An Effective Handover Strategy for Fast-moving Devices in LEO Satellite Network Based on Deep Reinforcement Learning

Yu-Hsuan Chu, Chuan-Wei Cho, and Meng-Shiuan Pan
Department of Electronic Engineering
National Taipei Univers
e-mail: {t111368093, t

***Abstract*—With the development of aerospace technology cost of deploying low Earth orbit (LEO) satellites is decrea LEO satellites can provide low latency and high throug networks on a global scale. Recently, LEO satellite netw are discussed to provide network services for user equipm (UEs) in airplanes. However, both airplanes and LEO sate are moving rapidly, resulting in dynamic changes to the net topology. This work proposes an efficient handover strategy b on deep reinforcement learning to achieve seamless net service in this scenario. In this approach, fast-moving de (e.g., airplanes) are treated as agents that learn to adap changes in the network environment. The designed Deep Network (DQN) model determines suitable handover tri timings and target satellites for these fast-moving devices, gu by comprehensive reward functions that considers multiple formance metrics. Simulation results indicate that, the prop scheme can significantly enhance network throughput and re latency.**

***Index Terms*—deep reinforcement learning, handover deci low earth orbit (LEO) satellite network, multi-agent**

I. Introduction

In recent years, low Earth orbit (LEO) satellite netw have attracted significant attention in both industry academia [1]. LEO satellites operate at altitudes ran from approximately 800 to 2000 km. Most LEO satel function in the Ku-band (12-18 GHz) and Ka-band (2 40 GHz), which enable real-time, high-throughput wir communication. Fig. 1 illustrates a LEO satellite networl this network, LEO satellites can be interconnected via i satellite links (ISLs). A device (or say user equipment) connect to a *host satellite* to access network services. host satellite relays data through ISLs and then connec the Internet through a ground station.

Relative to the ground, LEO satellites move at high spe Therefore, when connecting to a LEO satellite networ static user equipment (UE) must perform handovers pe ically. Recently, LEO satellite networks have been explore provide communication services for UEs in airplanes. Given the rapid movement of airplanes, the handover will frequently happen in this scenario. To minimize interruptions and enhance the quality of communication, designing an efficient handover mechanism is important in this network scenario.

Most previous handover schemes in LEO satellite networks are designed for user equipment (UE) on the ground. Typically,

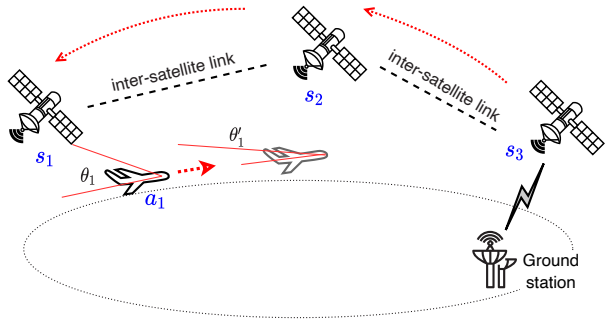

Fig. 1. A LEO satellite network.

when a UE connects to a satellite, a handover procedure is triggered when the elevation angle between the UE and the satellite falls below a predefined threshold. However, this handover triggering condition can be easily activated when applied to fast-moving devices, such as airplanes. For example, in Fig. 1, the airplane a_1 initially connects to satellite s_1. Assume that s_1 moves to the left while a_1 moves to the right. In this scenario, the elevation angle between s_1 and a_1 changes rapidly, degrading from θ_1 to θ_1'. As a result, a_1 will trigger a handover procedure within a short period. From this example, it is evident that handovers may be triggered easily and frequently in such scenarios. Therefore, this work aims to decide (i) the suitable handover timing and (ii) the appropriate handover target for fast-moving devices.

