# A GNN–DRL–ResNet-Based Dynamic Routing Algorithm for Low Earth Orbit Satellite Networks

Zhen Zhang*, Yunfei Xia
Shanghai Institute of Satellite Engineering, Shanghai Academy of Spaceflight Technology, Shanghai 201109, China
E-mail: zhangzhensast@163.com
*Corresponding author

**Keywords:** Low Earth Orbit (LEO) Communication Network, Graph Neural Networks, RESNETs, Deep Q Networks (DQNs) and Dijikstras Algorithm

**Received:** October 22, 2025

*There are a lot of nodes in a Low Earth Orbit (LEO) communication network, and their resource limits might change quickly. Because of these features, conventional routing techniques are not a good fit for LEO satellite networks. An inductive learning architecture called Graph Neural Network (GNNs) is proposed in this research to tackle this issue. The number of topological nodes that require training can be drastically decreased with the help of the suggested architecture. Because of this cut, the nodes' computational difficulty is reduced. In addition, routing methods are optimized and learned continuously using deep reinforcement learning (DRL), which is made even more generalizable by building the DRL agent with GNN. Every Deep Q-Network (DQN) agent manages its own tasks in the suggested algorithm for optimizing LEO satellite route planning based on graph neural networks which does not extracts the spatial and temporal information of networks that's why we planned to propose RESNET model to replace DQN. By using the GNN paradigm, it discovers the nodes' concealed states. To determine the best routes, the DRLmodel takes these hidden states into account. A comparison and simulation were run to assess the algorithm's efficiency. Finally, optimization technique is presented to choose the shortest route. The outcomes demonstrate that the suggested method mitigates average overall latency while simultaneously increasing total network throughput. When compared to the Deep Q Networks (DQNs) and Dijikstras Algorithm, the suggested approach achieves a 25% and 30% improvement in average throughput, respectively. Additionally, it can adjust to different topologies and lower average end-to-end latency by 44% and 22%, respectively.*

*Povzetek: Raziskava predlaga uporabo grafnih nevronskih mrež in globokega učenja za izboljšanje usmerjanja v LEO satelitskih omrežjih, kar poveča prepustnost in zmanjša zakasnitev.*

## 1 Introduction

As the need for communications continues to rise on a worldwide scale, LEO satellite networks have emerged as an effective supplementary infrastructure for terrestrial communications. Their high degree of versatility and extensive coverage make them highly desirable. However, the structure of the network and the condition of the links are constantly changing due to the extremely rapid movement of the satellites. Effective routing algorithms are difficult to build due to these dynamics [1], [2], [3]. Recent years have seen a meteoric rise in the popularity of Mega-Constellation Networks (MCNs). They provide low-latency, high-throughput data transfer and global coverage to people all over the world. Although multi-constellation networks (MCNs) greatly enhance global uninterrupted coverage, they also make operating a large number of satellites more difficult. The quantity of hops needed for inter-satellite links (ISLs) and the intricate nature of routing are both increased by this. Satellite design, environmental restrictions, and constant satellite mobility are the three main tenets within which effective packet routing must function [4], [5]. Because of these specifics, routing is extremely difficult and calls for fresh algorithmic strategies. More recently, occurrences in the actual world have shown that there have been hostile efforts to discredit and interrupt the potential of MCN [6].

There has been a marked uptick in interest in LEO satellite network development in the past few years. Starlink, OneWeb, and Telesat are among the many LEO network constellations now under construction worldwide. LEO satellites function in contrast to geosynchronous Earth orbit (GEO) networks [7]. As a consequence, channel conditions improve, transmission rates rise, and latency drops. With these updates, the user experience is

greatly improved. Yet, in comparison to Earth, these satellites are constantly in motion at great rates. Consequently, a significantly higher number of satellites must be deployed in LEO networks to achieve worldwide coverage compared to GEO networks. The network architecture gets increasingly intricate as the number of satellites increases. The network is already complicated, and the satellites' rapid orbits just make things worse. Effective routing algorithms are crucial for a network of this complexity to keep latency low and performance high. This highlights the critical need to create efficient routing algorithms for LEO satellite networks [8], [9].

It is very difficult for routing techniques to handle the complicated structure of LEO networks. The inaccuracy of forward-looking projections and the unreliability of satellite network architecture are both caused by the extremely rapid motion and huge number of satellites [10]. Presently, the majority of research is devoted to studying self-regulating networks, rather than routing techniques tailored for LEO networks via satellites. In order to develop LEO routing algorithms, the authors of reference [11] combined deep learning with neural network graphs. Unfortunately, the computational complexity induced by the changing topology during method execution and the vast number of nodes in LEO satellite constellations are not taken into consideration by the GNN method. Considering how to do distributed modeling on massive graph data is important for LEO satellite communications due to the high number of nodes. Acquiring knowledge of previously unseen nodes is also essential. The algorithms for routing are entirely designed by P. Zuo et al. [12] using deep learning. The difficulty of adjusting to the ever-changing topology of LEO satellite networks is, however, not tackled by their method. Many researchers have looked at self-organizing network routing algorithms and used them as a basis for LEO satellite networks. However, the features of LEO satellite nodes, possible unforeseen topological changes, and cases with scarce computational resources are not taken into account. When determining the shortest pathways, the majority of this research considers the nodes' states. Using deep learning as an example, [13] just takes into account the nodes' congestion states and channel quality. A graph neural network is presented in reference [14] as a tool for developing a deep learning-based routing system. Unfortunately, the features of nodes in wireless sensor networks are the only ones taken into account by their convergence mechanism. No mention of how difficult it is to finish computations for the entire topology due to the restricted processing power of nodes is found in the available research. References [15] and [16] employ deep learning techniques to construct routing schemes for wireless networks that are self-organized and ad hoc networks, respectively. While references [17] primarily address quality of service and end-to-end delay, they neglect to take the impact of sudden topological shifts into account.

One more thing: we figure out how to implement shared training into your routing method. This method is useful for decreasing the difficulty of computation and the number of training nodes. It is also important that the algorithm can learn from its mistakes and improve its path optimization techniques in response to changes in node and link statuses over time. When deciding how to route data across a network, one option is to employ a smart routing solution that draws on deep learning algorithms. To ensure that data travels efficiently across networks, it automates the process of choosing the best routing paths. To do this, one must study the habits and patterns of the network's data flow. A smart routing system that relies on deep learning requires a substantial quantity of data to train the model. For the purpose of developing the model, this data set contains details about the network's topology, as well as data on traffic demand and bottlenecks. Afterwards, routing decisions are made using the trained DL model [18], [19], and [20].

Smart routing techniques built on neural networks can keep tabs on network traffic in real time, unlike conventional techniques. Overall network efficiency can be enhanced by their use as well. But the algorithm must modify the training labels if the network topology changes. This modification is necessary for the conventional deep learning method to ensure accurate output pathways. Additionally, scalability is an issue with the conventional DL technique. A significant increase in processing time could result from training the model with updated input data in the event that the input network configuration changes. While both supervised and unsupervised learning have their place in the machine learning (ML) landscape, DRL stands out. Interactive learning with the environment is at the heart of DRL. In contrast to more conventional approaches to reinforcement learning (RL), DRL makes use of deep neural networks (DNNs) rather than tables. It avoids explicit listing in favor of function approximations when expressing policies. Because of this, DRL can handle complicated real-life tasks without the dimensionality issues that plague conventional tabular RL.

The route optimization issue is one area where trained DRL agents excel, but they struggle when faced with unfamiliar network configurations. The cause of this occurrence can be determined by studying the structure of DRL agents. The inputs and model dimensions of a conventional DQN are dictated by the size of the network, as measured by factors like the network layout. The input and output elements of the model are fixed once training is complete. So, the model might get inputs with unusually large dimensions when it encounters networks of varying sizes. Cutting or padding the input dimensions may alter their values, but doing so will remove any topological

information that may have been stored in the matrix. Moreover, graphs are the fundamental building blocks of computer networks. Current approaches that employ neural networks for processing state matrices fall short when it comes to capturing the nuances of graph architectures. When applied to fresh networks, DRL's efficiency is diminished due to this constraint. Finally, DRL thrives at optimizing routes, but it isn't up to the task of generalizing graph structures or reasoning relationally, which limits its applicability to novel network settings.

### Our contributions are multi-fold:

Our novel contributions are explicitly defined as:

1. **ResNet-Embedded DRL Policy Network:** We are, to the best of our knowledge, the first to integrate a **ResNet architecture** into the DRL policy network for LEO routing. This is crucial for:
    - **Stabilizing Deep Learning:** Preventing the vanishing gradient problem that occurs when trying to train deep networks necessary to process complex, high-dimensional embeddings generated by the GNN.
    - **Accelerating convergence:** Ensuring the DRL agent can achieve high episodic returns faster than a standard DNN. We will provide an **Ablation Study** (Section 6.2) comparing the performance and convergence speed against a standard GNN-DRL-DNN baseline.
2. **Integrated spatio-temporal feature generation:** Our **GNN input features** are meticulously engineered to implicitly capture temporal dynamics in a bandwidth-efficient manner (as argued in the previous response). This eliminates the need for complex, bandwidth-intensive sequence models like RNNs or Transformers in the *onboard inference path*, which are typically computationally prohibitive for satellite hardware. The GNN's role is not static graph representation, but **predicting link durability** using time-sensitive features (e.g., rate of change of distance, time-since-last-link-break).
3. **Distributed agent training with shared GNN embeddings (Future Work/Extension):** The current model assumes a centralized training approach. We propose future work focused on a distributed scheme where the **GNN feature extractor** is shared across local DRL agents, minimizing the communication overhead of policy synchronization.

## 2 Related work

Authors presented an architecture is an SDMCN, or software-defined MCN. By collecting data from satellites and allowing for dynamic global network tracking, this design makes network management very flexible. We offer a technique called Orbit-Grid-Based Dynamic Routing (OGDR) to improve routing in Walker Delta MCNs. Starting with a path model that takes the least hop metric and the ISL distribution into account, the technique chooses access satellites. Next, on an orbital grid, inter-plane and intra-plane links are chosen by utilizing the satellites' interval and the path model's properties. In order to guarantee the least amount of delay, this selection is made immediately according to the satellites' latitudes. Last but not least, we guarantee path dependability by introducing a path failure recovery method. The technique reduces the number of ISL hops and reduces computing complexity, in addition to achieving efficiency similar to the shortest path, as demonstrated experimentally [21].

