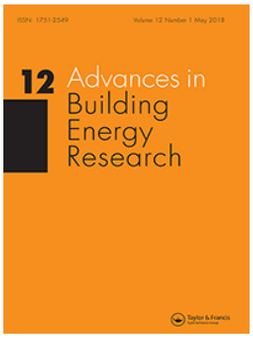

Advances in Building Energy Research

ISSN: 1751-2549 (Print) 1756-2201 (Online) Journal homepage: https://www.tandfonline.com/loi/taer20

Energy management in solar microgrid via reinforcement learning using fuzzy reward

Panagiotis Kofinas, George Vouros & Anastasios I. Dounis

To cite this article: Panagiotis Kofinas, George Vouros & Anastasios I. Dounis (2018) Energy management in solar microgrid via reinforcement learning using fuzzy reward, Advances in Building Energy Research, 12:1, 97-115, DOI: 10.1080/17512549.2017.1314832

To link to this article: https://doi.org/10.1080/17512549.2017.1314832

ADVANCES IN BUILDING ENERGY RESEARCH, 2018
VOL. 12, NO. 1, 97–115
https://doi.org/10.1080/17512549.2017.1314832

Taylor & Francis
Taylor & Francis Group

Energy management in solar microgrid via reinforcement learning using fuzzy reward

Panagiotis Kofinas[a,b], George Vouros[a] and Anastasios I. Dounis [b]

[a]Department of Digital Systems 80, University of Piraeus, Piraeus, Greece; [b]Department of Automation Engineering 250, Piraeus University of Applied Sciences (T.E.I. of Piraeus), Athens, Greece

ABSTRACT

This paper proposes a single-agent system towards solving energy management issues in solar microgrids. The proposed system consists of a photovoltaic (PV) source, a battery bank, a desalination unit (responsible for providing the demanded water) and a local consumer. The trade-offs and complexities involved in the operation of the different units, and the quality of services' demanded from energy consumer units (e.g. the desalination unit), makes the energy management a challenging task. The goal of the agent is to satisfy the energy demand in the solar microgrid, optimizing the battery usage, in conjunction to satisfying the quality of services provided. It is assumed that the solar microgrid operates in island-mode. Thus, no connection to the electrical grid is considered. The agent collects data from the elements of the system and learns the suitable policy towards optimizing system performance by using the Q-Learning algorithm. The reward function is implemented by fuzzy system Sugeno type for improving the learning efficiency. Simulation results provided show the performance of the system.

ARTICLE HISTORY
Received 29 January 2017
Accepted 24 February 2017

KEYWORDS
Reinforcement learning; Q-learning; microgrid; energy management; fuzzy reward

1. Introduction

Nowadays, one of the most concerning environmental issues is global warming due to the use of fossil fuels that emit greenhouse gases. To satisfy the continuously rising energy demand and simultaneously reduce greenhouse gas emissions, renewable energy sources are integrated in the overall energy production (Sechilariu, Wang, Locment, & Jouglet, 2014).

New technologies encourage distributed generation resources around the world. Microgrid is a group of interconnected loads and distributed resources within clearly defined electrical boundaries that act as a single controllable entity with respect to the grid (Smith, 2012). Microgrid can operate in either grid-connected or island-mode. Its ability to operate in island-mode makes it an exceptional energy solution in remote areas where the cost of the grid expansion is prohibitive.

The challenging task in microgrids is the energy management between its elements. This involves planning and control in energy production and consumption, in order to save energy. In grid tied microgrids the coordination of the distributed energy sources

CONTACT Panagiotis Kofinas panagiotis.kofinas@gmail.com

and the energy storage has to be optimum in order to minimize the energy consumption from the grid. In island-mode, the energy management is crucial for the reliability of the microgrid under the uncertainties of the renewable generators and the dynamic loads.

Many studies concerning the energy management of the microgrids have been presented in literature. Huang and Liu (2013) proposed a self-learning single neural network for energy management in residential applications. A two step ahead Q-learning algorithm for planning the battery scheduling in a wind microgrid has been proposed by Kuznetsova et al. (2013), while a three step ahead Q-learning for planning the battery scheduling in a solar microgrid has been proposed by Leo, Milton, and Sibi (2014). A multiagent system using Q-learning has been proposed by Raju, Sankar, and Milton (2015) in order to reduce the power consumption of a solar microgrid from the grid. Chung, Yoo, and Oh (2013) proposed a multi agent system with a central coordinator for optimal response in emergency power demand. An autonomous multiagent system for optimal management of buying and selling power has been proposed by Kim, Lim, and Kinoshita (2012) and an autonomous multiagent system for power generation and power demand scheduling has been proposed by Foo, Gooi, and Chen (2014).