In this paper, we propose a handover strategy based on deep reinforcement learning (DRL) techniques. The proposed approach employs two deep Q-networks (DQNs) to determine handover trigger timing and handover targets, respectively. First, the DQN for determining handover trigger timing considers changes in the network topology and identifies whether the link quality of the connected satellite is likely to degrade. Second, the DQN for selecting the host satellite evaluates multiple metrics, including visibility time, elevation angle, load of satellite, and hop count distance to the ground station. The simulation results indicate that the proposed scheme can effectively increase network throughput and reduce latency. The contributions of this work are summarized as follows:

- The proposed scheme accounts for changes in the network environment to trigger handovers by considering

the characteristics of fast-moving devices.
- The proposed method incorporates multiple metrics when selecting the target satellite. These metrics help stabilize the link to the satellite and reduce the hop count distance to the ground station.
- To the best of our knowledge, this is the first work to propose a handover scheme specifically for fast-moving devices in LEO satellite networks.

The remainder of this paper is organized as follows. Section II and Section III review some previous works and basic concepts of deep reinforcement learning, respectively. Then, Section IV introduces the proposed handover strategy. Next, Section V illustrates the simulation results. Finally, Section VI concludes this paper.

II. Related Works

In previous studies, reference [2] introduces key selection metrics for making handover decisions. These metrics include visibility time, elevation angle, and the number of free available channels. References [3], [4] propose methods to improve communication quality using different selection metrics. The authors in reference [3] design a graph-based satellite handover framework. This framework models the overlapping period of the satellite as nodes and the possible handovers between two overlapping periods as edges. Handover decisions can then be implemented by finding a path in the constructed directed graph. Reference [4] proposes a centralized handover strategy controlled by ground stations. The proposed method establishes a network flow model that considers users' requirements, allowing handover decisions to be made by finding the maximum flow in the network to satisfy multiple requirements. Both [3] and [4] achieve their objectives, but their schemes only consider single metrics when making handover decisions. Furthermore, references [5]–[7] design handover schemes by considering multiple metrics. For instance, reference [5] develops a strategy based on a weighted bipartite graph, which assigns weights based on satellites' qualities (including channel quality, remaining service time, satellite load, and power allocation). Although this scheme effectively makes handover decisions, it incurs high computational overhead. Reference [6] proposes a handover scheme for LEO networks using deep reinforcement learning. A deep Q-network (DQN) is utilized to solve a multi-objective optimization problem. However, this handover scheme requires periodic execution, leading to unnecessary computations. Reference [7] adopts a multi-agent reinforcement learning approach for handover. The designed scheme considers satellite load constraints while aims to reduce the number of handovers. However, the designed method has high computational complexity. In summary, in references [2]–[7], some works only consider single metrics when making handover decision, and some works incur high computational complexity when making handover decisions.

III. Deep Reinforcement Learning (DRL) Concept

Reinforcement learning (RL) is a branch of machine learning. In RL, an agent interacts with an environment to learn optimal actions, which can maximize cumulative long-term rewards. The RL involves four key components, i.e., *states*, *actions*, *rewards*, and *policies*. Based on the current state, an agent takes actions, and then the environment responds with a new state and a reward. Moreover, the Q-learning is a value-based RL algorithm, which aims to learn an action-value function, known as the Q-function. The Q-function estimates the expected long-term return from taking a specific action in a given state. Legacy Q-learning adopts a Q-table to store and update Q-values for all possible state-action pairs. However, as environments become more complex, it becomes impractical to use Q-table due to the exponential growth in state-action pairs. Therefore, the Deep Q-Networks (DQN), which adopts deep neural networks to approximate the Q-function, is proposed to handle large-scale and complex environments.

Instead of storing Q-values in a table, DQN uses a neural network to approximate the Q-function, which estimates the expected future rewards for each possible action given a state. Then, the agent can select the action with the highest Q-value. DQN stores past experiences (state, action, reward, next state) in a replay buffer. Then, it can randomly samples from this buffer to break temporal correlations between experiences. To prevent instability, DQN uses a primary network for learning and a target network for generating target Q-values. The target network will be updated periodically to stabilize the training process. During training, the agent can adopt an ϵ-greedy policy to balance exploration (trying new actions) and exploitation (choosing the best-known actions). This strategy facilitates learning and prevents convergence to suboptimal. By integrating the above techniques, DQN achieves efficient learning in complex environments.