Because it is simpler and uses fewer resources, geographical routing performs better than centralized techniques on this system's network infrastructure. But "dead-ends" can still happen with geographical routing due to holes in the network. Our proposed solution involves enhancing network accessibility and routing accessibility through the use of encountering inter-satellite links (eISLs). In addition, we analyze eISLs and provide a system model for them. To further facilitate the establishment of eISLs between encountering orbiting satellites, we provide our Dynamic eISLs Configuration (DeC) technique. Through our experiments, we have shown that DeC, when enabled in satellite networks, may drastically cut propagation delay in centralized routing schemes by 21% and path stretch by 14%. DeC can also raise the accessible ratio from 65% to 100% in regional routing while keeping an additional 30% in throughput, which makes it better than schemes that don't use eISLs. As a whole, the outcomes of improving LEO satellite systems' interconnectivity and communication performance using our suggested eISL-enabled satellite network architecture are encouraging [22].

Authors were started by presenting an efficient and adaptable software-defined networking (SDN) framework-based architecture for network administration in this article. Our zonal division strategy and DISNR protocol are built upon this foundation, aiming to reduce maintenance and routing complexity to a minimum. Limiting the extent of fault information flooding is achieved by DISNR through the development of intra- and inter-area link-state synchronization mechanisms. We provide a hierarchical shortest path forwarding (SPF) routing algorithm that significantly enhances efficiency in

the massive LEO satellite network while simultaneously decreasing the computational complexity of routing in terms of both time and space. At last, simulation findings show that the DISNR drastically cuts management expenses in comparison to other routing protocols [23].

Current methods, such as the Minimum Hop Path set, put an emphasis on reducing latency by decreasing hop counts, but they fail to take ISL switching costs into account, resulting in significant instability. The Adaptive Path Routing Scheme gets around this by implementing path similarity thresholds, which lower the frequency of ISL switching between shells. Adaptive Path Routing Scheme's greedy method, on the other hand, frequently gets stuck in local optima and neglects the effectiveness of inter-shell path distance. We present the DP-IRC method, which is built specifically for optimizing inter-shell routing, to overcome these restrictions. The DP-IRC algorithm finds a happy medium between hop counts and ISL stability by using an Integrated Routing Cost (IRC) measure that takes into account switching expenses, inter-/intra-shell hops, and the way multi-shell pathways are structured as a multiple-phase decision problem. Based on the experimental results, DP-IRC significantly lowers the inter-shell ISL switching rates in comparison with Adaptive Path Routing and Minimal Hop Path, respectively, by 21.2% and 40.2%. All the while, it keeps the end-to-end distances very close to ideal [24].

The present LOSN, however, is woefully inadequate to provide such high throughput for ubiquitous service provisioning. When ground gateways are deployed centrally, it leads to a large traffic load in the space section of the LOSN. This load eventually becomes the limiting factor in the expansion of the network's throughput. Secondly, the adaptability of routing computations is affected by the constant swings in traffic load caused by the high-speed movement of LEO satellites. This paper proposes two routing techniques for the dynamic LOSN topology that can boost throughput. When dealing with congestion on the links between satellites in a dynamic LOSN architecture, the congestion-aware load balancing (CALB) algorithm is suggested as a solution. Afterwards, a satellite-ground cooperation–based load-balancing routing method (SGC–LB) is suggested to further mitigate the effects of the network bottleneck. Our suggested routing system was tested on a 288-satellite Walker-Star constellation with satellite-to-satellite links through rigorous simulations. An evaluation of the service blockage rate and network capacity usage rate demonstrates the efficacy of the suggested routing method [25].

Unfortunately, current routing approaches result in unnecessary control overhead due to the topology's high degree of dynamic and unpredictable nature, as well as the restricted computing power available on board. Congestion also causes them to suffer from extremely high packet degradation rates and extremely high latency. Our proposed hybrid routing system in this research makes use of smart area segmentation. This strategy dynamically adjusts to crowded locations by combining centralized and decentralized approaches. In particular, controllers only get status reports from Low-Earth Orbit (LEO) satellites with overloaded communication channels when using software-defined features. Two components that make up these controllers are the Ground Computing Center and the Geostationary Earth Orbit (GEO) satellites. Using a Deep Q-Network (DQN) method, they accurately determine the crowded locations periodically. The suggested approach significantly decreases control overhead, according to numerical simulation findings. In comparison to more conventional methods, it reduces packet loss by about 40% in massive constellations, paving the way for 6G messaging networks that are both durable and scalable [26].

A clustered multi-criteria routing (CMCR) method was suggested by the authors of [27] for use in mega LEO satellite networks that provide multimodal data services. The first step is to create a method for grouping the large LEO satellite constellation, which takes into account the satellites' latitude, satellite connectivity link, line-of-sight (LoS), and flight direction. This grouping separates the CMCR into two types of routing: intra-cluster and inter-cluster. Afterwards, the ever-changing megacluster topology is depicted as a dynamic graph. In order to determine the most effective routes for multimodal communications, both intra- and inter-cluster routing start by finding satellites that can set up an ISL using the dynamic graphs. To show how much ISLs value different data services, we add the attribute consistency; CMCR then uses this consistency comparison to find the most popular pathways. After determining which of the dominant paths has the most desirable feature, the CMCR algorithm chooses one, and it can converge to the best route thanks to a defined routing algebra. The CMCR surpasses current routing techniques in terms of multimodal service preferences, as confirmed by the computational results.

But current routing algorithms aren't going to cut it with forthcoming LEO satellite constellations that are super dense, highly dynamic, and massive in size; they're more suited to grounded or smaller-scale satellite networks. In addition, the routing method must be dynamic since Free Space Optical (FSO) transmissions are anticipated for Inter-satellite Links (ISLs) and the quantity of built FSO ISLs is dependent on the geometrical perspectives and Acquisition, Pointing, and Tracking (APT) endpoints. This research explores a dual-layer network design that incorporates both Medium Earth Orbit (MEO) and LEO satellites to solve these problems. To make things easier and enhance routing effectiveness, the

LEO satellite layer uses local network segmentation. A multidimensional RL routing technique that takes local data into account is then suggested as a means to address the varied Quality of Service (QoS) needs of various grounded applications. To resolve the conflicts that arise from the routing architecture for various applications, a cooperation mechanism has also been meticulously created. Simulation outcomes show that the approach works better than benchmark techniques on varied QoS metrics and can handle different numbers of APT terminals [28].

In this work, we present a routing algorithm and a methodology for creating rules and inter-layer connections in a dual-layer LEO satellite network. The first step is to build a standard dual-layer constellation layout for low-Earth orbit satellites by choosing satellites to link layers. Secondly, a fast inter-layer link switching approach is suggested, which relies on the periodic relative movement of satellites, to guarantee reliable connections over the long term. With this technique, the network's inter-layer linkages are guaranteed to transmit data reliably and continuously. The link rules and constantly changing relationships within the satellite network are also used to construct the time slots for the dual-layer network. For efficient routing, Dijkstra's algorithm is used to calculate the shortest path between source nodes to destination nodes. Then, using the characteristics of the Kuiper constellation, a simulation model with two layers is constructed, and investigations are run in the laboratory. The dual-layer constellation routing algorithm (DCRA) decreases round-trip time (RTT) by 14.5% and 21.23% at 14,000 km relative to the single-layer routing algorithm (SLRA) [29].

This research introduces a new network coordinate system that is based on the ISL architecture and is used to enhance extremely durable LEO mega-constellation adaptive routing techniques. By standardizing on these coordinates, we can streamline the network's architecture and make more consequential routing decisions with less computing burden. We show a proof-of-concept, adaptable, compact routing algorithm based on our topology. For LEO MCN routing techniques, we suggest an evaluation system for robustness in order to facilitate conventional comparisons. An adversarial capability, essential performance indicators, and scenario tests are all defined by this framework. Several top-tier dynamic routing techniques are tested and compared with our routing approach using the suggested framework. At highly intense levels of adversarial interruption, the results reveal a 13% improvement in the packet delivery rate [30]. Table 1 shows the comparison of routing techniques in LEO satellite networks.

Table 1: Comparison of representative routing algorithms for LEO satellite networks

| **Algorithm / Work** | **Dataset / Topology** | **Dynamic Adaptation Mechanism** | **Metrics Used** | **Performance Summary** |
|---|---|---|---|---|
| **OGDR – Orbit-Grid–Based Dynamic Routing** [21] | Walker-Delta constellation; Orbit-grid model | • Grid-based satellite selection<br>• Hop-minimizing path model<br>• Latitude-based dynamic ISL selection<br>• Fast path failure recovery | • Delay<br>• Hop count<br>• Computational cost | • Achieves near-shortest-path delay<br>• Reduces ISL hops and switching<br>• Lower computational complexity than shortest path |
| **DeC – Dynamic Encountering ISL Configuration** [22] | LEO constellation with eISL capability | • Encounter-based ISL establishment<br>• Latency-aware link configuration<br>• Regional eISL discovery | • Propagation delay<br>• Path stretch<br>• Accessible ratio<br>• Throughput | • 21% lower delay<br>• 14% lower path stretch<br>• Accessible ratio increases from 65% → 100%<br>• +30% throughput improvement |
| **DISNR – Distributed Intra/Inter-Area Link-State Synchronization Routing** [23] | SDN-enabled LEO; Zonal segmentation | • SDN-based global control<br>• Intra/inter area link-state sync<br>• Hierarchical SPF routing | • Control overhead<br>• Routing complexity<br>• Delay | • Significantly reduces management overhead<br>• Improves routing efficiency in large-scale constellations |
| **DP-IRC – Dynamic Path Integrated Routing Cost** [24] | Multi-shell large-scale LEO constellation | • Multi-phase decision modeling<br>• ISL switching | • ISL switching rate<br>• End-to-end | • ISL switching reduced by 21.2% vs APRS |

