CO-QLR: Cooperative Q-Learning based Routing for IoT Networks

Shotaro Takahashi, Shota Inoue, Hiroyuki Ohsaki
School of Engineering
Kwansei Gakuin University
Hyogo, Japan
email: {shotaro-t,shota-i,ohsaki}@kwansei.ac.jp

***Abstract*—Low-power and Lossy Networks (LLNs) are characterized by constrained nodes with limited power, processing capabilities, and memory, leading to challenges such as high packet loss, low transmission rates, and network instability. While the standardized IPv6 Routing Protocol for LLNs (RPL) addresses some of these issues, it falls short in dynamic IoT environments, particularly under congestion. To enhance routing efficiency in LLNs, this paper proposes CO-QLR (Cooperative Q-Learning-based Routing), a novel distributed routing protocol that leverages cooperative reinforcement learning. Unlike independent Q-learning-based approaches, CO-QLR enables each node to share learned metrics from its Q-table with neighboring nodes, thereby improving load balancing and reducing packet loss. Simulation experiments are conducted to evaluate CO-QLR's performance against conventional independent Q-learning-based routing, demonstrating its effectiveness in reducing message loss rate. This study contributes a cooperative learning framework for routing in constrained networks, showing potential for enhancing the resilience and efficiency of LLNs in diverse network topologies.**

***Index Terms*—Low-power and Lossy Networks (LLNs), Routing protocol, Reinforcement Learning, Q-Learning, Cooperative Q-Learning**

I. Introduction

Low-power and Lossy Networks (LLNs) comprise constrained nodes with limited power, processing power, and memory. The transmission between LLN nodes is known to have high packet loss, low transmission rates, and instability [1]. Therefore, the routing protocol for LLNs must implement routing protocol that considers the limitations and characteristics of these network nodes.

Currently, the IPv6 Routing Protocol for LLNs (RPL) [2] is the standardized routing protocol for LLNs to fulfill the requirements of LLN applications. Although RPL has been standardized, it is insufficient for dynamic IoT environments. RPL is a single path routing protocol, wherein each node transmits messages to a preferred parent node chosen based on its objective function. The RPL standard defines Objective Function Zero (OF0) [3] and the Minimum Rank Hysteresis Objective Function (MRHOF) [4] as objective functions in its RFC. OF0 uses only hop count as a routing metric, while MRHOF relies solely on the expected number of transmissions (ETX). Since RPL is designed for low-traffic scenarios, it faces challenges with load balancing in congested environments, which can lead to issues such as high packet loss and increased power consumption [5].

To address the issue of congestion, proposed a reinforcement learning-based routing protocol in [5]. In their approach, each node maintains a Q-table representing routing information for neighboring nodes. This Q-table is updated using a feedback function that reflects both the congestion level and link quality to each neighbor. Each node identifies the neighboring node with the minimum Q-value in its table as the preferred forwarding node and probabilistically selects the next-hop node based on the values in the Q-table.

Here, instead of each node independently executing Q-learning as in the approach in [5], we anticipate that routing performance can be enhanced through cooperative Q-learning, where each node shares its learning state with neighboring nodes. To the best of our knowledge, there are two studies that have applied cooperative Q-learning to routing: [6, 7]. The cooperative Q-learning proposed in [6] enables newly joined nodes in a 6TiSCH network to acquire the learning state of their parent nodes. In [7], cooperative Q-learning is applied to ad-hoc networks, where, upon the arrival of a packet at a neighboring node, the neighboring node returns its Q-value to the sender, allowing the sender node to update its own Q-values based on those of the neighboring node.

However, in these studies, cooperative Q-learning is implemented by having each node update its own Q-values using the Q-values of neighboring nodes. Consequently, the learning state of neighboring nodes can only be applied to Q-values that share common information between the node itself and its neighbors.

Accordingly, the research questions for this study are summarized as follows.

