# Rapid Learning Optimization Approach for Battery Recovery-Aware Embedded System Communications

Mohammed Assaouy[1], Ouadoudi Zytoune[2] and Mohamed Ouadou[1]

[1]LRIT - CNRST URAC 29, Rabat IT Center, Mohammed V University in Rabat, Morocco
[2]System Engineering lab. (LGS), Ibn Tofail University in Kénitra, Morocco

***Abstract*:** To date, battery optimization for embedded systems still a crucial subject. Actually, the majority of carried out works focus on transmission controls without taking into account the specifications of the batteries themselves. Indeed, an improvement of 70% is reported by exploiting the battery recovery effect. In this paper, the recovery phenomenon is exploited to design an algorithm that optimizes both the lifetime of the battery and the performance of the studied system. The algorithms from Dynamic programming and Reinforcement learning fields are the first to be considered. When in Dynamic programming prior detailed information are assumed to be available, in reinforcement learning the information becomes unknown and long calculation times are needed to converge toward an optimal policy solution. The paper contribution is about designing a new Rapid Learning Algorithm (RLA) that combines both Dynamic programming and Reinforcement learning features. RLA exploits a reduced model of the system instead of exploring the whole and heavy system state model as Dynamic programming do. The RLA run-time is then shortened. Based on battery stochastic model, the simulation results obtained with RLA are compared to the Dynamic programming and Reinforcement learning algorithms under the same conditions. By taking into account the recovery effect this paper illustrates that the calculation time and the system performance are greatly improved when RLA is adopted.

***Keywords*:** Embedded system communications, Battery modeling, Battery recovery effect, Dynamic programming, Reinforcement learning.

## 1. Introduction

Internet of Things (IoT) has covered an important area of human being daily lives. All recent devices are to be able to communicate with each other by exchanging data of different using wireless communication capabilities. In fact, IoT offers a wide range of tools and technologies to extend connectivity to all devices and machines [1]. The efficiency of consumption has never reached a level of interest as high as today. Actually, many research groups focus their works on the need to reduce the energy consumption of battery-powered devices, such as sensors, phones, tablets, etc [2], [3]. In the field of embedded system communications such in wireless sensor networks (WSN), the problem is of high important level as the sensors are mostly distributed over large areas without the possibility of local access, especially when these areas are prone to danger, as in the case of the proximity of a volcano, war zones or in case of implemented sensors in the human body. Batteries are and remain the primary means to provide energy to the components of the wireless sensor networks. Optimal use of their energy is therefore the main goal that this paper is focusing on. Previous works have addressed this issue by providing protocols that operate at the MAC layer level to improve the consumption behavior of wireless sensors as in [4], [5] following a software vision. Routing mechanisms have also been approached to deal with energy consumption problems such in [6] At the hardware vision side, some works have considered solutions for the physical layer level as in [7] presenting new signal processing approaches. Batteries with harvesting capabilities of their surrounding energies have also been proposed to improve the performance of wireless sensors, as shown in [8]–[10]. Unfortunately, their impact is further mitigated by the limited and low energy quantities gathered by the harvesters. The battery recovery effect has been yet exploited in numerous previous works where scheduling algorithms as in [11], [12] were proposed to increase the lifetime of the WSN. Authors in [11] stated that, as consequences of the capacity recovery effect, up to 20% of the total cell capacity becomes available again with some rest time. Similar statements are proposed by [12] in the area of wireless body sensors, where an improvement of 70% is reported by exploiting the battery recovery effect.

Actually, RLA allows an optimal use of the available battery capacities. It considers a reduced model based on a partial known system state. The recovery effect as defined in [7] represents the main tool exploited by the RLA to achieve a high-performance level. This effect has been previously considered in similar works especially in the field of wireless communication with battery-powered devices [13]–[15].

This paper is based on the wireless sensor systems described in Figure 1.

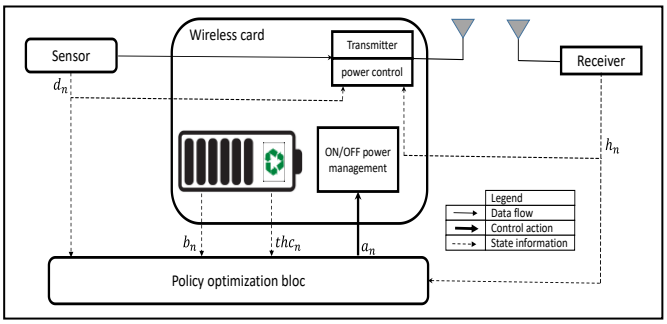


**Figure 1.** The recovery-aware battery powered study system

When the information is gathered by the sensor, the ON/OFF power manager decides according the policy being followed whether to switch on or off the wireless card. On position ON, the transmission occurs with a power level as fixed by the power control block. An optimizer block is in charge of making the right decision depending on the data packet size, the battery level, the available charges and the channel state. The operation mode is assumed to follow a time slotted way of duration $T_S$.

Three cases are then considered depending on the information availability. They are described **as follows:**

- In the first case, we assume that the optimizer has prior stochastic knowledge of data packet size and channel state transitions. Value Iteration from Dynamic programming field is appropriated for this situation [16] ;
- In the second case, the optimizer is assumed to have no prior information about system state transitions. Therefore, the optimizer performs some training episodes to know the system states before calculating the optimal strategy to incorporate using reinforcement learning algorithms such as Q-learning [15], [16];

- In the context of RLA, the optimizer can, therefore, exploit some partial information provided by system states and learn only unknown parts.