| | | | | |
|---|---|---|---|---|
| | | penalty<br>• Integrated Routing Cost (IRC) metric | distance<br>• Path stability | • Reduced by 40.2% vs Minimum Hop Path<br>• Maintains near-optimal path length |
| **CALB & SGC-LB – Congestion-Aware Load Balancing + Satellite-Ground Cooperative LB** [25] | 288-satellite Walker-Star constellation | • Congestion detection<br>• Cooperative routing via ground–satellite integration<br>• Dynamic load balancing | • Blockage rate<br>• Network capacity usage | • Higher network throughput<br>• Lower service blockage rate<br>• Effective under heavy traffic swings |
| **Hybrid DQN-based Area Segmentation Routing** [26] | Large LEO constellation with SDN controllers | • Smart area segmentation<br>• Deep Q-Network for congestion zone detection<br>• Dynamic centralized–distributed hybrid control | • Packet loss<br>• Control overhead<br>• Delay | • 40% reduction in packet loss<br>• Major control overhead reduction<br>• Suitable for massive-scale constellations |
| **CMCR – Clustered Multi-Criteria Routing** [27] | Mega LEO constellation for multimodal services | • Satellite clustering by LoS, altitude, direction<br>• Dynamic intra-/inter-cluster routing<br>• Attribute consistency for service-specific paths | • Path quality<br>• Service preference score<br>• Routing convergence | • Selects dominant multimodal paths<br>• Outperforms existing multi-criteria routing techniques |
| **Dual-Layer (MEO–LEO) RL-Based Routing** [28] | Dual-layer constellation with APT-based FSO ISLs | • Local segmentation in LEO layer<br>• RL-based QoS-aware routing<br>• Cooperation mechanism across apps | • QoS metrics<br>• Delay<br>• Throughput | • Outperforms benchmarks across multi-QoS criteria<br>• Handles varying APT terminal availability |
| **DCRA – Dual-Layer Constellation Routing Algorithm** [29] | Kuiper-style dual-layer constellation model | • Inter-layer link switching using relative motion<br>• Time-slot rule construction<br>• Dijkstra-based routing | • RTT<br>• Path reliability | • RTT improvement of 14.5%–21.23% over SLRA<br>• More stable multi-layer connectivity |
| **Topology Coordinate System + Robust Adaptive Routing** [30] | Large-scale LEO mega-constellation; adversarial test scenarios | • New coordinate topology system<br>• Compact routing via coordinate mapping<br>• Robustness evaluation framework | • Packet Delivery Rate<br>• Robustness metrics under adversarial interference | • 13% higher PDR under high-intensity adversarial disruption<br>• Better resilience and reduced computation |

# 3 Preliminaries

Satellites, base stations, user devices, and distant servers are all part of the LEO-MSN connection that is examined in this study. All the way from the user's interface to the distant server, informational packets can be transmitted through the ISLS of a LEO satellite in space. It is possible to express the LEO-MSN at each given time as a graph with no direction, $G(t) = (V, E(t))$, where V is the collection of vertices and E is the set of edges. Satellites, ground stations, user terminals, or remote servers are all represented by node $v_t \in V$. Each edge $e_{x,j} \in E$ in a network connects two nodes i and j. Using the input parameter TLE (Two-Line Element set), the SGP4 orbital

propagation model determines the positions of all the satellites. The following formula (1) is used to produce the most recent version of the satellite coordinates:

$$r_x(t) = SGP4(TLE_t, t), \Delta t = 60 \text{ s} \quad (1)$$

$TLE_t$ has several important orbital characteristics. Some of these parameters include the eccentricity, orbital orientation, ascending node position, and orbital halfway length axis. Any two low-Earth orbiting satellites can form an Inter-Satellite Link (ISL) with a pair of satellites in the neighboring or identical orbital planes. As long as the two satellites remain in an identical orbital plane at a constant distance, the satellite-to-satellite link (ISL) inside the orbit should remain stable throughout the system's lifetime. We can reliably assume stability for a system of satellites whose orbits are perfectly regular. In order to determine how far apart two nearby satellites are, one can use the following formula ().

$$L_{\text{neter}} = 2(R_{\text{earth}} + h) - \sin\left(\frac{\pi}{2N_L}\right) \quad (2)$$

To determine satellites on neighboring orbital planes, one can use the following equation (3):

$$L_{\text{ady}} = 2(R_{\text{earth}} + h) \cdot \sin\left(\frac{\Delta\Omega - \sin\theta}{2}\right) \quad (3)$$

In this case, the radius of the Earth (in kilometers) is denoted by ($R_{\text{earth}}$), and the height of the spacecraft's orbit is represented by (h). The quantity of satellites occupying the same plane of orbit is denoted as ($N_L$). ($\theta$) is the angle at which the orbit is inclined, and ΔΩ) is the change in the elliptical plane's longitude. Tabulated in Table 2 are the key concepts covered in this part.

## Channel model

The two primary types of transmission channels used by LEO gigantic satellite networks for communication are:

- Inter-satellite link channel model: The free-space path loss model is utilized as the satellite-to-satellite link channel in this paper. Here is one way to convey the path loss using the formula (4) [31]:

$$FSPL = \left(\frac{4\pi Rf}{c}\right)^2 \quad (4)$$

With R representing the distance between two points and c representing the speed of light (in m/s). The satellite-to-satellite link frequency, expressed in hertz (Hz), is denoted by f. The following formula (5) can be used to compute the received power, $P_R$:

$$P_R = P_{T_x} \cdot G_T \cdot G_R \cdot \frac{c^2}{(4\pi Rf)^2 L} \quad (5)$$

Where $P_{Tx}$ is the transmit power, and $G_T$ and $G_R$ are the transmit and receive antenna gains, respectively. $L$ is the additional system loss. The data rate (in bps) of inter-satellite links can be expressed as follows: With $P_{Tx}$ representing the power used to transmit and $G_T$ and $G_R$representing the receive and transmit antenna gains, respectively. Next, L denotes an extra loss to the system. Using equation, we can quantify the data transmitted via inter-satellite links in bits per second:

$$R_{ISL} = B \cdot \log_2\left(1 + \frac{P_R}{N_0 B}\right) \quad (6)$$

In this case, the channel bandwidth is denoted by B, and the noise spectral intensity is denoted by $N_0$. Satellite-ground link channel model included this category are the pathways that carry data from user devices to satellites and back again, as well as the pathways that connect satellites with base stations. The wireless signal travels through multiple settings on its way from the local node to the satellite node. The surface of the earth, the atmosphere, and space all fall within this category. A number of parameters need to be taken into account throughout this propagation. These consist of the ground shadow effect, the multiple path effect, the loss of signal in the atmosphere, and the loss of signal in space. This study builds the satellite-to-ground link channel model using the correct settings from 3GPP TR 38.811. Here are the parts that make up the Path Loss (PL) [32]:

$$PL = PL_b + PL_z + PL_z + PL_z \quad (7)$$

In which PL is the overall trajectory loss in decibels, and $PL_b$ is the fundamental path loss in decibels. The gas attenuation, measured in decibels, is $PL_Q$. The building entrance loss is denoted by $PL_e$ and measured in decibels, while $PL_s$ is the attenuation caused by atmospheric or tropospheric scintillation. Here is a simulation for the basic route loss, expressed in decibels:

$$PL_b = FSPL(s, f_c) + SF + CL(a, f_c) \quad (8)$$

$CL(\alpha, f_c)$ represents the clutter loss, while FSPL$(s, f_c)$ stands for the free space path loss. The shadow fading loss, denoted as SF, is actually a random value distributed normally, with a range of 0 to $\sigma^2$. Here is the formula for calculating the SNR:

$$SNR = EIRP - k - PL - B + \frac{G}{T} \quad (9)$$

So, EIRP stands for efficient isotropic radiated power, which is measured in decibels of power. Antenna gain relative to noise temperature $(\frac{G}{T})$ is measured in decibels/kelvin. In this equation, the Boltzmann constant, denoted by k, is equal to -228.6 dBW/K/Hz. The path loss, denoted by PL, is measured in decibels, while the channel bandwidth, denoted by B, is measured in decibels per Hz. The decibel scale ensures uniformity, even though the initial units of measurement for each component vary. Link quality is clearly shown in signal-to-noise ratio (SNR), which is calculated by adding or removing decibel values and integrating transmit power, path loss, interference, and receiver efficiency. The following is the formula (10) for EIRP:

$$EIRP = P_T - L_C + G_T \quad (10)$$

As shown, $P_T$ is the antenna's transmit strength measured in decibels. "$L_C$" stands for the cable loss, measured in decibels. A transmit antenna's gain, measured in decibels, is $G_T$. The following is the formula for the antenna gain in relation to noise temperature $\frac{G}{T}$

$$\frac{G}{T} = G_R - N_f - 10\log_{10}\left(T_0 + (T_a - T_0)10^{-0.1N_f}\right) \quad (11)$$

This is the result of expressing the satellite to ground link data rate (in bps) using the Shannon formula:

$$R_{\text{CSL}} = B\log_2(1 + SNR) \quad (12)$$

Here is a definition for the propagation delay of traffic d with Path $_{Src,Dst}$:

$$\text{Delay}_{\ d} = \sum_{e_{ij} \in P_{ath}} \frac{Bv_d}{R(i,j)} \quad (13)$$

In this equation, $Bv_d$ is the dimension of the traffic d data, and $R(i,j)$ is the connection's transfer rate (I,j).

## Constraints analysis

The following limitations can be defined using the aforementioned models' presumptions:

- **Link capacity constraint:** Every link must be able to handle no more traffic than it can handle at any one time [33]:

$$0 \le X_t^d(i,j) \le C_t(i,j), \forall e_{t,j} \in E(t) \quad (14)$$

Where the quantity of traffic sent on link (i,j) at time t is represented by $X_t^d(i,j)$, and the maximum bandwidth of link (i,j) at time t is represented by $C_t(i,j)$. Node capacity constraintIn other words, node u's cache capacity limit is the maximum quantity of data that may be maintained on the node.

$$0 \le \sum \int_t^{t+\Delta t} f_{v,\mu}(t)dt - \sum \int_t^{t+\Delta t} f_{u,w}(t)dt \le St(u) \quad (15)$$

In this context, St$(u)$ represents the dimension of node u's cache, $f_{v,u}(t)$ and $f_{u,w}(t)$ represent the flows into and out of node u at timestamp t, respectively.

