Distributed Dynamic Routing for LEO Satellite Networks With Temporal Graph Convolutions and Imitation Acceleration

Junyan Xiang, Xiaohe He, Yu Zhao, Zhuochen Xie, and Xuwen Liang

***Abstract*— Large-scale Low Earth Orbit (LEO) constellations offer promising solutions for global network access with low latency and high-speed transmission. However, their dynamic topology poses challenges for routing. Traditional centralized routing schemes enable global optimization but entail high overhead and computational complexity, making them unsuitable for large-scale LEO networks. We propose an algorithm named MATGCIR that leverages Multi-Agent deep reinforcement learning and Temporal Graph Convolution for distributed Intelligent Routing, extracting dynamic topology features and incorporating spatiotemporal information. A pretraining mechanism using shortest path rules enhances training efficiency and decision quality. Simulation results demonstrate that MATGCIR effectively addresses LEO network complexities and accelerates the training process for large-scale constellations.**

***Index Terms*— LEO satellite constellations, dynamic routing, multi-agent reinforcement learning, imitation learning.**

I. Introduction

LEO satellite systems, such as SpaceX's Starlink and the Iridium constellation, offer global low-latency, high-throughput communications. As 6G expands into remote and maritime areas, LEO satellites provide cost-effective alternatives to terrestrial networks [1], [2]. However, their dynamic topologies pose challenges to traditional routing, increasing latency and reducing throughput. Centralized methods struggle with real-time demands, while distributed ones often lack the reliability required for global positioning. This highlights the need for adaptive routing to maintain efficient and reliable communication amid frequent topology changes.

Existing routing algorithms for large-scale satellite constellations are broadly categorized into centralized and distributed approaches. Centralized optimization methods, such as Yuan et al. [3], extend software-defined satellite-terrestrial networks by jointly optimizing virtual network function placement and routing through time-evolving graph models, achieving lower latency and higher resource utilization. However, scalability remains a major challenge. To improve stability under frequent topology changes, Li et al. [4] proposed a hierarchical orbital-geodetic routing method that organizes LEO networks into structured domains. In contrast, distributed approaches focus on reducing computational complexity. Han et al. [5] introduced a distributed routing framework, and Feng et al. [6] further advanced this direction with a low-complexity satellite-terrestrial cooperative routing method, leveraging real-time position-based graphs to reduce delay and increase throughput.

Reinforcement learning (RL) has shown strong potential for complex optimization in dynamic networks [7]. However, single-agent RL in satellite constellations often suffers from slow convergence and low efficiency. Wei et al. [8] proposed Iris, a centralized deep reinforcement learning framework that improves adaptability through dynamic reward shaping, but centralized designs still face inherent scalability limitations. To address this, multi-agent reinforcement learning (MARL) has been adopted, enabling distributed decision-making and better adaptability. Xu et al. [9] proposed a spatial location-based MARL routing method, but its lack of spatiotemporal feature modeling limits adaptability to dynamic network changes.

Despite these advances, MARL-based distributed routing still face challenges such as local optimization and training inefficiencies. This letter makes the following contributions:

1) **Cooperative Routing and Load Balancing Optimization**: We propose a MAPPO-based routing framework under a centralized training with decentralized execution (CTDE) paradigm, improving load balancing, adaptability, and network stability in LEO satellite environments.
2) **Spatiotemporal Feature Extraction**: Temporal Graph Convolutional Networks (TGCN) are integrated with node embeddings to enhance spatiotemporal modeling and improve routing robustness in dynamic topologies.
3) **Accelerated Training with Imitation Learning**:To accelerate training in large-scale networks, we pre-train the model using expert policies via imitation learning, reducing early exploration and enabling faster convergence with higher-quality initial decisions.

