DCLA: A Duty-Cycle Learning Algorithm for IEEE 802.15.4 Beacon-Enabled WSNs

Rodolfo de Paz and Dirk Pesch

Nimbus Center for Embedded Systems Research
Cork Institute of Technology
Rossa Avenue, Cork Ireland
{rodolfo.depaz,dirk.pesch}@cit.ie

Abstract. The current specification for IEEE 802.15.4 beacon-enabled networks does not define how active and sleep schedules should be configured in order to achieve the optimal network performance in all traffic conditions. Several algorithms exist in the literature that dynamically vary these schedules based on traffic load estimations. But it is still uncertain how these adaptive schemes perform with regard to each other as their performance has only been compared with the standard beacon mode. In this paper, we compare the current state-of-the-art schemes, and with the objective of overcoming the performance deficiencies shown by previous approaches, we introduce DCLA, an adaptive duty-cycle scheme for IEEE 802.15.4 beacon-enabled Wireless Sensor Networks (WSN) that employs a reinforcement learning technique. Simulation results show that the proposed scheme achieves the best overall network performance for a wide range of traffic conditions and performance parameters when compared with existing IEEE 802.15.4 duty-cycle adaptation schemes.

Keywords: Wireless Sensor Networks (WSNs), IEEE 802.15.4, duty-cycle, energy efficiency, machine learning, reinforcement learning.

1 Introduction

A Wireless Sensor Network (WSN) is a collection of wireless sensor nodes deployed to measure and report through collaboration certain parameters such as temperature, pressure, humidity, etc. These nodes are usually battery operated and, in most of the deployments, replacing or recharging their batteries is infeasible. Consequently, the power consumption is considered as a primary requirement for WSN communication protocols. Specifically, at the medium access control (MAC) layer a balance needs to be struck between achieving high quality radio resource allocation and energy expenditure. For this, the idle listening problem must be solved as it has been identified as one of the major sources of energy expenditure. This is caused when nodes do not know when the data traffic is generated from other nodes and its transceiver continuously stays in the receiving mode even when there is no data traffic for them.

The IEEE 802.15.4 standard [1], which is currently the most commercially adopted MAC protocol for WSNs, specifies the beacon enabled mode for energy efficient operation. This mode is designed to support the transmission of beacon frames from

J. Zheng, D. Simplot-Ryl, V.C.M. Leung (Eds.): ADHOCNETS 2010, LNICST 49, pp. 217–232, 2010.

coordinator to end devices allowing node synchronization. Synchronization allows devices to sleep between coordinated transmissions avoiding idle listening, which results in prolonged network lifetimes.

The beacon mode employs the superframe structure depicted in Fig. 1. Its format is based on two fundamental parameters: the Beacon Interval (BI), which defines the time between two consecutive beacon frames, and the Superframe Duration (SD), which defines the nodes' active period in the BI. The superframe duration is composed by a contention access period (CAP) in which all devices use a slotted CSMA/CA protocol to gain access for time slots and a contention free period (CFP) for QoS demanding applications. The coordinator can introduce an inactive period to reduce energy consumption by choosing BI > SD. BI and SD are determined by two parameters, the Beacon Order (BO) and the Superframe Order (SO), respectively, as follows:

$$\left.\begin{aligned} BI &= aBaseSuperframeDuration \cdot 2^{BO} \\ SD &= aBaseSuperframeSuration \cdot 2^{SO} \end{aligned}\right\} \quad \text{for } 0 \le SO \le BO \le 14 \tag{1}$$

where aBaseSuperframeDuration is a parameter defined by the standard that depends on the data rate and frequency employed and denotes the minimum duration of the superframe which corresponds to SO = 0.

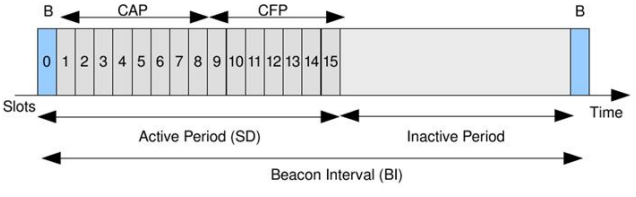

Fig. 1. Superframe structure of the IEEE 802.15.4 beacon enabled mode

As the energy savings in the beacon-enabled mode depend on the amount of periodic sleep periods introduced, it is important to control the fraction of the time that the node is active. This time, known as duty-cycle (DC), can be computed as the ratio between the superframe duration and the beacon interval that can be related to BO, SO as follows:

$$DC = \frac{SD}{BI} = 2^{SO-BO} \tag{2}$$

The smaller the duty-cycle the lower is the energy consumption. However, due to the inherent resource constrictions of sensor nodes, a small duty-cycle can cause buffer overflows and delays. On the other hand, although a high duty-cycle enables end devices to transmit a higher number of data frames and decrease delay it also increases the time the coordinator spends in idle listening. Consequently, duty-cycle adaptation is necessary to enhance the performance of IEEE 802.15.4 beacon-enabled networks.