- **Service function constraint:** Each packet of data traveling along the routing pathway Path $_{Src,Dst}$ from the original node Src must adhere to certain service function requirements, such as,

$$\text{Path}_{\ Src,Dst} = \{V_{Src} \to \cdots \to VF_t \cdots \to VF_{FN} \cdots \to V_{Dst}\} \quad (16)$$

The i[th] operational node that the complete service transfer goes through is denoted by $VF_t$. In this case, FN is the total quantity of SFC functional nodes along the route [34], [35].

# 4 System model

The graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ depicts the LEO satellite network, with V standing for the set of satellite nodes and E for the set of connections to satellites. We will use the coordinates (i_i,) to denote the location of a satellite in an Iridium-like system with M orbits and N spacecraft per orbit. The orbit number of the satellite is denoted by i, and the number of satellites in orbit is denoted by f, where $\mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{M}, \mathbf{1} \leq \boldsymbol{j} \leq \boldsymbol{N}$. (see fig 1).

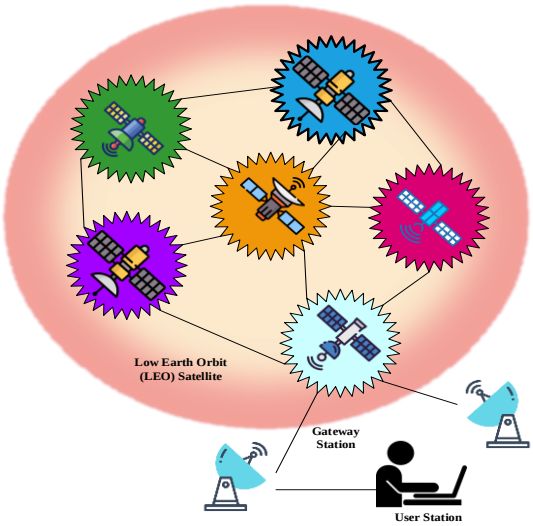


Figure 1: LEO network model

One kind of link connects satellites in the same orbit to another, while another type connects satellites in different orbits to each other. What this implies is that there can be up to four direct communications between any given satellite. We solely focus on the packet transfer technique between satellites in this paper. The packet begins at the first node, $\boldsymbol{N_4}$, and goes on to the next hop, N, by traversing the set of neighboring nodes, $\boldsymbol{N}_{\text{sis}}$ . We can see the overall count of satellite nodes in the variable (n). For each node, we used the value of its queue utilization ($\boldsymbol{QU_l}$) to represent $\boldsymbol{v_l}$. It can be calculated as the ratio of the number of packets presently in the queue of the satellite node ($\boldsymbol{v_l}$) to its queue bandwidth ($\boldsymbol{v_l}$) and stands for:

$$QU_i = \frac{\text{Number of packets in queue of node } i}{\text{Total queue size of node } i}. \quad (17)$$

Our DRL algorithm was implemented on the basic model described in the next section. For the collection of satellite-to-satellite links, denoted as $\boldsymbol{E} = \{\boldsymbol{e}_{(v_1,v_2)}, \boldsymbol{e}_{(v_2,v_3)}, \cdots, \boldsymbol{e}_{(v_1,v_1)}\}$, the satellite interaction link

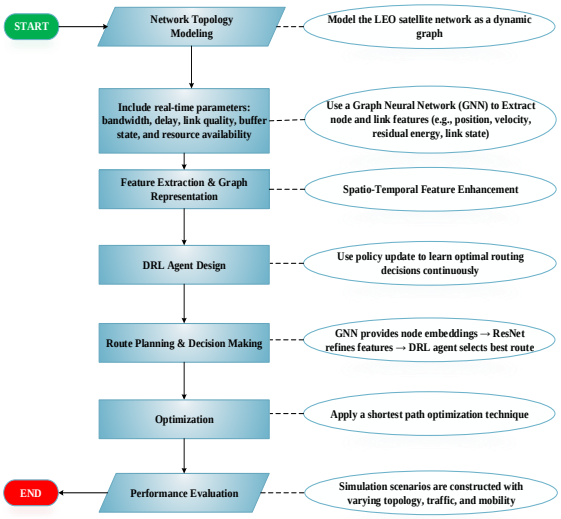


Figure 2: Proposed flowchart

between satellites $\boldsymbol{v_l}$ and $\boldsymbol{v_j}$ is represented by $\boldsymbol{e}_{(v_1,v_1)}$. The delay in time $\boldsymbol{C}_{(v_1,v_1)}$ and a bandwidth $\boldsymbol{B}_{(v_1,v_1)}$ were used to represent each intersatellite link, respectively. The foundational model that will be detailed in the following part was used to develop our proposed method which is shown in fig 2. After the current hop node sends the packet to its neighbor, the process is repeated by that node, which updates the packet's transmission delay D based on the delay accumulation rule. The packet's transmission delay is denoted as $\boldsymbol{D_{ij}}$. This process continues until the data packet reaches its final destination node. Developing a course of action that reduces D is the issue at hand. The algorithm must take into account the link's congestion in the real-life situation when numerous packets are transmitted via the network. To begin with, the algorithm needs to be able to plot out a reasonable route from the starting node to the ending node. Then, it needs to minimize the delays along that route, which includes both propagation and queuing delays. Consequently, the algorithm's end objective is to guarantee a high packet arrival rate while minimizing packet delivery delays.

## DRL for dynamic routing

For massive constellations of LEO satellites, the inter-satellite network connection problem is complex and highly dynamic. DRL provides a robust paradigm for solving this problem. Due to the rapid movement of the satellites, the network architecture is constantly evolving. This calls for a new approach to routing data, one that is more flexible than what conventional algorithms can offer. Through the use of Markov Decision Process (MDP) modeling, DRL enables every satellite to learn the best strategy for choosing the next hop in the network, just like an agent. Reducing end-to-end delay and improving congestion management are the primary objectives.

## DRL formulation

We now exclusively use standard notation:

- State: $s_t = \langle \mathcal{G}_t, \mathbf{L}_t, \mathbf{Q}_t \rangle$. Where $\mathcal{G}_t$the GNN-embedded is graph state, $\mathbf{L}_t$ is the vecto of link qualities, and $\mathbf{Q_t}$ is the task queue vector.
- Action: $a_t \in \{1, \ldots, |V|\}$. An action selects the next hop satellite.
- Reward: $R_t = -\left(\lambda_{\text{Delay}} \cdot D_t + \lambda_{\text{Congest}} \cdot C_t\right)$, where $D_t$ is the delay cost, $C_t$ is the congestion cost, and $\lambda$ values are weighting factors.

All variables are explicitly defined upon first introduction, and clear indexing relationships (e.g., $i, j \in V$ for nodes) are maintained.

Here, the ResNet (Residual Network) model operates as the DRL framework's neural network, much like a DQN or an Actor-Critic network. ResNet can train extremely deep networks because it makes use of residual blocks. By utilizing these blocks, the network is able to reliably extract intricate, non-linear state variables from inputs that are multidimensional and subject to time variation. The learning process cannot be stabilized without this capability. Additionally, the agent is able to apply its routing expertise to topological arrangements that have not been previously encountered. Therefore, the network can keep its effective routing even when things are changing. The result is a space-based network backbone that is both more robust and more efficient. GNN extracts relational structure and mitigates topological volatility. ResNet enhances feature propagation, enabling deeper temporal reasoning without vanishing gradients. DRL learns long-term routing gains, not just local shortest paths. Joint GNN–ResNet embedding improves prediction of future link reliability.

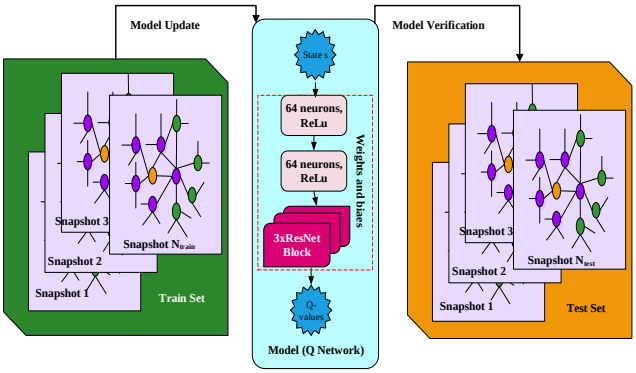


Figure 3: DRL with RESNET model

## ResNet attention mechanism

For the model to be able to understand the significance of the traffic data at each time, the researchers of this paper included temporal attention and spatial attention. This was utilized for the processing of space-and time-related traffic data from satellites in low Earth orbit. Its purpose is to capture the data's geographical and temporal connections to extract useful features. Here are the processes involved in designing the attention system: (B, T, N, F) is the structure of the input data, where B is the number of batches and T is the duration of the time sequence. F is the length of features, and N is the total quantity of nodes. In the adjacency matrix, spatial_neighbors is placed.

