SPECIAL SECTION ON ARTIFICIAL INTELLIGENCE AND COGNITIVE COMPUTING FOR COMMUNICATION AND NETWORK

IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received January 30, 2019, accepted February 20, 2019, date of publication March 4, 2019, date of current version March 20, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2902371

Lightweight Reinforcement Learning for Energy Efficient Communications in Wireless Sensor Networks

CLAUDIO SAVAGLIO[1], (Student Member, IEEE), PASQUALE PACE[1], (Member, IEEE), GIANLUCA ALOI[1], (Member, IEEE), ANTONIO LIOTTA[2], (Senior Member, IEEE), AND GIANCARLO FORTINO[1], (Senior Member, IEEE)

[1]Department of Informatics, Modeling, Electronics, and Systems University of Calabria, 87036 Rende, Italy
[2]Department of Electronics, Computing, and Mathematics, University of Derby, Derby DE22 1GB, U.K.

Corresponding author: Claudio Savaglio (csavaglio@dimes.unical.it)

This work was supported by the "COGITO—Sistema dinamico e cognitivo per consentire agli edifici di apprendere ed adattarsi" Project, funded by the Italian Government, under Grant ARS01_00836.

ABSTRACT High-density communications in wireless sensor networks (WSNs) demand for new approaches to meet stringent energy and spectrum requirements. We turn to reinforcement learning, a prominent method in artificial intelligence, to design an energy-preserving MAC protocol, with the aim to extend the network lifetime. Our QL-MAC protocol is derived from Q-learning, which iteratively tweaks the MAC parameters through a trial-and-error process to converge to a low energy state. This has a dual benefit of 1) solving this minimization problem without the need of predetermining the system model and 2) providing a self-adaptive protocol to topological and other external changes. QL-MAC self-adjusts the WSN node duty-cycle, reducing energy consumption without detrimental effects on the other network parameters. This is achieved by adjusting the radio sleeping and active periods based on traffic predictions and transmission state of neighboring nodes. Our findings are corroborated by an extensive set of experiments carried out on off-the-shelf devices, alongside large-scale simulations.

INDEX TERMS Wireless sensor network, artificial intelligence, reinforcement learning, energy-efficient network, medium access control.

I. INTRODUCTION

Due to their ability of collecting data from the physical world, elaborating and communicating it in a responsive manner, Wireless Sensors Networks (WSNs) represent essential building blocks for the Internet of Things (IoT) [1]. However, the limited computational resources and energy typically featuring the WSNs motes collide with the rising smart applications' demands as well as with the ballooning end-users' expectations. Therefore, in order to improve WSNs functionality, utility and survival aspects despite their intrinsic constraints, Artificial Intelligence (AI) techniques can be successfully applied both at network and node level, thus enabling intelligent behaviors and adaptivity to a variety of contexts (smart factory, structural health, etc., like the ones reported in Fig. 1) [2]. For example, especially when nodes deployment/replacement is not trivial (*e.g.,* large-scale industrial and outdoor monitoring applications, patients monitoring in small-scale indoor environments), network lifetime and operation greatly benefit from intelligent motes. Indeed, whatever the IoT application, motes require power of some sort, and energy-efficiency enables them to operate in a standalone manner, reducing managing costs and maintenance time. In particular, Reinforcement Learning (RL) is an AI-based approach that enables a decision maker to observe, learn, and take actions in its operating environment in order to increase its accumulated reward. RL promises to play a major role in AI-enabled cognitive networks of the future because, more than other paradigms (*e.g.,* neural networks, swarm intelligence, software agents), it demands low computation resources and implementation efforts thus providing high flexibility to topological changes and near-optimal results, without requiring any *apriori* network model [3], [4].

According to this challenging vision, this paper exploits RL techniques for WSNs to design a non-fixed and adaptive node duty-cycle at MAC layer that reduces the energy

The associate editor coordinating the review of this manuscript and approving it for publication was Yin Zhang.

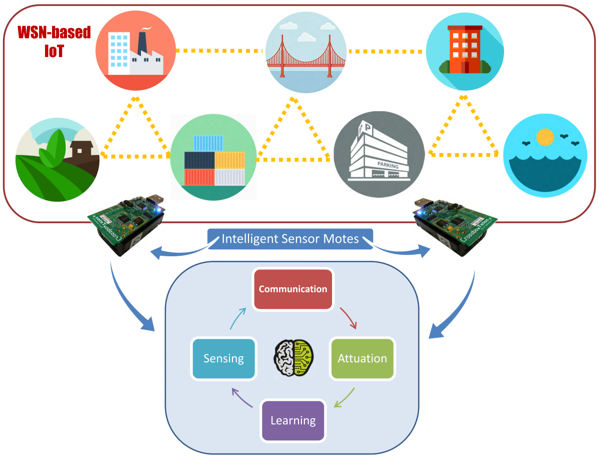

FIGURE 1. Intelligent sensor motes supporting different WSNs application scenarios.

consumption over time, without affecting the other network performances. In particular, Q-Learning [5], one of the most popular and powerful algorithms based on RL, has been used to develop the proposed QL-MAC protocol, whose aim is the optimization of the radio sleeping and active periods of network's nodes according to both the traffic condition and the neighbors transmission state. The performance of the QL-MAC protocol, implemented both on a real small-scale testbed using *TelosB* motes and on a large-scale scenario using the *Contiki Cooja* simulator, have been evaluated in terms of effectiveness and efficiency and compared to the conventional asynchronous CSMA-CA MAC protocol. The results show that, in small- as well as in large-scale scenarios, the adaptive behavior of QL-MAC guarantees better network performances compared to standard MAC protocols with respect to both Packet Delivery Ratio (PDR) and energy consumption.

Summarizing, the novelty of this work with respect to conventional networking is the introduction of an intelligent/predictive radio scheduling strategy for WSN's motes to minimize their energy consumption. The extensive set of comparative experiments represents a major contribution with respect to our previous work [6], in which the preliminary studies on the QL-MAC protocol were presented, providing only a partial parametric analysis of the configuration settings and a reduced performance study.

The rest of the paper is organized as follows: a brief analysis on the AI-oriented approach exploited to design smart communications and networks is reported in Sect. II, while a background on Reinforcement and Q-Learning along with QL-MAC protocol details are provided in Sect. III. Design and implementation choices of the proposed QL-MAC protocol at Application, Network and MAC layers as well as the Sink-Node communication phases are reported in Sect. IV. The proposed QL-MAC configuration (learning rate, frame dimension, etc.) and the performance analysis, both in small and large scale scenarios as well as in real and simulated environments, find place in Sect. V. The conclusions are given in Sect. VI, drawing future research directions.