Several works in the literature have been conducted for adjusting the duty cycle to the traffic load using the IEEE 802.15.4 beacon-enabled MAC. For simplicity, these works adapt the duty cycle fixing SO and adapting BO to traffic [2][3] or fixing BO

and adapting SO [4][5]. However, we observed that, with these techniques, if SO is fixed to a low value, the WSN produce lower network throughput than a higher SO value with the same duty-cycle ratio. Alternatively, algorithms that fix BO can suffer from unnecessary beacon overheads in idle WSNs. This motivated us to explore a new duty-cycle adaptation algorithm that could jointly adjust BO and SO values to find the optimal network performance. Also, as no comparison has been carried out among the different IEEE 802.15.4 adaptive duty-cycle adaptation schemes proposed by the community, there is still a doubt on how these approaches performs with regard to each other for different traffic loads. Therefore, it is also the objective of this work to carry out a performance comparison of previous IEEE 802.15.4 duty-cycle adaptation algorithms.

For the main goal of finding the optimal duty cycle, we have decided to employ a reinforcement learning (RL) technique known as Q-learning. RL is a machine learning (ML) approach that finds the optimum value through trial-error iterations. Within the ML field, reinforcement learning (RL) is the most widely used method for solving WSN problems as they incur only minimal communication overhead and achieve optimal results [6]. In particular the Q-learning algorithm is well suited to this problem as it does not require any prior information about the environment. Additionally, the only memory and computation requirements for the nodes are the values of the possible actions the algorithm can take. We thus propose in this paper a duty-cycle learning adaptation algorithm (DCLA) for IEEE 802.15.4 beacon-enabled networks that employs the Q-learning technique. Our simulation results show that the proposed algorithm outperforms the existing state-of-the-art schemes in terms of average drop rate, throughput, energy efficiency and end-to-end delay.

The rest of this paper is organised as follows. In Section 2, related works on duty-cycle adaptation schemes for IEEE 802.15.4 beacon-enabled networks are discussed. In section 3, we introduce the basis of the reinforcement learning technique employed. Section 4 describes in detail the design of the proposed duty-cycle learning algorithm which includes traffic estimation, design of the RL agent and the algorithm for BO and SO joint adaptation. Section 5 presents simulation results that accentuate the distinct advantages of the proposed approach when compared with the existing schemes. Finally, the conclusions are drawn and future works are proposed in Section 6.

2 Related Work

Recently, the performance trade-off in IEEE 802.15.4 beacon-enabled networks between power consumption, reliability and delay has produced attention within the research community. Inline with this, several MAC layer duty-cycle adaptation methods that try to balance those objectives with the ultimate purpose of increasing life-time have been proposed. We discuss in this section the four IEEE 802.15.4 duty-cycle adaptation schemes found in the literature.

The earliest work in 802.15.4 duty-cycle adaptation is known as the Beacon Order Adaptation Algorithm (BOAA) [2]. BOAA fixes SO to zero and adapts the beacon order using a coordinator buffer matrix $B(n_{ED}, l_b)$. The matrix records end devices' number of received messages n_{ED} at coordinator during a number of beacon intervals l_b. At the beginning of the network operation the buffer matrix is assumed to be empty

and a counter is initialized. Every beacon interval the number of messages received by every end device is stored in the matrix rows. After a l_b number of beacon intervals the maximum number of messages n_{max} from any of the end devices is extracted from the B matrix. Then, BO is increased or decreased depending if n_{max} is lower or greater than a pre-fixed lower or an upper bound. The B matrix gives memory to the algorithm as the messages are collected by a l_b number of beacon intervals. Although this is a nice feature, this matrix might not be scalable for large networks as the number of rows increases with the number of end devices in the topology.

Also, BOAA does not work well if the traffic is not uniformly distributed among the sensor nodes as the algorithm only takes into account the device with the highest number of messages sent. Aware of this, B. Gao and C. He proposed an Individual Beacon Order Adaptation Algorithm (IBOAA) [3]. Their proposal aims to improve BOAA in networks where the traffic is not uniformly distributed. With this aim, the algorithm adapts the beacon interval of each end device individually based on a traffic queue flag embedded by nodes in one of the reserved bits of the standard MAC header. Because of the difference in the lengths of the individual beacon intervals, a minimal beacon interval BO_{min} is fixed in the coordinator as the reference period of any possible value of the end devices' beacon intervals BI_i. Moreover, the algorithm sets a common maximum value BO_{max} to avoid unbearable delays. Therefore any BI_i set for a sensor device is limited in the range $[BO_{min}, BO_{max}]$. In most scenarios, this imposes the limitation for coordinator node to be plug powered as for low values of BO_{min} the coordinator cannot sleep even if the sensors become idle, while setting a high BO_{min} value would cause the loss of many packets in congestion situations. Finally, it is worth noticing that this scheme introduces extra control overhead in the beacon payload compared to the other approaches since the coordinator has to indicate the BO increase or decrease (1 byte) and the address list of end devices which BOs are going to be modified (variable size).

Authors in [4] propose an adaptive MAC for efficient low power communications, named as AMPE for the rest of the paper. A coordinator running AMPE fixes BO to a maximum BO_{max} and adapts SO to the superframe occupation. Then, it estimates the superframe utilization by performing CCA measurements every 30 beacon intervals or whenever the number of received packets is halved or doubled. Then SO is increased or decreased depending on comparisons between a fixed threshold and the measured superframe utilization. CCA measurements may give a more accurate value of the superframe occupation than estimations, but they would imply a great deal of work to the coordinator in traffic variant scenarios where the measurement needs to be frequently activated. On the other hand, in scenarios where a small gradual increase of the traffic exists, the estimation updates could be too infrequent which in turn would decrease the performance of the network.