In our previous work (Kofinas, Vouros, & Dounis, 2016), we proposed a single-agent system for performing energy management of a solar microgrid via Q-learning. The goal, closely related to this work, is to satisfy the energy demand in the solar microgrid, optimizing the battery usage, in conjunction to satisfying the quality of services provided by energy consumer units. The simulation results showed lack in covering the power demand in some situations, the state of the charge (SOC) of the battery remained in low levels and many deep discharges were observed. In this work, the proposed agent is enhanced with an improved exploration/exploitation strategy based on the number of times of being in a state. In addition to that, the learning mechanism is improved by adopting a fuzzy reward function. This is deemed necessary, given the discretization of states, while it presents improvement regarding the experimental results.

The main contributions of this paper are as follows: We address the problem incorporating consumer units that are responsible for providing services and/or goods at specific levels of demand. This provides more trade-offs beyond pure parameters for energy production and consumption. For this purpose as in previous work (Kofinas et al., 2016), we use a desalination unit. Additionally we introduce a fuzzy reward function in order to enhance the learning mechanism. Then, we show preliminary results on how the energy demand can be fulfilled with respect to providing the demanded quality of services by developing an agent using reinforcement learning (RL) techniques.

Experimental results are based on real data concerning the demanded load and the energy produced by a photovoltaic (PV) unit. The structure of this paper is as follows:

Section 2 provides preliminaries about RL and Q-learning. Section 3 provides preliminaries about Fuzzy System (FS) and fuzzy reward. Section, 4 presents the problem specification, and Section 5 presents the design of the Q-Learning agent. Section 6 presents and discusses simulation results. Finally, Section 7 concludes the paper sketching future work.

2. Reinforcement learning

Markov decision processes (MDPs) are widely used for modelling sequential decision-making problems. These can be discrete time processes based on state transitions via

actions. MDP can be described by a time parameter t, a set of states **S**, a set of actions **A** that can be performed in each state at any time, the transition probabilities $P = (s'|s,a)$ for a state s' to occur when performing the action a (in a time step) in state s; and the reward function $R(s,s')$ that describes the reward gained when the transition between states occurs. A policy is a decision-making function π:**S**→**A** that defines which action the agent should take at any state so as to maximize the expected sum of the discounted rewards.

Solving the MDPs results to computing the policy that maximizes the cumulative discounted expected reward (expected utility) $V(s_t)$.

$$V(s_t) = \sum_{t=0}^{\infty} \gamma^t R(s_t,\ s_{t+1}), \tag{1}$$

where γ is the discount factor determining the importance of future rewards, and t is the time step. When the transition probabilities are not known, model-free RL techniques can be used.

The term RL refers to a family of algorithms inspired by human and animal learning. The objective of RL is to discover a policy, that is, a mapping from states to actions, by learning through exploration in the space of possible state-action pairs. Actions that result in a good performance when performed in a specific state are 'rewarded' and actions leading to a poor performance are 'punished'. The goal of RL is to discover the policy that maximizes the reward received (Russel & Norving, 1995). Although there are many model-free RL algorithms to approach the problem, we may distinguish three main approaches for computing the optimal action at any state:

The first is called Q-Learning (Watkins, 1989). Q-learning aims at learning an action-value function that gives the expected utility of taking a given action in a given state and following the optimal policy thereafter. The second method uses gradient information (Berenji & Khedkar, 1992), assuming a differentiable utility function. The third method uses the temporal difference between the expected performance and the measured performance (Barto, Sutton, & Anderson, 1983). In this study we use Q-Learning as we do not have a model of the environment and our problem involves stochastic transitions and rewards, without requiring adaptations.

In Q-Learning, the value of each action a_k when performed in a state s_k is calculated as follows:

$$Q_{\kappa+1}(s_\kappa, a_\kappa) = Q_\kappa(s_\kappa, a_\kappa) + \alpha(R_{k+1} + \gamma \max Q_k(s_{k+1},\ a) - Q_k(s_k, a_k)), \tag{2}$$

where $Q_{k+1}(s_k, a_k)$ is the new value of the state-action combination, $Q_k(s_k, a_k)$ is the previous value for the same state-action combination, α is the learning rate, and $R_{k+1} = R(s_k,a_k,s_{k+1})$ is the reward for performing the action a_k in the state s_k. Q learning assumes that the agent continues by performing the optimal policy, thus $\max Q_k(s_{k+1},a)$ is the value of the best action performed in the state s_{k+1} (i.e. the state resulting after performing a_k in the state s_k). The learning rate α shows in what per cent the new information overrides the old one.