II. RELATED WORK

Recently, in order to design smart communications and networks with optimized resource management, dynamic device configuration and feasible service provision, intelligence has been pushed from the network's core (data centers) to the edge (nodes) by following decentralized, autonomic and cognitive approaches [7]–[9]. Indeed, network's elements, even the resource-constrained ones, have been upgraded with different degrees of smartness and provided with new capabilities of computation, reasoning and learning through AI-related and data mining techniques. In particular, several researches efforts have been focused on optimizing the node's task scheduling, routing paths, and computation-related aspects [10]–[12]. However, it has been also largely studied that most of node's energy expenditure is due to the radio activity (transmitting one bit may consume as much as executing a few thousands instructions), thus suggesting that communication should be traded for computation [13]. Therefore, a large number of contributions have been provided aiming to optimize *(i)* the network data traffic, thus minimizing the overall message load and, consequently, the individual nodes' transmission and receiving times, *(ii)* the behavior of communication-related node's components, abandoning fixed configurations for an efficient and adaptive management of the node's sleep and active periods, idle listening, transmission power, etc.

With respect to the first direction, the mainstream approach consists in software-level interventions to develop on-node and ad-hoc data mining techniques, leveraging distributed computation paradigms as enablers (*e.g.,* Software Agents [14], [15]). The pursued goal is extracting application-oriented models and patterns with acceptable accuracy from continuous and rapid sensors data streams, thus transforming (*i.e.,* preprocess, filter, aggregate) raw sensed data directly in-situ and reducing the overall data traffic [16]. To the same end but with a different strategy, intelligent nodes can autonomously and adaptively manage the logical network organization (*e.g.,* dynamic clustering of the nodes according to their properties such as position, available resource, residual energy) through genetic algorithms [17], perform event recognition or prediction (*e.g.,* inferring node faults) through neural networks [18], or locally cooperate for minimizing the expensive, multi-hop and fail-prone communications to the sink on behalf of short-range message exchanges [19], [20].

With respect to the communication-related components' behavior, instead, a well-established research line foresees the application of machine learning approaches to achieve autonomous behavior and operation of node's hardware for the sake of energy-efficiency [4]. The goal here is to automatically learn the properties of both the environment and the neighborhood in order to reactively adjust the radio activity and settings. For example, through RL, nodes can adapt

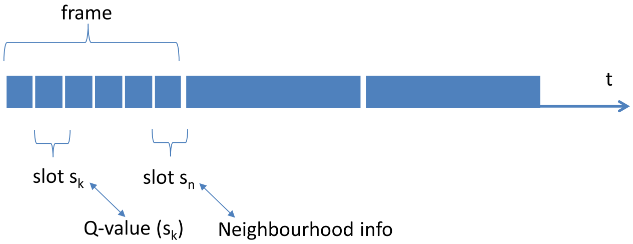

FIGURE 2. Slot and frames.

the radio scheduling (*i.e.,* sleeping and active periods) by actively inferring the state of nodes neighborhood through exchanging control messages about the number of waiting messages, ratio of successfully transmitted messages, residual energy, etc., as in [21] and [22]. Similarly, other works aim at transmitting only in slots with a lower probability of collision and selectively turning on/off their radio to save energy [23]. A different approach is adopted by cognitive algorithms which dynamically control the node's transmission power, perform spectrum sharing or intelligent antenna switching according to the current service requirements and environmental conditions [24].

Our proposal falls within the second category of approaches, aiming to optimize the radio sleeping and active periods through RL. Indeed, more than neural networks, swarm intelligence and software agents, reinforcement learning demands for less computation resources, it is easy to implement, highly flexible to topology changes and achieves optimal results even without an *apriori* network model [3], [4].

III. QL-MAC PROTOCOL

In this section we describe a MAC protocol, well suited for WSNs, that dynamically learns the traffic conditions over the time to better adopt the most suitable sleep/active scheduling policy. In particular, each node not only takes into consideration its own packet traffic due to the application layer, but also considers its neighborhood's state. The underlying behavior of the QL-MAC relies on a simple asynchronous CSMA-CA approach according to a frame-based structure dividing the time into discrete time units, the *frames*, which are further divided into smaller time units, the *slots* (see Fig. 2). Both frame length and slot number are fixed parameters of the algorithm and remain unchanged during the execution. In particular, within each frame, every slot stores a Q-value while the last one is used to store information exchanged among neighboring nodes that will affect the rewarding function.

In summary, the main beneficial effect of such Q-Learning based algorithm is the implementation of a non-fixed and adaptive duty-cycle that reduces the energy consumption over the time without affecting the other network performances, as shown by the simulations results discussed in Section V. Thanks to the QL-MAC protocol, each node can independently determine an efficient wake-up schedule to limit as much as possible the number of slots in which the radio is turned on, thus greatly increasing its own (and overall network) lifetime.

A. REINFORCEMENT LEARNING AND Q-LEARNING

Reinforcement Learning (RL) is a sub-area of machine learning related to the maximization of some long-term rewards according to the actions taken by a specific agent. In particular, the agent explores its environment by selecting at each step a specific action and receiving a corresponding reward from the environment. Since the best action is never known a-priori, the agent has to learn from its experience, by means of the execution of a sequence of different actions in order to infer what should be the best behavior from the obtained corresponding rewards. One of the most popular and powerful algorithm based on RL is Q-Learning [5], which does not need any a-priori knowledge of the environment to be modeled and whose actions depend on a so called Q-function, which indicates the quality of a specific action at a specific agent's state. Specifically, the Q-values are updated as follows:

$$\begin{aligned} &Q(s_{t+1}, a_t) \\ &\quad = Q(s_t, a_t) + \lambda[r_{t+1} + \psi \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \end{aligned} \tag{1}$$

where $Q(s_t, a_t)$ is the current value at state s_t, when action a_t is selected. At some state s_t, the agent selects an action a_t. It finds the maximum possible Q-value in the next state s_{t+1}, given that a_t is taken, and updates the current Q-value. The discounting factor $0 < \psi < 1$ gives preference either to immediate rewards (if $\psi \ll 1$) or to rewards in the future (if $\psi \gg 0$), whereas the learning rate $0 < \lambda < 1$ is used to tune the speed of learning.