IV. The Proposed Scheme

In this work, we assume that a fast-moving object (e.g., an airplane) is represented as an agent a, which is covered by at least one satellite. The agent a can select a satellite as the *host satellite* to access the Internet. In the proposed DRL-based scheme, two DQNs are employed: one is to determine whether to trigger a handover and the other is to select the host satellite.

A. DQN for Deciding Handover Triggering Time

In this DQN, the primary goal is to determine the optimal handover timing. Below, we introduce the state, action, and reward function. First, for an agent a at a time instant t, the system uses the following four parameters as the state:

- h_o^a: This parameter records the initial hop count distance from its host satellite to the ground station.
- θ_o^a: This parameter records the initial elevation angle of a to its host satellite.
- h_t^a: This parameter represents the current hop count distance from its host satellite to the ground station.
- θ_t^a: This parameter represents the elevation angle of a to its host satellite.

Then, the action for the agent a can be "trigger handover" or "not trigger".

Next, we introduce the design of our reward function. The reward function takes into account changes in both elevation angle and hop count distance. The idea is that when the hop count distance to the ground station increases or the elevation angle to the host satellite decreases, it implies that the network condition to the host satellite becomes worse. In such cases, if a handover is triggered, the reward should be higher. Conversely, if the network condition is still satisfactory but a handover is triggered, a negative reward will be assigned. To evaluate the changes in the network conditions, we define the following two parameters:

- Ratio of hop count distance change

$$r_{hop} = \frac{h_t^a - h_o^a}{h_o^a}.$$

- Rate of elevation angle change

$$r_{angle} = \frac{\theta_o^a - \theta_t^a}{\theta_o^a}.$$

We can see that the ratios of hop count and elevation angle change are used to evaluate the current network condition compared to the original state when connected to the host satellite. It is important to note that when a handover occurs, an agent aims to select a new host satellite that has a shorter hop count distance to the ground station and a higher elevation angle. Therefore, if the ratios become large, it indicates that the network condition becomes worse. The reward function, which assigns reward values based on whether a handover is triggered, is defined as follows:

- If a handover is triggered, the reward will be

$$(r_{hop} + r_{angle}).$$

- Otherwise, the reward will be

$$(1 - (r_{hop} + r_{angle})).$$

Based on this design, a large value of $(r_{hop} + r_{angle})$ implies a deteriorating network condition, and thus a higher reward for the DQN after making a handover decision should be given. This design effectively encourages triggering a handover in scenarios where the network condition is degrading. Conversely, if a handover is not triggered, the reward is higher when $(r_{hop} + r_{angle})$ is smaller. A small value of $(r_{hop}+r_{angle})$ implies a stable network; as a result, a handover is unnecessary.

B. DQN for Selecting Host Satellite

When an agent decides to perform a handover, it will select a new host satellite from a list of its candidate satellites. Assuming an agent a, the set C_t^a represents the candidate satellites available to agent a at time t. In this work, agent a considers its nearby m line-of-sight satellites that have larger elevation angles as its candidate satellites. At time instant t, for a candidate satellite $s \in C_t^a$, the agent a will maintain the following information: (i) elevation angle to s, say $\theta_t^{a,s}$, and (ii) visibility time of s, say $V_t^{a,s}$. Additionally, the agent a maintains a set N_t^a, which records the list of nearby agents and their corresponding host satellites, and the its hop count distance to the ground station, say h_t^a. In our design, the state of agent a will encompass the following three components:

- The list of candidate satellites and the associated information (visibility time and elevation angle) for each candidate satellite.
- The set N_t^a.
- The value h_t^a.

In this DQN, the action of a includes selecting any of the candidate satellites in C_t^a.

After selecting a new host satellite, the reward function is listed as below:

$$\frac{1}{\frac{1}{r_e} + \frac{1}{r_v} + \frac{1}{1 - r_h} + \frac{1}{1 - r_c}}. \tag{1}$$

In Eq. (1), we utilize the harmonic mean to define our reward function, which helps mitigate the influence of extreme values. The values of r_e, r_v, r_h, and r_c are normalized to fall within the range $(0, 1)$. In the reward function, r_e and r_v represent the rewards associated with the observed elevation angle and visibility time to the new host satellite, respectively. We can observe that higher values of r_e and r_v lead to a larger reward. The parameter r_h represents the reward for the hop count distance from the new host satellite to the ground station. Thus, when the hop count distance is lower, the reward is higher. Finally, r_c records the number of served agents at the new host satellite. In other words, if a satellite serves more agents, the value of r_c will be lower. Based on our design, the reward increases when r_c is lower. This approach ensures that the load on the selected host satellite remains manageable.