- Can the efficiency of network routing be enhanced through reinforcement learning by sharing learned knowledge among neighbors nodes? If yes, to what degree can the routing efficiency be increased?
- For routing in IoT networks using LLNs, what learned data that each node acquires through learning should be shared with its neighbors and how should it be shared?
- In which network topologies can routing protocol relying on cooperative Q-learning be most advantageous?

In this paper, we propose CO-QLR (Cooperative Q-Learning-based Routing), a routing method for LLNs that enables efficient routing by sharing metrics derived from each node's Q-table values with neighboring nodes. In CO-QLR, each node advertises the Q-values associated with its neighboring nodes, obtained through Q-learning. Using these cumulative Q-values as link costs, each node determines its upstream node through a distance-vector-based routing approach.

We also present experimental results of simulation, which quantitatively demonstrate the effectiveness of CO-QLR. This paper explores the efficacy of CO-QLR via simulation experiments. Specifically, this study evaluates and compares two routing methods: CO-QLR, which shares each node's learning state with its neighbors, and QLR, where nodes learn Q-values independently based on the method in [5]. The effectiveness of the proposed approach is demonstrated by evaluating and comparing CO-QLR and QLR performances performance in terms of the percentage of nodes having an available path to the sink, the average path length from every node to the sink node, and the message loss rate caused by the relay node.

The main contributions of this paper are summarized as follows.

- We propose CO-QLR, a cooperative distributed routing protocol that employs reinforcement learning. In CO-QLR, nodes disseminate a portion of the knowledge they have acquired through learning with their neighbor nodes.
- The performance of CO-QLR is quantitatively evaluated in different network topologies, which reveal that CO-QLR outperforms the conventional fully-independent Q-learning-based routing protocol in terms of the message loss rate.

The structure of this paper is as follows. In Section II, we provide an overview of related studies on routing in LLNs and cooperative Q-learning-based routing, establishing the background for our proposed method. Section III then presents the CO-QLR approach in detail, explaining its design and operational principles. Following this, Section IV describes simulation experiments conducted to assess the performance of CO-QLR. Finally, Section V summarizes our findings and discusses future directions for expanding this research.

II. Related Works

In RPL, OF0 and MRHOF are defined as objective functions; however, challenges have been reported in adapting to load balancing and network changes [1, 5, 8, 9].

To enhance the routing performance of RPL, several machine learning-based routing methods have been proposed in recent years. In [10], a new approach for selecting a parent node among nodes with the same rank is proposed using a random forest algorithm. In [11], ML-RPL, a method that selects a parent node using the CatBoost gradient-boosting algorithm for prediction, is introduced.

There are several studies on machine learning-based routing methods for LLNs, including reinforcement learning-based approaches, which are the focus of this study. In [5], Q-learning is applied to address the load balancing issue in RPL networks. In [12], Q-learning is used to solve the classic problem of finding the optimal parent node in a tree topology within WSNs. In [13], Q-RPL, a routing method for Advanced Metering Infrastructure based on Q-learning, is proposed. In [14], RI-RPL, which performs parent selection in the RPL routing protocol using Q-learning, is presented. Furthermore, the study [15] introduces RARI, a deep reinforcement learning-based RPL routing optimization algorithm designed to achieve load balancing and power-efficient communication for large-scale data transfers in RPL.

In [6], a cooperative reinforcement learning method, ACI-RPL, is proposed, where new nodes entering the network receive a Q-table from their parent nodes. In ACI-RPL, when a node receives a DIS(DODAG(Destination Oriented Directed Acyclic Graph) Information Solicitation) message, it responds to the sender node with a DIO(DODAG Information Object) message that includes its Q-table. The receiving node combines the Q-table from the DIO message with its own Q-table, enabling new nodes to efficiently begin learning with knowledge acquired from their parent nodes, rather than starting from scratch.

While not specifically targeting LLNs, the authors of [7] proposes a cooperative Q-learning routing method for ad-hoc networks. In this method, upon receiving packets from neighboring nodes, a node returns its Q-value to the sender node during communication. The sender node then uses the received Q-value to update its own Q-table.