B. PROTOCOL DETAILS

According to the depicted communication scenario and the related radio scheduling issues, the actions available to each agent/node consist in deciding whether it should stay in active or in sleep mode during each single time slot. Thus, the action space of a node is determined by the number of slots within a frame. Every node stores a set of Q-value, each of which is coupled to a specific slot within the frame. The Q-value represents an indication of the benefits that a node get when is awake during the related time slot. The Q-value is updated over the time according to some specific events occurring during the same slot at each frame. Furthermore, it is also related to the state information coming from the node's neighbors. Specifically, every Q-value of each specific node i is updated as follows:

$$Q_s^i(f+1) = (1-\lambda)Q_s^i(f) + \lambda R_s^i(f) \tag{2}$$

where $Q_s^i(f) \in [0, 1]$ is the current Q-value associated to the slot s on the frame f, $Q_s^i(f+1)$ is the updated Q-value, which will be associated to the same slot s but on the next frame, λ is the learning rate and R_s^i is the earned reward. Differently from the update rule shown in 1, the future reward is not considered and the discount factor ψ is set to 0. Following such decentralized approach, it is important to define a suitable reward function able to take into account both the condition node and its neighborhood conditions. In this direction, the QL-MAC protocol takes into considerations the events related to the packet traffic load, so that the reward function for the node i and related to a specific slot s is computed as follows:

$$R_s^i = \alpha\left(\frac{RP - OH}{RP}\right) + \beta S_i + \gamma\left(\frac{\sum_{j=1}^{|N_j|} P_j}{|N_j|}\right) \tag{3}$$

where OH is the number of over-heard packets representing the packets received but actually not intended for node i; RP is the total amount of packets received by node i during the slot s of frame f also including over-heard packets; S_i has a positive value equal to $+1$ if node i has at least one packet to broadcast during slot s, 0 otherwise; P_j has a positive value equal to $+1$ if the neighboring node j has sent at least one packet to node i during slot s, 0 otherwise; N_i is the set of neighbors of node i and the constants α, β, and γ weigh the different terms of the function accordingly.

It is worth noting that, at the beginning, all the Q-values on every node are set to 1, meaning that all nodes have their radio transceiver ON on every slot (*i.e.,* for the entire frame). During the learning process, the Q-values changes over the time accordingly to the variation of the reward function. A further parameter T_{ON} which represents a threshold value, has been used to properly set the state for the radio transceiver on the basis of the Q-values:

$$Radio_{[slot\ s]} = \begin{cases} On, & \text{if } Q_s^i(f) \geq T_{ON} \\ Off, & \text{otherwise} \end{cases} \tag{4}$$

According to this strategy, the generic node will switch to sleep mode for the duration of the whole slot if the quality value of a specific slot s is below threshold value; on the contrary, it will stay in active mode because most likely there will be communication activities directly involving the node. In such a decentralized learning approach, the main challenge is the definition of a suitable reward function for the individual node that will implicitly lead to a coordinated-group behavior by taking into consideration the current condition of both the node and its neighborhood. In particular, each node i will take into account the following information as reward signals for a specific slot s:

- *Transmitted packets*: the amount of packets the node has successfully transmitted to the intended receiver during the slot. In case of unicast communication in the MAC layer, successful data reception is directly acknowledged with an ACK packet.
- *Received packets*: the total amount of packets correctly received by the node from its neighbors during the slot;
- *Over-heard packets*: the amount of over-heard packets received during the same slot, *i.e.,* the packets received but actually not intended for the node itself. Again, in unicast communication the MAC layer is able to directly detect such packets;
- *Expected received packets*: the amount of packets a specific neighboring node has sent to node i during the slot; this is the only information explicitly exchanged by the protocol and it is necessary when the node is in sleep mode because it cannot perceive the communication activities of its neighborhood. Thanks to this information, the node is then able to figure out when it would be better to turn on the radio again during the slot because of a new packet traffic pattern. It is also used to compute the amount of packets not successfully received due to collisions.

When the packets exchange at MAC layer takes place in broadcast mode, it is necessary to get some extra information from the upper layers because a node is not able to understand whether each single received packet is actually destined for itself or not. In particular, the QL-MAC protocol uses a simple crosslayer communication by decapsulating every received packet and delivering to the network layer, which checks whether the packet is intended for the node. If the packet is discarded, the network layer informs the MAC protocol about the reception of an overheard packet, and the reward function is updated accordingly to (3).

In case the node needs to send a packet while the radio is turned off at a specific slot, during the same time window, the packet is buffered and the transmission id postponed to the next available slot in which the radio is switched "on". Finally, the last term of equation (3) provides an aggregated information about the state of the node neighborhood representing the packet traffic activity during a specific time slot. This protocol information is crucial for the generic node, working in sleep mode, to understand that it should be better to turn on the radio because of the presence of packets destined for it.

IV. QL-MAC DESIGN AND IMPLEMENTATION

To properly design and test the proposed QL-MAC protocol, the whole protocol stack to be implemented within each sensor node, has been carefully deployed to support the communication toward a central sink node of the WSN. A three layer protocol consisting of *Application*, *Network* and *MAC* layers depicted in Fig. 3, has been developed in *NesC* for the *TinyOS* operating system.

A. APPLICATION LAYER

The application layer has been designed to interact with the network layer to send data packets by determining the network load through the specific packet/rate of each node toward the sink. To accomplish this function, the application layer defines different data fields such as the source id, the destination id and the sequence number; moreover, it also sends to the sink all the summary statistics of each node to periodically compute the overall network performance analysis.

B. NETWORK LAYER

The network layer supports the routing of data packets towards the sink node by handling the discovery phase to

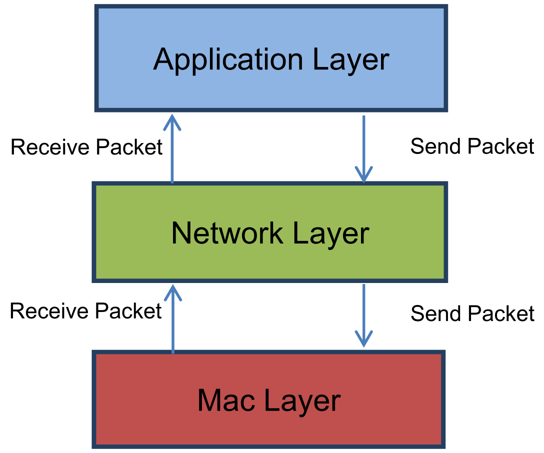

FIGURE 3. Different layers of the developed protocol stack.

define the real network topology in order to make each node aware of its own position and that of its neighbors. In particular, the Multi-Path ring routing scheme [25] that utilizes a ring level to separate sensor nodes into several sections, has been implemented due to the low overhead and high data reliability of such strategy.