## 2. Battery model and recovery effect description

Batteries, especially chemical, are mainly described by two important parameters, namely the theoretical and nominal capacities [17]. These parameters are defined as follows:

- The theoretical capacity of the battery, denoted $T_B$, is the initial total number of charge units available in the body of the battery, mainly at the level of the electrolyte and the electrodes before the first use ;
- The nominal capacity, denoted $B_{max}$, represents the effective amount of charges that can be supplied by the battery under a specific constant current until the battery reaches the cut-off voltage. When this cut-off value is reached, the battery runs out completely and can be declared out of order and then must be replaced.

The parameters $T_B$ and $B_{max}$ are both sensitive to current battery materials and discharge conditions.

The recovery effect is often defined as the selfreplenishment of the battery by active materials causing the voltage partially increasing when the current is interrupted.

Therefore, the intermittent discharge profile increases the use of the capacitance inside the battery, because the cells can feed themselves by recovering charge units from the electrolyte during periods of inactivity [17].

When considering pulsed current with periods of inactivity, as described in [18], a charge recovery phenomenon occurs in the battery volume during periods of inactivity, e.g. in a duty cycle operating mode.

These periods of inactivity (idle periods) are very important for the battery, they allow it to drain new charge units from the electrolyte to replace those sent to the outside. The new charges comes from the electrolyte solution according to a diffusion process given that the gradient of the active charges always decreases from the electrode deep inside the electrolyte. New charge units become then available for future transmission energy consumption cycles [17].

To take profit of the recovery effect we proceed by modeling the battery. Actually, many battery models are proposed in literature. We can find models such as electrochemical, Electrical circuit, Analytical and stochastic ones [19].

For our proper need, the stochastic model for the battery state behavior as described in [20] fit the best with our purpose. In our study, only the recovery effect will be taken into account, we will then neglect all other phenomena occurring inside the battery such as passivation as in [20]. According to the stochastic model, the battery capacity behavior will be represented by a Markovian process starting from a fully charged battery state and stopping when either all the theoretical capacity is consumed or the battery charge state felt into zero. This latter battery state is considered to be a trapping state where the battery becomes out of use even when quantities of charge units still available in the electrolyte.

It is well known, in the WSN communication field, that the transmission mechanism is the biggest source of energy consumption. According to this statement, we assume that the whole energy consumption is due to the data transmission process and will neglect the consumption of all other sensor radio positions such as receiving, idle listening or sleeping modes.

The battery model is represented by a set of possible states from 0 to the nominal capacity Bmax. At the beginning, the battery is fully charged and it state is Bmax. The shifting process between the battery states is done according to the decisions made by the optimizer. Data packet transmission occur at stochastic instant according to a Bernoulli process with a probability of $a_1 = q$ during the time slot. The data packet can also not be transmitted and then dropped, the recovery effect occurs with a probability of $a_0 = 1 - q$ since there is no discharge. As stated before, a battery state of zero is a trapping state. Such trapping state is a terminal one by which the battery is declared dead. The nominal capacity $B_{max}$ is an upper battery state bound and can be reached many times as the recovery charge process is going on. We can summarize all this in a graphical model of the battery as in Figure 2.

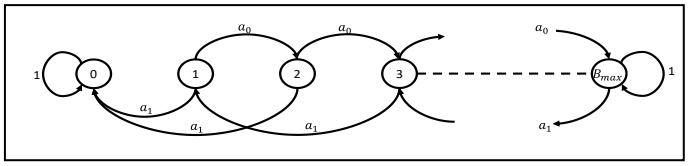


**Figure 2.** Battery operating model with Bernoulli process probability

In that graphic, each circle number represents the battery charge units available at the electrode. In this example, transmitting consumes one charge and dropping data allows the battery to recover one unit whenever the terminal state is not reached. Once the battery state fall into zero state it will never quit it any more like in a trap case.

## 3. System model presentation

The battery considered in our study has limited capacity and is not subject to any external feeding. We aim to exploit the whole theoretical capacity initially available inside the battery bulk. By the way we look forward to maximize the total amount of data packet supposed to be successfully transmitted to the receiver with a discounted time factor.

We consider that data is supposed to be gathered and ready for transmission at the beginning of each time slot $T_S$ with different sizes. This data is either transmitted or dropped during the next time slot.Data arriving process is stochastic and can be represented by mean of a first order Markov model as in [15]. The data packet size, denoted dn, belong to a set of $N_D$ possible data sizes $D = \{d_1, d_2, \dots, d_{N_D}\}$. The probability transition from $d_n$ in $T_{S_n}$ to $d_{n+1}$in time slot $T_{S_{n+1}}$ is equal to $p_d(d_n, d_{n+1})$.

The channel state is considered to be constant during a time slot interval $T_S$. It can change only from one $T_S$ to the next. The channel state transitions is supposed to occur following a Markovian process. The channel states are in the set $H = \{h_1, h_2, \dots, h_{N_H}\}$, where $N_H$ is the total number of possible channel states. The transition from one channel state to another occurs with a probability $p_h(h_n, h_{n+1})$.

Battery charge state bn and consumed theoretical capacity $thc_n$ are supposed to be known at the beginning of each time slot. They change only after performing a transmission or a charge recovery when trapping state has not been reached

yet. The battery charge state are in the set $B = \{0,1,2,3, \dots, B_{max}\}$.

When the wireless card is switched on, the transmitter consumes energy from the battery to perform the transmission of the data packet towards the receiver. The consumed energy is denoted $E_n^{T_S}$ when data size is $d_n$ and the channel state is $h_n$. We consider $e$ as the smallest battery energy which represents one charge unit.

A successful data packet transmission depends on the quantity of energy provided by the battery to the transmitter and weather it meets their needs based on the data sizes and the channel condition's state. As expected, the more data are important in terms of sizes and the channel attenuation is high the more important is the energy to be extracted from the battery.

As stated earlier, battery charge levels, data packet sizes and channel states are considered to be known at the beginning of each time slot. The information about the channel states is assumed to be shared between the optimizer, the transmitter and the receiver.