As illustrated above, RPL routing performance improvement has gained considerable attention, with various researchers proposing new routing methods through different approaches. However, to the best of our knowledge, no research has implemented a cooperative Q-learning routing approach that advertises metrics based on each node's Q-values.

III. CO-QLR (Cooperative Q-Learning-based Routing)

Each node within the LLNs can communicate with neighboring nodes that exist within its communication range via wireless communication. Messages generated by each node are delivered to the sink node through multihop communication via other nodes.

CO-QLR is an autonomous routing scheme for nodes in an LLNs that determines the optimal path from each node to the sink node by using Q-learning, a type of reinforcement learning. CO-QLR is built upon the routing scheme cited in [5]; however, it differs in that the Q-values are exchanged between neighboring nodes to improve upon the traditional method of independent Q-learning execution.

Each node on the network running CO-QLR maintains Q-tables that stores Q-values for its neighbors, and it periodically updates the Q-values based on a weighted sum of the message occupancy in its own buffer (which represents congestion in its neighboring nodes), the path length from its neighbors to the sink node, and the congestion level in the neighboring nodes [5].

In CO-QLR, each node maintains a Q-value for each neighbor, following the method used by [5]. Selecting an upstream node among its neighbors in CO-QLR corresponds to an action, a, in Q-learning. The calculated reward, r, in Q-learning for selecting node a as the upstream node is given as follows [5].

$$r = \max\left(\frac{\theta_a}{\beta}, 1 - \frac{\theta_a}{\beta}\right)\theta_a + H_a \tag{1}$$

In the above equation, θ_a is the buffer occupancy of the adjacent node a; H_a, the path length from adjacent node a to the sink node; and β, the backlog factor [5]. Unlike in general Q-learning, a smaller reward value r indicates a more favorable upstream node.

In CO-QLR, the upstream node (next-hop node) selection differs fundamentally from the method proposed by [5]. Instead of probabilistically selecting the upstream node based on Q-values at regular intervals, CO-QLR calculates a metric using the advertised Q-values from the upstream node and the node's own Q-value. The reciprocal of this metric is raised to the power of δ, and the upstream node is chosen with a probability proportional to the resulting value.

The distance metric, denoted by $d_{u,v}$, between node u and its neighboring node v can be calculated as

$$d_{u,v} = d_v^* + Q_u[v] \tag{2}$$

where d_v^* is the metric advertised by the neighbor node v and $Q_u[v]$, the metric advertised by node u to the neighboring node v. $Q_u[v]$ is the Q-value corresponding to the adjacent node v held by node u.

Node v then distributes

$$\min_{v \in N(u)} d_{u,v} \tag{3}$$

to its neighbors. $N(u)$ refers to the set of neighbors for node u.

IV. Experiment

This section evaluates the effectiveness of a routing method using cooperative Q-learning through simulation experiments. We implemented the proposed routing method, CO-QLR, based on cooperative reinforcement learning in Python, and developed a custom simulator to analyze its performance. Results from this simulator were used to assess the CO-QLR method.

As a comparison, we implemented a non-cooperative Q-learning routing method, QLR, in Python and conducted experiments under the same conditions as CO-QLR. Notably, we did not compare with routing methods that use the standard RPL objective functions, OF0 and MRHOF, as the effectiveness of reinforcement learning-based routing over traditional RPL has already been investigated by [5]. Since the aim of this study is to examine the potential efficiency of routing through cooperative reinforcement learning, we leave this comparison for future research.

To assess the effectiveness of CO-QLR in small- and medium-scale network environments, we conducted experiments with two different scenarios. The first scenario evaluates CO-QLR's effectiveness in relatively small networks, while the second scenario examines its performance in larger networks where message loss is more likely to occur. In a network comprising multiple nodes, each client node probabilistically generated messages and routed them to the sink node. In each experiment scenario, we employed a grid network topology (Fig. 1).

