A Reinforcement Learning-Based Scheduling Scheme for the IEEE 802.15.4e TSCH Network

Nadia Zerguine
LRSD Laboratory, Computer Science Department, Setif 1 University - Ferhat Abbas, Setif, Algeria
nadia.zerguine@univ-setif.dz (corresponding author)

Zibouda Aliouat
LRSD Laboratory, Computer Science Department, Setif 1 University - Ferhat Abbas, Setif, Algeria
zaliouat@univ-setif.dz

Samia Kharchi
LRSD Laboratory, Computer Science Department, Setif 1 University - Ferhat Abbas, Setif, Algeria
samia.kharchi@univ-setif.dz

Received: 10 May 2025 | Revised: 29 June 2025, 19 July 2025, and 22 July 2025 | Accepted: 27 July 2025

 DOI: https://doi.org/10.48084/etasr.12033

ABSTRACT

Time-Slotted Channel Hopping (TSCH), as defined in the IEEE 802.15.4e amendment, is currently considered the most widely used Medium Access Control (MAC) protocol in Industrial Internet of Things (IIoT) wireless networks. However, the absence of a defined scheduling process within the standard remains an open area of research. This paper presents a proposal for integrating Q-Learning (QL), a Reinforcement Learning (RL) method, into the TSCH protocol, allowing parent and channel selection in an intelligent manner. This intelligent allocation aims to optimize the performance of TSCH in the IIoT. The various in-depth simulations carried out show that the integration of QL significantly improves the performance of these networks, including radio reliability, Packet Delivery Ratio (PDR), and especially latency and energy consumption.

Keywords-Time-Slotted Channel Hopping (TSCH); Industrial Internet of Things (IIoT); Reinforcement Learning (RL); scheduling; Q-Learning (QL)

I. INTRODUCTION

The Industrial Internet of Things (IIoT) refers to the implementation of Internet of Things (IoT) technologies in manufacturing, allowing applications such as remote control, real-time monitoring, predictive maintenance, and data analysis, among others [1]. Wireless communication technologies are essential for ensuring rapid and economical deployment of IIoT systems. Among the Medium Access Control (MAC) modes defined by IEEE 802.15.4, Time-Slotted Channel Hopping (TSCH) [2] is a key protocol specifically designed for Low-Power and Lossy Networks (LLNs). Commonly used in Wireless Sensor Networks (WSNs) and IoT applications, TSCH combines Time-Division Multiple Access (TDMA) and Frequency Hopping Spread Spectrum (FHSS) [3, 4], enhancing network functionality through timeslots, channel hopping mechanisms, and time synchronization. These features improve reliability, energy efficiency, and resistance to interference, making TSCH well-suited for industrial environments (IEEE 802.15.4e).

IIoT applications have diverse and evolving requirements, ranging from reliability and energy efficiency to low latency and high throughput. To maintain optimal performance, dynamic network adaptation is essential [5].

The scheduler in a TSCH network has a significant impact on its efficiency. It provides nodes with the opportunity to transmit, receive, or sleep by assigning each one a discrete timeslot within a specific channel. This improves network reliability by avoiding collisions and interference, but it has an impact on energy consumption. In fact, TSCH does not automatically adapt to traffic or topology [6]. Consequently, the TSCH scheduler must be carefully designed to meet the needs of IIoT applications.

Although the IEEE 802.15.4e TSCH standard does not define a specific scheduling mechanism, an appropriate TSCH schedule must be chosen that considers the requirements of IIoT networks. Several approaches have been proposed and are generally classified into three categories: centralized [7, 8], distributed [7, 9] and autonomous [10]. Other approaches, such as static, neighbor-to-neighbor, and hop-by-hop, have also been explored [11].

Reinforcement Learning (RL) has attracted increasing interest in both academia and industry due to its remarkable success in various fields. It has been integrated into wireless communication technologies, particularly in the IoT, to exploit the advantages of artificial intelligence. In this context, various studies have integrated Q-Learning (QL) into TSCH networks to meet the requirements of different applications and to develop an efficient scheduler.

Authors in [12] propose HRL-TSCH, a Hierarchical RL (HRL)-based method to schedule TSCH schedules according to IIoT requirements. The method uses two policies: one for managing TSCH links and another for slot and channel allocation. Intelligent agents optimize throughput, energy consumption, and latency simultaneously. HRL-TSCH simulations show that it outperforms existing approaches by effectively balancing performance and energy efficiency.

Authors in [13] proposed a multi-agent scheduling scheme based on the QL technique of RL, which supports contention with the objective of minimizing collisions. QL is used to find the optimal transmission slots. The obtained results show improvements in various TSCH performance metrics.

In [14], the authors proposed QL Scheduling on TDMA (QS-TDMA), a QL-based task scheduling algorithm for WSNs. This algorithm considers packet priority, total hop count, and initial deadline. QS-TDMA accounts for the real-time evolution of packets in WSNs to enhance its performance. The simulation outcomes prove that QS-TDMA is an optimal task scheduling algorithm capable of enhancing the reliability and real-time performance of WSNs.

In [15], the authors proposed QL-TSCH-plus, an improvement to the existing QL-TSCH scheduler [13]. Instead of continuously listening for neighbors' communications, which consumes energy, QL-TSCH-plus allows nodes to broadcast learned transmission slots to update action overview tables and allocate reception slots. This reduces energy consumption by up to 47% compared to QL-TSCH while maintaining reliability and speed, making QL-TSCH-plus suitable for IIoT.