Finally, J.Jeon et. al. introduced a Duty Cycle Adaptation (DCA) Algorithm for 802.15.4 beacon-enabled WSNs in [5]. DCA assumes BO constant and adapts SO to the traffic according to superframe estimations. In order to gather the end devices traffic information, the scheme modifies the reserved frame control field in the MPDU header as was also proposed by IBOA. However, in this case the algorithm also embeds the queuing delay. Based on the collected information the DCA coordinator estimates the number of packets being queued in the end devices and varies SO accordingly. This is the first scheme that takes into account delay measurements in the

DC adaptation, however, a problem arises in idle networks as DCA does not adapt SO if no packets are received from end devices and it is limited to a fixed BO_{max} equal to seven due to the manner the superframe occupation is estimated in the scheme.

3 Reinforcement Learning

The idea behind designing any learning system is to guarantee robust behavior without the complete knowledge, if any, of the system/environment to be controlled. In reinforcement learning (RL) an agent takes actions and learns from its environment through the rewards received. A crucial advantage of RL compared to other machine learning approaches is that it requires no information about the environment except for the reinforcement signal. This is especially appropriate to embedded systems such as wireless sensor networks were resources are scarce.

The standard reinforcement-learning model is depicted in Fig. 2. An agent must learn the best behavior, formally called policy, through trial and error interactions with a random environment. At each step, the agent selects some possible action, *a,* and receives an immediate reward, *r*, from the environment for the current state *s*. If delayed reinforcement is employed, the action will not only affect the immediate reward but also future rewards. The agent then chooses an action *a* following a defined policy $\pi(s)$ to generate an output. The action changes the state of the environment, and the value of this state transition is communicated again to the agent through the reinforcement signal *r*. The process is then repeated. The agent's objective is to choose actions that tend to increase the long-run sum of values of the reinforcement signal.

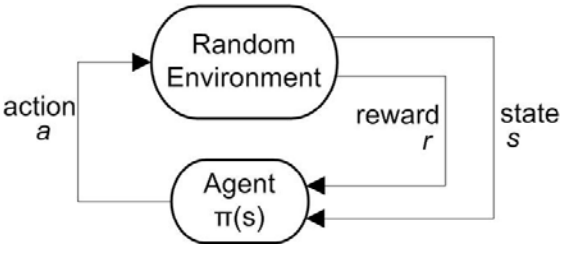

Fig. 2. The standard reinforcement learning model

Formally, the model is described by the experience tuple (S, A, T, R), where S is a discrete set of environment states $s_1, s_2,\ldots,s_n$, A is the set of possible actions from each state $a_1, a_2,\ldots,a_m$, T(s,a,s') is the transition probability from state *s* to a successor state *s'* when taking action *a* and R(s,a) is the reward function. The reward function in delayed reinforcement is computed as a combination of the immediate reward *r* plus future discounted rewards as follows:

$$R(s,a) = \sum_t \gamma^t r_t = r_0 + \gamma r_1 + \gamma^2 r_2 \ldots \quad (3)$$

The discount factor γ is a number in the range [0..1] and is used to weight near term reinforcement more heavily than distant future reinforcement. The closer γ is to 1 the greater the weight of future reinforcements is.

The agent's rule for selecting actions, which is the policy $\pi(s,a)$, is a mapping from each state *s* and action *a* to the transition probability T of taking action *a* in state *s*. The agent's goal is to find a policy π that maximizes the expected return by either maximizing the state-value $V^{\pi}(s)$ or action-value $Q^{\pi}(s,a)$ functions.

$$V^{\pi}(s) = E_{\pi}\{R \mid s, \pi\} \tag{4}$$

$$Q^{\pi}(s,a) = E_{\pi}\{R \mid s, a, \pi\} \tag{5}$$

The state-value function $V^{\pi}(s)$ estimates the expected return when starting from state *s* and following π thereafter whereas the action-value $Q^{\pi}(s,a)$ is the function that estimates the reward of taking action a in state *s* following policy π. Both functions can be related by means of the transition probability and reward functions as follows:

$$Q(s,a) = r_0 + \gamma \sum_{s' \in S} T(s,a,s') \cdot V(s') \tag{6}$$

In the literature, several researchers have applied reinforcement schemes to design WSN communication protocols. Among them, RL-MAC [7] applies reinforcement learning to adjust the sleeping schedule of a MAC protocol in a WSN setting. The MAC protocol is similar in its idea to DCLA and it is therefore worthwhile explaining their differences. RL-MAC employs a time frame based structure similar to S-MAC [8]. Time is divided into frames while each frame is further divided into time slots and updating schedules is accomplished by sending SYNC packets similar to the beacon frame of the IEEE 802.15.4 MAC. However, as opposite to the standard, if multiple neighbors want to transmit to a node, they contend for the medium using a mechanism similar to that in IEEE 802.11, i.e., using RTS (Request To Send) and CTS (Clear To Send) packets.

Although RTS/CTS can alleviate the hidden terminal problem, it incurs high overhead (40% to 75% of the channel capacity [9]) in WSNs because data packets are typically very small. Overhead has been identified as one of the sources that cause energy inefficiency in MAC protocols and for this reason the 802.15.4 MAC does not allow the transmission of RTS/CTS packets. However, this means that the current standard needs of some upper layer mechanism to avoid beacon collisions and allow multihop support. Adapting nodes' duty-cycles while supporting scheduled multihop IEEE 802.15.4 communications is therefore a whole and complex new problem out of the scope of this paper.