Three metrics were used to evaluate CO-QLR: path availability, average path length, and message loss rate. Path availability represents the percentage of nodes that successfully connected to the sink node across all nodes. The average path length is the mean path length from each node to the sink node. The message loss rate p

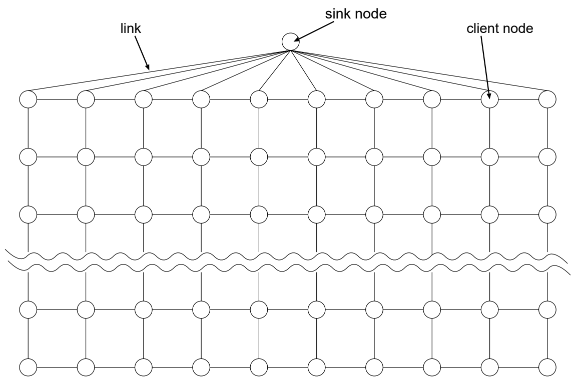

Fig. 1. Network topology of a grid comprising 100 nodes

approximates the proportion of messages lost during the simulation and is calculated as follows:

$$p = \frac{\hat{p}}{T} \lambda N \tag{4}$$

where $\hat{p}$ is the total message loss during the simulation, T is the number of simulation steps, λ is the message generation rate, and N is the number of nodes excluding the sink node.

While power consumption is a common performance metric for routing in LLNs, we did not include it in our evaluation due to existing studies on the overhead associated with reinforcement learning [6]. Instead, we consider it a subject for future research. Cooperative reinforcement learning could be realized by periodically advertising each node's metrics through DIO messages, as described in [6]. Investigations into the overhead induced by cooperative reinforcement learning were conducted by the study [6] and [5]. We defer the evaluation of power consumption; future work should investigate how cooperative learning influences power usage across nodes and examine the impact of metric advertisement frequency on routing performance.

In each experimental scenario, we performed 500 simulation runs, calculating and plotting the mean and 95 % confidence intervals for each evaluation metric. For each simulation, each slot was set to 100 [ms], totaling 5 [seconds] per run. Furthermore, each client ' s Q-learning learning rate was set to 0.3, with a buffer size of 5 [messages] and bandwidth of 1 [message/slot]. Each client was configured to send a message of size based on a uniformly distributed random value ranging from 0 to 2λ per slot to a designated parent client.

A. Small Scale Scenario

In the first experimental scenario, we utilized a grid network topology consisting of 100 client nodes (see Fig. 1). In this grid network, the top 10 clients were connected to the sink nodes, and the message generation rate λ for each client was set to 0.01.

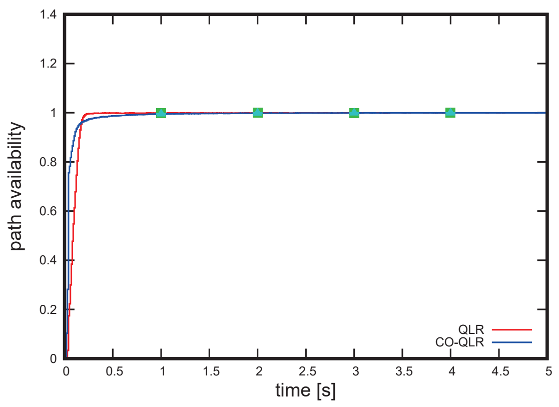

Fig. 2. Temporal variation of path availability rate

The figures display the temporal fluctuations in the path availability between client and sink nodes, the average hop count between them, and the cumulative number of message losses at relay nodes throughout the simulation. Figs. 2 to 4 illustrate these variations.

As shown in Figs. 2 and 3, there is little difference in path availability and hop count between QLR and CO-QLR within a grid topology composed of 100 clients. Fig. 2 depicts the temporal evolution of path availability for all clients throughout the simulation. The results show that path availability for both QLR and CO-QLR approaches nearly 1 within about 0.5 seconds, indicating that almost all clients are able to connect to the sink nodes. This suggests that collaborative Q-learning does not significantly impact path availability. Next, Fig. 3 illustrates the time-dependent variation in the path length from each client to the sink node. The results show that CO-QLR records a slightly lower path length compared to QLR at the beginning of the simulation, but this difference diminishes as the simulation progresses. Overall, there is no notable difference in path length due to the use of collaborative Q-learning.