Authors in [16] proposed the Low-Latency and QL-based Scheduling Function (LLQL-SF), designed for IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH) networks. This method uses QL to dynamically adapt cell allocation according to traffic needs, aiming to minimize latency. Performance was evaluated through simulation and on 30 real nodes of the FIT IoT-LAB testbed. The findings indicate that LLQL-SF outperforms existing methods with an 11% improvement in delivery rate, a 20% reduction in latency, and an 11% decrease in energy consumption. Table I summarizes the approaches mentioned above.

In this paper, we apply QL-based RL to optimize TSCH scheduling and frequency hopping. Unlike standard TSCH, which is static in Contiki-NG and dynamic in Orchestra [17], the proposal aims to intelligently assign communication slots between neighboring network nodes with frequency channel hopping independent of actual network parameters. It does so by selecting the best parent based on real network performance while also dynamically selecting the most suitable channel. This approach advances the field by offering a flexible and adaptable solution that optimizes network performance in IIoT environments.

TABLE I. SUMMARY OF RELATED WORK

Ref.	Year	Multi-agent	Latency -aware	Energy -aware	Simulation	Test bed
[12]	2024	✓	✓	✓	✓	✗
[13]	2020	✓	✓	✗	✗	✓
[14]	2018	✗	✗	✗	✓	✗
[15]	2023	✗	✗	✓	✓	✗
[16]	2024	✗	✓	✓	✓	✓

A. Primary Contributions

This paper makes the following key contributions:

- We design a QL-based RL solution for the TSCH scheduling problem in IIoT networks, which optimizes their performance.
- We design a TSCH selection algorithm that allows nodes to select the optimal scheduled link to the best parent with the most stable connection and the highest transmission success rate.
- We integrate a Routing Protocol for Low-power and Lossy Networks (RPL) neighbor balancing module to evenly distribute children among parents.
- We perform an in-depth performance analysis of the proposed approach in the Cooja network simulator, using different IIoT network parameters.

II. PROPOSED APPROACH

Integrating QL into TSCH networks for scheduling communications offers major advantages, such as the ability to dynamically adapt to changing network conditions. This is particularly useful in real-world environments where conditions are unpredictable. QL [18] is a model-free learning technique, making it suitable for complex networks where accurate network modeling is difficult or too resource-intensive. To contribute to the optimization of TSCH scheduling, our proposal uses QL to optimize the attribution of timeslots by selecting both the best parent and the best channel based on dynamic criteria such as Received Signal Strength Indicator (RSSI) and transmission success rate. Unlike standard TSCH, which assigns timeslots to all neighbors, in our solution, at each decision, a free timeslot and the best offset are assigned to the selected best parent, allowing efficient allocation of time resources without overlapping. This approach is local, distributed, and autonomous: each node makes its decisions independently, without explicit coordination with other nodes. Reception (RX) slots are therefore assigned implicitly; the parent is assumed to listen to the slots in which its children are likely to transmit.

Unlike QL-TSCH [13] and QL-TSCH-plus [15], that only address Transmission (TX) slot selection, our proposal focuses QL exclusively on parent selection and channel assignment, optimizing for both link quality and channel diversity. Slot allocation is handled separately by automatically assigning the first available (free) slot in the slotframe. This design simplifies the action space and decreases learning overhead, while remaining adaptable. The selected parent implicitly derives its RX slot from the child's TX choice, ensuring coherent scheduling without pre-synchronization or multi-slotframe logic. Additionally, our solution introduces a balanced distribution of children among parents, an aspect not addressed in either previous work. Table II presents a comparative overview of QL-TSCH, QL-TSCH-plus, and our proposed scheme.

Furthermore, the proposed technique avoids energy-intensive processes such as persistent neighbor listening and slot announcement packets. Instead, it uses local knowledge and link performance metrics (RSSI and TX success rate) to automatically make the best scheduling selections. As a result, energy consumption and complexity are decreased, while Packet Delivery Ratio (PDR) and latency remain competitive. The overall architecture of our proposal is illustrated in Figure 1.

TABLE II. COMPARATIVE SUMMARY OF QL-TSCH VARIANTS

Criteria	QL-TSCH [13]	QL-TSCH-plus [15]	Our proposal
Slotframe number	2(unicast, broadcast)	3 (unicast, RPL, broadcast)	1 (unicast optimized)
QL	TX slot	TX slot	Parent & channel
Reward function	TX success	TX success	RSSI & TX success
RX attribution	Action peeking	Broadcast	Implicit, derived from RPL link
Synchronization complexity	Medium (2 Slotframes)	High (3 slotframes)	Low (1 Slotframe)
Memory overhead	Medium	High	Low, suitable for Zolertia Z1
Dynamic adaptation	Channel	Channel & parent (according to RPL)	Channel & parent (learned)
Neighbor balancing	No	No	Yes
Platform tested	ARM Cortex-M4 MCU	Cooja (abstract simulation	Z1 (hardware emulation)

