Reinforcement Learning Strategies for Energy Management in Low Power IoT

Yohann Rioual, Johann Laurent, Eric Senn, Jean-Philippe Diguet

Reinforcement Learning Strategies for Energy Management in Low Power IoT

Yohann Rioual, Johann Laurent, Eric Senn and Jean-Philippe Diguet
CNRS UMR6285 - Lab-STICC, University of South-Brittany - UBS, Lorient, France
Email: {firstname.lastname}@univ-ubs.fr

***Abstract*—Energy management in low power IoT is a difficult problem. Modeling the consumption of a sensor node is complicated, they operate in a stochastic environment. They harvest energy in their environment but energy sources present time-varying behavior. It becomes hazardous to predict in advance the energy behavior of our system. In this paper we propose a new approach using both neural networks to estimate the harvesting energy and reinforcement learning algorithms to find the operating parameters to maximize the node's performance while preserving its energy resources.**

***Index Terms*—Energy management, WSN, Reinforcement Learning, Q-learning, neural network**

I. Introduction

Over the last few decades, sensor networks have emerged to meet the increasing needs for diffuse and automatic observation of complex physical and biological phenomena in various areas. This growth was made possible by many benefits of Wireless Sensor Networks (WSNs) such as unmanned operation, easy deployment, real-time monitoring, and relatively low cost. All these advantages have lead the development of different sensor nodes used in different applications from environmental monitoring to industrial facility surveillance through shark detection on the Australian coast. We can identify two main applications for sensor networks, monitoring applications and phenomenon detection. Depending on the application the node has several sensors and may not be allowed to switch off some sensors. So the large number of applications and sensors goes hand in hand with a large number of energy needs, since the applications' constraints and node architectures are disparate. The energy profile of each application depends of components used to conceive the node, constraint fixed by the end user, but the environment itself too.

Application	QoS	Delay tolerance
Monitoring	low	high
Event detection	high	low

Fig. 1: Applications constraints

WSN consist of spatially distributed autonomous sensor. Thus, the battery autonomy is critical, battery replacement is expensive or sometimes impossible. To outperform, most WSNs rely on the harvesting capability but these technologies have a low efficiency (only $15\mathrm{mW/cm^2}$ for solar technology [1]). However a oversized battery have a prohibitive cost and size. As nodes can recharge whenever energy is available and harvested energies vary greatly over a day, it is arduous to know in advance the harvested energy. The problem is to determine at design time the node autonomy. We need accurate model for the different components of a node such as harvesters, processor or RF front end. And model accurately a harvester like a solar panel or a wind turbine is difficult, several parameters need to be considered like the wind, the ambient temperature or the brightness. In addition, harvesters have specific parameters (Solar panels have parameters such as ideality factor, open-circuit voltage or short-circuit current). And these parameters change over time due to environmental conditions or aging. Similarly it is arduous to model precisely a processor or a RF front end, and it is time consuming to model the consumption of all sensors and components. So model a complex system such as a sensor node is hazardous.

In this paper, we present a new approach in order to realize an energy management system. The rest of this paper is organized as follows : The Section II presents related works on automatic learning for the energy management in sensor network, and the Section III presents our approach. The Section IV introduces a simulator developed to test our strategy and some results. Finally, Section V concludes this paper.

II. Related works

Since the node battery autonomy is critical. We need to have methods to determine before deploying a node, if there is enough energy in its battery. [2] model a sensor node at function level. Since the node's consumption depends on the hardware and the software, they narrow the node at 5 different units : acquisition unit, computation unit, reception unit, transmission unit and battery. For each of these units, they link the possible functioning states to a consumption profile. When simulating the consumption of a node, they recover an overall consumption profile by adding the consumption profiles of the various functions and thus obtains the energy behavior of its system. This modeling method allows the energy behavior of the system to be determined early in the design phase. This work was done under simulation and does not take into account harvesting.

[3] wants to model and simulate the operation of a mobile connected buoy for a pollution puddle monitoring application in the Mediterranean sea. To model his buoy, the author

used Fluid Stochastic Petri Nets and takes into account the different potential states of his system (data acquisition or transmission). He also takes into account the state of the environment, i.e. wind and current to evaluate the drift of his buoy and to estimate the energy that will be able to harvest with meteorological data. It seeks to determine the influence of an electric motor, used to move the buoy and track pollution puddles, on the overall consumption of the system.