While there is little difference in path availability and path length, Fig. 4 reveals that CO-QLR effectively reduces message loss rates among all clients compared to QLR. Fig. 4 shows the temporal change in the message loss rate for each client during the simulation. The results indicate that CO-QLR ultimately reduces the message loss rate to approximately half that of QLR. This suggests that collaborative Q-learning achieves efficient load balancing, as it maintains comparable performance in terms of path availability and path length while reducing the message loss rate.

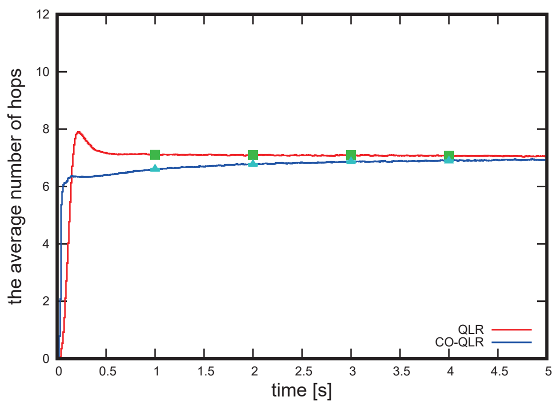

Fig. 3. Temporal variation of the path length to the sink node

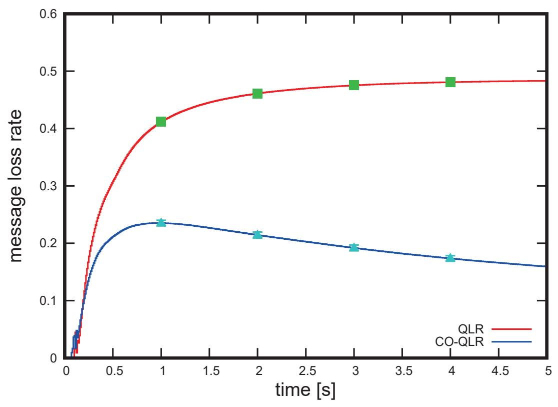

Fig. 4. Temporal variation in the message loss rate

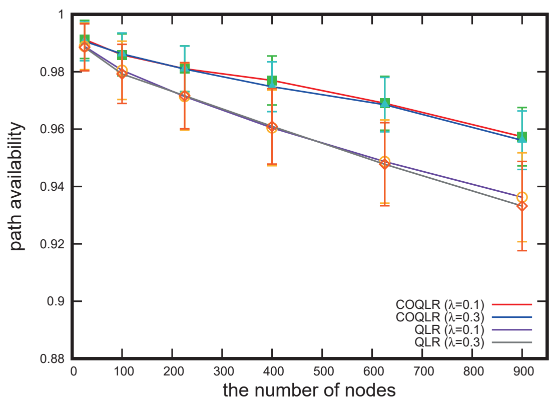

Fig. 5. Path availability changes in response to the increase in graph size

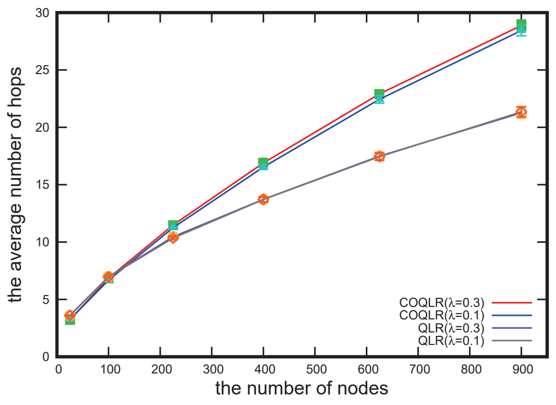

Fig. 6. Changes in the path length to the sink node in response to the increase in graph size

B. Large Scale Scenario

In the second scenario, the routing efficiency of CO-QLR was evaluated across grid topologies of varying scales, ranging from 3 × 3 to 30 × 30, using three metrics: path availability, path length to the sink node, and message loss rate throughout the entire simulation. To further investigate the effectiveness of CO-QLR in a more demanding environment, two message generation rates, $\lambda = 0.1$ and 0.3 [messages/slot], were set.