II. System Model and Problem Formulation

A. System Model

This study focuses on distributed routing, modeling the network as a system where each node makes independent routing decisions, unaffected by global control. To investigate the adaptability of this approach, we consider a highly dynamic polar orbit configuration with N_L orbital planes, each containing M_L satellites. Each satellite maintains four inter-satellite links (ISLs): two stable intra-plane links $(\mathcal{E}a, L)$ and two dynamic inter-plane links $(\mathcal{E}c, L)$, which are available only when adjacent satellites are within communication range. Ground stations are connected via feeder links that are established under line-of-sight (LoS) visibility constraints. These time-varying connections are encoded in the dynamic

Received 19 May 2025; revised 7 July 2025; accepted 13 August 2025. Date of publication 20 August 2025; date of current version 13 November 2025. This work was supported by the National Key Research and Development Program of China under Grant 2024YFE0200203. The associate editor coordinating the review of this letter and approving it for publication was J. Ben Othman. *(Corresponding authors: Xuwen Liang; Zhuochen Xie.)*

The authors are with the Innovation Academy for Microsatellites of Chinese Academy of Sciences, Shanghai 201304, China, also with the Key Laboratory for Satellite Digitalization Technology, Chinese Academy of Sciences, Shanghai 201210, China, also with the University of Chinese Academy of Sciences, Beijing 100039, China, and also with ShanghaiTech University, Shanghai 201210, China (e-mail: xiangjy2022@shanghaitech.edu.cn; hexh@shanghaitech.edu.cn; zhaoyu2023@shanghaitech.edu.cn; liangxw@shanghaitech.edu.cn; xiezc.ac@hotmail.com).

Digital Object Identifier 10.1109/LCOMM.2025.3601011

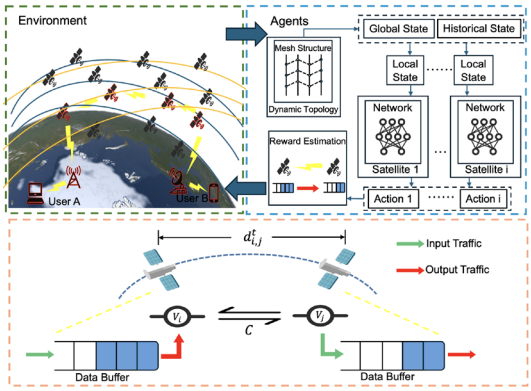


Fig. 1. Architecture of distributed routing system and transmission link model in large-scale low Earth orbit satellite networks.

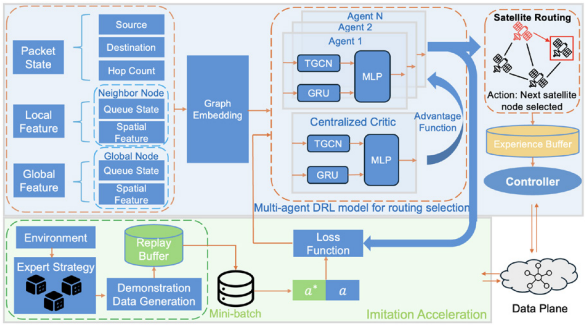


Fig. 2. Multi-agent DRL model for routing selection.

adjacency matrix $\mathcal{E}L = \mathcal{E}a, L \cup \mathcal{E}_{c,L}$ of the directed graph $\mathcal{G}_L = (\mathcal{V}_L, \mathcal{E}_L)$, where $\mathcal{V}_L$ denotes the set of satellite nodes.

B. Problem Formulation

In this study, we aim to optimize the overall transmission performance across the satellite network, considering dynamic topology changes and evolving traffic loads. Specifically, the objective is to minimize the cumulative end-to-end transmission cost for all data packets generated within a given time window, rather than focusing on a single source-destination pair. The communication model incorporates two critical delay components: *transmission delay* and *queuing delay*, which are jointly integrated into a unified routing cost function. Routing decisions are made at each satellite node in a decentralized manner, based solely on local observations, without requiring pre-computation of complete paths. This design enhances scalability and adaptability in large-scale satellite constellations. The delay models are detailed as follows:

1) Transmission Delay Model: At each time slot t, the transmission delay between satellite nodes includes two stages: (i) processing delay for sending and receiving operations, and (ii) propagation delay over the inter-satellite link. Given a data packet consisting of N_d bits, the processing delay is expressed as:$t'_{p,ij,t} = \frac{2\times N_d}{C}$, where C represents the channel capacity.

The propagation delay between satellite $v_{i,t}$ and $v_{j,t}$ depends on their spatial distance: $t''_{p,ij,t} = \frac{d^t_{i,j}}{v_c} = \frac{\|v^t_i - v^t_j\|}{v_c}$ where v_c denotes the speed of signal propagation in free space.