Also, some differences also exist in the manner the both RL agents are defined. In the RL-MAC, the agent which runs in every node, employs the number of packets queued for transmitting at the beginning of the frame as the state and the reserved active time as the action generated. The reward function depends on the number of waiting messages on the nodes and on the number of successfully transmitted messages during the reserved slot. However, in our case the agent is only present in the coordinator which has to rely on the information collected from end devices to derive

the optimal policy. This scenario is challenging in the way that coordinator cannot employ the local information of its transmitting queue to build the reward function but on some estimation of end devices traffic status. Further, the DCLA agent employs a heterogeneous reward function that aims to improve the multiple goals present in WSNs as it is explained in the following section.

4 DCLA Protocol Design

In this section, the design of the DCLA protocol is explained in detail. Specifically, we discuss the traffic estimation used by coordinator; we formulate the actions, states, reward and policy functions employed by the agent to find the optimum duty-cycle and we describe the algorithm for the selection of Beacon and Superframe Order values.

4.1 Coordinator Traffic Estimation

The coordinator node needs to employ some estimation of the end devices' traffic requirements to find the optimal duty-cycle. For this, the number of received messages during the active period is collected by the coordinator. On the other side, end devices embed their transmit queue occupation and delay values in the MAC header of sent data frames.

The number of messages m received from end devices is employed by coordinator to estimate the superframe utilization SF_u, which can be defined as the ratio of the superframe utilized for data communication as follows:

$$SF_u = \left(\frac{SD - T_{beacon}}{m \cdot T_s} \right) \tag{7}$$

This estimation is calculated by dividing the total time available for data transmission, which is the superframe duration SD minus the time T_{beacon} spent for beacon transmission, with the time portion of the superframe $m \cdot T_s$ end devices employed to transmit the m messages received by coordinator. The time used by coordinator to transmit a beacon is calculated by summing the 802.15.4 MAC and physical overhead (30 symbols) plus the beacon payload. Alternatively, the superframe portion utilized by a sensor node to send one message, denoted as T_s, is estimated as follows:

$$T_S = T_{CCA} + T_{DATA} + T_{IFS} + T_{ACK} \tag{8}$$

Where the time for clear assignment T_{CCA} is two backoff slots (40 symbols); the time to transmit the data frame T_{DATA} is the standard 802.15.4 MAC and physical headers (38 symbols) plus the payload length; the inter-frame spacing T_{IFS} is the sum of the turnaround time (12 symbols) plus the time to find the next backoff boundary (up to 20 symbols); and the T_{ACK} is the time to receive the acknowledgement frame from coordinator (equal to 22 symbols). This estimation does not consider the time the node is in backoff because during this time the radio is in idle mode and others are free to use the superframe to transmit.

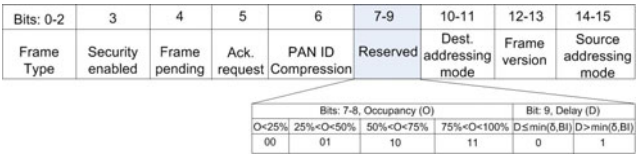

Fig. 3. The queuing occupancy and delay bits are embedded inside the reserved field of the MAC frame control field

On transmitter side, end devices embed the queue occupation and delay in the MAC header of the first frame sent within the superframe. These values represent the packet accumulation and delay caused by the last duty-cycle selection. Specifically, this information is embedded in the 3 reserved bits of the frame control field (as per Fig. 3). Two bits are used by the queue occupancy (O) to indicate 4 different levels whereas the queuing delay (D) is divided into 2 levels. The queuing delay bit indicates whether the delay is above '1' or below '0' an application defined delay threshold (δ).

4.2 The DCLA Agent

The DCLA agent is assumed to be run by a coordinator that does not have any knowledge with regard to the traffic generated by end devices. The coordinator should therefore determine the optimal duty-cycle without any prior knowledge of the environment by employing a RL technique known as Q-Learning [10]. In our agent, we select the energy saving level ℓ, defined as the difference between BO and SO values, as the state, and the actions are represented by the possible state transitions. Then, let L be the set of states $\{\ell_0, \ell_1, \ldots, \ell_n\}$ of our DCLA approach and let ρ be the number of possible states as follows:

$$L\{\ell_0, \ell_1, \ldots, \ell_n\} = \sum_{\ell=0}^{\rho} 2^{-\ell} \quad \rho \leq 14 \tag{9}$$

The number of states ρ must be lower to 14 due to the limitation of the SO and BO values in the 802.15.4 standard specification. The energy saving level ℓ of a sensor node can be computed at any time by employing e.q. (**3**) as follows:

$$DC = 2^{SO-BO} = 2^{-\ell} \rightarrow \ell = BO - SO \quad 0 \leq \ell \leq \rho \tag{10}$$

Thus, ℓ represents the level of energy the sensor node is saving which is inversely proportional to the activity of the node (duty cycle). A high ℓ value means that the duty cycle is low and the node is thus saving batteries. Alternatively, a low ℓ value implies that the battery will be rapidly depleted. When the DCLA agent moves to a new state ℓ, its corresponding duty cycle can be calculated using equation (**10**) whereas BO, SO values can be selected following the algorithm described in section 4.3.