For decision making purposes, the optimizer exploit all the system state components such as data sizes, channel states, battery charge and consumed theoretical capacity parts. Decisions made become the policy to implement inside the wireless sensor to keep its choices oriented towards maximizing charge units use and the quantity of data transmitted to the receiver under multiple constraints such as trapping state and nominal capacity limitation. The actions made by the optimizer are either turning on the wireless card or switching it off. In the first case $a_n$ is set to 1 to select the wireless card for transmission mode or $a_n$ is set to 0 otherwise. Those actions are picked from a set of actions $A = \{0,1\}$ with only two possible values.

The optimization problem will be conducted under some constraints that are summarized in the equations below (1), (2) and (3):

$$a_n E_n^{T_S} \leq b_n \quad , \qquad (1)$$

$$b_n \leq B_{max} \quad , \qquad (2)$$

$$\sum_{n=1}^{N_L} a_n E_n^{T_S} \leq T_B \quad , \qquad (3)$$

Where $N_L$ denotes the number of transitions performed by the considered system. $N_L$ depends firmly on the discounted factor value. The battery charge state $b_n$ and the consumed theoretical capacity $thc_n$ are updated after each transition according to Equations (4) and (5) respectively.

$$b_{n+1} = \begin{cases} b_n - E_n^{T_S} & if\ a_n = 1 \\ \min(b_n + e, B_{max}, T_B - thc_n) & if\ a_n = 0 \end{cases} \qquad (4)$$

$$thc_{n+1} = \begin{cases} \max(thc_n + E_n^{T_S}, T_B & if\ a_n = 1 \\ thc_n & if\ a_n = 0 \end{cases} \qquad (5)$$

The optimization objective, aiming to maximize the cumulative transmitted data packets over one episode, is described in Equation (6).

$$\max_{a_{1:N_L}} \sum_{n=1}^{N_L} \gamma^{n-1} a_n d_n \qquad subject\ to\ \ (1), (2), (3)\ and\ \ (4)\ , \qquad (6)$$

Where $0 < \gamma < 1$ is a discounted factor.

The system under study is operating in a discrete-time fashion. It is described by a finite sized state vector $S_n = \{b_n, thc_n, h_n, d_n\}$. The system is well modeled according a Markovian Decision Problem (MDP) since all the elements of $S_n$ are Markovians. We assume that the system is totally observable and either the next states and the rewards can be known by the optimizer who is the main decision maker agent. The set of system states is denoted $S = \{S_1, S_2, \dots, S_{N_S}\}$ where $N_S$ is the total number of all possible system states.

Policy $\pi$ denotes the decisions to be made by the sensor either to switch on or off the wireless card button. The reward value is obtained after a pair action-state having been performed. This reward is represented by a single number equal to the size of the packet data that has been successfully transmitted to the receiver. Actually, we can note that in each transition, $R_n = a_n \times d_n$. The total obtained reward will be then obtained by the accumulation of all rewards and undated by mean of the considered discounted factor $\gamma$.

The followed policy _ is evaluated for each state $S_n$ using the state-value function denoted by $V^{\pi}(S_n)$ as given in Equation (7).

$$V^{\pi}(S_n) \triangleq \sum_{\forall S_k \in S} p(S_n, \pi(S_n), S_k)[R(S_n, \pi(S_n), S_k) + \gamma V^{\pi}(S_k)] \qquad (7)$$

This function gives the sum of all the rewards obtained when starting by state $S_n$ and after follows policy $\pi$.

Another function can be defined the same way than state value function, we call it action-state value function and will be denoted $Q^{\pi}(S_n, a_n)$. This function evaluates the pair action-state and gives the cumulative amount of rewards obtained when starting by state $S_n$ and taking first an action $a_n$ before following the policy $\pi$ after. It is defined by Equation (8).

$$Q^{\pi}(S_n, a_n) \triangleq \sum_{\forall S_k \in S} p(S_n, a_n, S_k)[R(S_n, \pi(S_n), S_k) + \gamma V^{\pi}(S_k)] \qquad (8)$$

The optimization process proceed to a continuous improvement of the policy $\pi$. A policy $\pi'$ is better than $\pi$ when its state-value function is the higher one, which can be expressed as $V^{\pi'}(S_n) \geq V^{\pi}(S_n)$. Optimal policy $\pi^*$ is the best one or at least equal to any other policy in terms of state-values. The state value function can also be derived from the action-state value function by using Equation (9).

$$V^{\pi^*}(S_n) = \max_{a \in A} Q^{\pi^*}(S_n, a) \qquad (9)$$

The optimal policy is then obtained as given in Equation (10).

$$\pi^*(S_n) = \underset{a \in A}{\operatorname{argmax}}\, Q^{\pi^*}(S_n, a) \qquad (10)$$

## 4. Partially known-based model and the new Rapid Learning Algorithm

### *4.1 Partially known-based model optimization problem*

On the one hand, the conventional Q-learning method operates by estimating the value of the action state value vector during the learning phase. This is done considering an environment with dynamics completely unknown. Q-learning takes a lot of time because it must perform many episodes to have sufficient knowledge of the behavior of the system [21], [22]. On the other hand, Value Iteration requires full knowledge of the system which is quite impossible in real cases [16]. In several environments, some dynamics of the system may be partially known.

In Table I we list different types of dynamics of the system that can be established in a deterministic or stochastic way in advance or still completely unknown.

**Table 1.** Classification of the system state components

| Description | Known | Unknown |
|---|---|---|
| **Deterministic** | Battery charge state $b_n$ | N/A |
| | Consumed theoretical charge capacity $thc_n$ | N/A |
| **Completely unknown** | N/A | Data packet size $d_n$ |
| | N/A | Channel state $h_n$ |

Actually, as soon as the decision to transmit a data packet is made, the power control unit set the number of units to be consumed. Once the transmission performed, the next battery state and the consumed theoretical capacity can then easily be deduced according to Equation (4).