Another approach is the use of Neural Network to model a system. In [4] the authors use neural networks to model consumption, not of a complete node but only for analog components. For each component, the authors measure the instantaneous energy consumption according to inputs and operating parameters. Then use this data to train a neural network per component. Its allows approximating the function linking inputs and parameters to the consumption of the component. They then integrate these neural networks into a simulator to obtain the overall system consumption. They afterward perform an experimental validation by comparing the results they obtain with their method and the measurements on a node. The accuracy method differs from the reality only by 1.53% with a maximum error rate of 3.06%. The main problem is sensor nodes are heterogeneous and uses both analog and digital components, this method is difficult to extend. In addition it requires human experiments to train a neural network and it can't be automatized.

Modeling a entire node system is difficult. They are heterogeneous and deployed in stochastic environment. The energy that the node will be able to harvest is hard to predict, the best solution is to use real environmental data to reproduce the environment. Since we can't model our energy consumption precisely with a automatic method, we are looking for optimize our system without knowledge of its model.

[5] proposes a method for easily setting up a node in order to respect the operating constraints set by the end user. To do this, it models the problem of finding the optimal functioning parameters in the form of a Markov Decision Process (MDP are detailled in Section III-B). Its set of states and its set of actions are identical and correspond to all the possible operating states of the node. The evaluation function takes into account high level metrics such as latency or consumption with associated coefficients. Depending on the field of application of the sensors, the value of the coefficients varies according to the importance of the metric for the application. The MDP is resolved with the Bellman equations to obtain the optimal solution. The main limitation is computational, the Bellman equation needs computation power but the authors extend their work in [6] using heuristic to reduce the processing power and allows the research of the optimal parameter directly on the node. But the node changes its functioning parameters when the supervisor asks it and not dynamically depending on the environment.

There is a different method using neural network to find the optimal functioning parameters. For instance [7] uses neural networks to adjust the radio settings (transmission power) according to the quality of the communication channel. When the system is recharged, it carries out a learning phase during which it tests various settings by sending predetermined data packets and observing the energy consumption of its radio during transmission. It recovers the status of the packets transmitted and, depending on their condition, it deducts the status of its communication channel. It uses this knowledge to train two neural networks to create a map between setting to power consumption and channel state and setting to Vector Error Magnitude (EVM). By setting a maximum error vector, its neural networks can predict the best compromise between the energy consumed and the vector error of the transmitted packet in a particular channel state. Then it changes the settings according to the channel, so its system is able to use only the minimum amount of energy to guarantee the desired quality of service and at the end of the learning, it consumes up to 2.5 less energy depending on the state of the communication channel.

However training a neural network takes so times and the node need quickly a effecient energy management. So [8] proposes to adapt the duty cycling of a node using reinforcement learning. Its algorithm dynamically adapts its energy allocation policy according to environmental variations, it explores new policies while exploiting its already acquired knowledge in order to improve the quality of service. For a given energy recovery and battery charge, it allocates energy to the functioning of the node and according to the wasted energy, i.e. the energy that its node has collected but that it has not used, it modifies the energy that it will allocate to the next iteration of its algorithm. It tests its algorithm with several energy storage capacities and multiple harvesting sources. As a result, the learning algorithm achieves maximum energy efficiency and superior service quality for state-of-the-art power managers.

Modeling a entire node is difficult, its requires to much time and computation, especially if we want to be precise. However reinforcement learning algorithms do not need a model to find a optimal functioning policy. They can be used to teach sensor node how to manage their energy.

III. An approach by reinforcement learning algorithms

We identify to different problem in sensor nodes. Firstly sensor node harvest energy in its environment to extend its lifetime but the harvesters are aging due to environmental conditions (dust for a solar panel, for instance) and this aging is not taking account into harvesters models. Then we cannot extend a sensor node's lifetime using harvesting technologies without giving it the possibility to optimize the use of this energy. Since we have two different problems, we propose a approach with two strategies, one to improve harvesters model using neural network and one using reinforcement learning to optimize the sensor node energy use.

A. Overview of the approach

We propose to combine two different approaches. The first one is based on supervised learning and neural networks.

Indeed our harvesters alter during the node deployment, it is getting older. In the end sensor nodes have less energy available, it really depends of the node. Some of them can still have enough energy harvested.