Thus, the total transmission delay along a packet's path is:

$$t_1 = \sum_{i=1}^{H} \left(t'_{p,ij,t} + t''_{p,ij,t}\right) \tag{1}$$

where H denotes the total number of hops, dynamically determined as packets traverse the network.

2) Queuing Delay Model: The queuing delay captures the impact of local buffer congestion at each node. Assuming that each satellite maintains a buffer operating under the First-In-First-Out (FIFO) policy, the queuing delay at node i is given by:$t_{q,i} = \frac{q^t_i}{C}$ where q^t_i represents the total queued data length at time t. Accordingly, the total queuing delay encountered along a routing path is:

$$t_2 = \sum_{i=1}^{H} t_{q,i} \tag{2}$$

3) Overall Cost Function: Taking into account both transmission and queuing delays, the cumulative routing cost across all packets during the scheduling period can be formulated as:

$$\min \text{cost} = \sum_{\text{packets}} \sum_{i=1}^{H} \left(\omega_1 (t'_{p,ij,t} + t''_{p,ij,t}) + \omega_2 t_{q,i}\right) \tag{3}$$

where ω_1 and ω_2 are tunable weighting factors that balance the influence of transmission and queuing delays in the cost evaluation. The values of ω_1 and ω_2 can be adjusted based on network characteristics and performance goals, such as prioritizing real-time responsiveness or maximizing throughput under congestion. By jointly optimizing these two delay components, routing decisions minimize end-to-end latency and promote efficient traffic distribution, implicitly supporting load balancing and congestion mitigation.

III. Methodology

The MATGCIR framework combines multi-agent deep reinforcement learning for optimized routing and imitation learning for accelerated convergence, enhancing efficiency and training speed in large-scale LEO satellite networks.

A. Graph Embedding and Temporal Features

To model satellite network dynamics and structure, we integrate node embedding and temporal feature extraction.

1) Node Embedding with Node2Vec: To capture the structural properties of the satellite network, we use Node2Vec [10], which maps the network topology into low-dimensional vectors. The input is the network's adjacency graph, while the output is a four-dimensional vector for each node. Node2Vec employs biased random walks to extract both local and global features, training a Skip-Gram model to produce embeddings that reflect the network structure. To adapt to dynamic environments, we update embeddings incrementally by using the previous step's embeddings as initialization, adjusting only for connectivity changes. The resulting embeddings serve as compact inputs for routing decision modules, maintaining structural consistency despite frequent topology changes.

2) Spatial-Temporal Modeling with TGCN: We discretize time into fixed intervals and model the satellite network as a Discrete-Time Dynamic Graph $G_{t_i} = (V, E_{t_i})$, with node features X_{t_i} containing queue length, bandwidth, and orbit information. To capture spatial dependencies, we adopt a GCN module, where $\hat{A}_{t_i}$ is the normalized adjacency matrix:

$$f(X, A_{t_i}) = \sigma\left(\hat{A}_{t_i} \text{ReLU}\left(\hat{A}_{t_i} X_{t_i} W_0\right) W_1\right) \tag{4}$$

Algorithm 1 MATGCIR With Imitation Acceleration

Require: N agents, environment $\mathcal{E}$, demonstration buffer D, initial policy θ, critic ϕ, learning rate α, discount factor γ, clip parameter ϵ, pre-training steps T_D
Ensure: Trained policy parameters θ_i for each agent

```
Initialize replay buffer D and actor π_{θ_i} with θ_i for each agent
Imitation Learning Pre-training Phase:
for I = 1 to T_D do
    for each time step t, agent i do
        Observe state s_t^i, compute masked action probabilities P_i
        Select expert-guided action a_t^i ~ P_i, execute a_t^i
        Store transition (s_t^i, a_t^i, r_t^i, s_{t+1}^i) in D
    end for
    Sample batch from D, compute loss L_1(θ), update θ_i
end for
Reinforcement Learning Fine-tuning Phase:
Initialize centralized critic Q with parameters φ
while not converged do
    Reset environment E
    for each time step t, agent i do
        Observe state o_i, compute masked action probabilities P_i with action mask m_i
        Select and execute action a_i ~ P_i, store (o_i, a_i, r_i, o_i)
    end for
    for each agent i do
        Compute advantage estimate Â_i using Q
        Compute surrogate loss and update θ_i via clipped objective L^CLIP(θ_i)
    end for
    Update centralized critic parameters φ
end while
```

To model temporal correlations, we feed the graph-convolved features into a GRU, which captures the time evolution of link availability and network load. In LEO constellations, such dynamics are strongly influenced by the satellites' orbital positions, causing periodic variations in connectivity and transmission delay. By learning these temporal patterns, the TGCN module enables each agent to anticipate spatiotemporal fluctuations in connectivity and queue congestion, thus improving the robustness and foresight of routing decisions. future network conditions.