Thus, the Q-learning agent chooses the action a_k to be performed in the current state s_k (either via exploration or by exploitation) and identifies the resulting state s_{k+1}. It gets the reward from the transition from s_k to s_{k+1}, i.e. $R(s_k, a_k, s_{k+1})$, and recalculates the value for

the action-state pair (a_k, s_k) assuming that it performs the optimal policy from state s_{k+1} and on.

3. Fuzzy reward

In RL the reward signal is provided by the environment and indicates whether an agent performed a 'good' action in a given state. The reward function has to specify succinctly the constraints to be respected, also incorporating the parameters that should affect agents' decisions and their interrelationships in order to improve the learning efficiency (Zhai, Chen, Li, & Guo, 2009). In most of the cases an accurate reward function affects the complexity of the computations required. In addition to that, multi-criteria decision-making typically involves flexible requirements for the optimality (Kurano, Yasuda, Nakagami, & Yoshida, 2003). In order to balance between flexible requirements and the complexity of implementation a fuzzy reward is proposed based on FS. Two FS are used conventional, the FS type Mamdani (Mamdani & Assilian, 1975) and the FS type Sugeno (Takagi & Sugeno, 1985). Their main difference is the way the crisp output is generated from the fuzzy inputs; the Mamadani uses a defuzzifier while the Sugeno uses the weighted average. Due to Sugeno's computational efficiency we are using the zero-order type Sugeno where the consequents of the rules are constant numbers. FS employs a way of reasoning using fuzzy rules. The set of the fuzzy rules compose the rule base of the system and can embed the knowledge of the expert. The rules are of the form of

$$R^l: x_1 \text{ is } A_1^l \text{ and } x_2 \text{ is } A_2^l \ldots \text{and } x_n \text{ is } A_n^l \text{ then } y \text{ is } C^l,$$

$l = 1,2 \ldots M$ where M is the total number of the rules, A_n is a fuzzy set of the inputs, $X = (x_1, x_2 \ldots x_n)$ is the crisp input vector, y is the crisp output and C is a constant number (fuzzy singleton) defined by the expert and corresponds to the value of the FS output signal when the firing strength of this rule is maximum and the firing strengths of the other rules are zero. The block diagram of the Sugeno FS consists of four blocks (Figure 1): The fuzzifier which converts the crisp values of the input vector into fuzzy values. The knowledge base which defines the membership functions (MFs) and stores the fuzzy rules. The inference engine which performs the approximate reasoning exploiting the fuzzy rules and finally the weighted average block that calculates directly (without the need of a deffuzifier) the crisp output via the firing rules. Details for these computations are provided in Section 5.3.

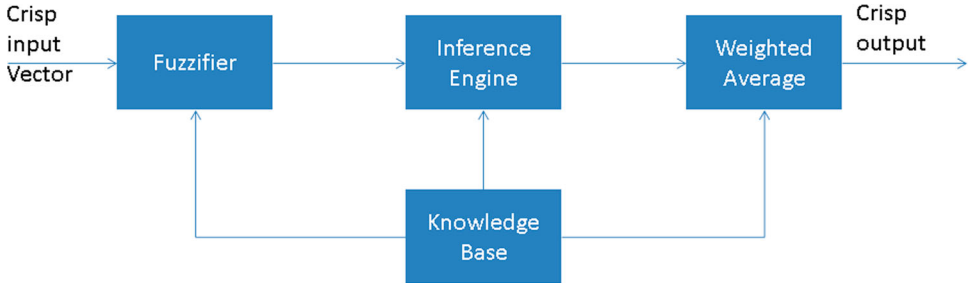

Figure 1. Block diagram of FS type Sugeno.

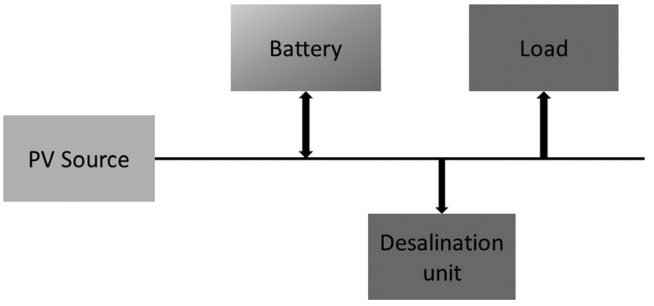

Figure 2. Energy system overview.

4. Problem specification

4.1. Energy system

The power system consists of a PV source 25 kW, a desalination unit, which delivers potable water into a tank of 1 m^3 capacity, a battery of 80 kWh and a dynamic load that represents a small residential building of four households.