V. Simulation Results

This work utilizes Python and the General Mission Analysis Tool (GMAT) [8] to simulate the environment of LEO satellite networks. In our simulation, satellites are deployed using the *walker star constellation* and *walker delta constellation*. Each orbit plane consists of 11 satellites, which are uniformly distributed. The detailed simulation parameters are presented in Table I. Initially, the simulation program loads data for the agent (i.e., airplane) and the satellites. The movement of the agent takes into account real airport locations; for example, airplanes might start their journey from a starting airport (e.g., Bangkok) to a destination airport (e.g., Los Angeles). The simulation program continuously updates the locations of the satellites using planetary ephemeris data. With this location information, the program evaluates the elevation angles between the agents and the satellites. Additionally, the simulation deploys 25 ground stations and continuously updates the routing paths for the satellites. Each satellite employs the A* algorithm to determine the routing path that minimizes the hop count distance to a ground station. Moreover, Table II presents the DQN parameters used in our design. Most of these two DQN models share similar configurations; however, the DQN for selecting the host satellite incorporates more hidden

TABLE I
SIMULATION PARAMETERS.

Parameter	Value
Orbital planes	12
Satellite per orbital planes	11
Orbital inclination	$[86.4°, 60°, 120°]$
Orbital altitude	780 km
Minimum elevation angle	$10°$
Center frequency	30 GHz
Antenna transmission power	30 dBm
Tx antenna gain	30 dBi
Rx antenna gain	40 dBi
Center frequency	30 GHz
Bandwidth	10 MHz
System noise temperature	290 K
Boltzmann constant	1.38^{-23} J/K

TABLE II
DQN MODEL PARAMETERS

Parameter	Selecting host satellite	Deciding handover triggering time
State space size	24	4
Action space size	4	2
Hidden layers	2	2
Hidden layer nodes	128	64
Conv layers	4	-
γ	0.99	0.99
ϵ-start	1.0	1.0
ϵ-end	0.01	0.01
ϵ-decay	0.995	0.995
α	0.0001	0.001
β	0.01	-
Optimizer	Adam	Adam
Loss function	HuberLoss	MSE
Dropout	0.2	-

layer nodes and utilizes the HuberLoss function to mitigate noise and outliers.

In this work, we compare the proposed scheme (denoted by OUR) with three host satellite selection strategies (as listed in [2], [9], [10]): (i) selecting the satellite with the maximum elevation angle (denoted by MEA), (ii) selecting the satellite with the longest visibility time (denoted by MVT), and (iii) selecting the satellite with the most free available channels (denoted by FAC). Note that in the MEA, MVT, and FAC strategies, an agent will trigger a handover when the observed elevation angle to its host satellite falls below 10 degrees.

Fig. 2 illustrates the average throughput between agents and their host satellites. We can observe that OUR outperforms the other three methods. Compared to MEA, MVT, and FAC, OUR achieves throughput increases of 3.65%, 4.17%, and 4.55%, respectively. This improvement is attributed to the joint consideration of multiple metrics when selecting host satellites, which enhances signal quality and visibility time. Next, Fig. 3 presents the average system throughput, measured between agents and ground stations. The system throughput can be influenced by ISLs between intermediate satellites. The results indicate that the system throughput is nearly consistent with the previous findings, and OUR continues to outperform the other methods. Generally, system throughput increases

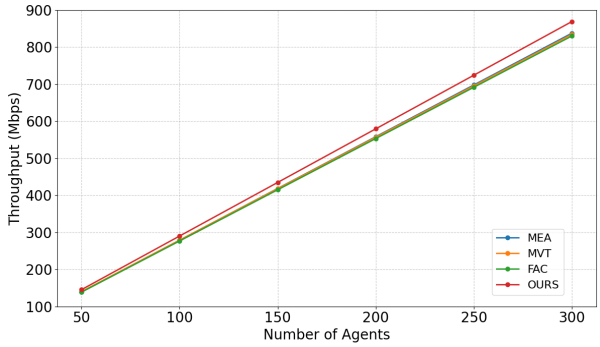

Fig. 2. Simulation result on throughput between agent and satellite.