The basic idea of multipath construction phase is to organize the network into levels according to the hop distance from the sink node to a sensor node *i.e.,* by the end of this phase each node will get a ring level which indicates how many hops away from the sink node; in this phase, the sink node broadcasts a packet with its ring level 0. The nodes which received the packet will increase their ring number and rebroadcast the packet to their neighbors and at the end, all the nodes in WSN will be separated into several levels.

C. MAC LEVEL

The MAC layer represents the core of this work because it embeds the learning algorithm to effectively schedule the on-off periods of the radio transmission module; in particular, it handles the access to the transmission medium and interact with the network layer not only to send and receive data packets but to detect if the overheard packets are destined to the current node.

It basically divides the time into discrete time units, the *frames*, which are further divided into smaller time units, the *slots* as depicted in Fig. 2, then the Q-values are computed for each slot according to the formulas described in section III.B to decide the best radio configuration for the next frame. This choice is taken by considering different information for each slot such as those related to the state of the current node and its neighborhood; as a consequence, to guarantee a correct messages exchange within the network nodes, the last slot of each frame has been dedicated to host these specific information that will impact on the reward function of the algorithm.

The Fig. 4 shows the changes in the on-off radio activation over the time due to the learning process; the sequences are shown for each time slot of different frames of the same node.

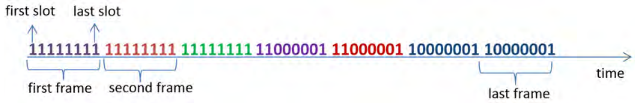

FIGURE 4. On-Off schedule of the radio transmission during the learning phase.

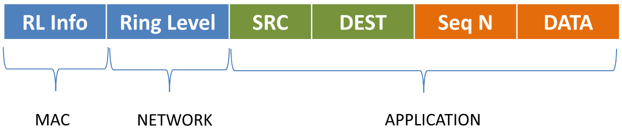

FIGURE 5. Data format included by each layer.

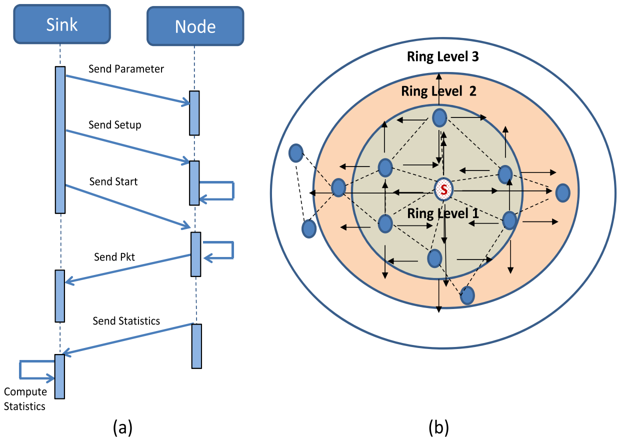

FIGURE 6. a) Node-sink communication phases; b) different network levels according to the hops distance from the sink.

The Fig. 5 summarizes the different data fields within the implemented communication protocol for each of the described layers.

D. SINK-NODE COMMUNICATION

The communication between the sensor nodes and the sink represents a key aspect due to the distributed characteristics of the transmission environments; for this reason we put attention on the design of both synchronization and transmission phases to guarantee the right transmission of all parameters needed to implement the QL-MAC protocol. In this context, the sink node plays the role of coordinator among the following different communication phases as shown in Fig. 6.a:

1) *Send Parameter* - A broadcast communication between the sink and the other nodes is implemented with the aim of setting the communication parameters such as the packet rate.
2) *Send Setup* - The sink sends low power broadcast message containing its own network level (assumed equal to 0) to reach the neighbor nodes in order to create the first level ring communication that will be used in the routing algorithm. In this way, the neighbor nodes are aware of the sink network level in terms of hops number

and they can update their level accordingly. Then, they can propagate the broadcast message to their neighbors belonging to the next level that can receive the massage and update their network level number. Thanks to this setup phase, each node can be aware of its distance from the sink as shown in Fig. 6.b.

3) *Send Start* - The sink, acting as a coordinator, sends a start broadcast message to activate the data sending action of each node. This phase guarantees that each node can receive the sending message at the same time also avoiding the presence of isolated nodes.
4) *Send Pkt* - Each node periodically sends data packets to the sink; this phase also exploits the functionalities of the MAC layer with particular focus on the new features of the presented QL-MAC protocol dividing the transmission time in frames and slots within which the nodes con send and receive data packets. During this transmission phase, all nodes adapt to network traffic using a radio on-off schedule, thanks to the learning techniques described in section III. Specifically, the packets are transmitted in broadcast and with low power to guarantees a multi hop communication towards the sink passing through the different network rings discovered during the *Send Setup* phase. It is clear that, due to the broadcast communication nature, data can reach nodes that are out of the paths towards the sink, thus generating a certain amount of overheard.
5) *Send Statistics* - This phase is implemented only for testing purpose to properly collect the statistics used to evaluate the system performance. Once the sink has collected all the received data, each node sends a summary of its own statistics that will be used by the sink to compute performance metrics such as the packet delivery ratio, the overhead and the energy consumption of each node; moreover, each node sends to the sink the specific information on the status of the transmitted frames that will be used to measure the time needed for all nodes to adopt the right on-off radio scheduling policy.

E. TINYOS COMPONENTS

The designed communication stack consisting of three layers (Application, Network and MAC) has been developed in NesC in order to deploy a testbed on real sensor devises supporting the TinyOS operating system. The implemented software is composed by four modules, three of which to support the three stack layers of the sensor node and one for the sink to support the routing strategy. The Fig. 7 shows the logical communication scheme of the software modules developed inside the generic sensor node also highlighting the logical exposed interfaces that allow the communications among the different software modules.

V. PERFORMANCE EVALUATION

In this section the performance of the proposed QL-MAC protocol have been evaluated, both in small and large scale scenarios, by focusing on two main metrics: the packet delivery ratio (PDR, which provides important indications about the protocol's effectiveness) and the energy consumption (which decisively impacts the nodes lifetime and hence the protocol's efficiency). Under these metrics, QL-MAC has been compared to the conventional asynchronous CSMA-CA MAC both in real and simulated cases. Referring to the first case, the mote considered for the tests is *TelosB*, an IEEE 802.15.4-compliant device whose main features are the TI MSP430 Microcontroller with 10kB RAM, an integrated onboard antenna, Integrated Temperature, Light and Humidity Sensors, a 250 kbps High Data Rate Radio, and the *TinyOS* open-source operating system. Due to its high versatility and usability, *TelosB* motes are widely spread in the WSN contexts. To the latter case, the *Contiki Cooja* simulator, which leverages the MSPSim to emulate TI MSP430-based devices such as TelosB motes [26], has been exploited. In the following, it will be firstly discussed the scenario design with its basic settings, then the small and large scale scenarios.