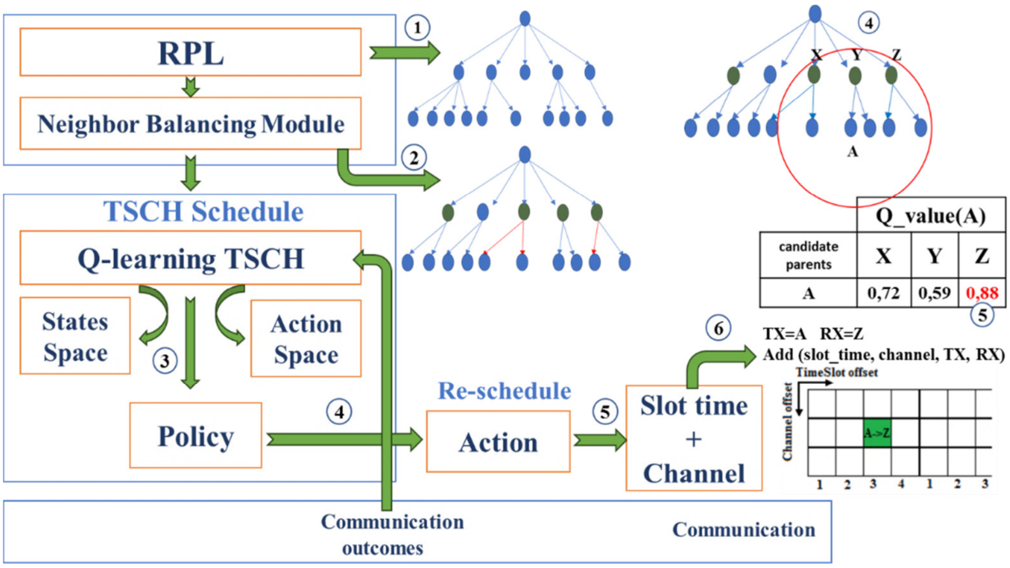

Fig. 1. Architecture of the proposed QL-based TSCH scheduling scheme.

A. Q-Learning System Space

The decision-making process of our QL-based scheduling system is formalized as a Markov Decision Process (MDP) [18]. This process provides a theoretical foundation for our approach, with the tuple $\langle S, A, P, R\rangle$ defined as follows:

- S (states): Let the global state space S be the cartisian product of $S1$ and $S2$ subspaces: $S = S1 \times S2$, where:
  - $S1$: Channel subspace, representing the quality of each available channel.
  - $S2$: Parent subspace, capturing the reliability of potential routing parents and load balancing indicators.

For each available channel $c_i \in \{1, .., M\}$:

$$s_1^i = \left(\mathrm{RSSI}(c_i), \mathrm{TXrate}(c_i)\right) \in \mathbb{R}^2$$

For each candidate parent $p_i \in \{1, .., N\}$:

$$s_2^i = \left(\mathrm{RSSI}(p_i), \mathrm{TXrate}(p_i)\right) \in \mathbb{R}^2$$

Although $S1$ and $S2$ share the same metric types (RSSI / TXrate), their semantic separation is essential: channel metrics model interference, whereas parent metrics model link reliability. This separation ensures that joint optimization can address both interference avoidance and link quality, which are crucial in dense and noisy IIoT environments:

- A (actions): The action space A consists of all possible (parent, channel) pairs: $A = \{(p_i, c_i) |\, p_i \in \{1, .., N\},\ c_i \in \{1, .., M\}\}$.

At each step t, the agent selects the action $a_t = (p_i, c_i)$ aiming to optimize:

$\max_{(p_i,c_i)}(\text{TXrate}(p_i) \times \text{RSSI}(p_i) \times \text{TXrate}(c_i) \times \text{RSSI}(c_i))$

- P (transitions): These are unknown and not explicitly modeled but are implicitly learned through interaction with the environment.
- R (reward): A weighted combination of reliability and link quality.

Although we do not perform a formal MDP analysis or provide convergence proofs, this formulation offers a theoretical foundation for our RL approach. It helps balance key IIoT trade-offs: energy, interference, and delivery reliability. This abstraction enhances both the interpretability and reproducibility of the proposed method.

B. Q-Learning Update Rule

Each node maintains a Q-table $Q(s,a)$ over the combined state s and action a. At each episode, after transitioning from state s to s' following action a and receiving reward r, the Q-values are updated via the Bellman rule [19]:

$$Q(s,a) = Q(s,a) + \alpha[r + \gamma \max Q(s',a') - Q(s,a)] \quad (1)$$

where α is the learning rate, γ is the discount factor, and r is the reward.

The agent follows an ε -greedy policy to balance exploration and exploitation during learning: with probability ε, a random (parent, channel) pair is chosen; otherwise, the action $a = \text{argmax}Q(s,a)$ is selected. Over time, ε decays, ensuring convergence to the optimal policy.

C. Reward Function

The reward function (2) is decomposed into two components: (1) link quality, measured via RSSI, and (2) transmission success rate (TXrate). This decomposition simplifies learning and supports the stable convergence of QL. Both RSSI and TXrate are directly measurable in real IIoT hardware and reflect essential physical-layer conditions.