The second approach is based on reinforcement learning algorithm. Our node have battery with a limited amount of energy, and we want to maximize its performance. The wanted performance depends of the sensor network application. It can be extend the network lifetime or maximize the quality of service (QoS). Since it is impossible to model the sensor node consumption, we are going to teach it to self-manage so it can adapt its consumption in accordance with its energy resources. In this paper, we focus on the RL based approach.

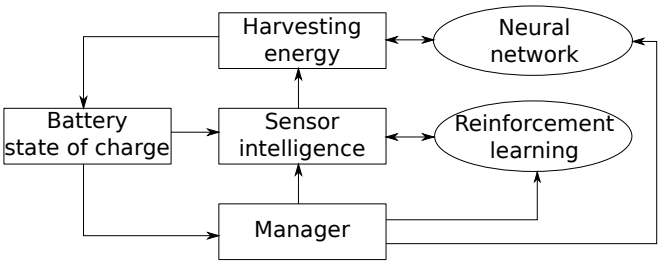

Fig. 2: Multi-strategies approach

B. RL based approach

Reinforcement Learning (RL) is a formal framework that models sequential decision problems [9], in which an agent learns to make better decisions by interacting with the environment (fig. 3). When the agent performs an action, the state changes and the agent receives a reinforcement called a reward, encoding information about the quality of the transition. The agent's goal is to maximize its total reward over the long term.

1) Definitions: We can model the energy management problem as a Markov Decision Process (MDP). A MDP provides a mathematical framework for modeling decision making in situations where outcomes are partly random and partly under the control of a decision maker. A MDP is formally defined as a n-uplet $< S, A, T, R >$ where S is a state space, A a set of possible actions, $T : S \times A \times S \rightarrow [0, 1]$ are the transition's probability between states ($T(s, a, s') = p(s'|a, s)$ is the probability to reach the state s' starting from s after taking the action a) and $R : S \times A \rightarrow \mathbb{R}$ is a reward signal. A politic $\pi : S \times A \rightarrow [0, 1]$ tell the agent how to act (the probability to take a action in a given state). A optimal politic π^* maximize the expected discounted reward,

$$\pi^* = \arg\max_{\pi} J(\pi) = \arg\max_{\pi} \mathbb{E}[\sum_{t=0}^{\infty} (\gamma^t \times R(s_t, \pi_t(s_t))] \tag{1}$$

where t is step of time (called decision epochs) and $0 < \gamma < 1$ is a discount factor. In the RL framework, the goal is to learn a optimal politic when the model, T and R, is unknown.

In our problem, the state space S is the different states the battery can take, the actions set A is the different operating mode our node can operate. R is a reward using the battery states and different parameters we can use to evaluate the node performance such as the sampling frequency, or the transmission frequency.

Since the environment is uncertain, we propose the sensor node control its operating mode itself and adapt to the random change of its environment. We use Reinforcement Learning (RL) algorithms to determine the best operating mode according several parameters. This parameters depend of our application. And depending on the application, sensor node have different constraints to respect. These constraints involve conflicting objectives such as maximizing the quality of service and minimizing consumption. Our energy management must find the best compromise between these different objectives.

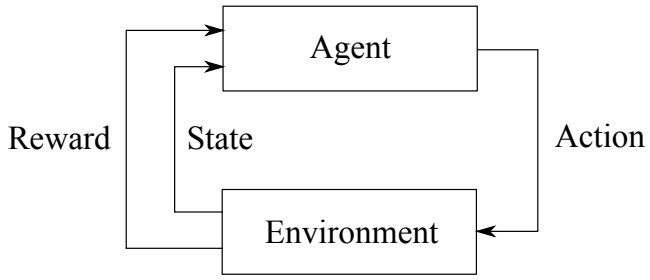

Fig. 3: Interaction between agent and environment

In a RL algorithm, we have a agent which can be a software or in our case a sensor node. So the agent tries a action and then observe how its environment react, if the reaction is good the agent have a good reward but in the other case he have a bad reward. So he tries a lot of action, observe and learn with the objective to maximize its rewards.

C. Q-learning algorithm

In this section we present a RL algorithm from the state-of the-art which is the Q-Learning algorithm [10]. Q-learning algorithm is widely used since it is easy to implement and its convergence is proven. So we test this algorithm for the energy management of a sensor node.