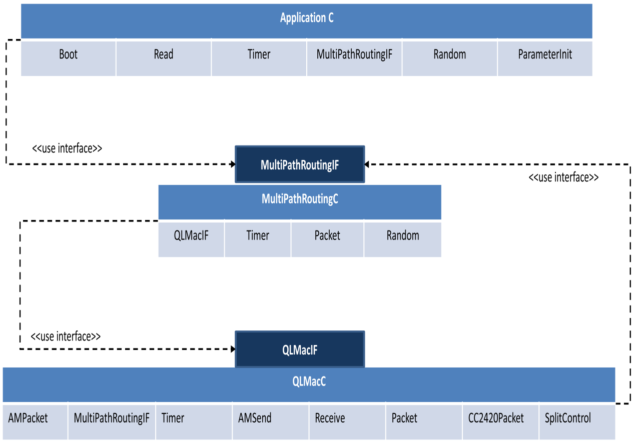

FIGURE 7. Software modules communication within the sensor node.

TABLE 1. Scenario and QL-MAC parameters.

Scenario Param.	Value	QL-MAC Param.	Value
#Nodes	7, 16, 49, 100	Frame Length	1,2s
Radio Device	CC2420	Slot #	4,6,8
Area	10000m^2	λ	0.95,0.5,0.05
Data Payload	32byte	α,β,γ	0.33
Tx Power	0dbm	Packet Rate	0.5, 1, 2pkt/s

A. SCENARIO DESIGN

A set of experiments has been carried out to evaluate the QL-MAC protocol with different traffic loads in a data-collection application, which is one of the most typical use cases of a WSN in real contexts, has been chosen as case study. The parameters related to the general scenario and specifically featuring the QL-MAC protocol are reported in Table 1 and discussed in the following.

With respect to the Scenario Parameters, seven nodes have been deployed in a small scale scenario as depicted

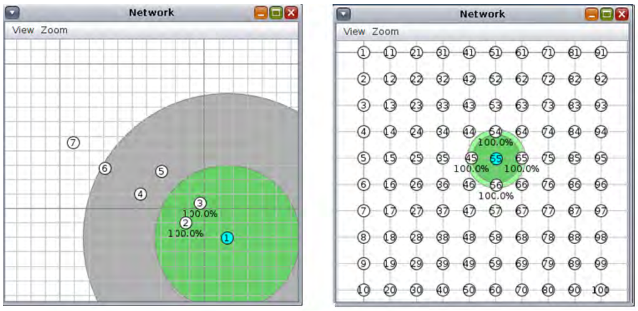

FIGURE 8. Simulation topologies: small-scale scenario (left side) and large-scale scenario (right side) with 100 nodes (10×10 grid).

in Fig. 8(a), while 16, 49 and 100 nodes in a large scale scenario have been placed in regular grids (4×4,7×7,10×10) as in Fig. 8(b) within an area of 10.000m^2. All nodes transmit messages with a fixed 32 bytes payload through the same CC2420 radio transceiver at a transmission power level of 0dBm (which allows roughly 46 meters transmission range) and employing the nodes-to-sink communication pattern described in Sec. IV-D (which is based on a simple multipath ring routing protocol at network layer because the sink is not in the transmission range of every node).

With respect to QL-MAC parameters, we experienced that the protocol is robust to different packet rates (0.5, 1 and 2 pkt/s) and frame lengths (1 and 2s). Instead, as the number of slots increases, PDR is stable but the energy spent by nodes tends to decrease thanks to a more fine-grained radio switch/slots management. Actually, with the use of 8 slots, QL-MAC exhibits the better trade-off, *i.e.,* similar PDR with respect to the cases with 4 and 6 slots (random variation in terms of ±3%), but minor energy expenditure. In this first study, we set the same value 0.33 to the three variables α, β, and γ of the equation 3 in order to fairly weight the components of the reward function, reserving the parametric analysis as a future work. An important setting concerns the value of λ, which rules the learning rate and impacts the protocol performance, especially when the number of nodes changes. Therefore, we separately reported a λ-analysis for small and large scale scenario in Tables 2 and 3.

B. SMALL SCALE SCENARIO

The small scale scenario consists of seven nodes deployed as in Fig. 8(a). Starting from the aforementioned preliminary considerations about the QL-MAC parameters settings (frame length=1s, packet rate=1pkt/s and slot#=8), we performed an initial set of simulations to figure out the most suitable value of λ to be adopted in the subsequent tests, see Table 2.

TABLE 2. λ analysis for small scale scenario (in bold, the optimum value for each metric).

	λ=0.95	λ=0.5	λ=0.05
PDR	70.5%	74.2%	**80.4%**
Slot ON	**23.5%**	24.2%	25.76%
Avg Pkt Lost (pkt/min)	16.7±4.5	14.3±4.6	**10.7±2.3**

As one could note, the $\lambda = 0.05$ case provides a higher PDR and, especially, it ensures greater stability to the protocol (the lowest average packet lost per minute with the smallest standard deviation). Indeed, after a quick transition phase, all the nodes find a radio scheduling alignment which is stably maintained over the time, thus minimizing collision and overheard packets. In detail, nodes initially hold the radio on in all the slots for some frames (transition phase, the whole 8 slots are consecutively on) until they get synchronized and converge to the optimal, energy saving, radio scheduling configuration (an average of 2 active slots), as reported in Fig. 9 (left side). Such configuration is "guessed" more than "learnt" with higher λ values, since nodes try to directly synchronize with each other minimizing the number of slot with radio on. However, this implies that a definitive radio scheduling configuration, in practice, will never be reached, with continuous misalignment among the nodes (see Fig. 9), ride side and, hence, a higher probability of packet lost. Indeed, the λ value used in the learning phase affects the Q-Value of each slot, as a high λ value gives greater importance to the current reward function with respect to the Q-value of the slot in the previous frame. So, if a node decides to send messages in a slot different from the one used in the previous frame, there will be a misalignment and a stable configuration will never be achieved.