RSSI is a low-cost, locally observable metric that reflects both interference and link attenuation. TXrate captures packet delivery reliability, including MAC layer success and acknowledgment reception.

$$Reward = \beta \times (TXrate(p_i) + TXrate(c_i)) \times f(RSSI(p_i), RSSI(c_i)) - \delta \quad (2)$$

where β and δ (set to 5 and 10, respectively, in the simulations) are the weights of the criteria , which are adjusted according to the desired objective (e.g., energy consumption or latency), and f is a function mapping RSSI values to a normalized score in [0, 1]:

$$f(\text{RSSI}) = \frac{90+\text{RSSI}}{60}, \quad \text{RSSI} \in [-90,-30] \quad (3)$$

This reward function balances three key objectives: ensuring link reliability through parent quality, reducing retransmissions through better channel selection, and implicitly promoting energy efficiency by avoiding weak or congested links.

D. Complexity Analysis

The proposed QL-based scheduler has a time and memory complexity of $O(N \times M)$, where N is the number of candidate parents and M is the number of available channels, corresponding to the size of the Q-table. The Q-table update operation occurs in constant time, $O(1)$, following each transmission feedback. Regarding communication overhead, the approach incurs $O(N)$ complexity, as it requires maintaining quality metrics for N neighbors; however, these metrics are passively extracted from standard TSCH control messages, resulting in no additional control messages, or signaling overhead.

The method provides several practical benefits in addition to computational efficiency. It dynamically adjusts parent and channel selections in response to network variations, making it highly adaptive. Additionally, it saves energy by avoiding the need for custom synchronization messages. Furthermore, the integration of a neighbor balancing mechanism ensures a fair and even distribution of children among available parents. This mechanism enforces a maximum number of X children per parent and has a time complexity of $O(N \times P)$, where N is the number of candidate parents, and P is the number of children. It iteratively checks and reassigns children from overloaded parents to underloaded ones based on candidate lists, without requiring additional messages. This controlled balancing reduces congestion and ensures stable and reliable parent-child associations.

E. TSCH Optimization Description

Our approach improves the TSCH protocol by combining a distributed neighbor balancing algorithm and RL-based channel and parent selection, enabling efficient, decentralized, and adaptive scheduling.

1) Neighbor Balancing

To ensure a fair distribution of children among candidate parents, we implemented a lightweight local balancing mechanism outlined in Algorithm 1. Each node selects a parent who has not exceeded the child limit C; otherwise, the node connects to the least loaded candidate to ensure a balanced load with minimal overhead.

```
Algorithm 1: Neighbor_Balancing
Input:
N:Children to distribute
P:Candidate parents
C:Max children per parent
Procedure:
- Initialize the number of children per
parent to 0
- For each child:
  1.Iterate through its list of candidate
    parents
  2.If a parent has less than C children,
    assign the child to that parent
  3.Else, assign the child to the least
    loaded available parent
```

```
- Update the number of children for each
  assigned parent
- Return Balanced_Neighbors
```

2) Channel and Parent Selection

Channel and parent selection are handled by the QL method described in Algorithm 2. This lightweight algorithm jointly selects the best parent-channel pair. Each node maintains a Q-table that is indexed by parent-channel pairs, and it uses an ε-greedy strategy to balance exploration and exploitation. At each transmission opportunity, the node selects the action (i.e., the parent-channel pair) with the highest Q-value unless it makes a random choice with probability ε to encourage exploration. After receiving transmission feedback (RSSI, ACK reception), the corresponding Q-value is updated according to a classical QL rule. This adaptive method allows nodes to gradually favor the most reliable links and least congested channels, promoting both energy efficiency and communication reliability.

```
Algorithm 2: QL_Selection
Input:
Q[P][C]: Q-table for all parent-channel
pairs (initialized to 0)
ε: Exploration rate
α: Learning rate
γ: Discount factor
Procedure:
- For each transmission occurrence:
  1. Action selection (ε-greedy)
     If random () < ε:  // Exploration
       p = random_select(P)
       c = random_select(C)
     Else:  // Exploitation: best known
     pair
       (p,c) = argmax(Q[P][C])
  2. Perform transmission:
       success = transmit (packet,p,c)
  3. Observe new state and calculate
     reward using (2)
  4. Update Q-table using (1)
- Return updated Q-table
```

3) Adaptative Scheduling Algorithm

TSCH scheduling, as detailed in Algorithm 3, determines which neighbor a node will communicate with and when.

```
Algorithm 3: TSCH_Schedule
Input:
P: Network nodes list
C: Available channels list
Slotframe: List of all empty timeslots
Procedure:
- Neighbor_Balancing (Algorithm 1)
- For each node:
  1. Assign a cell:
     - Assign a free timeslot in the
       slotframe
     - (Channel,Parent) = QL_Selection
       (Algorithm 2)
     - Assign TX to the node
     - Assign RX to its parent
  2. Add (slot, channel, TX, RX) to the
     planning
- Return the planning
```

The above algorithm integrates two key mechanisms that enable fully distributed scheduling:

- Local decision framework: Each node autonomously combines the neighbor balancing mechanism and QL-based channel and parent selection to create optimized transmission schedules without centralized coordination.
- Energy efficiency: Unlike standard TSCH, where parent nodes monitor all slots, energy consumption is reduced by limiting parent nodes to listening only to their assigned children's transmissions, thus minimizing unnecessary radio activity.