### *4.2 Partially known optimization problem: definition*

We define an intermediate state $\tilde{S}$ to describe the state of the system after an action has been taken based on the known parts of the system. The system state is, as soon as the unknown parts are communicated to the wireless card, updated accordingly. This mechanism has already been adopted in previous publications, see [16], [23], [24]. Based on optimization subareas of system states, the relationship established at anytime between the intermediate state $\tilde{S}_n$, the system state $S_n = \{b_n, thc_n, h_n, d_n\}$, the $a_n \in \{0,1\}$ and the next system state $S_{n+1}$ is given as follows:

- The intermediate system state is $\tilde{S}_n = (\tilde{b}_n, \widetilde{thc}_n, h_n, d_n) = (b_n - c_n, thc_n + c_n, d_n, h_n)$ ;
- The next system state $S_n = (b_{n+1}, thc_{n+1}, d_{n+1}, h_{n+1}) = (\tilde{b}_n, \widetilde{thc}_n, h_{n+1}, d_{n+1})$.

Where $c_n$ is the number of charge units having been used for the transmission in case where the wireless card was turned on (i.e. $a_n = 1$). Otherwise, $c_n = 0$.

Actually, the intermediate state is a fictitious state and the same value for the packet size of the initial state are maintained, even though this packet have been transmitted or dropped. The intermediate state considers only the known part of the studied system when transition from system state $S_n$ to the state $S_{n+1}$ occurs after taking the decision $a_n$. The unknown components of the system are not incorporated in the intermediate state (i.e. the following packet size and channel state values). Depending on the consumption of charge units in case of packet transmission, updating the battery levels and the consumed theoretical capacity is operated accordingly and the new packet size and channel state are integrated into the intermediate state to obtain the state $S_{n+1}$. By introducing this notion of intermediate state, we can split the transition probability function into two separated parts a known and unknown ones as in [16]. The known part informs about the transition from the current state to the intermediate state, i.e. $S_n \to \tilde{S}_n$, while the unknown part governs the transition from the intermediate state to the next state $\tilde{S}_n \to S_{n+1}$. In terms of probabilities, it can be established that:

$$p(S_n, a_n, S_{n+1}) = \sum_{\tilde{S}} p_u(\tilde{S}_n, a_n, S_{n+1}) p_k(S_n, a_n, \tilde{S}_n) \quad (11)$$

Where the index k and u respectively denote the the known and unknown parts of the transition probability functions. In this case, the reward is generated during the transition from the current state to the intermediate state according to the value of the action taken from the policy followed by the wireless sensor. The phase corresponding to the transition from intermediate to the next state can also generate an additional reward. We can then establish that:

$$R(S_n, a_n, S_{n+1}) = R_u(\tilde{S}_n, a_n, S_{n+1}) + R_k(S_n, a_n, \tilde{S}_n) \quad (12)$$

The unknown parts of the next system state do not depends on the action having been taken and do not contribute to the production of the reward. So we can write that:

$$p_u(\tilde{S}_n, a_n, S_{n+1}) = p_u(\tilde{S}_n, S_{n+1}) \quad (13)$$

$$R_u(\tilde{S}_n, a_n, S_{n+1}) = R_u(\tilde{S}_n, S_{n+1}) = 0 \quad (14)$$

However, this approach can easily be extended to cases where the unknown part depends on both the intermediate state and the action undertaken in the first phase. In this case an exploration of all possible actions will be necessary to determine the optimal policy.

The known and unknown transition probability functions can therefore be expressed as follows:

$$p_k(S_n, a_n, \tilde{S}_n) = p_b(b_n, a_n, \tilde{b}_n) p_t(thc_n, a_n, \widetilde{thc}_n) I(\tilde{h}_n = h_n) I(\tilde{d}_n = d_n) \quad (15)$$

$$p_u(\tilde{S}_n, S_{n+1}) = p_d(\tilde{d}_n, d_{n+1}) p_h(\tilde{h}_n, h_{n+1}) I(b_{n+1} = \tilde{b}_n) I(thc_{n+1} = \widetilde{thc}_n) \quad (16)$$

Where the operator I(.) is the indicator function that takes the value 1 if the argument in parentheses is true and the value 0 otherwise. The reward generated by the known part at each iteration is given as follows:

$$R_k(S_n, a_n) = a_n \times d_n \quad (17)$$

### *4.3 Dynamic programming based concept for Rapid Learning Algorithm*

Before proceeding with the description of the learning algorithm, we first define the state value function $V(S_n)$. This function will play the similar role as the action-state value function for the Q-learning algorithm. The optimal state value function, denoted $V^*_{S_r}$, is computed for each fixed pair (packet size $d_n$, channel state $h_n$) where $S_r$ is a reduced set of system states $S_r$. We have to go through all the possible combinations for these two elements in order to finally find $V^*$ for all possible system states. $V^*_{S_r}$ is computed according to the equation below:

$$V^*_{S_r}(S_n) = \max_{a_i} \left\{ R_k(S_n, a_i) + \gamma \sum_{S_{red} \in S_r} p_k(S_n, a_i, S_{red}) V_{S_r}(S_{red}) \right\} \quad (18)$$

Where $S_{red}$ is the possible future state in the reduced set of system states $S_r$. $S_r$ is based on the pair (packet size, channel state) as given in the current system state $S_n$. Having the optimal value $V^*_{S_r}$, the optimal policy is directly obtained by calculating:

$$\pi^*_{S_r}(S_n) = \underset{a_i}{\operatorname{argmax}} \left\{ R_k(S_n, a_i) + \gamma \sum_{S_{red} \in S_r} p_k(S_n, a_i, S_{red}) V_{S_r}(S_{red}) \right\} \quad (19)$$

The following proposition proves that $\pi^*_{S_r}(S_n)$, as defined in Equation (19), and $\pi^*(S_n)$, as defined in Equation (10) are equivalent.