This model does not take into account the quality of the power (voltage, current, frequency etc.) and the type of the power (AC/DC). It considers the delivered amount of power amongst the elements of the microgrid. This is an assumption that leads to good results in energy management applications as the information for taking a decision regarding energy management is not related with the way that this power is distributed through the power circuits. The power flow of the energy model is depicted in Figure 2.

4.2. Problem formulation

The PV source is the only source of the system. The battery can perform either as a source or as a consumer and the other consumers of the system are the desalination unit and the dynamic load. The uncertainties of power production (pp) and power demand make the management of such a system a challenging task. The difficulties of managing the system concern whether the battery must perform as a consumer or as a source, and whether the desalination unit must stop or operate. For example in a simple case where there is surplus in pp the battery should charge, otherwise the produced power is wasted. In a more complicated case where the power produced by the PV source is enough to supply the dynamic load, the rest of the power must be distributed to the battery and/or to the desalination unit. There is no certain decision about the power distribution as it is a multi-variable problem depending on the water demand, the SOC of the battery, the produced power and the power demand. The decision-making agent receives input from its environment (i.e. the microgrid) and decides on the most appropriate actions to react to the environment in order to achieve its goals (Dounis, 2010), that is, to satisfy the energy demand in the solar microgrid in conjunction to achieving the quality of services provided by energy consumer units, also optimizing the battery usage.

5. Design of the Q-learning agent

5.1. Agent overview

Towards developing the agent we need to specify PAGE, that is, percepts, actions, goals and the environment.

The environment of the agent is the microgrid system already described. The agent collects data from the microgrid to determine its current state: It collects the power demand of the consumers, the power produced by the PV source, the amount of the water into the tank and the SOC of the battery. The states of the agent are defined by the power balance (pb) between the produced power and the power demand, the SOC and the water into the tank.

The actions that the agent can perform affect the battery and the desalination unit. These actions are charging/discharging the battery and operation/nonoperation of the desalination unit (Figure 3). Thus the agent can decide whether the battery will be charged or discharged and whether the desalination unit will operate or not. In our study, the action unit consists of an exploration/exploitation strategy to explore the application of actions in different states and a Q-Learning method for learning the optimal mapping between system states and actions.

The goal of the agent is the energy management of the microgrid so as the battery to be fully charged, the water tank to be full of water and cover the power demand. Figure 4 presents the policy iteration scheme.

5.2. State space and actions

As already said, the state of the microgrid is determined by the SOC of the battery, the percentage of the potable water into the tank, with respect to the capacity of the tank,

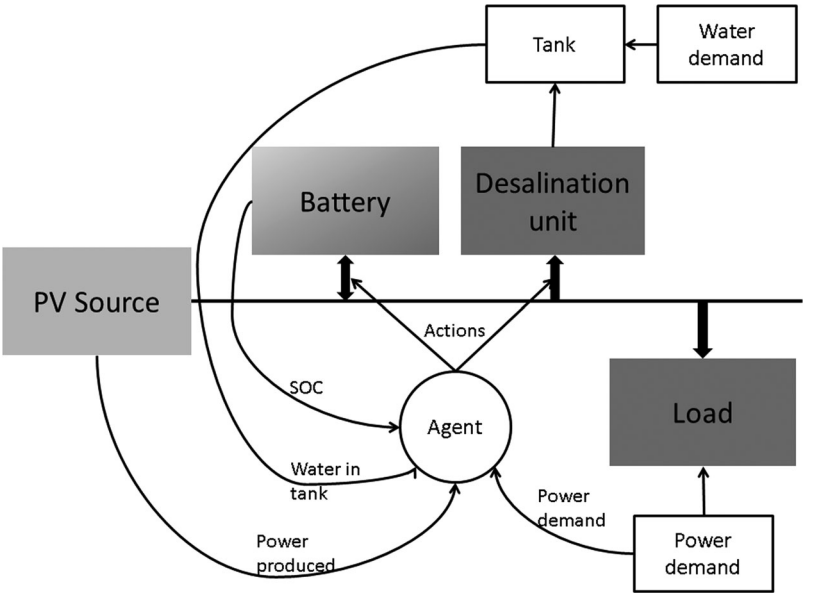

Figure 3. Agent interaction with the environment.