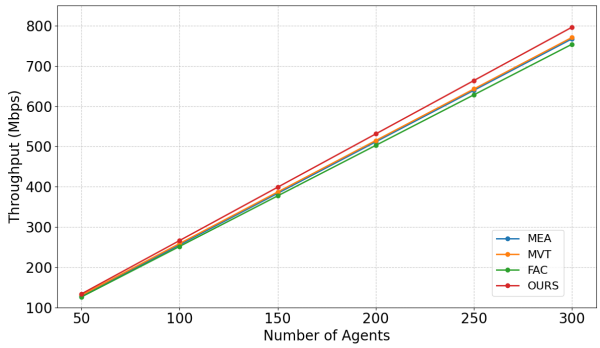

Fig. 3. Simulation result on system throughput.

proportionally with the number of agents. However, as the number of agents increases from 250 to 300, OUR demonstrates a more significant increase in throughput. This result demonstrates that the proposed strategy effectively facilitates communication between agents and ground stations.

Next, Fig. 4 presents the average hop count distance to the ground stations. Compared to MEA, MVT, and FAC, OUR can reduce hop count distance by 4.07%, 5.85%, and 11.94%, respectively. These results demonstrate that the designed policy effectively selects host satellites for agents, allowing OUR to reduce communication latency for agents. Moreover, Fig. 5 illustrates the average number of triggered handovers. Note that the MVT method selects the host satellite

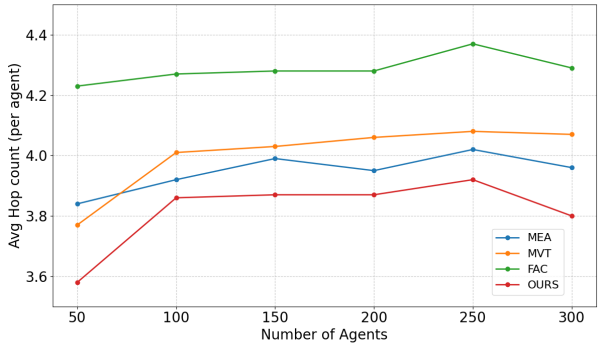

Fig. 4. Simulation result on hop count distance to ground stations.

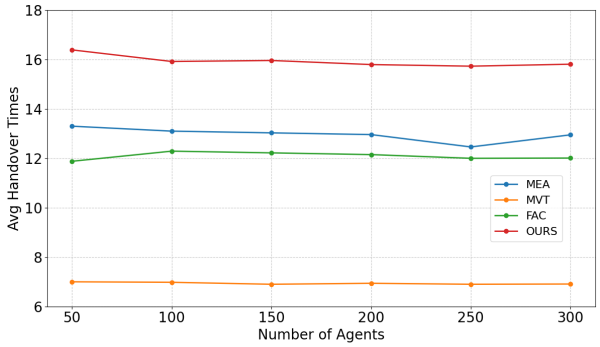

Fig. 5. Simulation result on average number of triggered handovers.

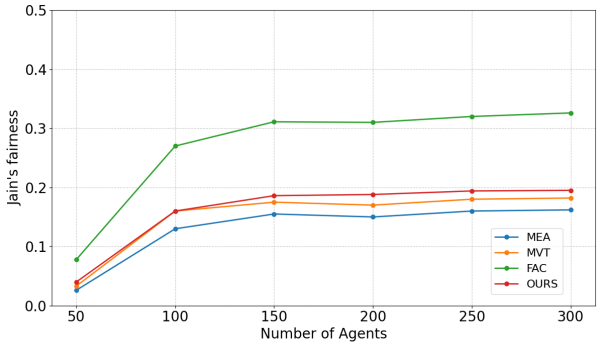

Fig. 6. Simulation results on Jain's fairness on the loads of satellites.

with longer visibility time, which consequently reduces the frequency of handovers. In contrast, OUR may induce more handover procedures; however, as mentioned earlier, OUR provides higher throughput and lower latency. Additionally, OUR triggers handovers before the elevation angles become too small, helping to preserve connections between agents and satellites.

