delayed routing table update and significant packet losses. Furthermore, because of the discrete nature of the constellation topology change, the packet loss rate increases nonlinearly with the route calculation interval. The route calculation interval in OPSPF is initially set to 1 second.
- **Dynamics of the hop number and packet latency:** Figure 18 depicts the dynamics of the ISL hop number and end-to-end packet latency during the satellite constellation's motion between two fixed-end hosts. It can be concluded that (1) the packet latency typically varies synchronously with the ISL hop number, and (2) the packet latency can vary even for the same ISL hop number due to occasionally changing ISL lengths or routing paths.

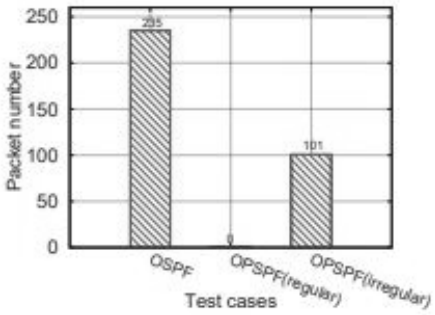

Figure 11. The number of packets generated during route convergence that is caused by a single change in constellation topology is examined in three scenarios: OSPF, OPSPF (regular), and OPSPF (irregular).

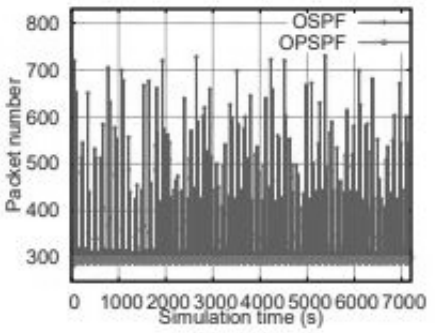

Figure 12. The number of packets generated in a single time interval of satellite motion (6900s) during route convergence caused by periodic changes in the constellation topology.

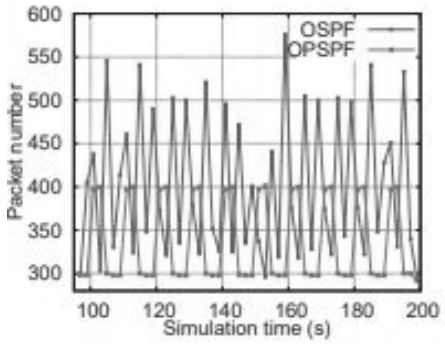

Figure 13. The number of packets generated during route convergence due to irregular changes in the convergence topology; the convergence topology changes every 5 seconds.

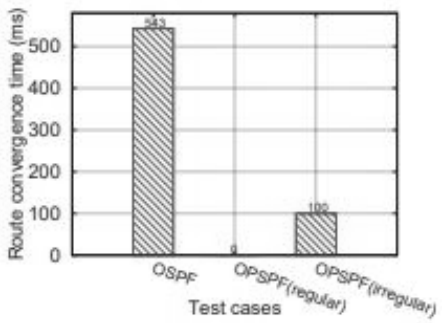

Figure 14. One constellation topology change causes the time consumption of route convergence to increase; three scenarios—OPSPF, OPSPF (regular), and OPSPF (irregular)—are examined.

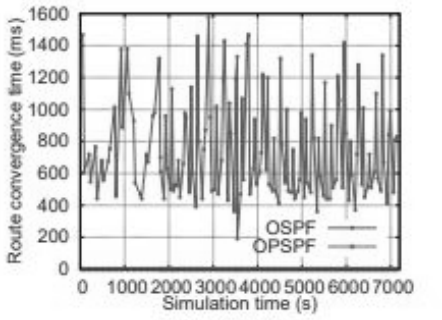

Figure 15. The duration of route convergence, caused by periodic changes in constellation topology, within a single satellite motion time interval (6900s).

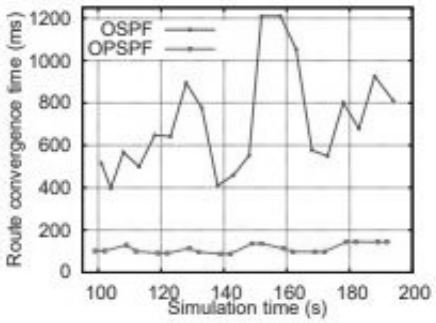

Figure 16. The irregular constellation topology changes, which happen every 5 seconds, cause the route convergence time.

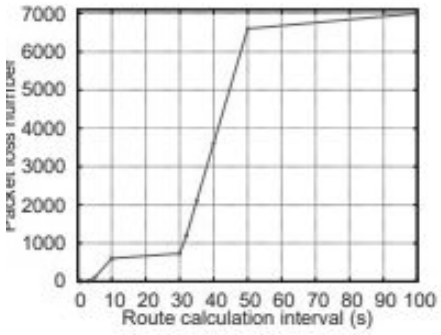

Figure 17. When the periodic route calculation interval increases, the number of packet losses (caused by a delayed routing table update) rises.