The integration of Node2Vec and TGCN enables robust modeling of spatial and temporal dynamics: Node2Vec efficiently updates node embeddings to accommodate link failures or reconfigurations, while TGCN captures temporal variations in features like bandwidth and orbital changes.

B. MARL Model for Routing Selection

To optimize routing in satellite networks, we propose a multi-agent reinforcement learning framework based on Proximal Policy Optimization (PPO), which stabilizes policy updates under dynamic conditions. As shown in Figure 2, PPO agents in the Routing Module interact with the environment through defined action and state spaces, guided by path-state data from the Processing Module. Given each node's partial observability, the node selection problem is formulated as a Partially Observable Markov Decision Process (POMDP), enabling agents to make decisions based on local observations.

1) Observation and State Spaces: Each agent's observation O^i includes information on the local network topology g^i, buffer queue status q^i, adjacent link states l^i, global sub-satellite tracking tr^i, and packet status p^i (specifying source, current node, and destination). Thus, the observation for agent i is represented as $O^i = [g^i, q^i, l^i, tr^i, p^i]$. The global state S, which is essential for coordinated decision-making, aggregates observations from all agents, forming $S = [O^1, O^2, \ldots, O^m]$.

2) Action Space: Each satellite agent selects the next-hop from its four neighboring nodes with established inter-satellite links, forming its local action space A^i.

3) Reward Function: All agents share a joint reward $R(t)$ that reflects the global performance of the constellation, guiding policy optimization to enhance throughput, reduce congestion, and minimize delay. Immediate feedback after each hop selection improves training efficiency and strategy quality. During packet forwarding, the reward considers successful delivery, queuing conditions, and link utilization status. To balance resource usage and network stability, two additional mechanisms are integrated: a link utilization penalty and a packet loss penalty. Specifically, the link utilization deviation from the target value μ_{target} is penalized quadratically to prevent both congestion and resource underutilization, while packet loss events due to hop limit, buffer overflow, or link failure trigger explicit penalties to encourage route reliability. The complete reward function is defined as:

$$R(t) = \begin{cases} \omega_1 \left(1 - \dfrac{h}{H_{s,d}}\right), & \text{if } N_t = N_d \\ -\delta_{\text{drop}}, & \text{if a packet loss occurs} \\ -\omega_2 \log\left(1 + \frac{1-q_t}{t_{\text{trans}}}\right) & \\ \quad -\lambda(\mu_t - \mu_{\text{target}})^2, & \text{otherwise} \end{cases} \tag{5}$$

where h is the number of hops used, $H_{s,d}$ is the maximum allowed hops based on source-destination distance, q_t is the queue occupancy ratio, μ_t and μ_{target} are the current and target link utilization rates, t_{trans} is the transmission time, ω_1 and ω_2 are weighting factors, λ is the penalty coefficient for utilization deviation, and δ_{drop} is the penalty for packet loss.

Each agent uses an actor-critic model, as shown in Figure 2. The actor network operates in a decentralized manner, receiving (1) local network topology from a TGCN (agent's observation O_i) and (2) historical actions processed by a GRU, enabling decisions based on immediate surroundings and past experiences. A centralized critic network is shared among all agents and plays a key role in coordinating their actions. The critic network receives (1) global network topology extracted by a multi-layer TGCN covering node and link features, and (2) the local request status of the agent. By leveraging the global network perspective, the critic provides a shared value function that aligns the agents' actions toward a common objective. During training, the parameters of the actor and critic networks are updated collaboratively, ensuring consistent policy optimization across all agents. The actor network focuses on optimizing individual agent decisions, while the

critic provides feedback based on the global state, enabling agents to adapt to dynamic and distributed environments effectively.