**Proposition 1**: $\pi^*_{S_r}(S_n)$ and $\pi^*(S_n)$ are equivalent.

**Proof**: To show that $\pi^*_{S_r}(S_n)$ and $\pi^*(S_n)$ are equivalent, we split the total set of system states S on subsets $S_{r_i}$ with $i \in \{1,2,\dots,N_D \times N_H\}$. If $N_D = 2$ and $N_H = 2$, then $i \in \{1,2,3,4\}$. The equivalence over $S_{r_i}$ for i=1 is given in Equation (20).

$$\pi^*_{S_{r_1}}(S_n) = \underset{a_i}{\text{argmax}} \left\{R_k(S_n,a_i) + \gamma \sum_{S_{red}\in S_{r_1}} p_k(S_n,a_i,S_{red})V_{S_{r_1}}(S_{red})\right\}$$
$$= \underset{a_i}{\text{argmax}} \left\{R_k(S_n,a_i) + \gamma \left(\sum_{S_{red}\in S_{r_1}} p_k(S_n,a_i,S_{red})V_{S_{r_1}}(S_{red}) + \sum_{S_{red}\in S_{r_2}} p_k(S_n,a_i,S_{red})V_{S_{r_2}}(S_{red}) + \sum_{S_{red}\in S_{r_3}} p_k(S_n,a_i,S_{red})V_{S_{r_3}}(S_{red}) + \sum_{S_{red}\in S_{r_4}} p_k(S_n,a_i,S_{red})V_{S_{r_4}}(S_{red})\right)\right\}$$
$$= \underset{a_i}{\text{argmax}} \{R_k(S_n,a_i) + \gamma(\sum_{S'\in S} p_k(S_n,a_i,S')V(S')\} = \pi^*(S_n) \quad (20)$$

Similarly for the cases of $\pi^*_{S_{r_2}}(S_n)$, $\pi^*_{S_{r_3}}(S_n)$ and $\pi^*_{S_{r_4}}(S_n)$ which can all be proved equivalent to $\pi^*(S_n)$. In other words, $\pi^*(S_n)$ is the concatenation of the set of $\pi^*_{S_{r_i}}(S_n)$ for all $i \in \{1,2,3,4\}$, since each one covers a reduced set of S.

**Proposition 1** is very important because it allows us to use the $\pi^*_{S_{r_i}}(S_n)$ to learn the optimal policy $\pi^*(S_n)$.

### *4.4 The proposed Rapid Learning Algorithm*

While Q-learning algorithm learns the action-state-value function to approach its optimal value $Q^*(S_n)$, the proposed method focus on state-value function and directly calculates its optimal value $V^*(S_n)$. However, even if the elements of the optimal vector $V^*$ are computed step by step while fixing the known parts of the system, the whole process converges quickly to a robust estimate of the global optimal policy __. The learning process includes exploring unknown parts of the system. Actually, for each combination of data packet size and channel status identified, a dynamic optimization is conducted to update the system state value vector $V(S_n)$. Schematically, the proposed method explores the episodes of learning phases as in Figure 3.

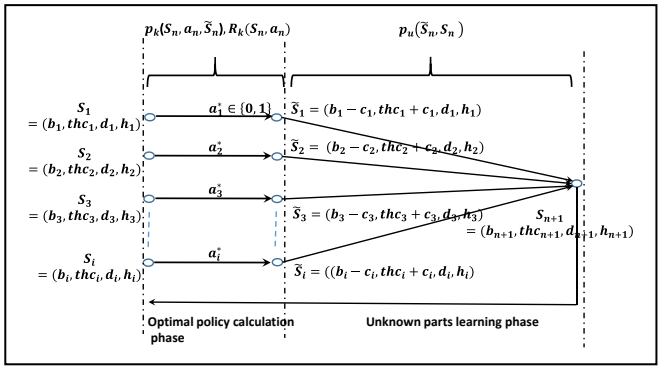


**Figure 3.** The proposed Rapid learning process description

In Value iteration case, the calculation complexity based on the entire system state S were bounded by $0\left(\frac{2^{N_S}}{N_S}\right)$ [25]. However, in the case of RLA, the new system model has become a concatenation of a smaller $S_r$ set that contributes to the reduction of complexity. Since the algorithm concept in RLA is similar to the Value iteration one, the complexity can be deduced directly as $0\left(\frac{2^{N_{S_r}}}{N_{S_r}}\right)$. Since the ratio of the system state dimensions is equal to $\frac{N_S}{N_{S_r}}$, the complexity is then reduced by a factor of $\frac{2^{N_D N_H}}{N_D N_H}$.

The Rapid Learning Algorithm is given in Algorithm (1).