After this analysis, it has been straightforwardly decided to fix the λ value to 0.05 since the only drawback of such configuration is a light increase of the energy expenditure (25.76% vs 23.5%), which, however, is fully compensated by a higher PDR and overall stability. Hence, the QL-MAC performances with the $\lambda = 0.05$ setting have been compared by means of simulation and real tests, with conventional asynchronous CSMA-CA MAC with duty cycle at 100% (in the following, MAC100) and 60% (in the following, MAC60). Results are reported in Fig. 10. Both in simulated and real tests, QL-MAC and MAC100 have almost the same PDR performance, but they differ for the node energy consumption. In fact, QL-MAC allows nodes to spend much less energy, as a result of the sleep/wake-up radio schedule. Moreover, QL-MAC notably outperforms the MAC60 especially in terms of PDR but also with respect to the energy consumption. In order to figure out the implication of the adaptive QL-MAC radio scheduling on the nodes lifetime, a long-time set of tests has been carried out (using new AA alkaline batteries - LR6 E91) aiming to determine the nodes' maximum working time (in hours). Fig. 11 shows that the TelosB cutoff voltage of 2.1V (namely, the energy threshold under which the node's chip radio stop operating even if batteries still have a few of residual energy [27]) is reached in 27 hours by the MAC100 (almost 28 hours in simulation), 45 hours by MAC60 (almost 47 in simulation), and in 104 hours by QL-MAC (almost 108 in simulation). It means that, in terms of network lifetime for both the real and simulated test, QL-MAC outperforms of 4.5× the MAC60 and of 26× the MAC100. In summary,

TABLE 3. λ **analysis for large scale scenario (in bold, the optimum value for each metric).**

	4x4 nodes			7x7 nodes			10x10 nodes		
	λ=0.95	λ=0.5	λ=0.05	λ=0.95	λ=0.5	λ=0.05	λ=0.95	λ=0.5	λ=0.05
PDR	67%	70%	**73%**	54%	61%	**67%**	42%	45%	**52%**
Slot ON	**38%**	**38%**	43%	**38%**	**38%**	43%	**38%**	**38%**	43%
Avg Pkt Lost (pkt/min)	24.3±5.3	22.3±3.6	**22.04±3.1**	29.7±5.9	27.3±4.1	**26.3±2.2**	34.7±7.2	31.6±6.1	**28.6±6.7**

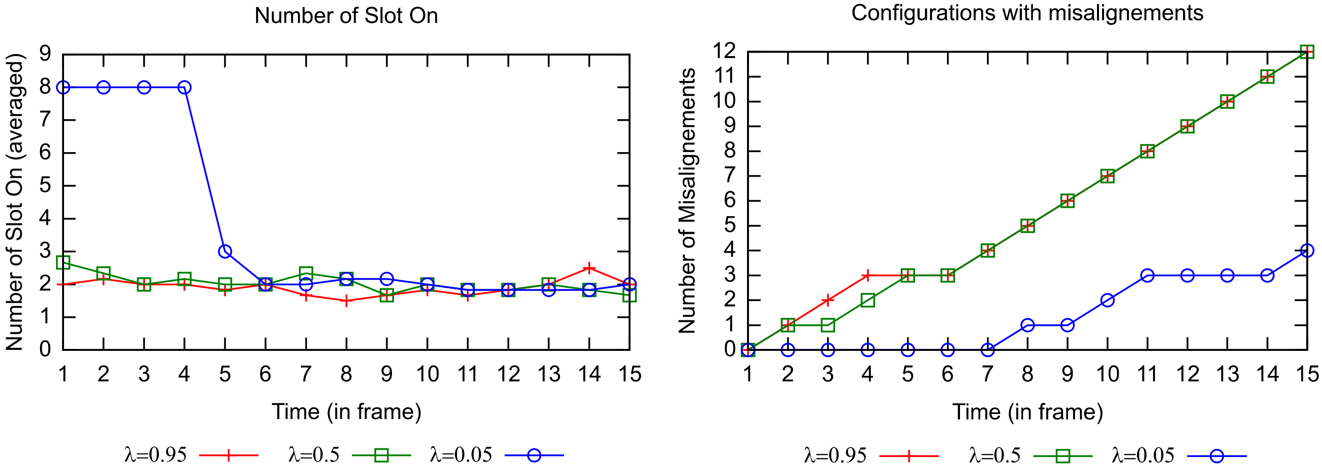

FIGURE 9. By choosing λ = 0.05, after a quick transition phase in which all the slots are consecutively on (left side), the QL-MAC finds a stable radio scheduling configuration which minimizes the number of misalignments (right side).

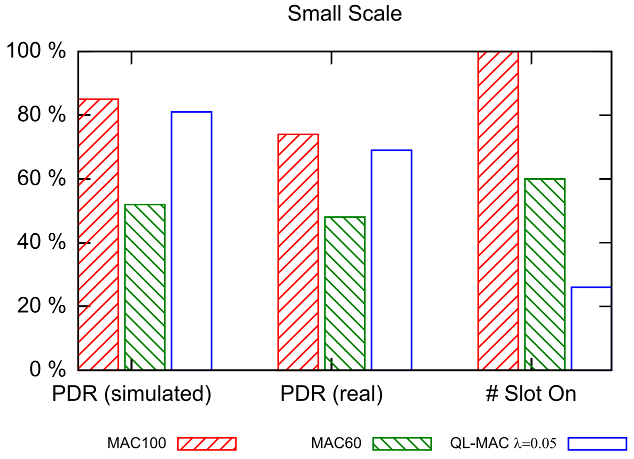

FIGURE 10. Small-scale: QL-MAC vs MAC in simulation and real tests. QL-MAC saves energy without worsening the PDR.

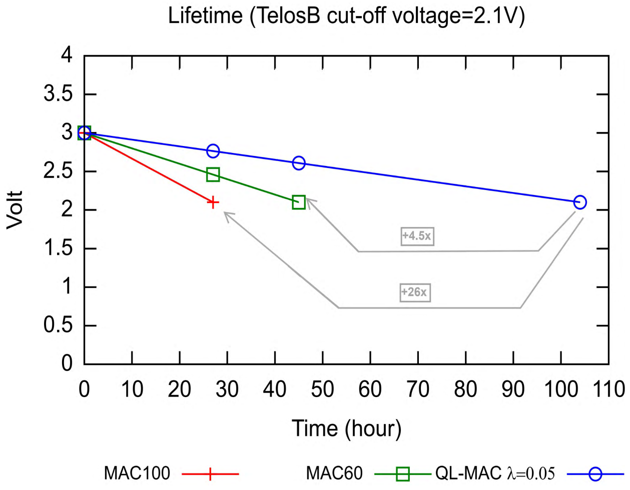

FIGURE 11. Hours spent before motes stop working. The QL-MAC self-adaptive radio scheduling greatly improves the mote's lifetime.

the QL-MAC scheme tested in small scale scenario provides near-optimal performance in terms of PDR and optimal results in terms of energy saving.

