Latency and Lifetime Enhancements in IWSN: a Q-Learning Approach for Graph Routing

Gustavo Künzel, Leandro Soares Indrusiak, Carlos Eduardo Pereira

***Abstract*—Industrial Wireless Sensor Networks usually have a centralized management approach, where a device known as Network Manager is responsible for the overall configuration, definition of routes, and allocation of communication resources. Graph routing is used to increase the reliability of the communications through path redundancy. Some of the state-of-the-art graph routing algorithms use weighted cost equations to define preferences on how the routes are constructed. The characteristics and requirements of these networks complicate to find a proper set of weight values to enhance network performance. Reinforcement Learning can be useful to adjust these weights according to the current operating conditions of the network. We present the Q-Learning Reliable Routing with a Weighting Agent approach, where an agent adjusts the weights of a state-of-the-art graph routing algorithm. The states of the agent represent sets of weights, and the actions change the weights during network operation. Rewards are given to the agent when the average network latency decreases or the expected network lifetime increases. Simulations were conducted on a WirelessHART simulator considering industrial monitoring applications with random topologies. Results show, in most cases, a reduction of the average network latency while the expected network lifetime and the communication reliability are at least as good as what is obtained by the state-of-the-art graph routing algorithms.**

***Index Terms*—Industrial Wireless Sensor Networks, Routing, Reinforcement Learning, Q-Learning, WirelessHART.**

I. Introduction

INDUSTRIAL Wireless Sensor Networks (IWSN) are an attractive technology for communication in process automation and allow the incorporation of Industrial Internet of Things (IIoT) and Industry 4.0 (I4.0) concepts [1], [2]. The global IWSN market size is anticipated to reach USD 8.67 billion by 2025 [3]. Flexibility, mobility, scalability, low maintenance and reduced infrastructure are advantages of IWSN [4], [5]. An IWSN consists of a set of wireless sensor devices (nodes) connected to a gateway through Access Points (AP). The gateway provides a connection with the plant automation network. A device known as Network Manager (NM) is connected to the gateway and is responsible for the management of the network, admission control, configuration, routing, and scheduling. Centralized management is used to allow better control of the network operation and also to simplify the hardware and software of the nodes. WirelessHART (WH), ISA SP100.11a and WIA-PA standards are being used in IWSN applications [4]. These standards usually form a mesh network, where nodes may act as routers to increase path availability for communications [6]. IWSN applications typically require reliable, low-latency and real-time communications. Low energy consumption is another requirement as batteries are often used to power devices [1]. Meeting these requirements and optimizing the network performance are often complex tasks because of the characteristics of the devices, topologies, and the wireless network properties (shared medium, interference, signal reflections, and signal strength) [5]–[8].

Routing is an essential task of the NM. The routes built by the NM influence the reliability of the communications, latency, energy consumption, transmission errors and resource usage [1], [6]. Path redundancy is used to increase reliability and is implemented using graph routing. A graph is a route that connects nodes in the network, and each intermediate node in a route to the destination may have multiple neighbors to forward a message to. If the communication with a neighbor fails, a node can try to send the message through another neighbor [9]. Graph routing algorithms for centralized management protocols were described over the last decade [9]–[17]. Parameters, heuristics, and weighted cost equations are used to choose the connections on the graphs. Usually, these weight values and parameters are statically defined and suitable only for certain network conditions [6]. Manually adjusting these parameters, aiming to improve the network performance, is inconvenient. It needs several tests, requires periodic monitoring of the network conditions, the user must know about the properties of the algorithms, and a network system representation is often unavailable [10]. It would be significant if these adjustments could be made in a way that achieves an adaptation according to the current network operational conditions while balancing or optimizing some performance metrics. Centralized routing algorithms that can optimize the performance of the IWSN are a relevant research topic.

In this context, the use of Machine Learning (ML) for creating and adjusting routes in a centralized way may be useful for IWSN and future IIoT and I4.0 protocols [10], [18]. ML provides a system the ability to learn and improve from experience, and Reinforcement Learning (RL) relies on the existence of an agent that acts in an environment and receives

Manuscript received February 25, 2019; revised May 20, 2019; accepted September 9, 2019. This study was financed in part by the Federal Institute of Science, Technology and Education of Rio Grande do Sul; in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Paper no. TII-19-2090. *(Corresponding author: Gustavo Künzel.)*

G. Künzel is with the Federal Institute of Science, Education and Technology of Rio Grande do Sul (IFRS), Farroupilha 95174-274, Brazil, (e-mail: gustavo.kunzel@farroupilha.ifrs.edu.br)

L.S. Indrusiak is with the Department of Computer Science, University of York, York YO10 5GH, UK (e-mail: leandro.indrusiak@york.ac.uk)

C.E. Pereira is with the Department of Electrical Engineering, Federal University of Rio Grande do Sul (UFRGS), Porto Alegre 90035-190, Brazil (e-mail: cpereira@ece.ufrgs.br)

Digital Object Identifier:

rewards based on the results of its actions. By exploring the environment, it learns which behavior it must take to maximize its rewards [19]. RL demands low computational resources and implementation efforts, thus providing high flexibility to topological changes and near-optimal results, without requiring any *apriori* network model [18]–[20]. RL algorithms like Q-Learning are being used for centralized and decentralized routing approaches in general-use network technologies and Wireless Sensor Networks (WSN) [20]–[22]. In decentralized approaches, each node is modeled as a learning agent that selects routes to forward its packets. These approaches are not suitable for IWSN, since they require nodes to exchange information independently, reconfigure and decide its routing strategies [10]. Similarly, the available centralized approaches are not suitable for IWSN since they are intended to be used for other protocols and do not build graphs or routes with path redundancy.