III. SIMULATION AND RESULTS

A. Simulation Setup

The simulation was conducted using the parameters listed in Table III and was performed in the Cooja simulator [20], which is provided by Contiki-NG [21]. This tool is widely adopted by the IoT research community. With over 1,500 citations, Cooja can fully emulate sensor network platforms such as Z1 and Tmote Sky. Due to the Z1 model's accuracy in this simulator, we chose this platform to achieve confirmed results and a more realistic IoT experimental setup in our simulations. The experiments were run on a PC with an Intel Core i7-1255U 1.70 GHz processor and 16 GB of RAM. The coordinator node, positioned at the center of the topology, acted as the RPL root and was surrounded by sensor nodes that periodically transmitted data to it.

TABLE III. SIMULATION PARAMETERS

Parameter	Value
Mote type	Z1 mote
Experiment duration	30 m
Data generation duration	30 m
Transport protocol	UDP
Routing protocol	RPL Classic
Timeslot duration	10 ms
Slotframe length	31
Channel number	16
TSCH maximum retries	5
TSCH queue size	12
Packet period	1 packet /2s
Number of nodes	10/20/30/40/50
Radio medium	UDGM
Distance between nodes	45 m
Interference range	100 m
TX range	50 m

UDGM: Unit Disk Graph Medium.

To evaluate the performance of our proposal (hereafter referred to as QL-TSCH), we conducted a comparative analysis based on a consistent simulation setting across all schemes. Specifically:

- We modified Orchestra [17] by replacing its default mechanism with our optimized channel selection function, resulting in the QL-Orchestra variant. Although this led to improved transmission performance, the overall gain remained modest.
- We conducted thorough implementations and simulations of all protocols (TSCH, Orchestra, QL-Orchestra, and QL-TSCH) under identical conditions to ensure a fair and rigorous comparison.
- We performed extensive simulations to tune the QL algorithm parameters to optimize the performance of our solution.
- To ensure statistical reliability, all simulations were repeated ten times using different random seeds. For each key performance metric, we report both the mean value and 95% Confidence Intervals (CIs). These CIs were calculated using the Student's t-distribution [22] applied to the standard deviation of the 10 simulations, providing a rigorous quantification of both central tendency and data dispersion. In addition, we conducted a one-way repeated-measures ANOVA ($\alpha = 0.05$) to evaluate whether the performance differences among QL-TSCH and the reference protocols were statistically significant. When significant differences were found, a Tukey post-hoc test was applied to identify the pairs of protocols that differed.

B. Q-Learning Parameter Selection

The QL parameters—learning rate (α), discount factor (γ), and exploration rate (ε)—were optimized through simulations of a 30-sensor node network transmitting data every 2 s. Parameter tuning was guided by end-to-end delay performance, as shown in Table IV.

TABLE IV. EVALUATION OF QL PARAMETERS BASED ON END-TO-END DELAY

ε	α	γ	End-to-end delay (s)
0.1	0.1	0.5	0.676
0.1	0.1	0.9	0.679
0.1	0.5	0.5	0.681
0.1	0.5	0.9	0.670
0.5	0.1	0.5	0.698
0.5	0.1	0.9	0.681
0.5	0.5	0.5	0.687
0.5	0.5	0.9	0.675

The lowest end-to-end delay, 0.67 s, was achieved with $\varepsilon = 0.1$, $\alpha = 0.5$, and $\gamma = 0.9$. The discount factor controls the relevance of future rewards, whereas the learning rate affects how quickly new values replace old ones.

The highest delay, 0.698 s, was observed with $\varepsilon = 0.5$, $\alpha = 0.1$, and $\gamma = 0.5$. High latency can lead to low PDR. A high PDR suggests that the network is stable and reliable, reducing delays through a reduced number of retransmissions. We concluded that the combination of $\varepsilon = 0.1$, $\alpha = 0.5$, and $\gamma = 0.9$ is significant for the remaining experiments.

C. Simulation Evaluation

The following metrics were used to assess and compare the performance of the proposed and reference protocols.

1) Retransmission Rate

The retransmission rate is the proportion of packets that must be resent at least once before being successfully delivered. As shown in Figure 2, the proposed QL-TSCH achieves a significantly lower average retransmission rate than other solutions starting from 20 nodes. For example, in the case of a network of 40 nodes, QL-TSCH achieves a mean of 9.67% with a 95% CI of $\pm$ 0.53. This is lower than the rates achieved by TSCH (11.09% $\pm$ 0.32), QL-orchestra (11.44% $\pm$ 0.25), and Orchestra (18.25% $\pm$ 1.13). Since the CIs do not overlap and the ANOVA test produces a p-value close to zero, the observed differences are statistically significant. These outcomes highlight the ability of QL-TSCH to significantly reduce retransmissions in denser network scenarios, despite the variability across topologies.

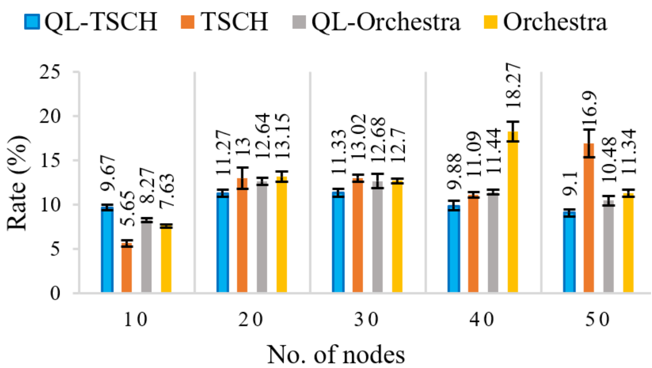

Fig. 2. Retransmission rate comparison for varying network sizes.