**Temporal attention calculation:** For each time lag i (ranging from 1 to num_lags. Here, the setting of the time step needs to consider the actual situation and the data collection frequency. In Section 4.1, as the collection frequency of the Abilene dataset was 5 min, it is reasonable to set the time step to $24(2\text{ h})$, and as the collection frequency of the traffic data in the internal low Earth orbit constellation business simulation system is 5 s , it is reasonable to set the time step to $30(1\text{ min})$. An overly long-time step will lead to high memory usage, introduce noise, and even cause model overfitting, while an overly short time step may limit the learning ability of the model. The historical data and the current data are extracted for the lagged data $\text{lagged\_x} = \text{x}[:, -\text{i}, :, \text{i}]$ and the current data $\text{current\_x} = \text{x}[:, \text{i}, :, \text{i}]$, both of which have the shape of (B, T-i, N, F). Next, the historical data and the current data are weighted using the temporal attention weights. The formula is as follows, and the resulting data also have the shape of (B, T-i, N, F). ":" indicates the selection of all elements in that dimension. For example, $\text{x}[:, :, :, :]$ means selecting all elements of the tensor x . " i " means taking the elements from the i -th position (including the i -th position) of that dimension to the end of that dimension, while " $= -\text{i}''$ means taking the elements from the starting

position of that dimension to the $i$-th position from the end (excluding the $i$-th position from the end).

$$\text{attention}^{(i)} = \sum_{f=1}^{F} W_{\text{temp}}^{(i-1)}\text{-lagged\_x....f-current\_x } x_{1,\ldots,f} \quad (18)$$

All time lags' attention scores are saved and then joined. A tensor with the form (B, T-num_lags, N, num_lags) is produced by applying the following formula.

$$\text{temporal\_attentions} = [\text{attention}^{(1)}, \text{attention}^{(2)}, \cdots, \text{attention}(\text{num\_lags})]$$
$$\text{temporal\_attention} = \text{concat}(\text{temporal\_attentions}, \text{dim} = 1)$$

**Spatial attention calculation:** To figure out spatial attention, we need to make the data set x bigger so that it can be increased by the spatial adjacency matrix. Apply the spatial attention weights $W_{\text{spatital}}$ to the input information and generate a geometric adjacency matrix. (B, T, N, 1) is a representation of spatial attention, and this equation is as follows. The variable "spatial_attention" is defined as the sum of all mappings from f=1 to 0.^F▒ 〖" " W〗 _"spatial " ^((f))·x·" spatial_neihbors "

$$\text{spatial\_attention} = \sum_{f=1}^{\text{F}} W_{\text{spatial}}^{(\text{f})} \cdot x \cdot \text{spatial\_neihbors} \quad (19)$$

The balanced spatial attention is expanded to (B, T-num_lags, N, num_lags), added to the temporal attention, and the attention weights are calculated using a softmax operation.

$$\text{combined\_attention} = \text{temporal\_attention} + \text{spatial\_attention}$$
$$\text{attention\_weights} = \text{softmax}(\text{combined\_attention}, \text{dim} = 1)$$

Expand the input data $x$ to the shape of (B, T-num_lags, N, num_lags, F), and then use the attention weights to perform a weighted sum on the input data. The formula is as follows, and the resulting data has the shape of (B, N, F), representing the weighted features of each node.

$$\text{output} = \sum_{t=1}^{T-\text{num\_lags}} \sum_{i=1}^{\text{num\_lags}} \text{attention\_weights}, t;,;X_{i,t,r} \quad (20)$$

The efficiency of the approach is highly dependent on the reasonableness of the configuration of auxiliary components connected to reinforcement learning. State space, action space, and reward settings are described in depth here, along with the suggested DRL smart routing approach for LEOs.

## State space

For the DRL model to work, the state must accurately and completely represent the agent's (the present LEO's) surroundings. In this study, we use 4 routing-specific metrics of nodes that are two hops away from the current node to help the agent learn more about its surroundings. The following section introduces these parameters, which make up the state space. One measure of channel quality is the signal-to-interference-noise ratio, or SINR. This study uses the signal-to-noise ratio (SINR) of the channel connecting the current node (c) and a one-hop node (i) at time (t) as represented by $\eta_{c,t}^{J}(t)$. Likewise, at time t, the SINR of the channel connecting a one-hop node (i) and a two-hop node (j) is represented by $\eta_{l,t}^{II}(t)$. These annotations record the signal strength over a series of network hops. This includes:

$$\eta_{c,t}^{l}(t) = \frac{g_{c,t}(t)p_{c,t}(t)}{\sigma_t^2(t)}$$
$$\eta_{t,t}^{II}(t) = \frac{g_{t,l}(t)p_{t,j}(t)}{\sigma_j^2(t)} \quad (21)$$

In this case, the average channel gain of the two connections, ($c \to i$ and $i \to j$), is represented by ( $g_{c,l}(t)$ and $g_{l,j}(t)$, respectively. In terms of transfer powers, node (c) has $p_{c,l}(t)$ and node (i) has $p_{t,j}(t)$ at any given time. At networking nodes (i) and (j), the variation of Gaussian white noise is represented by $\sigma_t^2(t)$ and $\sigma_j^2(t)$ accordingly. In the section that follows, the time attribution in the formulae is removed for clarity. After that, we may use the Shannon formula to find the channel capacity.

$$C_{c,t}^{I} = B_{c,t}\log_2\left(1 + \eta_{c,t}^{I}\right)$$
$$C_{t,j}^{II} = B_{i,j}\log_2\left(1 + \eta_{t,j}^{II}\right) \quad (22)$$

In this context, the accessible bandwidth of the two lines is represented by $B_{c,I}$ and $B_{I_dJ}$ correspondingly. We incorporate the channel's bandwidth ratio as part of the state to objectively assess the possibilities of various candidate nodes. What follows is a possible representation of this ratio.

$$\bar{C}_{c,t}^{I} = \frac{C_{c,t}^{I}}{\sum_{m\in N_c} C_{c,n}^{I}/|N_c|}$$
$$C_{c,t}^{II} = \frac{|M_c - M_c \cap N_c| \times \sum_{k\in\left(N_t - N_f \cap N_c\right)} C_{t,g}^{II}}{|N_t - N_t \cap N_c| \times \sum_{n,m\in(M_c - M_c\cap N_c)} C_{h,m}} \quad (23)$$

where $N_t$ is the set of all the nodes that are one hop away from node i, which is a node that is one hop away from node c, and ||| is the cardinality of the set. When determining the ratio of nodes with two hops, it is important to remember to omit those nodes that are part of both hops at the same time.

Distance, the second component of the suggested method's state set, is an essential component of nearly all route selection techniques. Distances between nodes might be readily determined with the positioning capability. Specifically, DRL-FIR used the following proximity ratios:

$$\bar{D}_{c,t}^{t} = \frac{D_{c,t}^{t}}{\sum_{m\in N_c} D_{c,n}^{t}/|N_c|}$$
$$\bar{D}_{c,t}^{tI} = \frac{|M_c - M_c\cap N_c|\times\sum_{f\in(N_t - N_t\cap N_c)} D_{t,g}^{tI}}{|N_t - N_t\cap N_c|\times\sum_{n,m\in(M_c - M_c\cap N_c)} D_{n,m}^{H}} \quad (24)$$

In this context, the initial distance $D_{c,I}^{l}$ and the second distance $D_{t,j}^{II}$ represent the distances among the present node (c) and the single-hop node (i) and the two-hop node (j), respectively. When calculating the best route to take, these variations are crucial for gauging the quality of the link and the latency in communication.

This study also considers the fact that node business load plays a significant role in determining routing efficiency, particularly latency and data loss rate. As part of the state configuration, we provide the load ratio of nodes that are within two hops. The following is a mathematical expression for this load ratio.

$$L_{c,t}^{I} = \frac{L_{c,t}^{I}}{\sum_{m\in N_c} L_{c,n}^{I}/|N_c|}$$
$$L_{c,t}^{II} = \frac{|M_c - M_c\cap N_c|\times\sum_{f\in(N_t - N_t\cap N_c)} L_{t,f}^{II}}{|N_t - N_t\cap N_c|\times\sum_{m,m\in(Mc - Mc\cap N_c)} L_{n,m}^{IH}} \quad (25)$$

As $L_{c,l}^{I}$ stands for the queue length in the MAC layer of node i, and $L_{t,l}^{II}$ for node j. Additionally, it is important to take into account the mobility properties of the nodes. The dependability of routing can be impacted by these features, which in turn affect the longevity of links. Using the following equation, we can determine the lifespan of the connection between nodes / and j, denoted as $T_{dj}$.

$$\left(x_{ij} + a_{ij}T_{ij} - x_j - a_{jk}T_{ij}\right)^2 + \left(y_j + a_{jk}T_{ij} - y_j - a_{kj}T_{ij}\right)^2 = R^2 \quad (26)$$

where $(x_a, y_a)$ represents the location vector of node a and $(v_{xa}, v_{ya})$ represents the speed vector. Next, the suggested approach's state was used to set the link lifespan ratios, which can be defined as follows:

$$T_u^u = \frac{T_d^g}{\sum_{ncN_c} T_s^g/|N_c|}$$
$$T_{cd}^g = \frac{|M_c - M_c\cap N_c|\times\sum_{n\in(N_c - N_c\cap N_c)} r_u^g}{|N_c - N_c\cap N_c|\times\sum_{n\in C|N_c, M_c\cap N_c)}} \quad (27)$$

The lifespans of the linkages $c \to i$ and $i \to j$ are denoted by $T_{cf}^{l}$ and $T_{l,}^{l}$ accordingly. Matrix $S = [S^I, S^{II}]$ can be used to express the status of the DRL-FIR procedure in the end.

## Action space

It seems reasonable that picking an action in the action space would be the same as picking a node to hop to next. In contrast to Q-learning, which dynamically changes the Q values of all nearby nodes, DRL-FIR is an example of an offline learning mode. Therefore, the quantity of one-hop nodes in the state space should be predetermined to match the total area of the action space. Our mathematical expression looks like this: $a$ is a member of the set {$X$ node $_{c1}$, node $c_{c2}, \cdots$, node $c_{c1}c_{c1}$}, where node $_c$ indicates that node * is the next routing node for node c.