This paper presents the Q-Learning Reliable Routing with a Weighting Agent (QLRR-WA) algorithm. QLRR-WA builds a routing graph used by nodes to send information to a gateway. During the construction of the graph, nodes and neighbors (edges of the graph) are selected through a weighted cost equation. A set of weights defines how some topology and device characteristics influence the cost values. Periodically, an agent acts in the environment trying to learn a set of weights that optimizes the network performance. The Q-Learning states are modeled as a fixed set of weights, and the actions represent the increase or reduction of weights. Rewards are given when the agent decreases the Average Network Latency (ANL) or increases the Expected Network Lifetime (ENL). To increase reliability, QLRR-WA tries to build an uplink graph where nodes have at least two neighbors to forward data to the gateway. QLRR-WA was evaluated using a WH stack implemented over the Network Simulator 2 (NS2) [23]. QLRR-WA was compared against other uplink routing algorithms using ANL, ENL, Packet Delivery Ratio (PDR) and the Percentage of Reliable Nodes (PRN) as performance metrics. The tested topologies consisted of nodes with different types of power sources randomly scattered over an area.

The main contribution of this paper is the QLRR-WA algorithm, which builds the uplink routing graph in a centralized manner suitable for IWSN protocols such as WH, using Q-Learning to enhance the network performance. Other contributions are the discussion of the implications of the use of Q-Learning for graph routing in centralized approaches, the performance evaluation and the comparison of the state-of-the-art routing algorithms. The remainder of this paper is organized as follows. Section II presents the literature review. Section III describes the QLRR-WA algorithm. Section IV presents the simulation setup and performance evaluation. Section V presents the conclusions and future works.

II. Literature Review

A. Network Manager tasks

In an IWSN, the NM has a sequence of management tasks that must be executed. These tasks run when the topology changes, when some operational conditions change, or periodically for optimization purposes. The routing algorithms that will provide the routes used by the gateway and nodes to exchange information are first executed. Then, the scheduling algorithms translate these routes in a sequence of timeslots (links) for communication. Routes and links are then converted into a sequence of commands sent to the nodes to update the network configuration [1], [9]. Sending these commands causes a communication overhead. Some implementations of the NM reduce overhead by comparing the old and new routes and links and updating the changes only [9], [23]. To ensure path availability during reconfiguration, the new routes and links are first configured, and only then the old ones are removed [23]. NM must keep information about the current network topology, nodes, and operating conditions to properly run these tasks. An overview of IWSN requirements, WH protocol, and management tasks can be found in [1], [6], [9], [23].

B. Graph routing algorithms

Three graph types are defined for IWSN: broadcast, allowing the gateway to send messages to all nodes; uplink, allowing nodes to send messages towards the gateway; and downlink, allowing the gateway to send messages to specific nodes [6], [9]. ELHFR [12] produces an uplink graph based on a Breadth First Tree (BFS) and uses the Received Signal Level (RSL) information to select neighbors. The distance in hops from the gateway is used to build graphs in [9], reducing latency and resource usage. Communication load, energy consumption, and residual energy are used by nodes to choose which neighbors they will use to forward a message, considering that a graph was already given [15]. Residual energy, link quality, and node degree are used to calculate a priority for neighbor selection [14]. A survey on routing and scheduling for WH suggests the use of adaptive weighted cost functions [6]. Network lifetime maximization is formulated as a greedy heuristic and as an optimization problem that requires intensive processing [13]. Quality of service is achieved by estimating the reliability and delay using link quality information [16]. A greedy algorithm builds graphs where nodes and edges are selected through a weighted cost equation that uses the number of hops, RSL, number of neighbors, and node power source type [11]. Primary and redundant paths are chosen based on the residual energy of neighbors to balance energy consumption in [17], but the configuration of a preferred neighbor for a node is not available in the WH standard. Few of these approaches adjust parameters autonomously or try to improve network performance over time. These approaches does not try to reduce latency and increase lifetime at the same time. Some of the algorithms require changes in the WH standard. Besides, the works are not compared using similar conditions or a simulator with a complete stack of an IWSN protocol such as WH.

C. Reinforcement Learning and Q-Learning

RL is an ML approach where a learning agent is usually modeled through a tuple consisting of a set of states S, actions A, and rewards R. At each iteration t, the agent takes an action $a_t \in A$ that leads the environment to the state $s_{t+1} \in S$. The

agent receives a reward $r_{t+1} \in R$ based on the effects of the last action taken and associates the reward with the action a_t and state s_t. The main goal of the agent is to maximize its long-term rewards, and after several iterations, the agent learns which actions it should take on each state to achieve this goal. Fig. 1 presents the interaction between an agent and the environment.

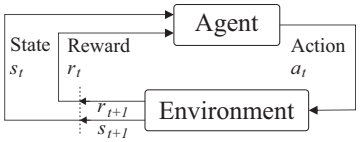


Fig. 1: Agent-environment interaction in RL [19].

The selection of the action to be taken is made through exploration or exploitation. When the agent explores the environment, it selects random actions to extend the knowledge concerning the rewards. When the agent exploits, it chooses actions where the expected rewards are already known. The balance between exploration and exploitation helps to increase the rewards accumulated over time, and the agent tries a variety of actions and progressively favors those that seem to give better rewards. ε-greedy is an approach to balance exploration and exploitation where the agent explores with a probability $0 \leq \varepsilon \leq 1$ all available actions in a state and exploits with a probability $1 - \varepsilon$ the best action.