Algorithm 1 Q-learning algorithm

Initialize $Q(s, a)$ arbitrarily
The agent observe the initial state s_0
Initialize s
for each decision epochs **do**
  Choose a from s using policy derived from Q
  Take action a, observe the new state s' and the associated reward r
  $Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a') - Q(s, a))$
  $s \leftarrow s'$
end for

The aim of this algorithm is to optimize the reward function for each couple state-action. So it uses the temporal difference principle to update the reward function. The temporal difference corresponds to the difference between two successive estimates of a couple's gain expectation.

a) Evaluation function: In the classical version of Q-learning, the Q values are in a two dimensions ($S \times A$) table called Q-table. The goal of the evaluation function is to evaluate each couple state-action (s, a).

The Q-learning updates the evaluation function using the equation below,

$$Q(s_t, a) = Q(s_t, a) + \alpha(r + \gamma \max_a Q(s_{t+1}, a'') - Q(s_t, a)) \quad (2)$$

In equation 2, the term $r + \gamma \max_a Q(s_{t+1}, a'') - Q(s_t, a)$ correspond to the difference between the new and the old estimation of $Q(s_t, a)$, it's the temporal difference.

b) Reinforcement function: The reinforcement function of Q-learning algorithm is based on the value iteration algorithm. The goal of this algorithm is to optimize the evaluation function for each pair state-action (s, a) in order to deduce an optimal control strategy.

For this purpose, Q-learning uses the temporal difference principle [11] to update the evaluation function. The temporal difference corresponds to the difference between two successive estimates of a couple's gain expectation.

c) Learning rate α: The learning rate α determines how fast the new information will surpass the old one. A factor of 0 would not teach the agent in question anything, whereas a factor of 1 would only teach the agent the latest information. $\alpha \in]0, 1[$

d) Discount factor γ: The discount factor γ determines the importance of future rewards. A factor of 0 would make the agent myopic by considering only current rewards, while a factor of close to 1 would also involve more distant rewards. If the discount factor is close to or equal to 1, the value of Q may never converge. $\gamma \in [0, 1]$

Our goal is to use this algorithm to taught our sensor node to learn to manage its energy by choosing the best operating mode for its micro-controller and sensors. In the next section we present a simulator we propose to test our approach and in addition we present some results.

IV. Simulator architecture and preliminary results

This section presents the simulator we conceived to test the energy management algorithms and some results of the use of Q-learning algorithm to manage our node energy.

A. Simulator architecture

The simulator has been designed to be as flexible as possible. It takes into account the component consumption data provided by the manufacturer's datasheet.

The production from solar panels is estimated using the surface power coming from sun radiations. It uses state-of-the-art model for photovoltaic panel [12]. The photovoltaic panel's temperature is influenced by the solar irradiation, the ambient temperature and the wind velocity.

Climate data come from the meteorological station of Lann Bihoué at Lorient, France. It includes an archiving of wind (angle, speed) and ambient temperature. These data are integrated to the simulator which selects the data that fit the date and time of simulation. These data are updated every 30 minutes.

A sunshine model is also implemented to estimate the surface power of the sun radiations. It depends on the date and node position on the globe. All the details can be found in [13].

The performance of a battery is modeled. The battery is represented by one capacitor with variable capacitance and one resistor which allows representing the batteries self-discharge when they are not in use. The potential VSOC varies between 0 and 1 V and represents the battery charge level.

$$\frac{dx}{dt} = \frac{-1}{R_{SD} \cdot C_{CAP}} x - \frac{1}{C_{CAP}} u \quad (3)$$

where x denotes the voltage that represents the state of the charge and u represents the current to the battery terminals. R_{SD} is the self-discharge resistor and C_{CAP}, the initial battery's capacitance.

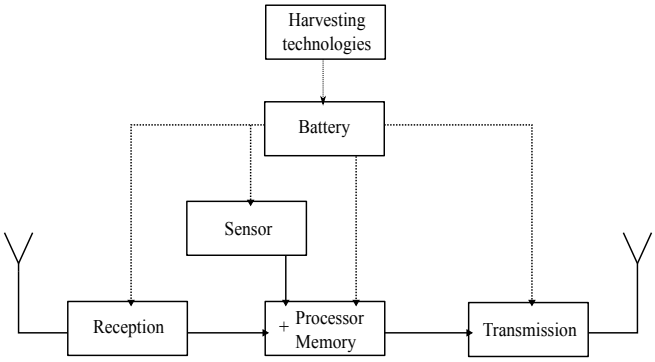

Fig. 4: Hardware of a sensor node

The simulator takes account into the beginning date of the node's deployment, and the deployment duration.