Finally, it is worth nothing that simulated and real tests show same trends with slight variations (PDR from simulation is around 10% higher of PDR from real test; energy consumption from simulation is around 2.5% lower with respect to real consumption) and this is due to the radio interference unavoidably featuring the real physical environment. Such data confirms the good reliability of the Contiki Cooja simulator and encouraged us to repeat the tests set for larger scales in the following subsection.

C. LARGE SCALE SCENARIO

The large scale scenario consists of 16, 49 and 100 nodes deployed in a regular grid of 4×4, 7×7, and 10×10 nodes located in a simulation area of 10.000m^2. As made for the small scale scenario, we carried out a λ-analysis to determine the best QL-MAC configuration, see Table 3.

Likewise the λ-analysis of the small scale scenario, the best trade-off PDR/Energy expenditure is reached by using λ = 0.05 and the trends reported in Table 2 are maintained (results provided by λ = 0.5 are intermediate, λ = 0.95 suffers of higher average packets lost and standard deviations).

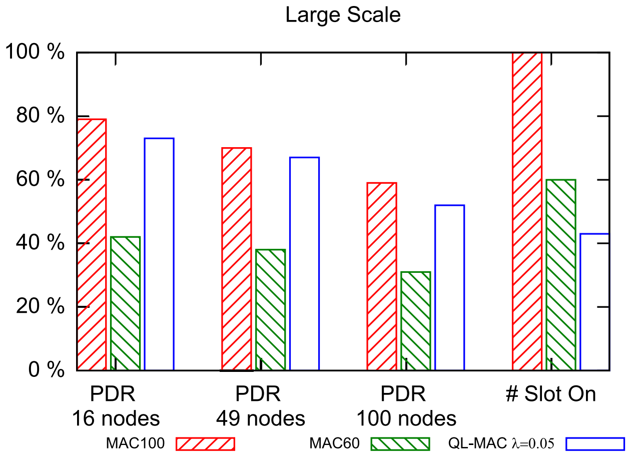

FIGURE 12. Large-scale: QL-MAC vs MAC in simulation tests. As for the small scale scenario, QL-MAC saves energy without worsening the PDR.

Therefore, given the learning rate $\lambda = 0.05$, we compared the QL-MAC performance with conventional asynchronous MAC100 and MAC60 in the cases of 16, 49 and 100 nodes (see Fig. 12). Obtained results showed that, for all the three considered protocols, the PDR decreases and energy consumption increases (the optimal scheduling configuration in large scale requires an average of 3 slots in which the radio is on) as consequence of the higher number of nodes to be synchronized. In particular, the QL-MAC PDR values are close to the optimum PDR of MAC100 but the QL-MAC saves almost the 60% of energy. With respect to MAC60, QL-MAC PDR is almost double in all the three considered network topologies but QL-MAC energy consumption is about 15% higher (60% vs 44%). In conclusion, as for the small scale, also in large scale scenario the QL-MAC achieves the sub-optimal PDR and, at the same time, minimizes the fraction of time in which the radio is switched on and hence optimizes energy consumption of the nodes, thus increasing the network lifetime.

VI. CONCLUSION

AI is a valuable source of algorithms, technologies and paradigms enabling the development of cognitive devices and networks. Following on, the paper investigated the potential of RL techniques to develop an enhanced and intelligent MAC protocol for WSNs. Through subsequent trial-and-error learning, QL-MAC allows each node to independently predict an efficient wake-up schedule to save energy by limiting the period in which the radio is in active mode. Both the experimental and simulation results corroborate our initial hypothesis that RL may be successfully used to realize energy-efficient MAC protocols. The superiority of learning-capable nodes is demonstrated through a comparative study with conventional, reactive MAC. The additional benefit is the full adaptivity of QL-MAC to changes in network topology and other external factors. Through a prototypical implementation on TELOS-B motes, we have also been able to demonstrate the viability of RL in thin computing architectures, which has significant implications for IoT services and, generally, high-density wireless communications.

As a future step, we intend to explore QL-MAC in heterogeneous WSNs, considering also the coexistence of intelligent and conventional nodes. This also opens an avenue to additional issues, like WSNs with hybrid levels of intelligence and the presence of non-collaborative or even malicious nodes. Security and trust are certainly of high priority in this context.

Having stepped away from conventional networking, with the introduction of intelligent/predictive protocols, it will also be interesting to further explore how cross-layer information may further improve the ability of QL-MAC through application- and network-layer information profiling.

ACKNOWLEDGMENT

The authors would like to thank Dr. Stefano Galzarano, Francesco Di Mare, and Luana Giordano for their contributions to the project which has eventually led to this paper.

REFERENCES

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of things (IoT): A vision, architectural elements, and future directions,” *Future Generat. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] X. X. Wang Li and V. C. M. Leung, “Artificial intelligence-based techniques for emerging heterogeneous network: State of the arts, opportunities, and challenges,” *IEEE Access*, vol. 3, pp. 1379–1391, 2015.

[3] N. Morozs, T. Clarke, and D. Grace, “Heuristically accelerated reinforcement learning for dynamic secondary spectrum sharing,” *IEEE Access*, vol. 3, pp. 2771–2783, 2015.

[4] A. Forster, “Machine learning techniques applied to wireless ad-hoc networks: Guide and survey,” in *Proc. 3rd Int. Conf. Intell. Sensors, Sensor Netw. Inf.*, Dec. 2007, pp. 365–370.

[5] W. Qiang and Z. Zhongli, “Reinforcement learning model, algorithms and its application,” in *Proc. Int. Conf. Mechatronic Sci., Electr. Eng. Comput. (MEC)*, Aug. 2011, pp. 1143–1146.

[6] S. Galzarano, G. Fortino, and A. Liotta, “A learning-based MAC for energy efficient wireless sensor networks,” in *Internet and Distributed Computing Systems*, G. Fortino, G. Di Fatta, W. Li, S. Ochoa, A. Cuzzocrea, and M. Pathan, Eds. Cham, Switzerland: Springer, 2014, pp. 396–406.

[7] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge computing: Vision and challenges,” *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[8] M. Chen and V. C. M. Leung, “From cloud-based communications to cognition-based communications: A computing perspective,” *Comput. Commun.*, vol. 128, pp. 74–79, Sep. 2018.

[9] A. Liotta, “The cognitive NET is coming,” *IEEE Spectr.*, vol. 50, no. 8, pp. 26–31, Aug. 2013.

[10] M. C. Huebscher and J. A. McCann, “A survey of autonomic computing—Degrees, models, and applications,” *ACM Comput. Surv.*, vol. 40, no. 3, Aug. 2008, Art. no. 7.