In Q-learning, the objective is to find the optimal policy π^*, by means of delayed reinforcement. The policies and the value function are represented by a two-dimensional lookup table, known as the Q table, indexed by state-action pairs. During

the Q-learning process, a learned action value function Q directly approximates Q^* through value iteration. Mathematically, the optimal policy and Q value are defined as:

$$\pi^*(s) = \arg\max_{a \in A(s)}(Q^*(s,a)) \tag{11}$$

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \cdot \max_{a' \in A(s')}(Q^*(s',a')) \tag{12}$$

And the rule to update the Q value after each time step *t* that the agent takes is:

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha[R(s,a) - Q_t(s,a)] \tag{13}$$

As we can see, every new Q value is computed as the sum of the old value and a correction term. This term consists of the total received reward and the last Q value. The learning rate α is a number in the range [0..1] that prevents the Q values from changing too fast and thus oscillating. This value is set to 0.1 in our tests.

For the DCLA agent, we employ one step Q-learning technique with discount factor γ equal to 0.5. Following the general delayed reinforcement definition of the reward function (see equation **3**), the total received reward is obtained as the sum of the immediate reward r_t and the discounted future rewards (Q values in this case). Due to sensor nodes' memory constraints we only account for the immediate and next future reward as follows:

$$R(s,a) = r_t + \gamma \max_{a' \in A(s')} Q(s_{t+1}, a_{t+1}) \tag{14}$$

The design of the immediate reward r_t is crucial as it allows the DCLA agent to learn the optimal duty-cycle. In our case, the agent objective is to improve the general overall WSN performance, which can be viewed as the sum of multiple goals such as: reduce energy consumption, increase throughput, decrease end-to-end delay and packet drop. To reflect this aim, we define a heterogeneous reward function in which the immediate reward is defined as the sum of four components: the energy r_e, superframe utilization r_u, delay r_d, and queue occupation r_o rewards as shown in equation **15**. The presence of positive and negative reinforcement in the continuous learning algorithm guarantees that the learner will not get stuck in local optimum.

$$r_t = r_e + r_u + r_d + r_o = \begin{cases} r_e = (1 - DC) \\ r_u = SF_u \\ r_d = -\overline{D} \\ r_o = \overline{O}(t-1) - 2 \cdot \overline{O}(t) \end{cases} \tag{15}$$

The energy reward r_e represents the amount of energy that the network saves through the duty-cycle selection. The lower is the duty-cycle the higher is the energy savings for the nodes. This reward is thus defined as a positive function that depends on the duty-cycle value *DC* as per equation **15**.

The utilization reward r_u is equal to the ratio of the superframe utilized for data communication as calculated in equation **7**. This is also a positive reward as the greater the superframe utilization SF_u is, the less time the coordinator is assumed to be in idle listening.

On the other hand, the delay reward r_d is calculated as a negative function by averaging the delay bits D received from end devices during the last active period. An increase in the delay thus represents a penalty from the environment in the computation of the new Q-value.

A final reward is introduced to avoid possible packet drops due to queues overloads. For this, we control end devices' transmit queues occupation by averaging the received O values in the headers of the data frames sent by end devices. The resulting value is used to calculate the occupation reward r_o as the difference between the occupation in the last *(t-1)* and current t steps of the algorithm. This reward can therefore oscillate among positive and negative values depending on previous and current queue occupation. It is seen as a penalty function if the number of queued packets has increased during the last beacon interval, whereas will be a positive reward function if lesser packets have been queued during the last interval.

We next discuss how the DCLA agent selects the next action to take. As in the Q-algorithm the best possible action is never known a-priori, the agent starts with an empty Q-table and it starts trying several actions to learn the optimal value. The most straight forward rule for selecting actions would be the greedy policy, which selects the action that maximizes the Q-value for the current state. However, if this policy is employed, the algorithm could get stuck at a local optimum before finding the global optima. In order to overcome this situation, an exploration strategy is defined within the DCLA policy.

The idea is thus to design a policy that can find an optimal balance between exploring and exploiting as the training progresses by employing the increasing amount of learnt knowledge. DCLA initially maximizes the exploration by selecting actions randomly. Then, the responses from the environment are observed, action probabilities are updated based on that responses and the procedure is repeated until all the states in the Q table have been visited. When all these positions in the table are filled, the exploration rate is decreased to a probability ε in which the best action is chosen following an epsilon greedy policy. In our experiments, ε is set to 0.01. Thus, the DCLA algorithm follows the next policy:

$$\pi(s,a) = \varepsilon \cdot \mathrm{rand}[Q(s,a)] + (1-\varepsilon) \cdot \arg\max_a [Q(s,a)] \tag{16}$$

The pseudo-code for the Q-learning algorithm that is run by the DCLA agent every beacon interval to decide the optimal duty-cycle is defined as follows:

FOR each <s,a> **DO**
Initialize table entry: Q(s,a)←0
Observe current energy level state $s_t = \ell_t$
WHILE (true) **DO**
Select action a_t according to π(s,a) and execute it
Observe the new state s_{t+1} and receive the immediate reward r_t
Update table entry $Q(s_t,a_t)$ as follows:
$Q(s_t,a_t) = Q(s_t,a_t) + \alpha[r_t + \gamma \cdot \max_a Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)]$
Move to the next energy level state $s_{t+1} = \ell_{t+1}$

4.3 Beacon Order and Superframe Order Selection

Immediately after the DCLA agent decides to move to a new energy saving state ℓ_{t+1}, the beacon order *BO* and superframe order *SO* need to be selected. The simplest way to do this, it is to fix one of both parameters and select the other according to equation **10**. However, some observations lead us to take a slightly different approach. According to this equation, for a specific energy level ℓ a total number of $k = \rho - \ell$ combinations of SO, BO are possible. Simulations showed that, among them, those that produce low SO, BO values obtain lower network throughput and higher energy expenditure than combinations with higher SO, BO values and same duty-cycle.