2) Packet Delivery Ratio

PDR is defined as the number of data packets received by the destination nodes relative to the total number of packets. Figure 3 shows that our QL-TSCH solution consistently achieves a higher average PDR, especially in networks with 10, 20, or 30 nodes (91.37% $\pm$ 0.08, 91.17% $\pm$ 0.5 and 92.48% $\pm$ 0.17, respectively). Although the performance analysis reveals that QL-TSCH achieves a higher PDR than the other protocols, regardless of the network size, ANOVA and Tukey tests revealed no statistically significant difference between these protocols (p = 0.079). Nevertheless, the results obtained by QL-TSCH are noteworthy and suggest a trend in favor of this protocol, despite the lack of statistical significance, probably due to the limited number of simulations. These results highlight the need to consider both statistical results and observed performance and encourage further testing with larger samples to confirm the potential benefits of QL-TSCH.

3) End-to-End Delay

End-to-end delay is defined as the time interval between a packet being sent by a source node and its reception by the destination node. As shown in Figure 4, the QL-TSCH solution achieves a lower average end-to-end delay (e.g., 0.69 $\pm$ 0.009 s for 50 nodes) than TSCH (0.7 ± 0.009 s), QL-orchestra (0.7 ± 0.011 s), and Orchestra (0.69 ± 0.026 s). Despite this favorable trend, the overlapping CIs indicate that, without formal hypothesis testing, the statistical significance of these differences cannot be conclusively established. The results of the ANOVA performed across all network sizes reveal a significant difference among the protocols ($p < 0.01$). The Tukey post-hoc test further confirms that the pairwise differences are statistically significant, with QL-TSCH outperforming both QL-Orchestra and TSCH protocols ($p < 0.05$), whereas no significant difference was observed between QL-Orchestra and TSCH. The low latency achieved by QL-TSCH is attributed to (1) efficient slot scheduling that minimizes idle listening and (2) interference-aware channel selection. These optimizations are realized without compromising PDR, demonstrating a good trade-off between speed and reliability.

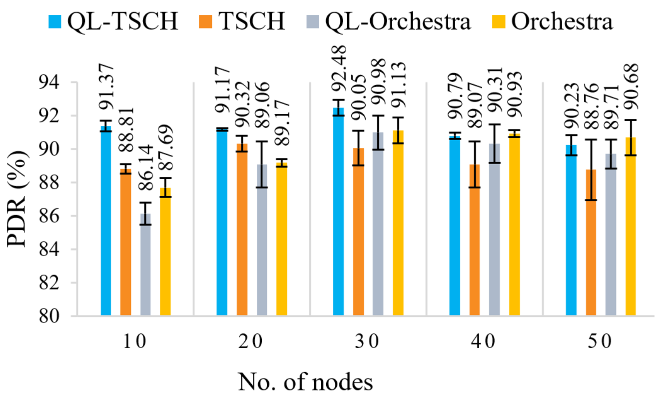

Fig. 3. PDR comparison for varying network sizes.

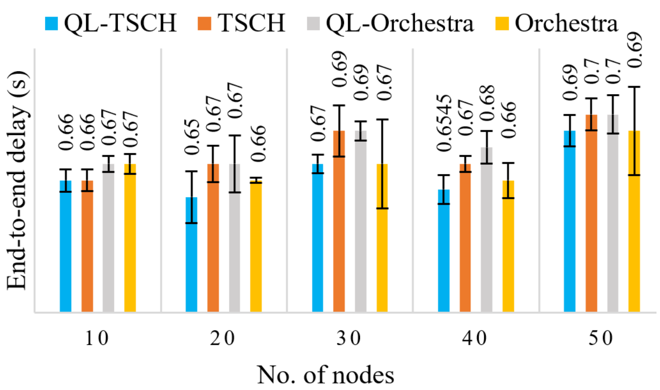

Fig. 4. End-to-end delay comparison for varying network sizes.

4) *Energy Consumption*

In this study, we only evaluated the energy consumption only through the radio ON time [20], which represents the main source of consumption in TSCH networks. The radio is the most energy-consuming hardware element in a Z1 mote.

As shown in Figure 5, energy consumption increases with network density. However, our QL-TSCH solution reduces energy consumption consistently across all simulated networks. The 95% CI of QL-TSCH in the networks with 20, 30, 40, and 50 nodes (21.75 ± 1.50 J, 26.67 ± 0.98 J, 27.73 ± 2.53 J, and 31.56 ± 1.2 J, respectively) shows no overlap with the other protocols. Statistical analyses using ANOVA followed by a Tukey post-hoc test confirm that the differences among all protocols are statistically significant ($p < 0.001$) for each network size. The consistent results across configurations highlight the robustness and statistical significance of our proposal as the most energy-efficient protocol.

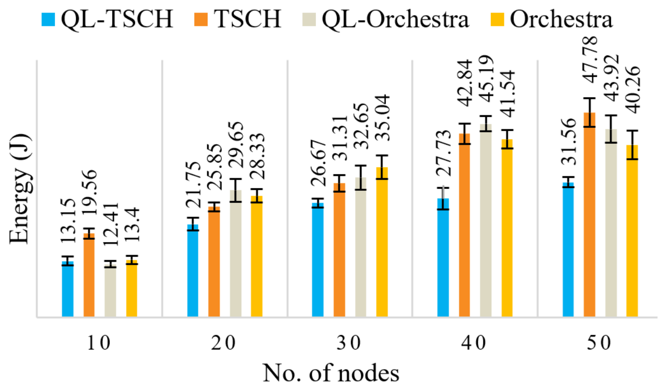

Fig. 5. Energy consumption (radio ON time) comparison for varying network sizes.