Q-Learning is an RL algorithm where a table with size $|S| \times |A|$ store Q-values, which are the long-term rewards that an agent can expect to receive by taking action a_t in state s_t. The Q-values are updated at each iteration $t + 1$, using the previously stored Q-value and the new reward r_{t+1} received from the environment following Eq. 1.

$$Q_{t+1}(s_t, a_t) \leftarrow \quad (1-\alpha)Q_t(s_t, a_t) + \alpha\left[r_{t+1}(s_{t+1}) + \gamma \max_{a \in A} Q_t\left(s_{t+1}, a\right)\right] \tag{1}$$

In Eq. 1, $0 \leq \alpha \leq 1$ is the learning rate, which makes the agent give preference for immediate rewards. Higher values of α tend to make the learning process susceptible to environmental perturbations. The discount factor $0 \leq \gamma \leq 1$ allows the agent to adjust its preference for long-term rewards because the future reward expected in the state s_{t+1} is considered. The Q-table usually initiates with zero values or random values [19]. The states, actions, rewards, and exploration approach for a specific problem are defined during design time and must consider the problem characteristics [19]. Further details of RL and the Q-Learning algorithm can be found in [18]–[21].

D. RL applied to routing

A survey on RL routing approaches for networks was recently presented in [21], but the centralized approaches described are not related to IWSN. A literature review in RL approaches for WSN networks is presented in [24] and a survey in [20] describes three main decentralized RL approaches for WSN, where each node has an agent to choose routes. In Q-routing, states are defined as the destination, actions are the next-hop neighbor, and rewards are calculated based on the information exchanged by nodes. In Multi-Agent Reinforcement Learning, nodes exchange information about its rewards with neighbors. In Partial Observable Markov Decision Processes, a node estimates its state using data from neighbors. In [25], a source node broadcasts a transmission request to the destination node, which collects topology information, simulates the transmission of the packet on a virtual topology and creates a path using Q-Learning. The path is then sent to the source node. In [10], an agent changes the value of the weight of the power source type of nodes in a cost equation used to build a broadcast graph. States store the current weight, actions keep or change the state, and rewards are given only when the agent reduces latency and increases lifetime. The use of actions that lead to the same state and the given rewards increase the worst-case complexity of the RL problem and require more exploration [26]. Also, link quality information is not used to define routes, and the simulations do not use an error model in the physical layer, and therefore do not provide proper information about reliability. To the best of our knowledge, the current works are not suitable for centralized IWSN because the decentralized approaches require nodes to choose routes independently, and the centralized approaches are used for other communication technologies and do not build uplink graphs that address the IWSN requirements.

III. Q-Learning Reliable Routing with a Weighting Agent

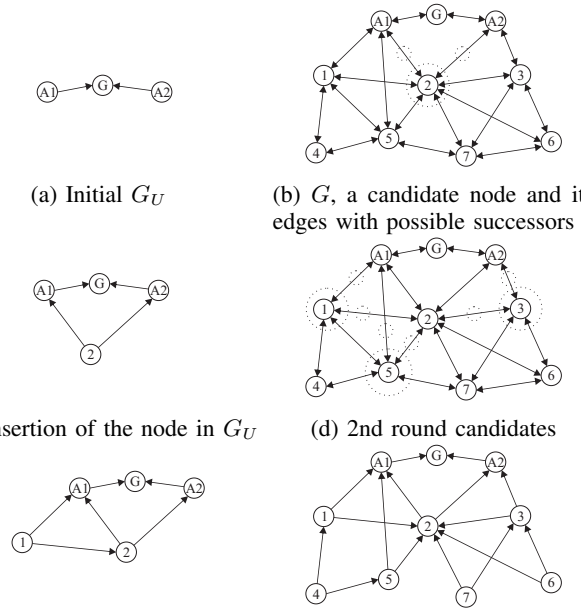
A. Scope and definitions

We focus on the construction of the uplink graph used by nodes to send sensor readings towards the gateway in IWSN monitoring applications. Static topologies are considered because IWSN are commonly planned topologies, having low-mobility nodes, and some nodes may be powered by batteries [6]. The network operates with a given topology during the simulations. Nodes can inform the NM about poor connections with neighbors. NM removes these connections from the network topology. We evaluate QLRR-WA considering three requirements of IWSN applications: low latency, low energy consumption, and high reliability [1], [6]. The following metrics are defined to evaluate these requirements, considering IWSN monitoring applications and the literature. The latency of a data packet is defined as the time interval between the generation of the packet at the sensor's Network Layer and the reception at the gateway's Medium Access Control layer. The ANL is calculated by measuring the latency of all data packets received at the gateway over a time interval t_s. The energy consumption is evaluated using the ENL, defined as the minimum expected lifetime value between all battery-powered nodes at a specific instant [13]. Reliability is evaluated using PDR, which is the percentage of data packets received at the gateway in comparison to the ones generated at the nodes, and PRN, which is the percentage of nodes which have at least two neighbors to forward data in the uplink graph [9], [13].

B. Uplink Graph Construction

The greedy algorithms in [9]–[11] are used as a baseline for QLRR-WA, where nodes and edges are iteratively added to the uplink graph and selected through a weighted cost equation.

The current network topology graph is $G(V, E)$, where V represents the devices such as nodes, V_{AP} is the set of AP, and g the gateway. E contains edges representing the available connections between devices. Fig. 2b depicts an example of G. The uplink graph $G_U(V_U, E_U)$ consists of a set of nodes V_U added while building G_U, and E_U, which is a subset of E with edges connecting the nodes towards g. An edge from a node v to a node u (successor) is represented by $e_{v,u}$. Fig. 2 depicts some of the G_U construction steps.