B. Results

We want to simulate a sensor node with a energy management algorithm using the Q-learning algorithm. So we define our sensor node's components and our application is a sensor buoy system monitoring the environment. Figure 6 details the elements used in the design of the sensor buoy system. It includes an antenna for RF transmission, a harvesting system (solar panel) to supplement the power supply, and two sensors to monitor meteorological data (wind speed, wind direction, atmospheric pressure, etc). Moreover we include a beacon light to make the buoy visible at night.

The application is environment monitoring, and we have a sampling frequency between 0.01Hz and 1Hz. We transmit the data as soon as we have it.

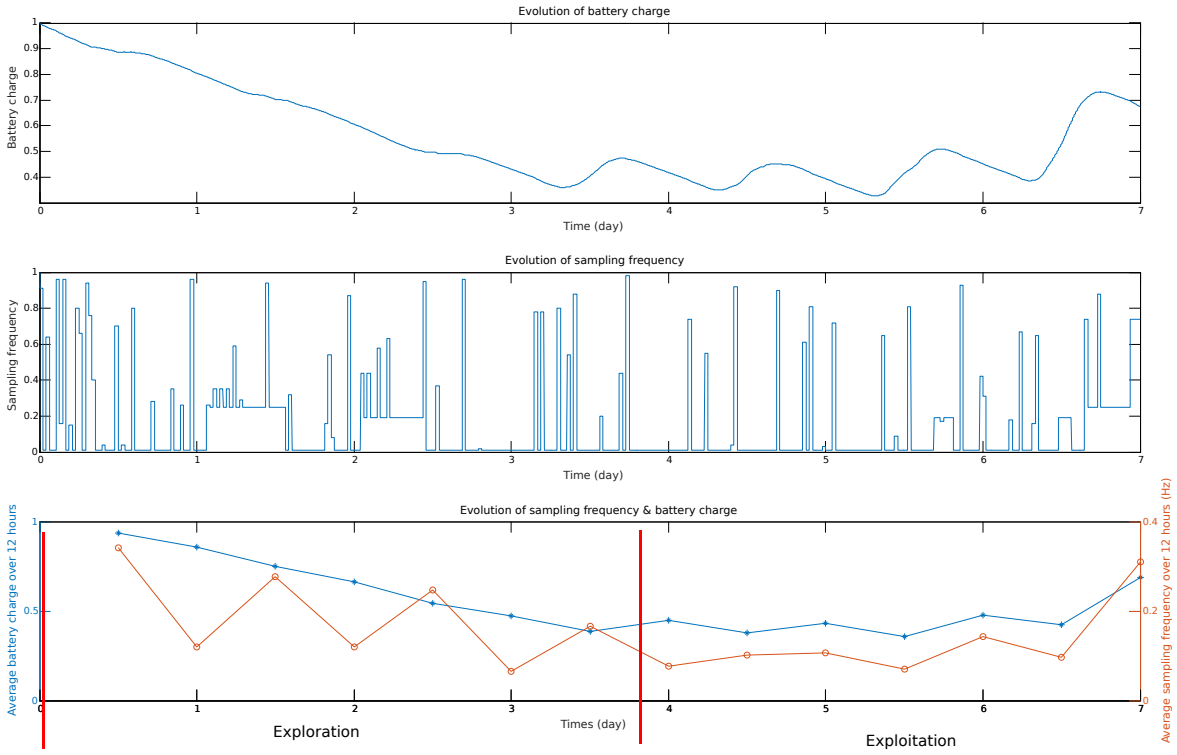

Fig. 5: Q-learning results

Components	Characteristic
Anemometers	WindMaster HS
Atmospheric sensor	YOUNG61302L
Processor unit	Cortex-M4 MCU
Communication unit	CC1000
Solar panel	2×10W
Battery capacity	5200mA

Fig. 6: Sensor buoy's components

We use a Q-learning algorithm to manage the energy in the buoy. We choose to run the Q-learning algorithm every 30 minutes. We have state space with 100 different states which correspond to the percent of energy left in the battery. Our set of action is composed of 10 different actions (1Hz, 0.9Hz, $\cdots$, 0.1Hz). In this work, the reward is computed as a function of both f and e_R :

$$R = \phi f \tag{4}$$

where ϕ is the feature, which correspond to the normalized residual energy.