**Algorithm 1** Proposed Rapid-learning algorithm

1. ***Initialization step:***
$n = 0$, read the initial system state $S_n = (b_0, thc_0, h_0, d_0)$
Initialize $V = zeros(N_S, 1)$ and $\pi = zeros(N_S, 1)$ set $i = 0$
and set $W = \Phi$ and $c = (h_n, d_n)$
2. ***Learning step:***
**if** $c \notin W$ **then**
Set the sub system states $S_r = \{B \times T, c\}$
$\epsilon \leftarrow 0.001$
**while** $\Delta > \epsilon$ **do**
$\Delta \leftarrow 0$
$temp \leftarrow V$
**for** each state $S_m \in S_r$ **do**
$V(S_m) \leftarrow \max_{a\in A} \sum_{\forall S_j \in S_r} p(S_m,a,S_j)[R_k(S_m,a,S_j) + \gamma V(S_j)]$
$\pi(S_m) \leftarrow \underset{a\in A}{\text{argmax}} \sum_{\forall S_k \in S_r} p(S_m,a,S_j)[R_k(S_m,a,S_j) + \gamma V(S_j)]$
**end for**
$\Delta \leftarrow max(\Delta, \| V(S_m) - temp \|_\infty)$
**end while**
**end if**
Apply action $a_n = \pi(S_n)$
Observe the system states experience results $S_n \rightarrow a_n \rightarrow R_k \rightarrow b_{n+1}, thc_{n+1}$
$c \rightarrow W$
read the next data packet size $d_{n+1}$ and channel state $h_{n+1}$
set $n \leftarrow n + 1$,
set $c = (h_n, d_n)$ go to 2
End of the episode

## 5. Simulation results

In this section, we define all the features of the considered battery recovery-aware embedded system. For comparison purposes by simulation, we consider a greedy algorithm where the wireless card is always switched on and the transmitter was sending continually the data packets it receives, which means that the decision being permanently made in that case is $a_n = 1$ $(\forall\, n = 1{:}\infty)$ all the time.

The performances obtained from the Value iteration, Rapid Learning Approach, Q-learning and Greedy algorithms are compared both in terms of the amount of data transmitted and the consumed theoretical capacity of the battery.

We consider episodes with duration's length of 100xTs, given that a discounted factor of 0,9 concentrates more than 99% of the whole significant amount of transmitted data in the interval 1 to 100.

2000 episodes are generated to determine an average of the results. These episodes focus primarily on varying the size of data packets and channel states that depend on their proper transition probabilities.

We set the nominal capacity $B_{max}$ = 5. A sequence of $N_L$ = 200000 episodes will be considered for Q-learning and Rapid Learning Algorithms to let them achieve sufficient knowledge of the system.

As additional values of other elements of computation we use parameters that are based on IEEE802.15.4e [26] for the time slot duration which will be fixed at $T_S$ = 10 ms, the transmission period will be set to $T_x$ = 5 ms. We assume that the data packet sizes are in the set $D = \{300,600\}$ with only two possible realizations. They may vary according to a probability transition matrix that is equal to $p_d = [v_1 \; v_2]$, where $v_1 = [0.9 \;\; 0.1]$ and $v_2 = [0.1 \;\; 0.9]$. The channel

state set is $H = \{1.655 \times 10^{-13}, 3.311 \times 10^{-13}\}$ which may account for an indoor channel model in urban scenarios cases as in [27] with dist = dindoor = 55, wall = 3, WPin = 5, and 5 dBm standard deviation, where dist is the distance in meters, wall the number of walls, and WPin the wall penetration losses. The transition state probability of the channel is given by the matrix $p_h = [v_1 \ v_2]$. One elementary charge units *e* are required for a successful transmission of a data packet of size $d_n$ = 300 bits over a channel with state $h_n$ = 3.311 x $10^{-13}$ with a noise power density set to $N_0 = 10^{-20,4}$ (W=Hz). *e* will then be equal to $\frac{d_n \log(2) N_0}{h_n} = 2.5\ \mu J$.

The energy needed for successful transmission depends on the data packet sizes and channel states. Hence, it is a multiple integer of the energy unit *e* that belongs to the set $\{1,2,4\}$. This quantities correspond to an energy consumption of 2.5, 5 and 10μJ and are equivalent to power of 0.5, 1 and 2 mW respectively.

In Figures 4 and 5, for a theoretical capacity value of $T_B$ = 20 charge units and nominal capacity $B_{max}$ = 5, Rapid Learning Algorithm reaches 99.8% of the optimal expected total transmitted data and consumes up to 94.5% of the available theoretical capacity which is equal to 96.16% of the optimal consumed capacity by the 37th iteration. Q-learning needs to run over than 200000 iteration to reach only 87.7% of the optimal expected total transmitted data and to consume only 60.5% of the available theoretical capacity which is equal to 61.6% of the optimal consumed capacity.

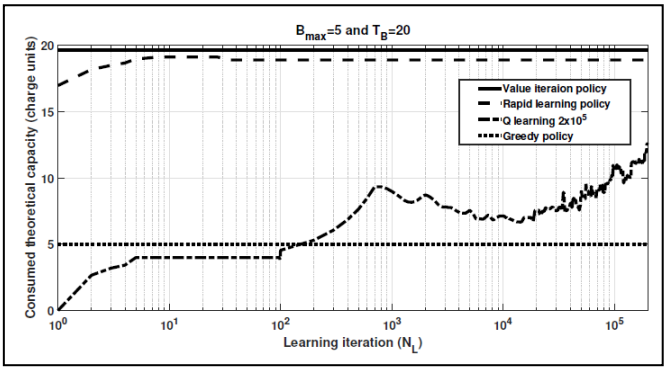


**Figure 4.** Expected total transmitted data vs $N_L$ with $T_B$ = 20 and $B_{max}$ = 5

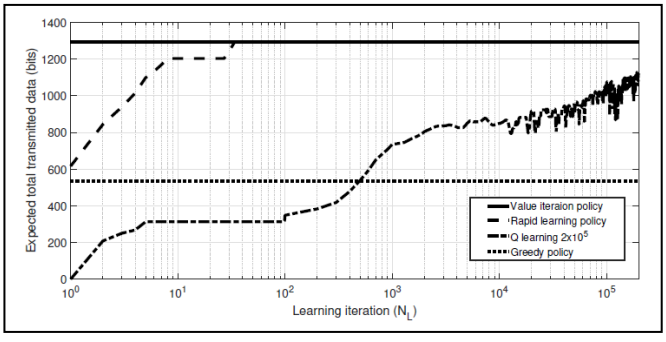


**Figure 5.** Consumed theoretical capacity vs $N_L$ with $T_B$ = 20 and $B_{max}$= 5

In Figure 6, we notice that the optimal policy generated by the Value iteration Algorithm sends an average amount of data of up to 1297.9 bits per episode of 100 iterations. The Rapid Learning Algorithm reaches an average amount of transmitted data of 1295.5 bits, while Q-learning achieves a transmission performance of 1138.9 transmitted bits. The Greedy algorithm arrives at the bottom position with the lowest performance by sending an average amount of data of only 535.6 transmitted bits.