(a) Initial G_U

(b) G, a candidate node and its edges with possible successors

(c) Insertion of the node in G_U

(d) 2nd round candidates

(e) Insertion of the lowest-cost candidate in G_U (2nd round)

(f) Finished G_U

Fig. 2: Uplink graph construction sequence in QLRR-WA [11]

Alg. 1 describes the construction of G_U. It is assumed that nodes disconnected from the topology are previously removed from G. At line 4, g, V_{AP}, and the edges from V_{AP} to g are added to G_U, as depicted in Fig. 2a. QLRR-WA then goes to a loop that adds all nodes in $V - V_U$ to G_U. First, the candidate nodes S' which have at least two possible successors in G_U are identified, as depicted in Fig. 2b. The successor nodes U_v of a node v and the candidate nodes in S' are assigned a cost c using Eq. 2.

$$c = w_h \frac{\overline{h}}{\overline{h}_{max}} + w_p p + w_s \frac{\min(s - s_d, 0)}{s_d} \tag{2}$$

The average number of hops $\overline{h}$ of a successor or node is given by the average hops of its successors in G_U plus one. When evaluating the successors in U_v, $\overline{h}_{max}$ stores the highest value of $\overline{h}$ between all successors, p is a constant value that is associated with the energy source type of the successor, s is the RSL value of the edge between node v and the successor. s_d is a constant value which gives a desirable level for the RSL. When all successors in U_v have their costs evaluated, they are sorted and then the two lower-cost successors u_1 and u_2 are selected for candidate node v. Then, the candidate nodes in S' are evaluated to choose one of them to be added to G_U with the edges to their selected successors. For the evaluation of a candidate node, $\overline{h}_{max}$ is the maximum value of $\overline{h}$ between all the candidate nodes in S', p is associated with the energy source type of v, and s is given by the average RSL of the edges with u_1 and u_2. If $|S'| = 0$, the set of nodes S'' with only one successor is identified and then the node which has the maximum number of ingoing edges from $V - V_U$ is chosen to maximize the chance of $|S'| > 0$ in the next round. By changing the values of the weights w_h, w_p, and w_s, it is possible to define how nodes and successors will be selected. Increasing w_h will reduce the distance in hops from nodes to g and thus the use of communication resources [9]. Increasing w_p will make nodes avoid choosing battery-powered nodes as successors [10]. Increasing w_s will make nodes connect to successors with greater RSL, thus reducing the probability of packet transmission failures [11], [27].

Algorithm 1: Building the Uplink Graph with QLRR-WA

1 Calculate reward r_{t+1} according to Eq. 3
2 Update $Q_{t+1}(s_t, a_t)$ according to Eq. 1
3 Select next action a_{t+1} using ε-greedy and set the weights w_h, w_p, w_s according to a_{t+1}
4 $V_U = g \cup V_{AP}$ and E_U contains all edges from V_{AP} to g.
5 **while** $V_U \neq V$ **do**
6 Find $S' \subseteq V - V_U : \forall v \in S'$, v has at least 2 edges to V_U
7 **if** $S' \neq \emptyset$ **do**
8 **for all** $v \in S'$ **do**
9 Store in U_v the successors of the edges $e_{v,u}$ to V_U
10 Sort U_v with Eq. 2, choose e_{v,u_1} and e_{v,u_2}
11 $\overline{h}_v = 1 + \frac{\overline{h}_{u_1} + \overline{h}_{u_2}}{2}$
12 Choose the node v with min c using Eq. 2
13 Add v to V_U and add e_{v,u_1} and e_{v,u_2} to E_U
14 **else**
15 Find $S'' \subseteq V - V_U : \forall v \in S''$, v has 1 edge $e_{v,u}$ to V_U
16 **for all** $v \in S''$ **do**
17 $\overline{h}_v = \overline{h}_{u_1} + 1$
18 Calculate n_v, the # of ingoing edges from $V - V_U$
19 Choose the node v with maximum n_v

C. Q-Learning Weighting Agent model

At each periodic execution of the tasks of the NM, the agent acts choosing a set of weights to be used before rebuilding G_U (Alg. 1, lines 1-3). At the next periodic execution, the agent access the performance metrics and receives a reward. A set of states was defined, and each state has a fixed set of values for w_h, w_p, and w_s, and $w_h + w_p + w_s = 1$. A step factor $0 \leq s_f \leq 1$ defines how much the value of the weights changes from state to state and is given by $s_f = m^{-1}$, where m is an integer that represents how many transitions between values each weight will have. This allows a trade-off between the number of available states (influencing the exploration time) and the change of the cost values (influencing the number of changes in G_U from state to state). Actions represent the increment or reduction of the weights from one state to another, but actions available in one state only allow transitions to states where the values of the weights change $\pm s_f$. Fig. 4 depicts an example of the states, the effect of the actions, and the weights when $m = 7$ (states where a weight assumes zero value were suppressed). The available actions are numbered according to the increment or reduction of the weights.

In our approach, the reward has the main objective of reducing the ANL while increasing the ENL. The ANL in

a specific iteration t of the agent is indicated as d_t, while the ENL is indicated as l_t. The current ANL and ENL values are indicated as d_{t+1} and l_{t+1}. Two arrays D and L keep the last measurements of d and l. D and L store the values of the least k iterations of the agent. The use of these arrays allows the agent to receive rewards not only based on the current ANL and ENL, but to compare the current values over a time window. A positive reward of value R is given if the ANL has decreased and the ENL has increased in comparison with $\min(D)$ and $\min(L)$ over the last iteration of the agent; a positive reward of value $R/2$ is given if ANL has decreased or ENL has increased; no reward otherwise. Eq. 3 describes the rewards given to the agent.

$$r_{t+1} = \begin{cases} R, & \text{if } d_{t+1} < \min(D) \text{ and } l_{t+1} > \min(L) \\ \frac{R}{2}, & \text{if } d_{t+1} < \min(D) \text{ or } l_{t+1} > \min(L) \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

D. QLRR-WA use considerations

The impacts of exploration and the learning time must be considered when using QLRR-WA. Fluctuations in the latency are expected because of the following reasons: the Q-Learning parameters and randomness of the exploration; changes in G_U require reconfiguration commands to be sent over the network, causing overhead; commands messages may have priority over data messages [7]; the number of hops that a message takes to reach the gateway may change; the number of messages waiting in the transmission stacks of the nodes may increase; and poor connections with neighbors may be chosen. When considering these fluctuations, several data process messages should be received to measure the ANL. The exploration will also reduce the ENL because of the energy spent with overhead, and this influence will be more significant when low-capacity batteries are used. Fluctuations may be tolerable only during the network's startup and maintenance, but not during process monitoring and control. In applications where variations in the latency are acceptable, the agent can continuously explore.