This realized that these results are basically due to two factors. First, the overhead of the beacon frame is more significant with low BO values since beacons are more frequent. Therefore, if this extra control overhead is not controlled may result in higher energy expenditure. Second, CCA deference is also more frequent with lower SO values, leading to more collisions at the start of each superframe. Let us explain this problem. Two or more nodes defer their transmissions when the remaining time in the current superframe is insufficient. All such nodes will start their CCAs immediately following the beacon frame. In the first two backoff periods, the channel will be found idle. Consequently, all the nodes will conclude that the channel is free and start their transmission in the third backoff period resulting in a collision. This contention problem is obviously more pronounced for small superframe values i.e. SO = 0 or 1 causing lower throughput and an increased number of retransmissions.

However, we should be careful with increasing BO and SO values since average delays and packet drops can be proportionally increased. This is because high BO values lead to longer sleep times between beacons. We then decided to increase the BO and SO values only if the same duty-cycle is selected a consecutive number of times (denoted as *B* in Fig. 4) unless the average delay and/or queue occupation parameters are above some pre-defined threshold or the network is working at maximum duty-cycle. Experimental tests suggested 5 as a proper value for this threshold. The exception of the maximum duty-cycle is because in this case BO is equal to SO which means there cannot be any harm in terms of delay or packet drop when increasing the beacon periodicity. Alternatively, BO and SO values are reduced while maintaining the same duty cycle, anytime delay or queue occupation is above the threshold as it means.

This beacon order and superframe order selection allows devices to reduce beacon overhead and CCA deference without increasing delay or packet drop as results will show later. The flow chart for the BO/SO selection algorithm is depicted in Fig. 4.

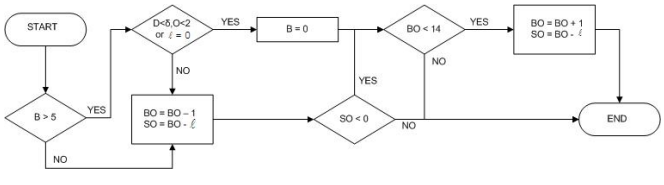

Fig. 4. Flow chart for the Beacon Order and Superframe Order selection algorithm. B is incremented each time the DCLA agent selects the same duty cycle ($\ell_t = \ell_{t-1}$).

5 Simulation Environment and Results

Several simulations were carried out to compare DCLA performance with the state-of-the-art IEEE 802.15.4 duty-cycle adaptation schemes described in Section 2. All the algorithms were implemented using as a base the Open-ZB IEEE802.15.4 simulation model [11] developed with OPNET Modeler [12] and following the 2.4GHz physical layer definition. The battery model was modified to more accurately estimate the radio chip consumed energy during the simulation. The computation was done by summing up the energy spent in the different power modes (transmitting, receiving, idle and power down) of the CC2420 datasheet [13].

The network topology used for our simulations is a star topology of MicaZ nodes [14]. The WSN is composed by a central coordinator surrounded by 8 equispaced sensor nodes. The coordinator node employs any of the described adaptation algorithms to determine the duty cycle and periodically broadcasts the result inside the beacon. The other nodes act as end devices without the capacity of beacon transmission. End devices periodically listen for beacons and generate data frames containing 40 bytes of payload following a Poisson distribution. Several tests, lasting 10 hours each, were carried out with different traffic conditions by varying the mean of the statistical distribution within a range from 0.1 to 10 seconds. End devices transmit the data frames during the contention access period (CAP) of the superframe using CSMA-CA channel access technique. If the transmitting queue is found empty, the end device goes into sleep mode to save energy. The contention free period of the superframe is not used for transmissions in this scenario. Every time the coordinator correctly receives a data frame an acknowledgement is sent to the end device as described by the 802.15.4 standard specification.

In the proposed scenario the delay threshold δ for the WSN application running in the end devices is set to 1 second. BO and SO values are initially set to 7 and 0 respectively for all the duty-cycle adaptation schemes. The only exception for that is the individual beacon order adaptation (IBOA) in which the coordinator's beacon order is set to 0 as it is used by end devices as the reference time. End Devices' transmitting queues have a size of 1Kbyte which allows them to accumulate around 18 data frames.

5.1 Simulation Results

Simulations results present a comparison between the state-of-the-art duty-cycle adaptation schemes and the proposed learning technique for duty cycle adaptation for different traffic conditions. The WSN performance metrics considered are: drop rate, throughput, end-to-end delay and energy efficiency.

Fig. 5 shows the duty-cycle selected by a coordinator that employs the DCLA algorithm described in this paper. The resulting values are in the range 0.1-100% reflecting the different traffic loads generated. In addition, it can be clearly seen that the higher is the traffic load the higher is the duty cycle selected, as would be expected.