[11] S. Galzarano, C. Savaglio, A. Liotta, and G. Fortino, “Gossiping-based AODV for wireless sensor networks,” in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2013, pp. 26–31.

[12] V. K. Sharma, S. S. P. Shukla, and V. Singh, “A tailored Q- learning for routing in wireless sensor networks,” in *Proc. 2nd IEEE Int. Conf. Parallel, Distrib. Grid Comput.*, Dec. 2012, pp. 663–668.

[13] G. J. Pottie and W. J. Kaiser, “Wireless integrated network sensors,” *Commun. ACM*, vol. 43, no. 5, pp. 51–58, 2000.

[14] M. Chen, S. Gonzalez, and V. C. M. Leung, “Applications and design issues for mobile agents in wireless sensor networks,” *IEEE Wireless Commun.*, vol. 14, no. 6, pp. 20–26, Dec. 2007.

[15] C. Savaglio, G. Fortino, M. Ganzha, M. Paprzycki, C. Bădică, and M. Ivanović, “Agent-based computing in the Internet of Things: A survey,” in *Intelligent Distributed Computing XI*, M. Ivanović, C. Bădică, J. Dix, Z. Jovanović, M. Malgeri, and M. Savić, Eds. Cham, Switzerland: Springer, 2018, pp. 307–320. doi: 10.1007/978-3-319-66379-1_27.

[16] A. Mahmood, K. Shi, S. Khatoon, and M. Xiao, "Data mining techniques for wireless sensor networks: A survey," *Int. J. Distrib. Sensor Netw.*, vol. 9, no. 7, Jul. 2013, Art. no. 406316.

[17] S. Hussain, A. W. Matin, and O. Islam, "Genetic algorithm for energy efficient clusters in wireless sensor networks," in *Proc. 4th Int. Conf. Inf. Technol. (ITNG)*, Apr. 2007, pp. 147–154.

[18] M. Bahrepour, N. Meratnia, and P. J. M. Havinga, "Sensor fusion-based event detection in wireless sensor networks," in *Proc. 6th Annu. Int. Mobile Ubiquitous Syst., Netw. Services, MobiQuitous*, Jul. 2009, pp. 1–8.

[19] W. Li, J. Bao, and W. Shen, "Collaborative wireless sensor networks: A survey," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 2011, pp. 2614–2619.

[20] F. Guerriero and V. Loscrí, P. Pace, and R. Surace, "Neural networks and SDR modulation schemes for wireless mobile nodes: A synergic approach," *Ad Hoc Netw.*, vol. 54, pp. 17–29, Jan. 2017.

[21] M. Mihaylov, K. Tuyls, and A. Nowé, "Decentralized learning in wireless sensor networks," in *Adaptive and Learning Agents*, M. E. Taylor and K. Tuyls, Eds. Berlin, Germany: Springer, 2010, pp. 60–73.

[22] Z. Liu and I. Elhanany, "RL-MAC: A reinforcement learning based MAC protocol for wireless sensor networks," *Int. J. Sensor Netw.*, vol. 1, nos. 3–4, pp. 117–124, Sep. 2006.

[23] Y. Chu, P. D. Mitchell, and D. Grace, "Aloha and Q-learning based medium access control for wireless sensor networks," in *Proc. Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2012, pp. 511–515.

[24] G. P. Joshi, S. Y. Nam, and S. W. Kim, "Cognitive radio wireless sensor networks: Applications, challenges and research trends," *Sensors*, vol. 13, no. 9, pp. 11196–11228, Aug. 2013.

[25] G. M. Huang, W. J. Tao, P. S. Liu, and S. Y. Liu, "Multipath ring routing in wireless sensor networks," *Instrum., Meas., Electron. Inf. Eng.* , vols. 347–350, pp. 701–705, Aug. 2013.

[26] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with COOJA," in *Proc. 31st IEEE Conf. Local Comput. Netw.*, Nov. 2006, pp. 641–648.

[27] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw.*, Apr. 2005, pp. 364–369.

CLAUDIO SAVAGLIO received the B.S., M.S., and Ph.D. degrees in computer engineering from the University of Calabria, in 2010, 2013, and 2018, respectively, where he is currently a Postdoctoral Researcher.

He was a Visiting Researcher with The University of Texas at Dallas, TX, USA, in 2013, the New Jersey Institute of Technology, NJ, USA, in 2016, and the Universitat Politecnica de Valencia, Valencia, Spain, in 2017. His research interests include the Internet of Things, edge computing, network simulation, and agent-oriented middleware and development methodologies.

PASQUALE PACE (M'05) received the Ph.D. degree in information engineering from the University of Calabria, Italy, in 2005, where he is currently an Assistant Professor in telecommunications.

He was a Visiting Researcher with the CCSR, Surrey, U.K., and the Georgia Institute of Technology. He has authored more than 90 papers in international journals, conferences, and books. His research interests include cognitive and opportunistic networks, sensor and self-organized networks, and interoperability of the Internet of Things platforms and devices.

GIANLUCA ALOI (M'02) received the Ph.D. degree in systems engineering and computer science from the DEIS Department, University of Calabria, in 2003.

In 2004, he joined the University of Calabria, where he is currently an Assistant Professor in telecommunications with the Department of Informatics, Modeling, Electronics and System Engineering. His main research interests include spontaneous and reconfigurable wireless networks, cognitive and opportunistic networks, sensor and self-organizing wireless networks, and the Internet of Things technologies.

ANTONIO LIOTTA (SM'15) is currently a Professor of data science and the Founding Director of the Data Science Research Centre, University of Derby, U.K. He is also the Director of the Joint Intellisensing Lab (Europe, Asia, and Australia) and a Guest Professor with Shanghai Ocean University, China. His team is at the forefront of influential research in data science and artificial intelligence, specifically in the context of smart cities, the Internet of Things, and smart sensing.

Dr. Liotta is the Editor-in-Chief of the *Internet of Things* book series (Springer), an Associate Editor of JNSM, IJNM, JMM, and IF, and an Editorial Board Member of six more journals.

GIANCARLO FORTINO (SM'12) received the Ph.D. degree in computer engineering from Unical, in 2000. He is currently a Professor of computer engineering with the Department of Informatics, Modeling, Electronics, and Systems, University of Calabria, Italy.

He is also the Co-Founder and the Chief Executive Officer of SenSysCal S.r.l., a Unical spin-off focused on the innovative Internet of Things (IoT) systems. He has authored over 300 papers in international journals, conferences and books. His research interests include agent-based computing, wireless sensor networks, and the IoT technology.