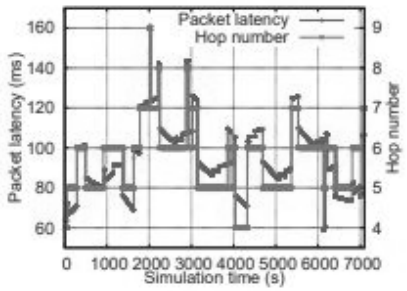

Figure 18. During a satellite constellation's motion, there are fluctuations in the number of ISL hops along the forwarding path and the end-to-end packet latency.

5. Conclusion

The advancement of routing mechanisms for Low Earth Orbit (LEO) satellite networks has been greatly facilitated by the creation of the OPSPF routing protocol. Through the integration of both static and dynamic routing algorithms, OPSPF provides a comprehensive solution that addresses the particular issues brought about by these networks' dynamic character. Static routing works by calculating routing tables inside each satellite on a regular basis. This provides a consistent foundation for network operations and may adapt well to changes in topology. Furthermore, by strategically flooding link state updates throughout the network, the on-demand dynamic routing method enables OPSPF to react quickly to irregular topology changes, such as network failures or recoveries. During rigorous testing, OPSPF has continuously shown better performance than the conventional OSPF protocol, displaying.

OPSPF's future potential is promising, and the protocol could be improved by more study and development to take into account newly developed technology and changing network topologies. Furthermore, investigating machine learning and adaptive algorithmic approaches may improve OPSPF's capacity to forecast and adjust to network dynamics instantly. Furthermore, expanding OPSPF's functionality to accommodate hybrid satellite-terrestrial networks may present fresh possibilities for cohesive and integrated communication systems. In the end, OPSPF serves as evidence of the continuous innovation in satellite routing protocols, opening the door for future satellite communication systems that are more effective, dependable, and flexible.

Author Contributions

V. Kishen Ajay Kumar supervised the project, contributed to the conceptualization of the study, methodology development, and writing of the original draft. Sukerthi Sutraya conducted the literature review and contributed to the writing of the review and editing. C. Sateesh Kumar Reddy conducted the formal analysis and prepared the visualizations. K. Roopa assisted with project administration and provided support throughout the study. N. Balakrishna was involved in the investigation and validation of the research. Finally, G. Sujatha provided necessary resources and helped with the final editing of the manuscript.All authors have read and agreed to the published version of the manuscript

Funding

This research did not receive any specific funding from public, commercial, or non-profit organizations.

Conflicts of Interest
The authors declare that there is no conflict of interest.

Data Availability Statement
Not applicable. No new data were generated or analyzed during the study.