We choose the parameters of the Q-learning algorithm as follow:

a) Learning rate: Our problem is stochastic, so the algorithm converge under some conditions that require it to decrease to zero. But as used in [8] we choose to set it to a constant value 0.1.

b) Discount factor: We want to take account into previous experiments, so we set the discount factor to 0.8.

Parameter	Value
α	0.1
γ	0.8
Time slot duration	30 minutes

Fig. 7: Q-learning parameters

The Q-learning follows a exploration policy, to test new action. The exploration function used is :

$$1 - \frac{1}{log(2+t)} \tag{5}$$

where t is the current time slot.

We simulate one week of operation. As can be seen in fig. 5, at deployment, the node has no prior knowledge of the energy management policy to follow. So it does a lot of exploration. After a period of time, it explores less and begin to exploit the information collected at the beginning of the deployment. It is fairly easy to see that it does more testing at first and then calms down. It can also be observed that the average frequency of measurement is lower when the battery charge is reduced and increases when it increases. The node therefore

adapts its energy management policy according to the load of its battery. And without initial knowledge, it is able to operate during the whole simulated week.

V. Perspectives and conclusion

In this paper, an approach to energy management in a sensor node was presented. This approach is based on two different strategies. An approach using neural networks to adapt the parameters taken into account in harvester model. And a second approach based on Reinforcement Learning to optimize the operating mode of a node so that it can adapt to the energy resources at its disposal.

We have detailed the second part of the approach. We formulated the energy management problem in the form of a Markov Decision Process, which we solved using a state-of-the-art learning algorithm : Q learning. We presented the simulator we use to test this approach and some preliminary results. These results show us that our approach is justified and relevant. It allows the node to find a stand-alone energy management policy without initial knowledge.

In our future work, we will try to take into account more operational parameters of our node in order to optimize its energy management policy. We will also detail the neural networks-based approach for harvesters.

References

[1] Zhou, G., Huang, L., Li, W., & Zhu, Z. (2014). Harvesting Ambient Environmental Energy for Wireless Sensor Networks: a survey. Journal of Sensors

[2] Randrianarisaina, A., Pasquier, O., & Charg, P. (2014). Energy consumption modeling of smart nodes with a function approach. In 2014 Conference on Design and Architectures for Signal and Image Processing (DASIP) (pp. 16).

[3] DArienzo, M., Iacono, M., Marrone, S., & Nardone, R. (2013). Petri net based evaluation of energy consumption in wireless sensor nodes. Journal of High Speed Networks, 19(4), 339358.

[4] Suissa, A., Romain, O., Denoulet, J., Hachicha, K., & Garda, P. (2010). Empirical Method Based on Neural Networks for Analog Power Modeling. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 29(5), 839844.

[5] Munir, A., & Gordon-Ross, A. (2012). An MDP-Based Dynamic Optimization Methodology for Wireless Sensor Networks. IEEE Transactions on Parallel and Distributed Systems, 23(4), 616625.

[6] Munir, A., Gordon-Ross, A., Lysecky, S., & Lysecky, R. (2012). Online algorithms for wireless sensor networks dynamic optimization. IEEE Consumer Communications and Networking Conference (CCNC) (pp. 180187).

[7] Banerjee, D., Sen, S., & Chatterjee, A. (2015). Self learning analog/mixed-signal/RF systems: Dynamic adaptation to workload and environmental uncertainties. IEEE/ACM International Conference on Computer-Aided Design (ICCAD) (pp. 5964).

[8] Aoudia, F. A., Gautier, M., & Berder, O. (2017). Learning to Survive : Achieving Energy Neutrality in Wireless Sensor Networks Using Reinforcement Learning. IEEE International Conference on Communications (ICC).

[9] Sutton, R. S., & Barto, A. G. (2013). Reinforcement learning : an introduction. Neural Networks IEEE Transactions on, 9(5), 1054.

[10] Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. Machine Learning, 8(3–4), 279–292.

[11] Sutton, R. S. (1988). Learning to Predict by the Methods of Temporal Differences. Machine Learning, 3, 9–44.

[12] Bellia, H., Youcef, R., & Fatima, M. (2014). A detailed modeling of photovoltaic module using MATLAB. NRIAG Journal of Astronomy and Geophysics, 3(1), 53–61.

[13] Piedallu C, & Ggout, J.C. (2007) Multiscale computation of solar radiation for predictive vegetation modelling. Annals of Forest Science, Springer Verlag/EDP Sciences, 64, pp.899-909.