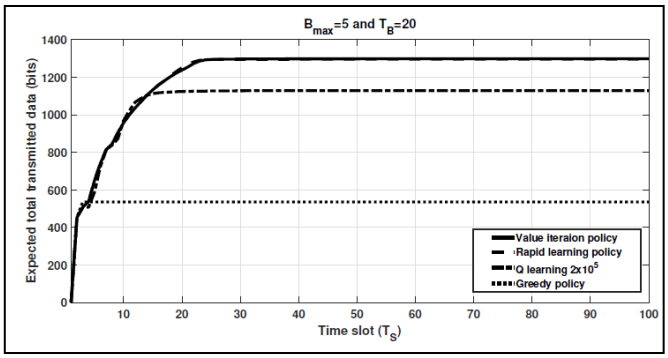


**Figure 6.** RLA comparison results for Expected total transmitted data vs time slots given $B_{max}$ = 5 and $T_B$ = 20

In Figure 7, the optimal Algorithm achieves an average consumption of 19.6 charge units from the 20 available theoretical capacity TB. The Rapid Learning Algorithm reaches an average amount of consumed charge units of 18.9, while Q-learning achieves a performance of 12.1 consumed charge units. The greedy algorithm arrives at the last position, achieving the worst performance when it consumes a quantity of units of charge of only 5 units leaving 15 units as unused capacity.

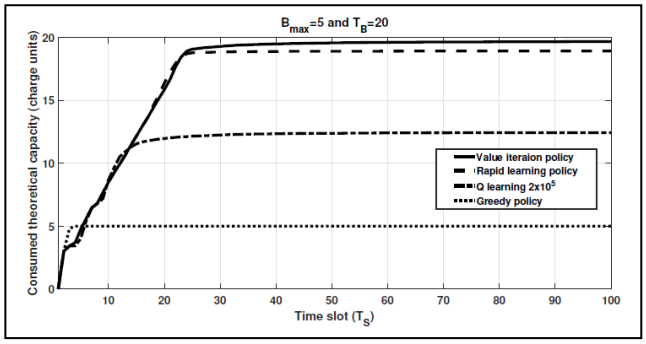


**Figure 7.** RLA comparison results for Consumed theoretical capacity vs time slots given $T_B$ = 20 and $B_{max}$ = 5

In Figure 8, optimal algorithm manages to avoid the battery trapping state as long as charge units are still available in the battery's electrolyte.

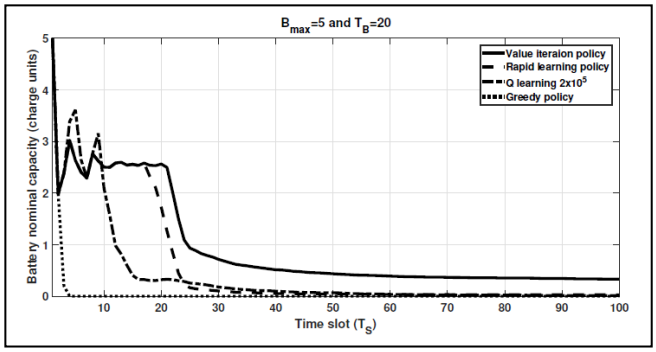


**Figure 8.** RLA comparison results for Nominal capacity vs time slots given $T_B$ = 20 and $B_{max}$ = 5

The Rapid learning algorithm takes the same challenge and tries to perform an optimal use of the capacity of the battery as well as the Q-learning algorithm. For those three algorithms the nominal capacity of the battery is kept greater than 0 for all the episode duration leaving an average quantity of 0.332 , 0.0275 and 0.059 charge units

respectively for the optimal, rapid and Q-learning algorithms by the end of the episode. For the greedy algorithm, the trapping state is quickly reached after an average number of 3 transitions only, making the battery unusable while many units of charge are still unused.

## 6. Conclusion

In our paper, we proposed a new Rapid learning algorithm, adapted to energy optimization for communications taking into account the battery recovery effect for embedded systems. The chosen simulation platform was based on a wireless sensor that collects information and sends it to a base station in peer to peer model. Three scenarios were considered in our study based on the availability of information. We first considered a complete stochastic availability of the information, which we processed using Value Iteration algorithm. In the second scenario, we assumed that no information was already known by the optimizer unit and we used the Q-learning algorithm to learn the studied system and find out the optimal policy. We noticed that, among the information available on the studied system, the Q-learning approach did not take into account the battery transitions and consumed charges levels that we can predict after each data packet transmission. This observation led us to propose the third type of scenario where we assumed that we have a partial knowledge of the studied system. We then introduced the Rapid Learning Algorithm that has operated faster than Q learning, as shown by the results of the simulation. In RLA, the underlying known parts of information were taken into account for optimal policy calculation. Good results were obtained from simulations, which can insure an optimized use of batteries when RLA is adopted. The investment costs on batteries was considerably reduced (e.g. the battery investment for the greedy algorithm costs 4 times more than in the case of Rapid Learning Algorithm), which produce a positive impact on the ROI (return on investment) ratio of the equipments. In addition, less batteries waste was produced when RLA is implemented, which better protects the environment and mitigates the impact of battery chemical waste on the planet.