References

1. V. L. Foreman, A. Siddiqi, & O. De Weck, (2017). Large satellite constellation orbital debris impacts: Case studies of oneweb and spacex proposals. In *AIAA SPACE and Astronautics Forum and Exposition*, pp. 5200.
2. X. Zhang, P. Dong, X. Du, H. Zhang, & M. Guizani (2020). Space-ground integrated information network enabled internet of vehicles: architecture and key mechanisms. *IEEE Communications Standards Magazine*, *4*(4), 11–17. IEEE.
3. F. Vatalaro, G. E. Corazza, C. Caini, & C. Ferrarelli (1995). Analysis of LEO, MEO, and GEO global mobile satellite systems in the presence of interference and fading. *IEEE Journal on selected areas in communications*, *13*(2), 291–300. IEEE.
4. K. Li, Z. Li, H. Wang, K. Zhao, W. Li, & Y. Fang (2023). An OLSR-based Multipath Routing Algorithm for LEO Satellite Network. In *Proceedings of the 7th International Conference on Algorithms, Computing and Systems*, pp. 171–177.
5. Y. Lu, F. Sun, & Y. Zhao (2013). Virtual topology for LEO satellite networks based on earth-fixed footprint mode. *IEEE communications letters*, *17*(2), 357–360. IEEE.
6. E. Ekici, I. F. Akyildiz, & M. D. Bender (2001). A distributed routing algorithm for datagram traffic in LEO satellite networks. *IEEE/ACM Transactions on networking*, *9*(2), 137–147. IEEE.
7. J. Li, H. Lu, K. Xue, & Y. Zhang (2019). Temporal netgrid model-based dynamic routing in large-scale small satellite networks. *IEEE Transactions on Vehicular Technology*, *68*(6), 6009–6021. IEEE.
8. W. Huang, R. Andrada, K. Holman, D. Borja, & K. Ho (2024). A Preliminary Availability Assessment of A LEO Satellite Constellation. In *2024 Annual Reliability and Maintainability Symposium (RAMS)*, pp. 1–6. IEEE.
9. A. Al-Hourani (2021). Optimal satellite constellation altitude for maximal coverage. *IEEE Wireless Communications Letters*, *10*(7), 1444–1448. IEEE.
10. H. W. Lee, S. Shimizu, S. Yoshikawa, & K. Ho (2020). Satellite constellation pattern optimization for complex regional coverage. *Journal of Spacecraft and Rockets*, *57*(6), 1309–1327. American Institute of Aeronautics and Astronautics.
11. Y. Chen, M. Zhang, X. Li, T. Che, R. Jin, J. Guo, W. Yang, B. An, & X. Nie (2022). Satellite-Enabled Internet of Remote Things Network Transmits Field Data from the Most Remote Areas of the Tibetan Plateau. *Sensors*, *22*(10), 3713. MDPI.
12. L. Zhen, A. K. Bashir, K. Yu, Y. D. Al-Otaibi, C. H. Foh, & P. Xiao (2020). Energy-efficient random access for LEO satellite-assisted 6G internet of remote things. *IEEE Internet of Things Journal*, *8*(7), 5114–5128. IEEE.
13. E. Yaacoub, & M.-S. Alouini (2020). A key 6G challenge and opportunity—Connecting the base of the pyramid: A survey on rural connectivity. *Proceedings of the IEEE*, *108*(4), 533–582. IEEE.
14. A. Al-Hourani (2021). An analytic approach for modeling the coverage performance of dense satellite networks. *IEEE Wireless Communications Letters*, *10*(4), 897–901. IEEE.
15. X. Qiu, Y. Wang, X. Xie, & H. Zhang (2020). Resilient model-free adaptive control for cyber-physical systems against jamming attack. *Neurocomputing*, *413*, 422–430. Elsevier.
16. M. Manulis, C. P. Bridges, R. Harrison, V. Sekar, & A. Davis (2021). Cyber security in new space: Analysis of threats, key enabling technologies and challenges. *International Journal of Information Security*, *20*, 287–311. Springer.
17. H. Yan, L. Qiao, W. Wu, J. A. Fraire, D. Zhou, L. Li, & Y. Xu (2024). Routing in future space-terrestrial integrated networks with SATNET-OSPF. *International Journal of Satellite Communications and Networking*, *42*(1), 1–25. Wiley Online Library.
18. H. S. Chang, B. W. Kim, C. G. Lee, S. L. Min, Y. Choi, H. S. Yang, D. N. Kim, & C. S. Kim (1998). FSA-based link assignment and routing in low-earth orbit satellite networks. *IEEE transactions on vehicular technology*, *47*(3), 1037–1048. IEEE.
19. Y. Zhao, & Q. Zhu (2022). Autonomous and resilient control for optimal LEO satellite constellation coverage against space threats. *arXiv preprint arXiv:2203.02050*.

20. J. Yang, D. Li, X. Jiang, S. Chen, & L. Hanzo (2020). Enhancing the resilience of low earth orbit remote sensing satellite networks. *IEEE Network*, *34*(4), 304–311. IEEE.
21. S.-C. L. Shih, C.-H. L. Chia, S.-Y. L. Shao, & others (2022). Towards Resilient Access Equality for 6G Serverless p-LEO Satellite Networks. *arXiv preprint arXiv:2205.08430*.
22. C. Lei, T. Wang, J. Wang, S. Cheng, & Z. Wang (2023). Inter-satellite Routing Study for LEO Constellations Based on Orbit Prediction. In *Journal of Physics: Conference Series*, Vol. 2640(1), p. 012016. IOP Publishing.
23. T. Pan, T. Huang, X. Li, Y. Chen, W. Xue, & Y. Liu (2019). OPSPF: Orbit prediction shortest path first routing for resilient LEO satellite networks. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pp. 1–6. IEEE.
24. Y. Cui, Q. Zhang, Z. Feng, Z. Wei, C. Shi, & H. Yang (2022). Topology-aware resilient routing protocol for FANETs: An adaptive Q-learning approach. *IEEE Internet of Things Journal*, *9*(19), 18632–18649. IEEE.
25. E. W. Dijkstra (2022). A note on two problems in connexion with graphs. In *Edsger Wybe Dijkstra: His Life, Work, and Legacy*, pp. 287–290.
26. D. Liu, J. Cui, J. Zhang, C. Yang, & L. Hanzo (2021). Deep reinforcement learning aided packet-routing for aeronautical ad-hoc networks formed by passenger planes. *IEEE Transactions on Vehicular Technology*, *70*(5), 5166–5171. IEEE.
27. A. Nahar, & D. Das (2020). SeScR: SDN-enabled spectral clustering-based optimized routing using deep learning in VANET environment. In *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*, pp. 1–9. IEEE.