## Reward

By carefully selecting nodes, the suggested approach sought to ensure the security, consistency, and trustworthiness of data transfer. Following this objective, the agent should get the highest reward if the next hop is the final node. And then it gets

$$F_l = \begin{cases} 1, \text{ Node } j \text{ is the destination node} \\ 0, \text{ Node } j \text{ ts not the destination node} \end{cases} \quad (28)$$

At the same time, we point out that conventional wisdom holds that ordinary relay node selection ought to be incentivized in order to steer routing convergence. In designing the incentive, these four factors were thus taken into account. The channel capacity ($C_f$) and link lifespan ($T_f$) between the currently selected node and the next node j that has been chosen are shown explicitly. The MAC queue length of node j and its proximity to the endpoint are denoted by $D_j$ and $L_j$, respectively. The corresponding reward was determined using the suggested procedure.

$$r = -\left(\mu_1 e^{-\frac{c_9}{c_{\max}}} + \mu_2\frac{D_j}{D_{\max}} + \mu_3\frac{L_j}{L_{\max}} + \mu_4 e^{-\frac{T_j}{T_{\max}}}\right)F_l, \quad (29)$$

This is where the highest channel bandwidth, link lifespan, length to the final point, and MAC queue length among the surrounding single-hop nodes are represented by $C_{\max}, T_{\max}, D_{\max}$, and $L_{\max}$, respectively. The weight factor is represented by a variable called $\mu_s$. The weight factors $\mu_1$, $\mu_2$, $\mu_3$, and $\mu_4$ all add up to 1, which is an additional requirement.

## Dynamic route update

Nodes in the inter-satellite network update the global topology data at regular intervals. In response to changes in the status of link connectivity among nodes, they determine the present link adjacency matrix. We use the notation ({AdjMatr}[i][j]) to portray the status of the link that connects nodes (i) and (j). When two mobile nodes are connected, it indicates that they are neighbors. Here, we see that the adjacency matrix has a value denoted as (m). Values are denoted as (n) if not.

$$\text{AdjMatr}\,[n]/] = \begin{bmatrix} n & m & \cdots & n & \cdots & n & n \\ m & n & \cdots & m & \cdots & n & n \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ n & n & \cdots & m & \cdots & n & n \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ n & n & \cdots & n & \cdots & n & m \\ n & n & \cdots & n & \cdots & m & n \end{bmatrix} \quad (30)$$

### Optimal path selection for inter-satellite links

After collecting all possible paths, the one with the fewest obstacles is chosen for data transmission. First, out of all the possible shortest paths, the one with the most bandwidth capacity of nearby nodes is chosen, followed by the complete path. By taking this route, you can stay away from the crowded areas. Doing so can enhance the system's transmission throughput. Furthermore, it contributes to a more even distribution of network traffic. If you pick nodes that use less data as neighbors, you won't have to worry about paths with low overall load where one node uses a ton of data—maybe even more than its threshold—while the rest of the path stays well below it. Data congestion is quite probable if it exclusively uses the variable with the path's lowest occupancy rate. We take into account the nodes that are neighboring the source node as our goal to minimize this issue. These neighbor nodes often play a somewhat essential role, according to experience, since they are the initial hop for services to be transmitted from the source node to distant destination nodes. This is particularly the case if the source node needs to transmit data to numerous nodes that are located at great distances from each other. Selecting an appropriate neighbor node can efficiently meet the service requirements of each destination node if there are numerous such nodes. In addition to enabling shunting and improving overall system throughput, it can help balance the load. To avoid transmission issues, system throughput issues, and ultimate delay, shunting is used to stop several services from choosing the same neighbor nodes.

Keep track of how many transmission links there are between nodes in an intersatellite network by using (s). Here, we can think of node bandwidth utilization as a weight, with a larger value equating to a higher node bandwidth capacity. A node's connection data use value is raised by 1 in response to a data service demand. If the node has an outage or congestion, its weight is reset to zero, and it is thereafter disabled from transmitting data. Take into consideration that there are a total of (m) nodes in the network and (n) shortest paths that have been found. Among these (n) links, the best one is chosen when the ratio of the number of node connections to their weight is the smallest. The number of links is denoted by c, the weight by w, the selected node by s, and the not selected node by sn. We determine the link state and the link between the chosen nodes by dividing c(s) by w(s). S needs to meet: If c(s) divided by w(s) is less than c(sn) divided by w(sn).

While the training of the GNN-DRL model is computationally expensive and is performed offline or via federated learning, the real-time inference is a simple forward pass through the trained network. This fixed, fast computation time is the primary advantage over traditional algorithms whose runtime must constantly increase as the LEO constellation scales to thousands of nodes ($N \to \infty$).

# 5 Results and discussion

## 5.1 Simulation network model

A feature vector $x_f(f = 0,1 \cdots N-1)$ represents the network properties of each node. The equation $x_t = [q_t, v_t']$ signifies the volume of traffic flow given by $q_t$. The aggregated attributes of the edges associated with this node, including details like link bandwidth and delay, are shown by $v_t'$. The following is the expression for the system feature matrix X of the entire network configuration:

$$X = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix} \quad (31)$$

Distributing servers to other locations makes use of the same data as the Starlink base station. One of the user endpoints is randomly selected as the source node. One of the distant servers is picked at random to be the final destination node. The data flow dimension is lognormally distributed and falls between 5 and 100 Mb. Our 1000 simulation runs proved that the suggested algorithm worked as expected. The techniques were coded in Python with the latest version being 3.13.2. We find the key parameters for the simulation in Table 1.

Table 1: An overview of the parameters used in the simulation.

| **Parameters** | **Values** |
|---|---|
| Count of satellites | 6051 |
| Time taken for the Simulation | 5450 s |
| Steps in Simulation | 62 s |
| Satellite orbit height | 554 km |
| Count of orbit planes | 85 |
| Count of satellites for single plane | 71 |
| Orbit Inclination | 55∘ |
| Orbit Eccentricity | 0 |
| Phase factor | 3 |
| Count of ground stations | 120 |
| Count of function nodes | [3, 4, 5] |

| | |
|---|---|
| Count of user terminals | [2100, 4200, 6050, 8100] |
| Minimal elevation angle of base station | 40∘ |
| Minimal elevation angle of user device | 20∘ |
| Central frequency | 30 GHz(UL), 40 GHz(DL) |
| Satellite Transmitter gain | 40 dBi |
| EIRP density of Satellite | 4 dBW/MHz |
| Capacity | 260 MHz(UL), 63.2 (DL) |
| Satellite G/T | 14 dB $K^{-1}$ |
| Satellite Receiver gain | 39.5 dBi |
| Receive antenna gain for user endpoints | 41.2 dBi |
| Figure showing user endpoint noise | 1.3 dB |
| Temperature of the user's endpoint antenna | 152 K |
| The room's temperature | 295 K |
| Antenna gain for the user endpoint Tx | 41.4 dBi |
| Transfer power for the user endpoint Tx | 2W (34 dBm) |
| | |

**Proximal Policy Optimization (PPO) DRL algorithm**

We use the PPO with DRL algorithm for its stability and on-policy learning efficiency.

**Hyperparameters:**

- Learning Rate (Actor/Critic): $1 \times 10^{-4}$
- Optimizer: Adam
- Discount Factor ( $\boldsymbol{\gamma}$ ): 0.99
- Exploration Strategy: Gaussian noise is added to the action output during training, standard for PPO.
- Training Episodes: 50,000 total episodes.
- Convergence Criteria: Training stops if the average episodic return's 50-episode moving average does not increase by more than 0.5% over 500 episodes.

## 5.2 Performance comparison

The theortical comparison between the DRL-based ResNet and Graph Neural Network Routing and Conventional Routing (e.g., SPF or Dijkstra) is given in Table 2.

Table 2: Advantages and Trade-Offs

| Feature | DRL-based ResNet and Graph Neural Network Routing | Conventional Routing (e.g., SPF or Dijkstra) |
|---|---|---|
| **Adaptivity** | Very high. Adapts in real-time to factors such as traffic volume, queue congestion, and network failures (load-sensitive or dynamic routing). | Very low. No matter how much traffic there is, static/delay-only routing only looks at the delay or number of hops. |
| **Performance** | Best in situations with a lot of foot traffic. Much improved network performance and much reduced total latency. | Inadequate during heavy usage because of the development of hotspots and the resulting queue delay. |
| **Scalability** | Very well done. Reduced need for specialized local knowledge is a benefit of distributed multi-agent systems. With GNN, features may be efficiently extracted from massive graphs. | Very bad. The most efficient central computation necessitates worldwide, real-time topology revisions, which causes a great deal of signaling clutter. |
| **Overhead** | Great difficulty in training and inferring. The system of neural networks needs robust hardware (or well efficient integrated systems) to function in real-time. | Little Computing Requirement. Easy path-cost computation, but heavy global state transmission overhead. |
| **Convergence** | Learn the policy using rigorous offline simulation, and then keep it up-to-date through online/continuous learning. | Rapid route planning using up-to-the-minute topographic data |

We tested all three of these approaches and compared their results. In this comparison, we will be looking at the following metrics: average execution duration, average complete path delay, average network bandwidth, and the mean traffic access success rate.

- **Traffic access success rate:** We can use this formula to find out what percentage of traffic satisfies the SFC constraint communication path compared to the entire traffic:

$$R_{\text{suc}} = \frac{N_{\text{SFC}}}{N_{\text{total}}} \quad (32)$$

The amount of traffic that meets the SFC constraint transmission path is represented by $N_{SPC}$, whereas the overall quantity of traffic is represented by $N_{\text{fotal}}$ .

- **Average network load:** Each node's burden on the ground and in orbit took an average.
- **Average end-to-end path delay:** All the transmission networks that satisfy the SFC requirement have an average latency.
- **Average running time:** The proportion of the total volume of traffic that is represented by the amount of time it takes for the technique to finish calculating all of the data transfer paths for the traffic.