## References

[1] L. Atzori, A. Iera and G. Morabito, “The internet of things: A survey”, Computer networks, 54(15), 2787-2805, 2010.

[2] A. Goldsmith, ”Wireless Communications”, 1st edition, Cambridge University Press, 2005.

[3] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, ”Instrumenting the world with wireless sensor networks”, Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2001), Salt Lake City, Utah, USA, vol. 4, pp. 2033-2036, May 2001.

[4] I. Bennis, O. Zytoune, D. Aboutajdine and H. Fouchal, ”Low energy geographical routing protocol for wireless multimedia sensor networks”. In Wireless Communications and Mobile Computing Conference (IWCMC), Sardinia, Italy, pp. 585-589, July 2013

[5] I. Dbibih, O. Zytoune and D. Aboutajdine, ”On/off markov model based energy-delay aware mac protocol for wireless sensor network”. Wireless personal communications, 78(2), pp. 1143-1155, 2014.

[6] S. El Khediri, R.U. Khan and W. Albattah, ”An optimal clustering algorithm-based distance aware routing protocol for wireless sensor networks”, International Journal of Communication Networks and Information Security, vol. 11, no 3, pp. 391-396, 2019.

[7] O. Zytoune, D. Aboutajdine, ”Energy usage analysis of digital modulations in wireless sensor networks with realistic battery model”. Wireless Networks, 22(8), pp. 2713-2725, 2016.

[8] M. K. Sharma, C. R. Murthy, ”On the Design of Dual Energy Harvesting Communication Links With Retransmission”. IEEE Transactions on Wireless Communications, v. 16 (6), pp. 4079-4093, June 2017.

[9] R. J. Vullers, R. Van Schaijk, H. J. Visser, J. Penders, C. Van Hoof, ”Energy harvesting for autonomous wireless sensor networks”, IEEE Solid-State Circuits Magazine, 2(2), pp. 29-38, 2010.

[10] M. Assaouy, O. Zytoune and D. Aboutajdine, ”DP and RL Approach Optimization for Embedded System Communications with Energy Harvesting”, I4CS 2017, CCIS 717, Darmstadt, Germany, pp. 167–182, 2017.

[11] C. Maurer, W. Commerell, A. Hintennach, A. Jossen, ”Capacity recovery effect in Litium Sulfure Batteries for electric vehicles”, World Electric Vehicle Journal, pp. 9-34, 2018.

[12] Y. Chenfu, W. Lili, L. Ye, ”Energy Efficient Transmission Approach for WBAN Based on Threshold Distance”, IEEE Sensors Journal 15, pp. 5133–5141, 2015.

[13] C.K. Chau, F. Qin, S. Sayed, M.H. Wahab, Y. Yang, ”Harnessing battery recovery effect in wireless sensor networks: experiments and analysis,” IEEE J. Sel. A. Commun., vol. 28, no. 7, pp. 1222-1232, 2010.

[14] L. He, G. Meng, Y. Gu, C. Liu, J. Sun, T. Zhu & K. G. Shin, ”Battery-Aware Mobile Data Service”, Mobile Computing IEEE Transactions on, vol. 16, pp. 1544-1558, 2016.

[15] M. Assaouy, O. Zytoune and M. Ouadou, ”Battery Recovery-Aware Optimization for Embedded System Communications”. Wireless Personal Communications, vol. 110, no 4, pp. 1929-1946, 2020.

[16] R. S. Sutton, A. G. Barto, ”Reinforcement Learning: An Introduction”, A. B. Book, Ed. Cambridge, MA: MIT Press, 1998.

[17] C. F. Chiasserini, R. R. Rao, ”Pulsed battery discharge in communication devices”, in Proc. MobiCom, Seattle, WA, pp. 88-95, Aug. 1999.

[18] C. F. Chiasserini, R. R. Rao. ”Improving battery performance by using traffic shaping techniques”, IEEE Journal on Selected Areas in Communications, 19(7), pp. 1385-1394, 2001.

[19] M. R. Jongerden, B. R. Haverkort, ”Which battery model to use?”, IET Softw., vol. 3, no. 6, pp. 445-457, Dec. 2009.

[20] C. F. Chiasserini, R. R. Rao, ”A model for battery pulsed discharge with recovery effect”, in Wireless Communications and Networking Conference, New Orleans, LA, USA pp. 636-639, 1999.

[21] E. Evan-Dar and Y. Mansour, “Learning rates for Q-learning”, Journal of Machine Learning Research, vol. 5, pp. 1-25, 2003.

[22] P. Auer, T. Jaksch, and R. Ortner, “Near-optimal regret bounds for reinforcement learning,” Advances in Neural Information Processing Systems, vol. 21, pp. 89-96, 2009.

[23] N. Salodkar, A. Bhorkar, A. Karandikar, V. S. Borkar, “An on-line learning algorithm for energy efficient delay constrained scheduling over a fading channel,” IEEE Journal on Selected Areas in Communications, vol. 26, no. 4, pp. 732-742, Apr. 2008.

[24] W. B. Powell, ”Approximate Dynamic Programming: Solving the Curse of Dimensionality”, 2nd edition. New York: Wiley, 2011.

[25] Y. Mansour and S. Singh, ”On the complexity of policy iteration”, in Proceedings of the 15th International Conference on Uncertainty in AI, Stockholm, SE, pp. 401-408, 1999.

[26] IEEE 802.15.4e Draft Standard: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for

Low-Rate Wireless Personal Area Networks (WPANs), IEEE Std., March 2010.

[27] A. Galindo-Serrano, L. Giupponi, and M. Dohler, “Cognition and docition in ofdma-based femtocell networks,” in IEEE Globecomm, Miami, Florida, USA, pp. 6–10, Dec. 2010.