Fig. 5 to 7 illustrate the average path availability from client nodes to the sink node, the average path length, and the message loss rate for the entire simulation.

As shown in Fig. 5, unlike in the first experimental scenario where no differences in path availability were observed, increasing the graph scale led to a growing disparity in path availability. In the experiment condition with the largest topology, involving 900 clients, a comparison between CO-QLR and QLR revealed that approximately 20 more clients were able to connect to the sink node under CO-QLR.

According to the results in Fig. 6, CO-QLR exhibited a higher average hop count than QLR when the network size increased. However, as indicated by Fig. 5, there were more nodes that could not find a path to the sink node under QLR-based routing compared to CO-QLR. Since unreachable paths are excluded from the hop count observation, it is inferred that CO-QLR recorded a higher average hop count than QLR due to its greater path availability.

As seen in Fig. 7, similar to the first scenario, CO-QLR was able to maintain a lower message loss rate per client compared to QLR. The results indicate that, across all graph scales and message generation rates, CO-QLR achieved lower message loss rates compared to QLR, particularly when the message generation rate was 0.1. This suggests that CO-QLR can maintain lower message loss rates under high-load and large-scale conditions than QLR.

In conclusion, within grid network topologies, CO-QLR enables communication with lower message loss

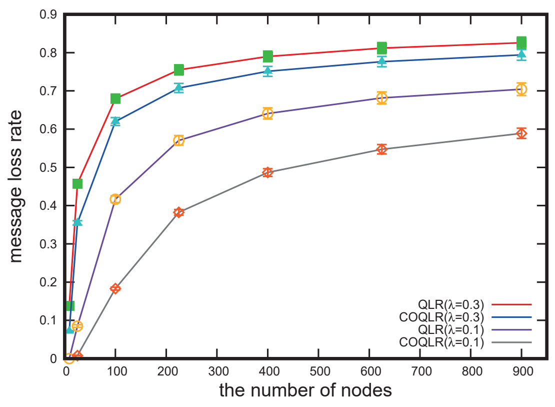

Fig. 7. Changes in the message loss rate in response to the increase in graph size

rates than QLR while maintaining comparable path availability and path length to the sink node. This outcome remained consistent as graph size and the total traffic load in the network increased.

V. Conclusion

In this study, we propose CO-QLR for efficient routing using Q-learning for LLNs by sharing the Q-table value of each node with its neighbors. Simulation experiments show the effectiveness of the proposed scheme in terms of the messages loss rate by the relay nodes.

Future research will examine the effectiveness of CO-QLR from an energy consumption perspective, evaluating its performance in more realistic environments that consider the impact of wireless links and MAC protocols, thereby clarifying the effectiveness of cooperative Q-learning. Additionally, optimizing the frequency of cooperation and control parameters for each node will be pursued.

Moreover, apart from the experiments conducted in this study, 500 simulation trials were performed on a single randomly generated topology where sink nodes and clients were randomly placed to assess the effectiveness of CO-QLR. However, no significant difference was observed when compared to QLR. Therefore, further studies will include experiments with a variety of randomly generated topologies to investigate whether the proposed method shows effectiveness in random topologies. This exploration will address whether applying the proposed method to network topologies other than grid networks is feasible and beneficial.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number 24K02936.