Fig. 6 illustrates Jain's fairness regarding the loads of satellites. Let x_i represent the number of times that a satellite s_i is selected as the host satellite. The equation to derive fairness is given by:

$$J(x_1, x_2, \ldots, x_n) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}.$$

According to this equation, the result of the above equation ranges from 0 to 1. A result closer to 1 indicates a more equitable distribution of network load among satellites. From the results, we observe that the FAC method allows more equally sharing of network load. However, this fairness comes at the expense of performance (i.e., throughput and hop count distance), which is lower compared to other methods. In contrast, when compared to MEA and MVT, OUR improves network throughput and hop count distance while maintaining a more balanced load across satellites.

In summary, the simulation results indicate that OUR can enhance network performance in expense of triggering slightly more handovers. Additionally, the results demonstrate that the designed DQNs operate effectively in managing handover decisions and host satellite selections, leading to improved overall system performance.

VI. Conclusions

In this work, we propose a handover strategy for fast-moving devices in a LEO satellite network utilizing DRL. The designed strategy jointly considers several metrics, including elevation angle, visibility time, hop count distance, neighbor status, and changes in the network environment. We implement two DQNs for agents to determine handover triggering times and select host satellites. The simulation results demonstrate that the proposed method effectively improves network throughput and reduces latency. In the future, we plan to further optimize our DQN models, collect more training datasets, and incorporate additional metrics into our models.

Acknowledgement

This work is sponsored by NSTC 112-2221-E-027-052-MY2.

References

[1] Y. Su, Y. Liu, Y. Zhou, J. Yuan, H. Cao, and J. Shi, "Broadband LEO satellite communications: Architectures and key technologies," *IEEE Wireless Communications*, vol. 26, no. 2, pp. 55–61, 2019.

[2] P. K. Chowdhury, M. Atiquzzaman, and W. Ivancic, "Handover schemes in satellite networks: state-of-the-art and future research directions," *IEEE Communications Surveys & Tutorials*, vol. 8, no. 4, pp. 2–14, 2006.

[3] Z. Wu, F. Jin, J. Luo, Y. Fu, J. Shan, and G. Hu, "A graph-based satellite handover framework for LEO satellite communication networks," *IEEE Communications Letters*, vol. 20, no. 8, pp. 1547–1550, 2016.

[4] S. Zhang, A. Liu, C. Han, X. Ding, and X. Liang, "A network-flows-based satellite handover strategy for LEO satellite networks," *IEEE Wireless Communications Letters*, vol. 10, no. 12, pp. 2669–2673, 2021.

[5] S. Zhang, A. Liu, and X. Liang, "A multi-objective satellite handover strategy based on entropy in LEO satellite communications," in *Proc. of IEEE International Conference on Computer and Communications (ICCC)*, 2020.

[6] J. Wang, W. Mu, Y. Liu, L. Guo, S. Zhang, and G. Gui, "Deep reinforcement learning-based satellite handover scheme for satellite communications," in *Proc. of International Conference on Wireless Communications and Signal Processing (WCSP)*, 2021.

[7] S. He, T. Wang, and S. Wang, "Load-aware satellite handover strategy based on multi-agent reinforcement learning," in *Proc. of IEEE Global Communications Conference (GLOBECOM)*, 2020.

[8] "NASA Goddard Space Flight Center, GSC-17177-1: A software framework for the integration of human factors in systems engineering," https://software.nasa.gov/software/GSC-17177-1, 2021, accessed: 2024-10-02.

[9] E. Papapetrou, S. Karapantazis, G. Dimitriadis, and F.-N. Pavlidou, "Satellite handover techniques for LEO networks," *International Journal of Satellite Communications and Networking*, vol. 22, pp. 231 – 245, 2004.

[10] A. Bottcher and R. Werner, "Strategies for handover control in low earth orbit satellite systems," in *Proc. of IEEE Vehicular Technology Conference (VTC)*, 1994.