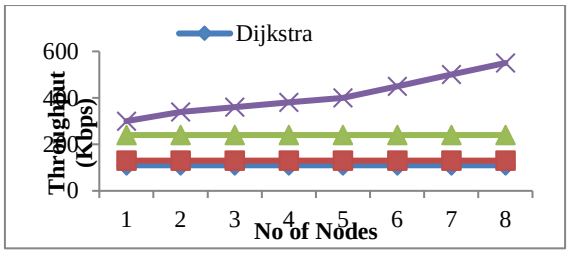


Figure 4: Throughput vs. No of nodes

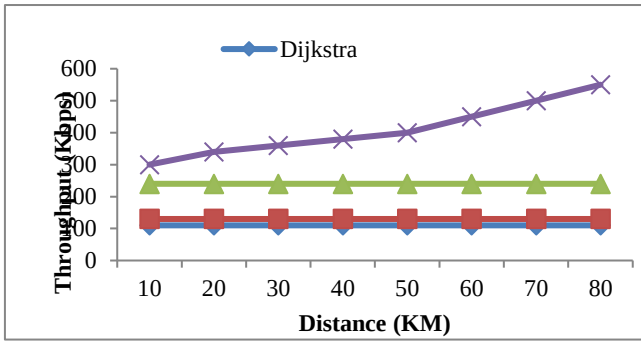


Figure 5: Throughput vs. distance

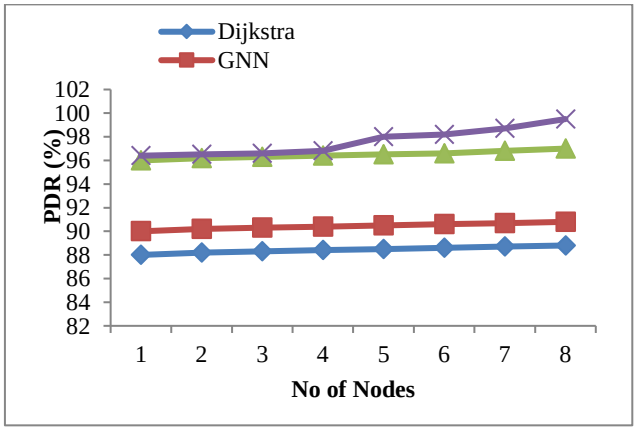


Figure 6: PDR vs. nodes

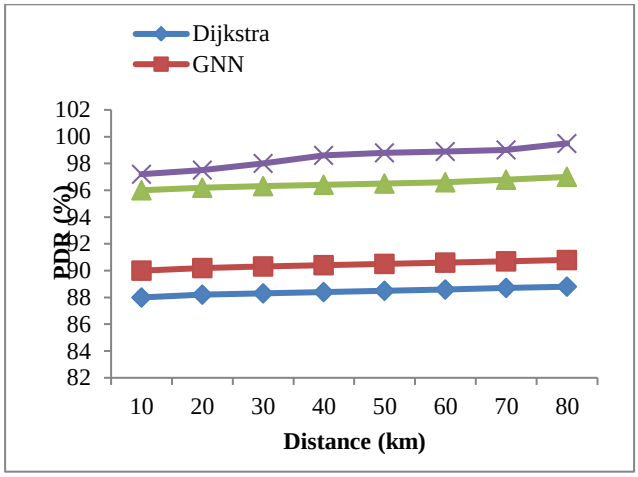


Figure 7: PDR vs. distance

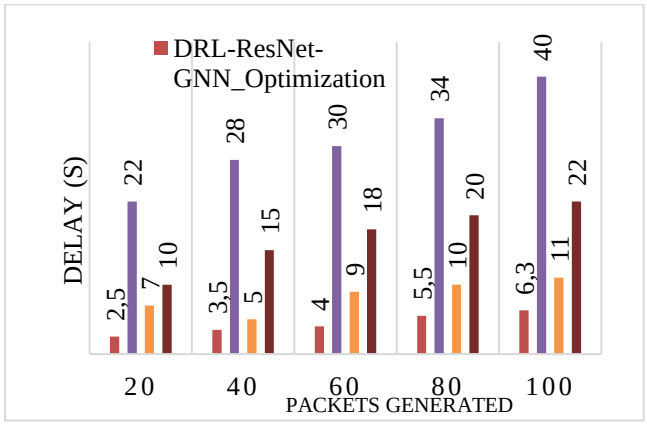


Figure 8: Delay vs. packets generated

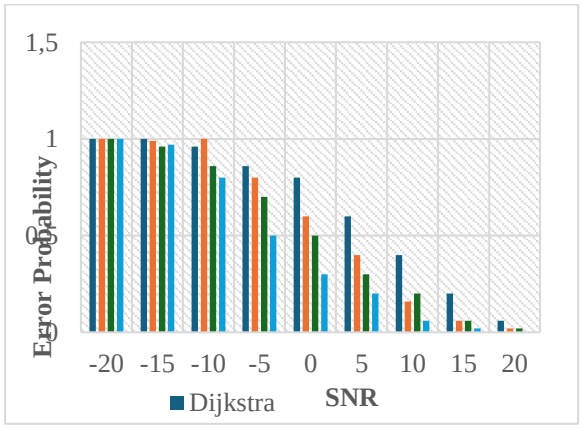


Figure 9: Error probability vs, SNR

Figure 4 shows the throughput versus the number of nodes, fig 5 illustrates the relationship between distance and throughput. Fig 6 illustrates the relationship between nodes and PDR. Fig 7 illustrates the relationship between distance and PDR. Fig 8 illustrates the relationship between the number of packets generated and the delay. Fig 9 illustrates the relationship between error probability and SNR. Table 3 shows the summaries of the performance.

Table 3: Performance summary

| Method/ Performance | Dijkstra | GNN | DQN | Multi-criteria | Geographical | DRL-ResNet-GNN_Optimization |
|---|---|---|---|---|---|---|
| PLR | 21.2% | 36.3% | 23.6% | 25.6% | 19.4% | 13.8% |
| Delay (s) | 2.72 | 2.96 | 2.75 | 2.54 | 2.94 | 2.31 |
| Network Load | | | | | | |
| Average running time (s) | 10.2 | 10.8 | 11.2 | 12.45 | 13.45 | 9.25 |

When dealing with heavy traffic or a constantly changing topology, you will notice the biggest improvements in Total Delay and Network Throughput.

**Table.4. Evaluation Measures Comparison**

| Evaluation Measures | Standard or Static Routing (e.g., Dijkstra) | DRL/GNN-driven Routing (Close to DRL-ResNet) | Improvement in Relationship |
|---|---|---|---|
| Average Total Delay | From 70−120 ms | From 55−80 ms | Between 30 and 40% |
| Average Network Throughput | From 75−100 Mbps | From 75−130 Mbps | Between 25 and 30% |

The advantages of combining a DRL framework with a deep learning architecture are demonstrated by these computational findings. With the use of a GNN, which could include ResNet principles, the routing agent can learn dynamic, intricate interactions inside the LEO network graph. As a result of this integration, the general efficiency of the network is improved.

1. **Delay reduction:** There have been reports that DRL-GNN techniques greatly enhance network performance. In comparison to the shorter-path Dijkstra method, they can cut typical end-to-end delays by as much as 38.74%.
2. **Throughput increase:** Net throughput as a whole is much enhanced by these smart routing strategies. To be more precise, as contrasted with Dijkstra and a simple DRL (DQN) routing agent, their throughput increases can reach 30.23% and 19.12%, respectively.

With the help of the GNN, the DRL agent can factor in demand and congestion data from the entire network when deciding which hop to take next. Having this functionality enhances the overall network's routing decisions. Because it solely takes hop count or static delayed propagation into account, a basic Dijkstra algorithm is unable to accomplish this. Similarly, a simple DRL agent has its limitations due to its exclusive focus on local queue duration. In order to optimize the routing of low-orbit satellite connections, this research employs a graph neural network-based approach. For low-orbit satellite networks, a technique based on graph neural systems is developed to adapt to networks that undergo recurrent topological alterations. The issue of network congestion is another target. This study proposes the GNN-DRL-RESNET-OPTIMIZATION technique, which applies a GNN feature construction model to the network's graph data in order to improve the routing of the network. In order to represent its own nodes, it uses GNNs to create hidden vector feature representations. A completely distributed agents DRL routing model subsequently makes use of these representations. A method for deep reinforcement learning is used to decide on the routes. The paper's suggested GNN-DRL-RESNET-OPTIMIZATION method outperforms Dijkstra's approach in terms of general network throughput. It slows down the average end-to-end time as well. In a similar vein, the GNN-DRL-RESNET-OPTIMIZATION method outperforms the classic DQN algorithm. When compared to Dijkstra, the average throughput of GNN-DRL-RESNET-OPTIMIZATION is 29.47% higher. When compared to DQN, it shows a rise of 18.42%. When contrasted with Dijkstra, the average total delay is 397.6 percent lower. There is a 15.29% decrease when compared to DQN. The GNN-DRL-RESNET-OPTIMIZATION model is also more suited to actual networks since it can handle topological modifications like traffic fluctuations, connection failures, and node malfunctions.

# 6 Conclusion

For satellite networks operating in low Earth orbit, this study applies a graph neural network-based routing optimizer technique. Modular algorithms based on graph neural networks are developed for low-orbit satellite connections to accommodate networks that undergo frequent topological changes. Its secondary objective is to alleviate the issue of overloaded networks. Using a GNN feature modelling to improve network routing is the goal of the GNN-DRL-RESNET-OPTIMIZATION technique presented in this research. The program uses GNNs to

create hidden feature vector models of its own nodes. A completely decentralized distributed DRL routing architecture subsequently takes advantage of these characterizations. An algorithm based on DRL determines the routes. In comparison to Dijkstra's algorithm and the conventional DQN technique, the suggested GNN-DRL-RESNET-OPTIMIZATION algorithm decreases average overall latency while simultaneously increasing total network performance.

## Future work

Centralized management or pre-calculated pathways are the backbone of traditional LEO network routing, leaving them open to external threats, sluggish authentication, and potential single points of failure. To control the network, blockchain adds a decentralized, verifiable layer. The unchangeable distributed ledger of the blockchain verifies the identification of users and new satellites whenever they need to join the network or exchange satellites, which happens often in low Earth orbit (LEO). This makes authentication more secure and expedites future considerations by doing away with the necessity for a central Certificate Authority (CA).