References

[1] B. Ghaleb, A. Y. Al-Dubai, E. Ekonomou, A. Alsarhan, Y. Nasser, L. M. Mackenzie, and A. Boukerche, "A Survey of Limitations and Enhancements of the IPv6 Routing Protocol for Low-Power and Lossy Networks: A Focus on Core Operations," *IEEE Communications Surveys & Tutorials*, vol. 21, pp. 1607–1635, Oct. 2018.

[2] P. Levis, K. Pister, J. Vasseur, R. Alexander, and R. Struik, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," *Request for Comments (RFC) 6550*, Mar. 2012.

[3] P. Thubert, "Objective Function Zero for the Routing Protocol for Low-Power and Lossy Networks (RPL)," *Request for Comments (RFC) 6552*, Mar. 2012.

[4] O. Gnawali and P. Levis, "The Minimum Rank with Hysteresis Objective Function," *Request for Comments (RFC) 6719*, Sept. 2012.

[5] H. Farag and Čedomir Stefanović, "Congestion-Aware Routing in Dynamic IoT Networks: A Reinforcement Learning Approach," in *Proceedings of the 2021 IEEE Global Communications Conference (GLOBECOM 2021)*, pp. 1–6, Dec. 2021.

[6] D. Z. Fawwaz and S.-H. Chung, "Intelligent Parent Change to Improve 6TiSCH Network Transmission using Multi-Agent Q-Learning," *IEEE Access*, vol. 12, pp. 102862–102879, July 2024.

[7] R. Desai and B. P. Patil, "Cooperative reinforcement learning approach for routing in ad hoc networks," in *Proceedings of the 2015 International Conference on Pervasive Computing (ICPC 2015)*, pp. 1–5, Jan. 2015.

[8] H.-S. Kim, H. Kim, J. Paek, and S. Bahk, "Load balancing under heavy traffic in RPL routing protocol for low power and lossy networks," *IEEE Transactions on Mobile Computing*, vol. 16, pp. 964–979, June 2016.

[9] K. A. Darabkh, M. Al-Akhras, J. N. Zomot, and M. Atiquzzaman, "RPL routing protocol over IoT: A comprehensive survey, recent advances, insights, bibliometric analysis, recommendations, and future directions," *Journal of Network and Computer Applications*, vol. 207, p. 103476, Nov. 2022.

[10] C. L. Duenas Santos, J. P. Astudillo León, A. M. Mezher, J. Cardenas Barrera, J. Meng, and E. Castillo Guerra, "RPL+: An improved parent selection strategy for RPL in wireless smart grid networks," in *Proceedings of the 19th ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks (PE-WASUN 22)*, pp. 75–82, Oct. 2022.

[11] C. L. D. Santos, A. M. Mezher, J. P. A. León, J. C. Barrera, E. C. Guerra, and J. Meng, "ML-RPL: Machine Learning-Based Routing Protocol for Wireless Smart Grid Networks," *IEEE Access*, vol. 11, pp. 57401–57414, June 2023.

[12] Kim, Beom-Su and Suh, Beomkyu and Seo, In Jin and Lee, Han Byul and Gong, Ji Seon and Kim, Ki-Il, "An Enhanced Tree Routing Based on Reinforcement Learning in Wireless Sensor Networks. Sensors," *Sensors*, vol. 23, p. 223, Dec. 2022.

[13] C. L. Duenas Santos, A. M. Mezher, J. P. Astudillo León, J. Cardenas Barrera, E. Castillo Guerra, and J. Meng, "Q-RPL: Q-Learning-Based Routing Protocol for Advanced Metering Infrastructure in Smart Grids," *Sensors*, vol. 24, July 2024.

[14] N. Zahedy, B. Barekatain, and A. A. Quintana, "RI-RPL: a new high-quality RPL-based routing protocol using Q-learning algorithm," *The Journal of Supercomputing*, vol. 80, pp. 7691–7749, Nov. 2023.

[15] J. Lei and J. Liu, "Reinforcement learning-based load balancing for heavy traffic Internet of Things," *Pervasive and Mobile Computing*, vol. 99, p. 101891, July 2024.