5) *Throughput*

Throughput is defined as the amount of data successfully received by the destination nodes per unit time. As shown in Figure 6, the throughput achieved by our solution is slightly lower than that of other protocols across all network sizes (e.g., in a 30-node network, QL-TSCH achieves a mean throughput of 627 ± 11.46 B/s, whereas TSCH, QL-Orchestra, and Orchestra reach 627 ± 11.46 B/s, 726 ± 12.92 B/s, 616 ± 17.34 B/s, and 788 ± 17.45 B/s, respectively).

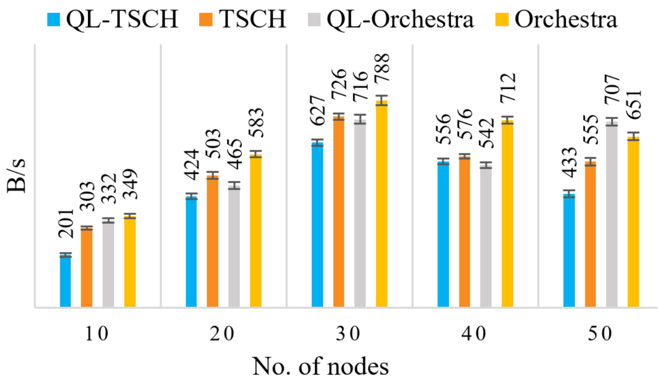

Fig. 6. Throughput comparison for varying network sizes.

Moreover, the statistical tests confirm the significance of these differences: ANOVA results indicate that QL-TSCH performs significantly worse than TSCH and Orchestra across all network sizes ($p < 0.01$). This decrease is attributed to contention on transmission cells when multiple children communicate with the same parent. Although this affects peak throughput, the trade-off prioritizes load balancing and transmission reliability over row data rate in constrained environments. This choice aligns with the critical requirements of IIoT networks, where stability and energy consumption often outweigh high throughput. To address this limitation, future work will explore a two-tier approach. After parent selection, a local cell management mechanism will be introduced to reduce contention. This mechanism could include dynamic slot reallocation or transmission cell duplication. Additionally, we plan to extend the reward function by incorporating slot utilization metrics, enabling more efficient use of underloaded cells.

IV. CONCLUSION AND FUTURE WORK

This paper presented a fully distributed, autonomous learning approach for Time-Slotted Channel Hopping (TSCH) scheduling that requires no central coordinator. For the first time, the proposed approach integrates a neighbor-balancing parent selection mechanism. Specifically, our approach optimizes two critical tasks simultaneously: neighbor balancing and parent and channel assignment. Each node maintains a dynamically updated list of candidate parents that is balanced to prevent any single parent from becoming overloaded. A distributed Q-Learning (QL) agent then learns, in real time, which channel maximizes link quality for the best parent. By adapting the Reception (RX) cell allocation to restrict each parent's listening to only its own children's cells, unnecessary idle listening is eliminated, and energy waste is reduced.

The proposed solution achieves significant improvements in radio efficiency: retransmissions are greatly reduced, Packet Delivery Ratio (PDR) exceeds 90%, average latency remains below 0.7 s, and power consumption decreases appreciably across various Industrial Internet of Things (IIoT) topologies. The superiority of QL-TSCH was statistically validated through ANOVA and Tukey tests ($p < 0.05$), confirming that the observed improvements are significant and not due to random variation. These results demonstrate the suitability of our QL-TSCH method for IIoT environments, highlighting the benefits of jointly optimizing all scheduling components within a Markov Decision Process (MDP)-based QL framework beyond previous independent efforts.

Future work will focus on addressing throughput limitations by integrating channel and slot optimization, while preserving energy-efficient parent selection. Physical deployments on industrial testbeds will complement simulations to provide quantitative validation of robustness, latency, and dynamic adaptation.

REFERENCES

[1] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, Opportunities, and Directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, Nov. 2018, https://doi.org/10.1109/TII.2018.2852491.

[2] *IEEE Standard for Low-Rate Wireless Networks*, Apr. 2016. https://doi.org/10.1109/IEEESTD.2016.7460875.

[3] A. Vatankhah and R. Liscano, "Comparative Analysis of Time-Slotted Channel Hopping Schedule Optimization Using Priority-Based Customized Differential Evolution Algorithm in Heterogeneous IoT Networks," *Sensors*, vol. 24, no. 4, Feb. 2024, Art. no. 1085, https://doi.org/10.3390/s24041085.

[4] A. R. Raut, S. Khandait, and D. Theng, "A Priority based Self-Organised MAC Protocol for Real Time Wireless Sensor Network Applications," *Engineering, Technology & Applied Science Research*, vol. 14, no. 6, pp. 18600–18607, Dec. 2024, https://doi.org/10.48084/etasr.8459.