C. Imitation Acceleration Scheme

In large-scale scenarios, distributed MARL demands substantial data and training time due to its trial-and-error nature, limiting its efficiency for large-scale optimization. To address this, we introduce an imitation acceleration scheme that combines imitation learning with DRL. By pretraining agents with expert demonstrations, imitation learning reduces exploration iterations and significantly speeds up the early training phase.

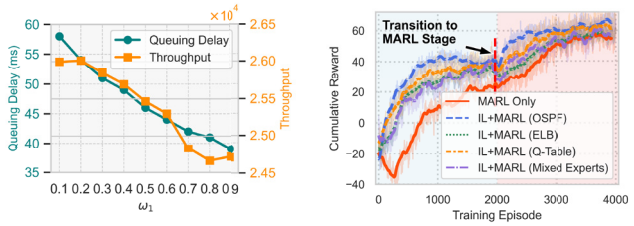
In our framework, we adopt three expert policies to generate supervision data for pretraining: the widely used Shortest Path First (OSPF) algorithm, based on Dijkstra's strategy and known for its simplicity and stability; an explicit load balancing (ELB) algorithm, which uses node queue lengths to evenly distribute traffic under dynamic conditions; and a Q-table-based method, a basic reinforcement learning approach with strong adaptability in small networks but requiring more environment interaction. For each network topology state, the corresponding expert decision is recorded to construct the expert dataset. In addition, we design a Mixed Experts scheme that integrates the three strategies into a unified dataset by randomly sampling expert transitions, aiming to encourage the model to reconcile conflicting or complementary decision patterns and learn a more generalizable and robust routing policy. We denote this method as IL+MARL (Mixed Experts).

The agent models are pretrained on the expert dataset by minimizing the following loss function: $L_1(\theta) = L_D(\theta) + \lambda\|\theta\|^2$, where $L_D(\theta) = \frac{1}{N}\sum_{i=1}^{N}|\pi_\theta(s_i) - a_i|^2$ denotes the supervised learning loss over demonstration data, and $|\theta|^2$ is an L2 regularization to improve model generalization. Upon completion of the offline pretraining phase, agents switch to online interaction using reinforcement learning. The pretrained policy serves only as an initialization to accelerate early convergence. During fine-tuning, the policy parameters are continuously updated based on environmental feedback, allowing the agents to adapt beyond expert behaviors. This ensures that rule-based efficiency from imitation learning is seamlessly integrated into a fully data-driven optimization process.

D. Complexity Analysis

In this subsection, we analyze the time complexity of the MATGCIR routing model, which consists of $\mathcal{L}_G$ TGCN layers followed by $\mathcal{L}_F$ fully connected layers. Each TGCN layer performs three gated graph convolutions, and the per-layer complexity per time step is given by: $\mathcal{O}(3kEd+3Nd^2)$, where d is the feature dimension, N is the number of nodes, E is the number of edges, and $k = 1$ is the convolution order. Over a time window of length T, the total TGCN complexity is: $\mathcal{O}\left(T\sum_{l=1}^{\mathcal{L}_G}(E_l d + Nd^2)\right)$. The FC module contributes: $\mathcal{O}\left(\sum_{l=0}^{\mathcal{L}_F-1} N\cdot\zeta_l\cdot\zeta_{l+1}\right)$, where ζ_l is the number of neurons in the l-th layer. Hence, the overall model complexity is:

$$\mathcal{O}\left(T\sum_{l=1}^{\mathcal{L}_G}(E_l d + Nd^2) + \sum_{l=0}^{\mathcal{L}_F-1} N\cdot\zeta_l\cdot\zeta_{l+1}\right) \tag{6}$$



(a) (b)

Fig. 3. (a) Impact of weight parameters ω_1 and ω_2 on throughput and delay. (b) Convergence of the MATGCIR, showing two training stages: imitation learning (blue background) followed by MARL training (red background).

For a 66-node network with $\mathcal{L}_G = 2$, $d = 64$, and FC structure $64 \rightarrow 512 \rightarrow 128 \rightarrow 4$, the total cost is approximately 23.8 MFLOPs. On the High Performance Spaceflight Computing (HPSC) platform with 256 GFLOPS peak performance, the theoretical inference time is about 0.093 ms, far below inter-satellite link delays (5)–10 ms), and thus introduces negligible overhead to real-time routing decisions.