The learning time is influenced by the reconfiguration and the agent model. The time between the iterations of the agent must allow the network to reconfigure properly and the measurements of ANL and ENL to represent stabilized conditions. In the current protocols, reconfiguration may take from seconds to tens of minutes. Changes in the topology are usually informed by health reports and path down alarms, also influencing the time needed for reconfiguration. The number of iterations required depends on the complexity of the learning model, which is influenced by the number of states, actions, and the choice of the rewards function [19], [20], [26]. The agent should explore several times the available actions in each state, requiring many iterations to converge [21], [26]. The actions of the agent will also be associated with the schedule changes because the changes in G_U will cause different reconfiguration patterns on the timeslots depending on the scheduling strategy. Thus, a proper combination of routing and scheduling algorithms is needed to enhance performance [6].

IV. Performance Evaluation

A. Simulation setup

We conducted simulations using the WH simulator over NS-2 [23], which was validated through comparison with real WH networks [6], [23]. The simulation parameters used are similar to the works described in Section II related to IWSN monitoring applications using WH. The energy model [6] and the simulator changes [10] were used to allow the NM to poll the battery type and the current expected battery lifetime from nodes. In the WH protocol, the battery lifetime is reported through an integer value that represents days and reduces slightly from one report to another. Because IWSN are subject to different conditions of the wireless channel, we included a general path loss model for RSL estimation (Two-Ray Ground-Reflection) with power transmission of 0 dBm with a maximum communication range of 40 m [23]. A packet loss probability model for indoor environments was included on the physical layer and uses the same transceiver family of the energy model [27]. We added to the application layer the Alarm Path Down command, so nodes can report broken links with neighbors to the NM when a Keep-Alive message is not exchanged between linked nodes after a certain amount of time. We also adapted the simulator to measure latency and PDR by monitoring the Absolute Slot Number snippet available in the Network Layer of the WH, and to calculate the PRN of G_U. Topologies were built consisting of: one NM/gateway positioned in the center of a 100 by 100 m area; two APs located 5 m to the left and 5 m to the right of the gateway; and several randomly-positioned nodes. The connections between the gateway and the APs are considered wired and reliable [9]. The nodes were numbered according to the distance from the gateway. 50 % of the nodes were powered with industrial-standard batteries (3.6 V, 17 Ah). Simulations were conducted with 20 and 40 nodes to verify QLRR-WA's performance when IWSN have different density of nodes. Fig. 3 depicts a 40-node topology used as an example for the performance evaluation. Black nodes are line-powered nodes, gray nodes are battery-powered nodes, and lines represent nodes within the communication range of each other.

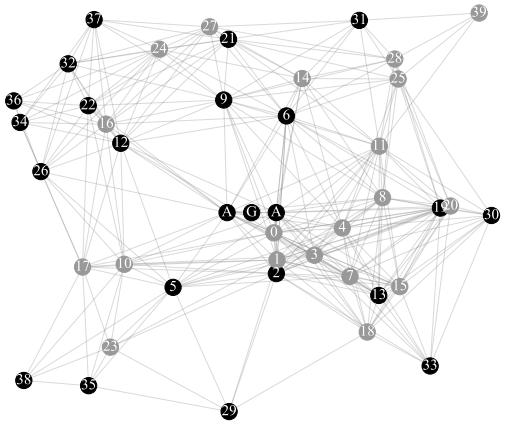


Fig. 3: A network topology example with 40 nodes.

Each simulation starts with the startup of the NM/gateway and APs. The first node is turned on after 5 minutes, and the

other nodes are turned on in 1-minute intervals, according to the sequence number given during the topology construction. A node listens to the channel looking for an advertisement packet from its neighbors and then beigns the join process. After joining, it requests bandwidth to the NM, receives configurations (routes and links), and starts sending sensor readings towards the gateway with a period of 32 seconds. Health reports are sent every 15 minutes and the NM polls the battery lifetime in 1-minute intervals. The management routines are executed when a new device joins the network or in 10-minute intervals. The simulations run for 12 hours. Other simulation parameters follow the work of [10], [23].

We used $p = 1$ for battery-powered nodes and $p = 0$ for line-powered nodes; $s_d = -45$ dBm; $t_s = 5$ min; $k = 2$; and $R = 1$. The Q-Learning parameters follow the suggested values from the works in Section II-C, where $\alpha = 0.1$, $\gamma = 0.8$, and $\varepsilon = 0.3$. The Q-values were set to 0 at startup. We set $m = 7$ and removed states with weights with value 0 to keep all weights influencing the cost equation and to reduce the number of states and the exploration time required. Fig. 4 depicts the states and actions used. The initial state was arbitrarily set to s_7, where all weights have similar values. The exploration phase starts at the beginning of the simulation and ends after 8 hours, allowing the agent to iterate approximately 48 times. The number of iterations was chosen because it represents the average number of iterations needed considering the complexity of a similar RL problem for goal-oriented domains [26]. After the exploration phase, the agent goes to the next state related to the state-action pair with greater value in $Q(s, a)$, where it exploits the best set of weights found. QLRR-WA executes only with the periodic NM management routines (10-minute intervals) to avoid network reconfiguration when a node is joining. This time interval between iterations was chosen based on simulation tests which allowed the evaluation of the network reconfiguration time for those topologies. Periods of hours are required for exploration in WH because of the time needed for network reconfiguration and stabilization.