[5] M. Kherbache, O. Sobirov, M. Maimour, E. Rondeau, and A. Benyahia, "Decentralized TSCH scheduling protocols and heterogeneous traffic: Overview and performance evaluation," *Internet of Things*, vol. 22, Jul. 2023, Art. no. 100696, https://doi.org/10.1016/j.iot.2023.100696.

[6] A. R. Urke, Ø. Kure, and K. Øvsthus, "A Survey of 802.15.4 TSCH Schedulers for a Standardized Industrial Internet of Things," *Sensors*, vol. 22, no. 1, Jan. 2022, Art. no. 15, https://doi.org/10.3390/s22010015.

[7] F. Righetti, C. Vallati, S. K. Das, and G. Anastasi, "Analysis of Distributed and Autonomous Scheduling Functions for 6TiSCH Networks," *IEEE Access*, vol. 8, pp. 158243–158262, 2020, https://doi.org/10.1109/ACCESS.2020.3020153.

[8] P. Li, Y. Xiao, J. Yan, X. Li, and X. Wang, "Reinforcement Learning for Adaptive Resource Scheduling in Complex System Environments," in *2024 5th International Symposium on Computer Engineering and Intelligent Communications*, Wuhan, China, 2024, pp. 92–98, https://doi.org/10.1109/ISCEIC63613.2024.10810157.

[9] T. Chang, T. Watteyne, Q. Wang, and X. Vilajosana, "LLSF: Low Latency Scheduling Function for 6TiSCH Networks," in *2016 International Conference on Distributed Computing in Sensor Systems*, Washington, USA, 2016, pp. 93–95, https://doi.org/10.1109/DCOSS.2016.10.

[10] I. F. Vieira Junior, J. Granjal, and M. Curado, "Insights for metrics in assessing TSCH scheduling efficiency," *Internet of Things*, vol. 29, Jan. 2025, Art. no. 101440, https://doi.org/10.1016/j.iot.2024.101440.

[11] P. Thubert, "An Architecture for IPv6 over the Time-Slotted Channel Hopping Mode of IEEE 802.15.4 (6TiSCH)," Internet Engineering Task Force, Request for Comments RFC 9030, May 2021. https://doi.org/10.17487/RFC9030.

[12] F. F. Jurado-Lasso, C. Orfanidis, J. F. Jurado, and X. Fafoutis, "HRL-TSCH: A Hierarchical Reinforcement Learning-Based TSCH Scheduler for IIoT," *IEEE Transactions on Cognitive Communications and Networking*, vol. 10, no. 6, pp. 2102–2118, Dec. 2024, https://doi.org/10.1109/TCCN.2024.3408459.

[13] H. Park, H. Kim, S.-T. Kim, and P. Mah, "Multi-Agent Reinforcement-Learning-Based Time-Slotted Channel Hopping Medium Access Control Scheduling Scheme," *IEEE Access*, vol. 8, pp. 139727–139736, 2020, https://doi.org/10.1109/ACCESS.2020.3010575.

[14] B. Zhang, W. Wu, X. Bi, and Y. Wang, "A Task Scheduling Algorithm Based on Q-Learning for WSNs," in *Communications and Networking: 13th EAI International Conference, ChinaCom 2018*, Chengdu, China, 2018, pp. 521–530, https://doi.org/10.1007/978-3-030-06161-6_51.

[15] M. Kherbache, M. Maimour, and E. Rondeau, "QL-TSCH-plus: A Q-learning Distributed Scheduling Algorithm for TSCH Networks," in *2023 IEEE 9th World Forum on Internet of Things*, Aveiro, Portugal, 2023, pp. 1–6, https://doi.org/10.1109/WF-IoT58464.2023.10539552.

[16] Y. H. Pratama, S.-H. Chung, and D. Z. Fawwaz, "Low-Latency and Q-Learning-Based Distributed Scheduling Function for Dynamic 6TiSCH Networks," *IEEE Access*, vol. 12, pp. 49694–49707, 2024, https://doi.org/10.1109/ACCESS.2024.3384869.

[17] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne, "Orchestra: Robust Mesh Networks Through Autonomously Scheduled TSCH," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, Seoul, South Korea, 2015, pp. 337–350, https://doi.org/10.1145/2809695.2809714.

[18] N. Zerguine, M. Mostefai, Z. Aliouat, and Y. Slimani, "Intelligent CW Selection Mechanism Based on Q-Learning (MISQ)," *Ingénierie des Systèmes d'Information*, vol. 25, no. 6, pp. 803–811, Dec. 2020, https://doi.org/10.18280/isi.250610.

[19] C. J. C. H. Watkins and P. Dayan, "Technical Note: Q-Learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1992, https://doi.org/10.1023/A:1022676722315.

[20] C. Thomson, I. Romdhani, A. Al-Dubai, M. Qasem, B. Ghaleb, and I. Wadhaj, *Cooja Simulator Manual*, 2016, https://doi.org/10.13140/RG.2.1.4274.8408.

[21] G. Oikonomou, S. Duquennoy, A. Elsts, J. Eriksson, Y. Tanaka, and N. Tsiftes, "The Contiki-NG open source operating system for next generation IoT devices," *SoftwareX*, vol. 18, Jun. 2022, Art. no. 101089, https://doi.org/10.1016/j.softx.2022.101089.

[22] D. C. Montgomery and G. C. Runger, *Applied Statistics and Probability for Engineers*, 7th ed. Hoboken, NJ, USA: Wiley, 2018.