Fig. 6 shows the average drop rate for all the different approaches. This parameter represents the average number of bits per second dropped in end devices transmitting queue. A frame in the simulator can be dropped due to the following reasons: (i) the frame cannot be inserted in the transmitting queue because it is already full; (ii) the device fails more than 4 times to acquire the channel, which corresponds to the default

value for the *macMaxCSMABackoffs* attribute; (iii) the channel has been acquired but the data transfer attempt fails more than 4 times which is the default value for (1+*macMaxFrameRetries*). We only show in Fig. 6 the drop rate results for the case (i) as it is the situation that mostly depends on the duty-cycle selection. Inter-arrival times greater than 1 second are omitted in the figure as the number of dropped packets was nearly zero. On the other hand, when the nodes' packet inter-arrival time is at its maximum (0.1sec) the network is overloaded. This means that end devices are not able to transmit all the generated data to coordinator even if it would be awake the 100% of the time. In this case, we have therefore high drop rates for all the schemes. But due to the proposed algorithm's ability to control the occupation of end devices transmit queues through the optimum duty-cycle selection, the drop rate is much lower than with the other schemes. The figure also shows that the worst performance is given by AMPE and BOAA as they do not consider the end devices queues status for the duty-cycle selection.

Fig. 7 depicts the average throughput. DCLA also outperforms the other schemes studied here as it presents the maximum throughput for all the range of traffic loads presented. This is mainly due to two reasons, firstly the coordinator is able to find a near optimal duty-cycle thanks to the Q-learning algorithm employed, and secondly the BO, SO selection algorithm avoids low BO, SO values that could result in CCA deferences when occupation and queuing delay permits it.

Fig. 8 shows the average end-to-end delay. This value is obtained as the delay in seconds since the frame was generated by the end device's application layer until is received by the physical layer at coordinator node. As we explained in Section 4.2 , DCLA considers a reward component that accounts for application delay. The figure clearly shows how DCLA meets the 1sec delay imposed in our scenario. In this case DCA also shows good performance as it also considers queuing delays in the duty-cycle decision process. Other algorithms increase their delays as the traffic in the network decreases as they try to minimize energy consumption without taking into account the delay requirement. Specifically AMPE presents the highest delay as it updates the duty-cycle decision after longer time intervals than the other schemes. We can say from the results that AMPE, IBOA and BOAA would not be suitable for WSN applications with specific delay bounds.

Perhaps, the most important metric for the wireless sensor network is the energy efficiency. For its computation in the simulator, we calculate the number of bits correctly delivered to coordinator per Joule spent by WSN nodes' batteries during each simulation run. The results for this performance parameter are shown in Fig. 9. DCLA presents the best energy efficiency when compared to the other approaches. This shows again that the learning algorithm in combination with the proposed BO/SO selection mechanism shows better performance than current state-of-the-art approaches. It is worth noticing that the energy efficiency results show the problem that IBOA presents in managing the coordinator duty-cycle. Because in this scheme the end devices can select different beacon intervals, a minimal beacon interval had to be set in the coordinator as the reference time period. This means that compared to the other approaches, the coordinator consumes much more energy as it cannot sleep even the traffic in the network is reduced being thus less energy efficient.