Downlink graphs without path redundancy were used in the simulations to reduce the number of links required during the scheduling process. They were created using a BFS tree, where edges are chosen based on the RSL of neighbors. The scheduling algorithm in [9] (implemented in [23]) was used. Links are allocated on the paths from the source to the destination in a depth-first manner and the traffic is split from a node among all its successors by reducing the bandwidth required on each successor [6], [9]. Links are scheduled in the following sequence: Advertisement links are allocated for all devices currently in the network with a period of 8 s; For the downlink graphs, one permanent link between two neighbors (for management communications) is allocated with a period of 4 s, and normal links are allocated for management (limited to 6 links between two neighbors) with a period of 2 s. The same allocation is used for the uplink graph for permanent and management links. Finally, links for publishing periodic data are allocated. Downlink graphs were scheduled first to reduce reconfiguration, because they rarely change during the simulations. We compared QLRR-WA with the following baseline uplink algorithms: Han [9], which builds graphs trying to reduce the number of hops from the gateway; ELHFR [12], which chooses edges based on the RSL of neighbors; Künzel [11], where weights were set $w_h = 0, w_p = 1, w_s = 0, w_n = 0$ on all related equations trying to build graphs that avoid battery-powered nodes as routers; and Künzel [10], which uses RL to adjust the weight of the power type of nodes in a cost equation, where we used $\alpha = 0.1$, $\gamma = 0.8$, $\varepsilon = 0.3$ and $s_f = 0.25$. We ran several repetitions of the simulation for each algorithm for each topology to get statistics of ANL, ENL, PDR, and PRN.

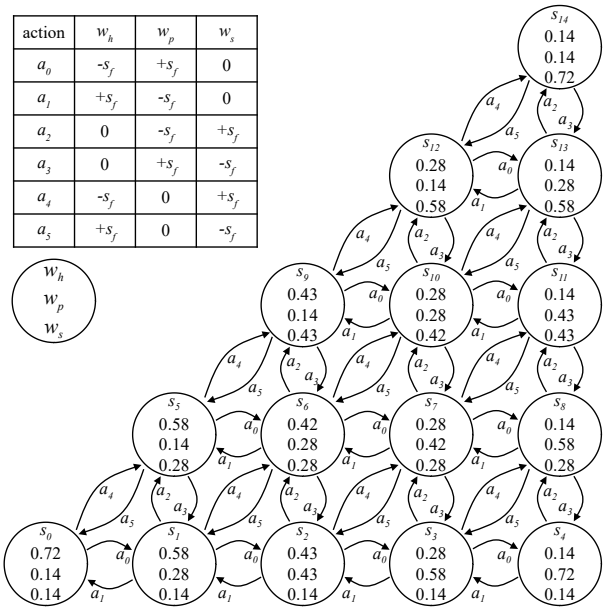


Fig. 4: Set of Q-Learning states, actions and weights used in the simulations (rounded values)

B. Results

Fig. 5 depicts the ANL of the 40-node topology over time (over several repetitions of the simulation). During the startup of the network (from 0 to 4 hours), the ANL increased because nodes were joining the network, exchanging commands and sending process data. The ANL stabilized in the Han, ELHFR, and Künzel [11] after the join process because the topology stopped changing. Künzel [10] continued to reduce ANL as it was exploring, but it had an average performance in these simulations. In QLRR-WA, the ANL started with reduced values during exploration because of the topology characteristics and weights used, and also presented slight variations from 0 to 8 hours because of the exploration. After exploration, QLRR-WA stabilized the ANL in a reduced value.

Fig. 6 depicts the boxplots of the ANL, ENL, and PDR at the last hour of the simulations. Samples of the ANL, ENL, PDR, and PRN were collected, for several simulations of the same topology, over the last hour of simulation (1 sample every 10 minutes). In both 20 and 40-node topologies, QLRR-WA presented the lowest ANL. Regarding ENL, it can be seen that QLRR-WA kept values as good as the other routing algorithms in both topologies. QLRR-WA presented the highest PDR for both topologies. All algorithms kept a PRN of 100 % on

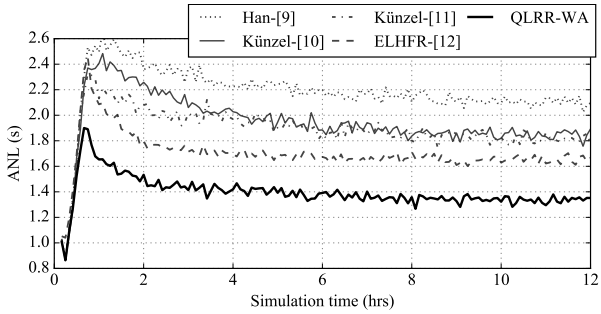
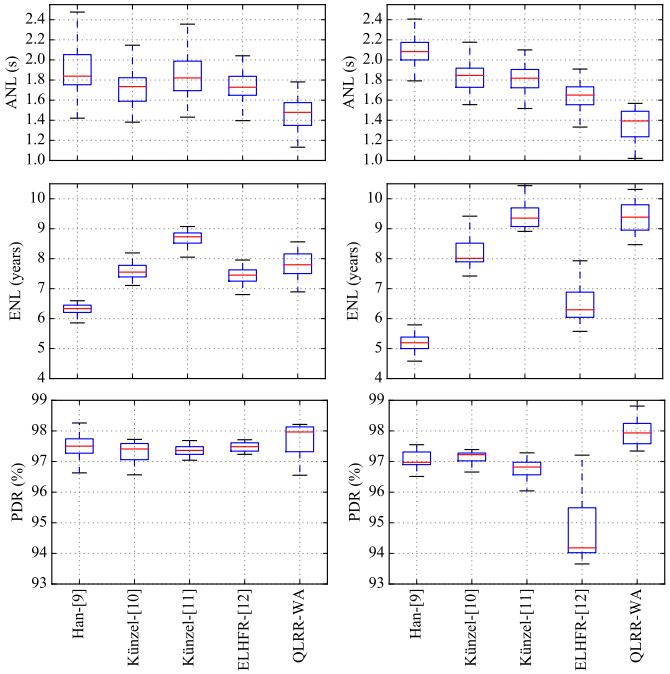


Fig. 5: ANL for 40-node topology example

both topologies, except ELHFR which presented 80 % (20 nodes) and 85 % (40 nodes). ELHFR presented the lowest PRN because it allows nodes to connect only to neighbors from lower levels in a BFS tree, and usually a few neighbors are available in lower levels.