IV. Performance Evaluation

A. Simulation Environment Settings

To realistically simulate the constellation environment, we consider scenarios where data packet requests are concentrated in specific regions. During network initialization, data packets are randomly generated and the traffic distribution during the simulation follows a Poisson process. To emulate the dynamic nature of real-world satellite networks, each link is assigned a 1% failure rate. Additionally, we implement a graded performance model, where each ISL is assigned one of six capacity levels (10, 5, 2.5, 1.25, 0.625 Gbps, and complete outage) to reflect realistic link variability.To account for polar orbit characteristics, inter-orbit links are temporarily disconnected when satellites enter polar regions and are restored upon exit. Table I summarizes the key experiment parameters.

B. Baseline

The proposed MATGCIR algorithm is compared against the following baselines: OSPF and ELB, introduced earlier in Section III-C, serve as traditional routing references. FDR-MARL [9] is a spatial location-aided, fully distributed routing algorithm based on MARL, utilizing DQN with fully connected neural networks for feature extraction.

C. Results and Analysis

This section presents simulation results evaluating MATGCIR's performance across five metrics: 1) Throughput, measuring total data transmitted successfully; 2) Average Latency, the mean packet delivery time; 3) Packet Loss Rate, the percentage of packets lost during transmission; 4) Load Balance Index, using Jain's Fairness Index [11] to assess resource allocation fairness. Near 1 values indicate balanced distribution; 5) Link utilization, measuring the proportion of available link capacity effectively used during data transmission.

We analyze the multi-objective cost function in Eq.3 with varying weight parameters. As shown in Fig. 3a, increasing

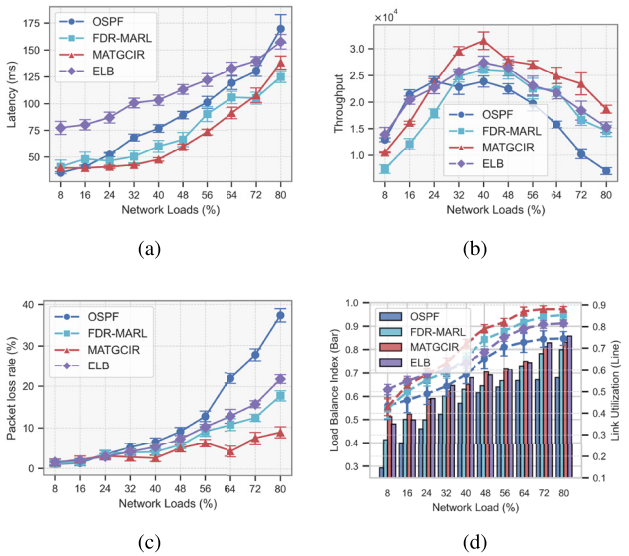


Fig. 4. Performance metrics under varying network loads: (a) delivery time, (b) throughput, (c) loss rate, and (d) load balance index and link utilization.

TABLE I
PARAMETER SETTINGS

Parameter	Value	Parameter	Value
Orbital Altitude	540 km	Ground Stations	20
Orbital Planes (N_L)	6	Satellites per Plane (M_L)	11
Time to live (H_{max})	20 hops	Weight factors (ω_1, ω_2)	0.6, 0.4
Actor TGCN Layers	2	Critic TGCN Layers	7
Dense Layer Size	[512,128,4]	Recurrent Layer Size	64
Learning Rate (α)	10^{-4}	Discount Factor (γ)	0.95
GAE (λ_{GAE})	0.95	Mini-batch Size	3

ω_1 enhances throughput by favoring high-capacity paths, while increasing ω_2 ($\omega_2 = 1 - \omega_1$) prioritizes paths with lower queuing delays, supporting low-latency transmission. Through Pareto analysis, we evaluated the trade-off between throughput and delay from a "benefit/cost" perspective. The results indicate that the optimal balance is achieved at $\omega_1 = 0.6$, which we select for subsequent experiments to maintain a reasonable compromise between throughput and delay.