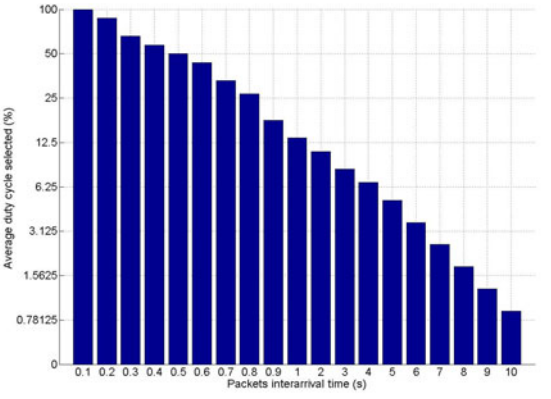

Fig. 5. DCLA average duty-cycle selection

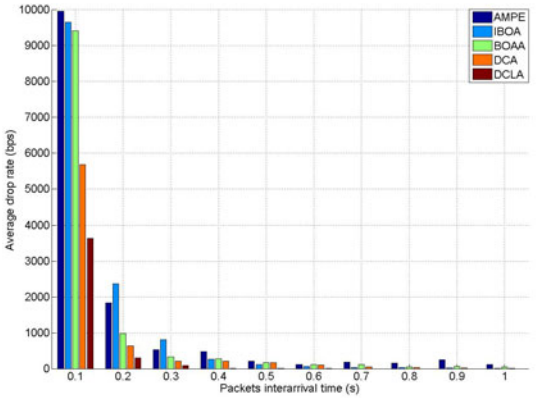

Fig. 6. Average drop rate measured in bps

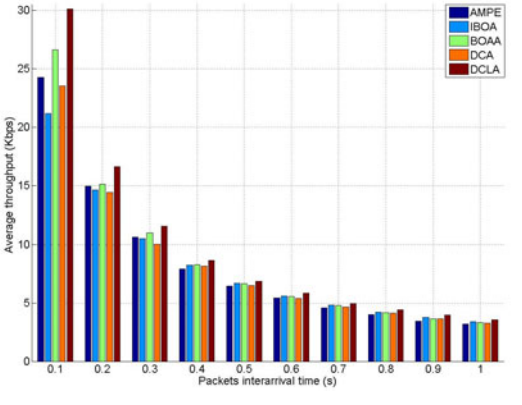

Fig. 7. Average throughput measured in Kbps

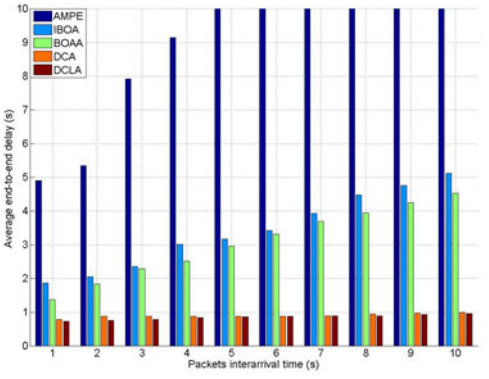

Fig. 8. Average end-to-end delay measured in seconds

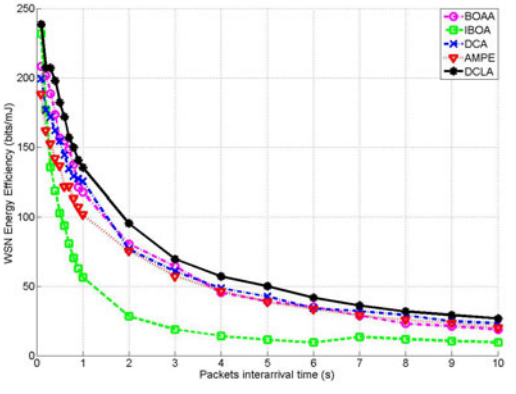

Fig. 9. WSN energy efficiency measured in correctly delivered bits per mJ

6 Conclusion and Future Work

This paper proposes a duty-cycle learning algorithm (DCLA) to enhance the overall network performance of IEEE802.15.4 beacon-enabled networks. We employ a reinforcement learning algorithm to solve the problem of adapting the coordinator's duty-cycle according to end devices' traffic conditions with the objective of minimizing the energy consumption while balancing at the same time other important WSN performance parameters. Once the RL agent has reached the duty-cycle decision, beacon order and superframe order are selected trying to find a compromise among beacon overhead and queuing delay. Simulation results show that DCLA outperforms current state of the art on duty-cycle adaptations for IEEE 802.15.4 beacon-enabled networks in terms of average drop rate, throughput, energy efficiency and end-to-end delay.

As a future direction of this work other scenarios with more rapid traffic fluctuations may be studied. In these situations and with the purpose of accelerating the RL adaptation process, it will be explored how the choice of different initial Q-values,

progress estimators or reward shaping could be used. We also think that an RL agent could be created at end devices to deal with collisions as this could reduce even more the energy expenditure. The new agent could control the minimum backoff exponent of the CSMA-CA according to the number of failed transmissions and the duty cycle chosen by DCLA. Finally, we also plan to expand this work to IEEE 802.15.4 multi-hop topologies by employing our solution on distributed beacon scheduling [15] with DCLA.

References

1. IEEE 802.15.4 Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low-Rate Wireless Personal Area Networks, LR-WPANs (2007)
2. Neugebauer, M., Plonnigs, J., Kabitzsch, K.: A new beacon order adaptation algorithm for IEEE 802.15.4 networks. In: Proceedings of the Second European Workshop on Wireless Sensor Networks, pp. 302–311 (2005)
3. Gao, B., He, C.: An individual beacon order adaptation algorithm for IEEE 802.15.4 networks. In: Proceedings of the 11th IEEE Singapore International Conference on Communication Systems, pp. 12–16 (November 2008)
4. Barbieri, A., Chiti, F., Fantacci, R.: Proposal of an Adaptive MAC Protocol for Efficient IEEE 802.15.4 Low Power Communications. In: Proceedings of the IEEE Global Telecommunications Conference, pp. 1–5 (December 2006)
5. Jeon, J., Lee, J.W., Ha, J.Y., Kwon, W.H.: DCA: Duty-Cycle Adaptation Algorithm for IEEE 802.15.4 Beacon-Enabled Networks. In: Proceedings of the 65th IEEE Vehicular Technology Conference, pp: 110–113 (April 2007)
6. Di, M., Joo, E.M.: A survey of machine learning in wireless sensor networks. In: Proceedings of the 6th International Conference on Information, Communications and Signal Processing (ICICS), pp: 1–5 (2007)
7. Liu, Z., Elahanany, I.: RL-MAC: A reinforcement learning based MAC protocol for wireless sensor networks. International Journal on Sensor Networks, 117–124 (2006)
8. Ye, W., Heidemann, J., Estrin, D.: Medium access control with coordinated adaptive sleeping for wireless sensor networks. IEEE/ACM Transactions on Networks 12(3), 493–506
9. Polastre, J., Hill, J., Culler, D.: Versatile low power media access for wireless sensor networks. In: Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems (SenSys), Baltimore, MD (November 2004)
10. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
11. Jurčík, P., Koubâa, A.: The IEEE 802.15.4 OPNET simulation model: reference guide v2.0. IPP-HURRAY Technical Report, HURRAY-TR-070509 (May 2007)
12. OPNET Modeler, OPNET Technologies Inc. Version 15.0., `http://www.opnet.com`
13. CC2420 radiochip datasheet: 2.4Ghz IEEE 802.15.4 Zigbee Ready RF Transceiver. Texas Instruments, `http://www.ti.com`
14. Low Power 2.4Ghz MicaZ mote for Wireless Sensor Networks, `http://xbow.com`
15. Muthukumaran, P.S., de Paz, R., Špinar, R., Pesch, D.: MeshMAC: Enabling Mesh Networking over IEEE802.15.4 through distributed beacon scheduling. In: Zheng, J., et al. (eds.) ADHOCNETS 2009. LNICST, vol. 28, pp. 561–575 (2010)