(a) 20 nodes (b) 40 nodes

Fig. 6: ANL, ENL and PDR boxplots for topology examples

Because each topology and application have different characteristics (i.e. spatial distribution and power source of nodes), the performance metrics will present different values. We conducted simulations to verify if QLRR-WA presented similar performance over 30 random topologies. We collected the ANL, ENL, PDR, and PRN samples of the last simulation hour for several repetitions of the simulations for each topology, and then used One-way Analysis of Variance (ANOVA) with a 95 % significance to verify if QLRR-WA improved those metrics on each topology when compared side-by-side to the other routing algorithms. Improvement is considered to exist when, for a given topology, the ANL is less or ENL, PDR, PRN are great than the other algorithm being compared, and the null hypothesis of the ANOVA test is rejected. Table I shows the percentage of topologies where QLRR-WA improved the ANL, ENL, and PDR when compared to the baseline algorithms. PRN was always around 100 % for all algorithms except ELHFR which was always above 75 %. Table I also shows the average relative decrease of the ANL and increase ENL for the topologies where improvements occurred when compared to the values presented by the other algorithms. PDR and PRN were omitted because these values were similar for all compared algorithms.

TABLE I: Performance comparison of the QLRR-WA with the state-of-the-art algorithms

Topologies where QLRR-WA improved ANL, PDR, ENL (%)					
Nodes	Metric	Han [9]	Künzel [10]	Künzel [11]	ELHFR [12]
20	ANL	86	63	76	80
	ENL	46	50	26	23
	PDR	33	6	56	96
40	ANL	100	100	100	93
	ENL	90	63	10	86
	PDR	3	0	16	100
Average decrease of ANL, increase of ENL values for QLRR-WA (%)					
Nodes	Metric	Han [9]	Künzel [10]	Künzel [11]	ELHFR [12]
20	ANL	14	10	10	16
	ENL	34	9	9	28
40	ANL	26	16	17	14
	ENL	76	11	11	62

V. Conclusions and Future Work

QLRR-WA is a centralized routing algorithm that builds uplink graphs. An agent using Q-Learning learns a set of weights to apply on a cost equation that is used to choose nodes and connections during the graph construction. Rewards are given when the agent reduces ANL or increases ENL. Simulations were conducted on a WH simulator considering IWSN applications and random network topologies. Results showed a reduction of the ANL in most topologies, while the ENL, PDR, and PRN were as good as the other state-of-the-art algorithms. Considerations must be made regarding the characteristics of the protocol and application, the time interval for network reconfiguration, the parameters used, the number of iterations necessary during the learning process, and the rewards of the agent. All these aspects influence the performance of QLRR-WA. Open issues are: how to estimate and reduce the required learning and exploration times, and how to quantify the expected performance improvements. These issues could be tackled by implementing a simulation/optimization module inside the NM, which could use simulation or real data to provide information about the network performance and an optimized configuration to be used. Other research possibilities include the development of QLRR approaches able to build routes considering adjustments on the transmission power, mobility, coexistence, scheduling, and real-time requirements; simulation and experiments using other centralized network protocols; to evaluate the state-of-the-art routing algorithms in real IWSN, specific IWSN applications and physical channel conditions.

Acknowledgment

The authors would like to thank the University of York, UFRGS, and IFRS for the support provided, and also the editors and anonymous referees for the contributions.

REFERENCES

[1] M. Sha, D. Gunatilaka, C. Wu, and C. Lu, "Empirical study and enhancements of industrial wireless sensor-actuator network protocols," *IEEE Internet of Things Journal*, vol. 4, no. 3, pp. 696–704, June 2017.

[2] L. D. Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, Nov 2014.

[3] G. V. Research. (2018) Industrial wireless sensor network market worth $8.67 billion by 2025. [Online]. Available: https://www.grandviewresearch.com/press-release/global-industrial-wireless-sensor-networks-iwsn-market

[4] J. M. Winter, I. Muller, G. Soatti, S. Savazzi, M. Nicoli, L. B. Becker, J. C. Netto, and C. E. Pereira, "Wireless coexistence and spectrum sensing in industrial internet of things: An experimental study," *International Journal of Distributed Sensor Networks*, 2015.

[5] J. Niu, L. Cheng, Y. Gu, L. Shu, and S. K. Das, "R3e: Reliable reactive routing enhancement for wireless sensor networks," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 784–794, Feb 2014.

[6] M. Nobre, I. Silva, and L. A. Guedes, "Routing and scheduling algorithms for wirelesshart networks: A survey," *Sensors*, vol. 15, no. 5, pp. 9703–9740, 2015.

[7] W. Shen, T. Zhang, F. Barac, and M. Gidlund, "Prioritymac: A priority-enhanced mac protocol for critical traffic in industrial wireless sensor and actuator networks," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 824–835, Feb 2014.

[8] W. Ikram, S. Petersen, P. Orten, and N. F. Thornhill, "Adaptive multi-channel transmission power control for industrial wireless instrumentation," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 978–990, May 2014.

[9] S. Han, X. Zhu, A. K. Mok, D. Chen, and M. Nixon, "Reliable and real-time communication in industrial wireless mesh networks," in *2011 17th IEEE Real-Time and Embedded Technology and Applications Symposium*, April 2011, pp. 3–12.

[10] G. Künzel, G. P. Cainelli, I. Müller, and C. E. Pereira, "Weight adjustments in a routing algorithm for wireless sensor and actuator networks using q-learning," in *3rd IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control (CESCIT)*, vol. 51, no. 10, 2018, pp. 58 – 63.