## References


[1] Shi, Y., Yuan, Z., Zhu, X., & Zhu, H. (2023). An Adaptive Routing Algorithm for Inter-Satellite Networks Based on the Combination of Multipath Transmission and Q-Learning. Proces**ses**. https://doi.org/10.3390/pr11010167

[2] Iskandar, M.I., & Asvial, M. (2025). Development of Physarum Routing Algorithm in Low Earth Orbit Satellite Network. 2025 International Seminar on Intelligent Technology and Its Applications (ISITIA), 397-402. DOI:10.1109/ISITIA66279.2025.11137520

[3] Jin, J., Shang, L., Yang, Z., Wang, H., & Li, G. (2024). A Local Pre-Rerouting Algorithm to Combat Sun Outage for Inter-Satellite Links in Low Earth Orbit Satellite Networks. Applied Scienc**es**. https://doi.org/10.3390/app14041625

[4] Hu, H., lv, S., He, J., & Feng, S. (2024). A distributed on-demand routing algorithm for large-scale low Earth orbit constellation. International Conference on Algorithms, Microchips and Network Applications. DOI:10.1117/12.3031951

[5] Wang, L., Xu, Z., Zhi, R., & Wang, J. (2024). Adaptive Load Balancing Routing Algorithm for Low Earth Orbit Satellite Cluster Networks. 2024 9th International Conference on Computer and Communication Systems (ICCCS), 666-671. DOI:10.1002/itl2.70031

[6] Hou, C., & Zhu, Y. (2023). The QoS Guaranteed Routing Strategy in Low Earth Orbit Satellite Constellations. 2023 IEEE/CIC International Conference on Communications in China (ICCC Workshops), 1-6. DOI:10.1109/ICCCWorkshops57813.2023.10233775

[7] Zhang, B., & Yang, Z. (2024). The Shortest Path Algorithm Based on Geometric Symmetry for Low Earth Orbit Satellite Network. 2024 5th Information Communication Technologies Conference (ICTC), 254-263. DOI:10.1109/ICTC61510.2024.10601677

[8] Wang, C., & Luo, Z. (2024). A DRL-Based Dynamic Resource Allocation and Task Offloading Algorithm for LEO Satellite Network. 2024 International Conference on Satellite Internet (SAT-NET), 59-65. DOI:10.1109/TVT.2025.3549119

[9] He, C., Zhang, Y., Ke, J., Yao, M., & Chen, C. (2024). Digital Twin Technology-Based Networking Solution in Low Earth Orbit Satellite Constellations. Electronics. https://doi.org/10.3390/electronics13071260

[10] Liu, X., Zhang, Z., & Yang, Y. (2025). A Multipath Routing Algorithm Based on Data Replication for Low Earth Orbit Satellite Networks. Int. J. Inf. Syst. Model. Des., 16, 1-20. https://doi.org/10.4018/IJISMD.373198

[11] Wang, F., Yao, H., He, W., Chang, H., Xin, X., & Guo, S. (2024). Time-Sensitive Scheduling Mechanism Based on End-to-End Collaborative Latency Tolerance for Low-Earth-Orbit Satellite Networks. IEEE Transactions on Network Science and Engineering, 11, 5149-5162. 10.1109/TNSE.2023.3342938

[12] Yuan, S., Sun, Y., & Peng, M. (2023). Joint Network Function Placement and Routing Optimization in Dynamic Software-Defined Satellite-Terrestrial Integrated Networks. IEEE Transactions on Wireless Communications, 23, 5172-5186. https://doi.org/10.48550/arXiv.2310.13940

[13] Zhao, J., & Pan, J. (2024). Low-Latency Live Video Streaming over a Low-Earth-Orbit Satellite Network with DASH. Proceedings of the 15th ACM Multimedia Systems Conference. DOI:10.1145/3625468.3647616

[14] Roth, M.M., Brandt, H., & Bischl, H. (2022). Distributed SDN-based Load-balanced Routing for Low Earth Orbit Satellite Constellation Networks. 2022 11th Advanced Satellite Multimedia Systems Conference and the 17th Signal Processing for Space Communications Workshop (ASMS/SPSC), 1-8. DOI:10.48550/arXiv.2209.05984

[15] Bhattacharjee, D., Madoery, P.G., Chaudhry, A.U., Yanikomeroglu, H., Kurt, G.K., Hu, P., Ahmed, K., & Martel, S. (2024). On-Demand Routing in LEO Mega-Constellations With Dynamic Laser Inter-Satellite Links. IEEE Transactions on Aerospace and

Electronic Systems, 60, 7089-7105. DOI: 10.1109/TAES.2024.3415571

[16] Xie, X., Huang, L., Tang, C., & Ning, Q. (2023). Multi-objective routing algorithms for low-earth orbit satellite network. International Journal of Satellite Communications and Networking, 41, 427 - 440. DOI: 10.1109/TAES.2024.3415571

[17] Chen, C., Liao, Y., & Chen, J. (2024). Congestion Avoidance Geographic Routing in a Large-Scale Multiple Shell Low Earth Orbit Satellite Constellation. 2024 10th International Conference on Applied System Innovation (ICASI), 383-385. DOI:10.1109/ICASI60819.2024.10547783

[18] Shake, T.H., Sun, J., ThomasC.Royster, I., & Narula-Tam, A. (2022). Failure Resilience in Proliferated Low Earth Orbit Satellite Network Topologies. MILCOM 2022 - 2022 IEEE Military Communications Conference (MILCOM), 828-834. DOI:10.1109/MILCOM55135.2022.10017632

[19] Cheng, H.T. (2022). Research on equalization algorithm of routing jump for low-orbit micro-satellites. 2022 IEEE 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), 10, 1202-1207. https://doi.org/10.3390/fi14070207

[20] Chu, J., & Chen, X. (2023). Robust precoding design for inter-satellite cooperation-based low-earth orbit satellite Internet of Things. DOI:10.1109/JIOT.2021.3055776

[21] Wang, M., Wei, L., Wang, Y., & Liu, Y. (2023). Orbit-Grid-Based Dynamic Routing for Software Defined Mega-Constellation Network. *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, 844-849. DOI:10.1109/GLOBECOM54140.2023.10437440

[22] Wang, X., Li, W., Han, S., Yang, M., & Jiang, Z. (2023). Enabling High-Connectivity LEO Satellite Networks Via Encountering Inter-Satellite Links. *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, 4883-4889. DOI:10.1186/s13677-025-00808-y

[23] Liao, X., Liu, J., Man, O., Du, J., Zhao, Y., & Zhang, R. (2025). DISNR: A Low-Overhead Dynamic Routing Protocol for Large-Scale LEO Satellite Networks. IEEE INFOCOM 2025 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 1-6. DOI:10.1109/INFOCOMWKSHPS65812.2025.11152774

[24] Wang, Y., Zhang, Q., Qiu, K., & Gao, Y. (2025). Stabilizing and Optimizing Inter-Shell Routing in LEO Networks with Integrated Routing Cost. *2025 IEEE/CIC International Conference on Communications in China (ICCC Workshops)*, 1-6. DOI:10.1109/ICCCWorkshops67136.2025.11148095

[25] Ning, Y., Yi, L., Zhao, Y., Qi, K., Wang, H., Rahman, S., & Zhang, J. (2023). Load-balancing routing algorithms for service congestion avoidance in LEO optical satellite networks. *Journal of Optical Communications and Networking*, *15*, 1038-1049. DOI:10.1364/JOCN.489919

[26] Wang, K., Zhang, J., Zheng, S., Wang, P., Zhang, X., & Evans, B. (2023). An Intelligent Area-segmentation Enabled Hybrid Routing Method in Mega-constellations. 2023 IEEE Globecom Workshops (GC Wkshps), 443-448. DOI:10.1109/GCWkshps58843.2023.10465201

[27] Jiao, J., Yang, P., Du, Z., Wang, Y., & Zhang, Q. (2024). Clustered Multi-Criteria Routing Algorithm for Mega Low Earth Orbit Satellite Constellations. *IEEE Transactions on Vehicular Technology*, *73*, 13790-13803. DOI:10.1109/TVT.2024.3396350

[28] Mao, B., Zhou, X., Liu, J., & Kato, N. (2024). On an Intelligent Hierarchical Routing Strategy for Ultra-Dense Free Space Optical Low Earth Orbit Satellite Networks. IEEE Journal on Selected Areas in Communications, 42, 1219-1230. DOI:10.1109/JSAC.2024.3365880

[29] Cheng, P., Du, P., Zhang, Y., Dong, M., Liang, Y., & Zhu, Y. (2024). Inter-Satellite Routing Algorithms for Dual-Layer Low Earth Orbit Satellite Internet. *2024 8th International Conference on Communication and Information Systems (ICCIS)*, 89-95. DOI:10.1109/ICCIS63642.2024.10779428

[30] Kedrowitsch, A., Black, J., & Yao, D. (2024). Resilient Routing for Low Earth Orbit Mega-Constellation Networks. Proceedings 2024 Workshop on Security of Space and Satellite Systems. https://doi.org/10.3390/s25041232

[31] Sun, S., Zhang, R., Liu, K., Sun, Z., Tang, Q., & Huang, T. (2025). LMSR: A Low-Jitter Multiple Slots Routing Algorithm in LEO Satellite Networks. 2025 IEEE Wireless Communications and Networking Conference (WCNC), 1-6. DOI:10.1109/WCNC61545.2025.10978608

[32] Chen, X., Ji, Z., Wu, S., Jia, H., Xiao, A., & Jiang, C. (2025). A Distributed Routing Algorithm for LEO Satellite Networks: A Multiagent Transformer-MIX Learning Approach. IEEE Internet of Things Journal, 12, 15748-15763. DOI:10.1109/JIOT.2025.3530919

[33] Wang, Y., Zhu, Z., Wu, K., Hou, Y., He, H., & Yang, J. (2025). Spatio-Temporal Correlated Network State Prediction and Dynamic Routing for Satellite Networks. 2025 IEEE Wireless Communications and Networking Conference (WCNC), 1-7. DOI:10.1109/WCNC61545.2025.10978270

[34] Xiang, J., He, X., Zhao, Y., Xie, Z., & Liang, X. (2025). Distributed Dynamic Routing for LEO Satellite Networks With Temporal Graph Convolutions and Imitation Acceleration. IEEE

Communications Letters, 29, 2521-2525. DOI:10.1109/LCOMM.2025.3601011

[35] Xia, L., Lin, B., Zhao, S., & Zhao, Y. (2025). A Centralized–Distributed Joint Routing Algorithm for LEO Satellite Constellations Based on Multi-Agent Reinforcement Learning. Applied Sciences. https://doi.org/10.3390/app15094664