Fig. 3b compares the training reward curves of MATGCIR under various expert strategies. The first half involves imitation learning from expert datasets, while the second half relies on reinforcement learning via environment interaction. The MARL-only model, trained solely through reinforcement, serves as the baseline. Imitation learning notably improves exploration efficiency, accelerates convergence, and enhances final policy performance. Among expert strategies, OSPF provides the fastest and most stable convergence. Q-table achieves similar final rewards but slower initial convergence. ELB performs well during imitation but degrades post-transition due to reward misalignment. Mixed Experts reach high final rewards but suffer from training instability caused by conflicting expert signals. Considering convergence efficiency, stability, and final performance, OSPF is selected for subsequent experiments.

As illustrated in Figs.4a and 4b, the proposed MATGCIR algorithm achieves the lowest end-to-end latency under varying network loads and the highest throughput under medium to high traffic. This is due to its routing policy that favors next-hop satellites with lower delays and lighter traffic. MATGCIR also shows small performance fluctuations across load scenarios, indicating strong consistency and robustness. As shown in Figs.4c and 4d, it delivers the lowest packet loss rate and favorable trade-offs between load balance and link utilization, enhancing overall resource efficiency. Under light traffic, MATGCIR prioritizes load balancing over aggressive latency minimization. While this slightly increases delay, it improves long-term stability and throughput. These results validate MATGCIR's ability to balance efficiency and reliability in dynamic LEO satellite networks.

V. CONCLUSION

This research presents MATGCIR, an online routing method using multi-agent deep reinforcement learning for large-scale satellite constellations. By combining centralized learning, distributed decision-making, dynamic modeling via TGCN, and imitation learning, MATGCIR achieves adaptive routing and efficient training. Simulations demonstrate its superior efficiency and scalability compared to existing methods.

REFERENCES

[1] L. Zhi et al., "Self-powered absorptive reconfigurable intelligent surfaces for securing satellite-terrestrial integrated networks," *China Commun.*, vol. 21, no. 9, pp. 276–291, Sep. 2024.

[2] C. Han, K. An, Z. Lin, S. Chatzinotas, and J. Wang, "Endogenous anti-jamming communications for SAGIN: A network perspective," *IEEE Netw.*, early access, Mar. 13, 2025, doi: 10.1109/MNET.2025.3551248.

[3] S. Yuan, Y. Sun, and M. Peng, "Joint network function placement and routing optimization in dynamic software-defined satellite-terrestrial integrated networks," *IEEE Trans. Wireless Commun.*, vol. 23, no. 5, pp. 5172–5186, May 2024.

[4] Y. Li et al., "Stable hierarchical routing for operational LEO networks," in *Proc. 30th Annu. Int. Conf. Mobile Comput. Netw.* New York, NY, USA: ACM, May 2024, pp. 296–311, doi: 10.1145/3636534.3649362.

[5] C. Han, A. Liu, L. Huo, H. Wang, and X. Liang, "Anti-jamming routing for Internet of Satellites: A reinforcement learning approach," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 2877–2881.

[6] X. Feng, Y. Sun, and M. Peng, "Distributed satellite-terrestrial cooperative routing strategy based on minimum hop-count analysis in mega LEO satellite constellation," *IEEE Trans. Mobile Comput.*, vol. 23, no. 11, pp. 10678–10693, Nov. 2024.

[7] Y. He, B. Sheng, H. Yin, D. Yan, and Y. Zhang, "Multi-objective deep reinforcement learning based time-frequency resource allocation for multi-beam satellite communications," *China Commun.*, vol. 19, no. 1, pp. 77–91, Jan. 2022.

[8] W. Wei et al., "Iris: Toward intelligent reliable routing for software-defined satellite networks," *IEEE Trans. Commun.*, vol. 73, no. 1, pp. 454–468, Jan. 2025.

[9] G. Xu, Y. Zhao, Y. Ran, R. Zhao, and J. Luo, "Spatial location aided fully-distributed dynamic routing for large-scale LEO satellite networks," *IEEE Commun. Lett.*, vol. 26, no. 12, pp. 3034–3038, Dec. 2022.

[10] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 855–864.

[11] R. K. Jain, D.-M. W. Chiu, and W. R. Hawe, "A quantitative measure of fairness and discrimination," Dept. Eastern Res. Lab., Digit. Equip. Corp., Hudson, MA, USA, Tech. Rep., 1984, vol. 21, p. 1.