[11] G. Künzel, G. P. Cainelli, and C. E. Pereira, "A weighted broadcast routing algorithm for wirelesshart networks," in *2017 VII Brazilian Symposium on Computing Systems Engineering (SBESC)*, Nov. 2017, pp. 187–192.

[12] Z. Jindong, L. Zhenjun, and Z. Yaopei, "Elhfr: A graph routing in industrial wireless mesh network," in *2009 International Conference on Information and Automation*, June 2009, pp. 106–110.

[13] C. Wu, D. Gunatilaka, A. Saifullah, M. Sha, P. B. Tiwari, C. Lu, and Y. Chen, "Maximizing network lifetime of wirelesshart networks under graph routing," in *2016 IEEE First International Conference on Internet-of-Things Design and Implementation (IoTDI)*, April 2016, pp. 176–186.

[14] Q. Zhang, F. Li, L. Ju, Z. Jia, and Z. Zhang, *Reliable and Energy Efficient Routing Algorithm for WirelessHART*. Cham: Springer International Publishing, 2014, pp. 192–203.

[15] S. Zhang, A. Yan, and T. Ma, "Energy-balanced routing for maximizing network lifetime in wirelesshart," *International Journal of Distributed Sensor Networks*, vol. 9, no. 10, p. 173185, 2013.

[16] M. Sepulcre, J. Gozalvez, and B. Coll-Perales, "Multipath qos-driven routing protocol for industrial wireless networks," *Journal of Network and Computer Applications*, vol. 74, no. Supplement C, pp. 121 – 132, 2016.

[17] X. Han, X. Ma, and D. Chen, "Energy-balancing routing algorithm for wirelesshart," in *2019 15th IEEE International Workshop on Factory Communication Systems (WFCS)*, May 2019, pp. 1–7.

[18] C. Savaglio, P. Pasquale, G. Aloi, A. Liotta, and G. Fortino, "Lightweight reinforcement learning for energy efficient communications in wireless sensor networks," *IEEE Access*, vol. PP, pp. 1–1, 03 2019.

[19] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018.

[20] H. A. Al-Rawi, M. A. Ng, and K.-L. A. Yau, "Application of reinforcement learning to routing in distributed wireless networks: A review," *Artif. Intell. Rev.*, vol. 43, no. 3, pp. 381–416, Mar. 2015.

[21] Z. Mammeri, "Reinforcement learning based routing in networks: Review and classification of approaches," *IEEE Access*, vol. 7, pp. 55 916–55 950, 2019.

[22] M. A. Habib, M. Y. Arafat, and S. Moh, "Routing protocols based on reinforcement learning for wireless sensor networks: A comparative study," *Journal of Advanced Research in Dynamical and Control Systems*, pp. 427–435, 01 2019.

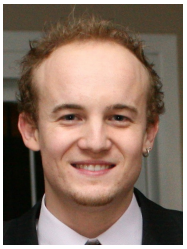
[23] P. Zand, E. Mathews, P. Havinga, S. Stojanovski, E. Sisinni, and P. Ferrari, "Implementation of wirelesshart in the ns-2 simulator and validation of its correctness," *Sensors*, vol. 14, no. 5, pp. 8633–8668, 2014.

[24] W. Guo, C. Yan, and T. Lu, "Optimizing the lifetime of wireless sensor networks via reinforcement-learning-based routing," *International Journal of Distributed Sensor Networks*, vol. 15, no. 2, p. 1550147719833541, 2019.

[25] Z. Jin, Y. Ma, Y. Su, S. Li, and X. Fu, "A q-learning-based delay-aware routing algorithm to extend the lifetime of underwater sensor networks," *Sensors*, vol. 17, no. 7, 2017.

[26] S. Koenig and R. G. Simmons, "Complexity analysis of real-time reinforcement learning applied to finding shortest paths in deterministic domains," Carnegie Mellon University, School of Computer Science, Pittsburg,, Tech. Rep. CMU-CS-196, 12 1992.

[27] A. Bildea, O. Alphand, F. Rousseau, and A. Duda, "Link quality metrics in large scale indoor wireless sensor networks," in *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Sep. 2013, pp. 1888–1892.



Gustavo Künzel received the B.Eng. degree in control and automation engineering from UNIVATES in 2010, a MSc from Federal University of Rio Grande do Sul (UFRGS) in 2012. He is a Professor at Federal Institute of Science, Technology and Education of Rio Grande do Sul (IFRS) and is currently working toward the Ph.D. degree in the Department of Electrical Engineering, UFRGS. His current research interests include industrial wireless networks and artificial intelligence.

Leandro Soares Indrusiak received the B.Eng. degree in electrical engineering from the Federal University of Santa Maria (UFSM) in 1995, obtained a MSc in Computer Science from UFRGS in 1998, and was issued a binational doctoral degree by UFRGS and Technische Universitt Darmstadt in 2003. He is a faculty member of University of York's Computer Science department since 2008, working on real-time systems and networks, distributed embedded systems, on-chip multiprocessing, energy-efficient computing, cyber-physical systems, cloud and high-performance computing, and several types of resource allocation problems. He has more than 150 technical publications on conferences and journals (nine of them received best paper awards).

Carlos Eduardo Pereira received the Dr.-Ing. degree in Electrical Engineering from the University of Stuttgart, Germany in 1995 and has MSc in Computer Science and B.S. degree in Electrical Engineering both from UFRGS. He is a Full Professor at UFRGS and Director of Operations at EMBRAPII in Brazil. He has more than 400 technical publications on conferences and journals. He is Associate Editor of the Journal "Control Engineering Practice" and Annual Reviews in Control from Elsevier and Council Member of IFAC. He received in 2012 the Friedrich Wilhelm Bessel Research Award from the Alexander von Humboldt Foundation - Germany.