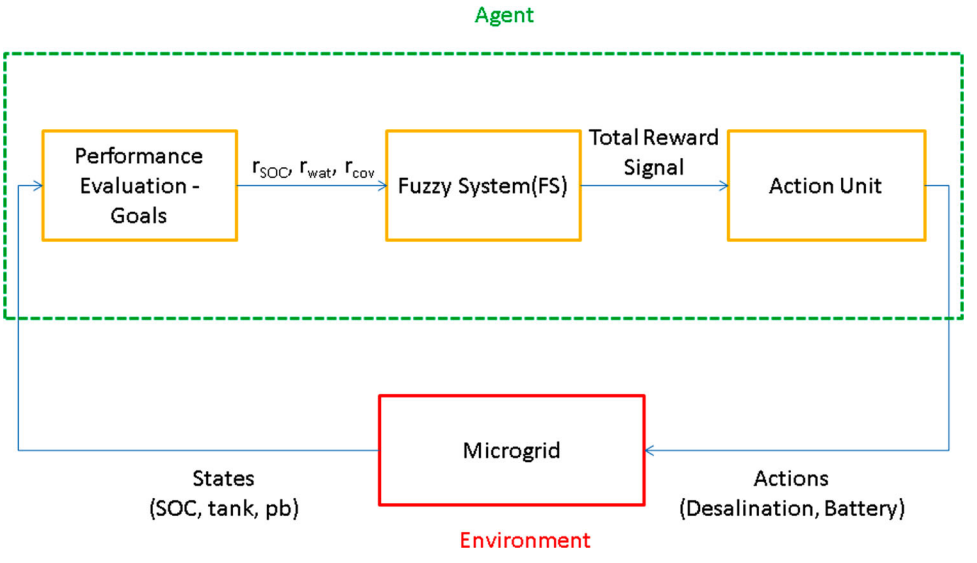

Figure 4. Q-Learning agent scheme.

and the pb. Although these measures are continuous, we discretize the state space by considering different ranges of variable's values. Thus:

The discrete state of battery Charge (dSOC) is considered to range in {1,2,3,4} as follows:

$$\begin{aligned}
&\text{dSOC} = 1 \text{ if } 0\% \leq \text{SOC} < 25\% \\
&\text{dSOC} = 2 \text{ if } 25\% \leq \text{SOC} < 50\% \\
&\text{dSOC} = 3 \text{ if } 50\% \leq \text{SOC} < 75\% \\
&\text{dSOC} = 4 \text{ if } 75\% \leq \text{SOC} \leq 100\%.
\end{aligned}$$

The discrete state of the tank (dtank) is considered to range in {1,2,3} as follows:

$$\begin{aligned}
&\text{dtank} = 1 \text{ if } 0\% \leq \text{tank} < 25\% \\
&\text{dtank} = 2 \text{ if } 25\% \leq \text{tank} < 50\% \\
&\text{dtank} = 3 \text{ if } 50\% \leq \text{tank} \leq 75\% \\
&\text{dtank} = 4 \text{ if } 75\% \leq \text{tank} \leq 100\%.
\end{aligned}$$

The discrete power balance (dpb) is divided in four states where the fraction of the difference between the pp and the power consumption (pc) divided by the pp can be positive high, positive low, negative high and negative low:

$$\begin{aligned}
&\text{dpb} = 1 \text{ if } -1 \leq \text{pb} < -0.5 \\
&\text{dpb} = 2 \text{ if } -0.5 \leq \text{pb} < 0 \\
&\text{dpb} = 3 \text{ if } 0 \leq \text{pb} < 0.5 \\
&\text{dpb} = 4 \text{ if } 0.5 < \text{pb} < 1,
\end{aligned}$$

where pb = (pp−pc)/pp.

The combination of discrete variables' values result in a discrete state space of sixty four states. It must be noted that although Q-learning methods for continuous state variables

and state spaces do exist (Angelidakis & Chalkiadakis, 2015), we plan to study these in our future work.

We assume that the time interval between two consecutive instants is 50 s. At any time instant the agent performs one action for the battery and one action for the desalination unit. The actions which can take place in the battery are two: charging ('C') and discharging ('D'). If there is an energy surplus and the battery is in charging mode, then it absorbs the remaining power. If there is surplus in energy and the battery is in discharging mode then the battery does not charge or discharge. If there is deficit in energy and the battery is in charging mode, the battery does not charge or discharge too. If there is energy deficit and the battery is in discharging mode, the battery offers the power that is needed. The actions that can take place in the desalination unit are two. Stop ('S') and operate at maximum power of 550 W ('P'). If the desalination unit is in operation, the desalination unit consumes 550 W and potable water is added into the tank with a rate of about 100 lt/h. If the desalination unit is in no operation mode the desalination unit consumes 0 W and no water is added in the tank. Thus, the agent at any time instant may decide any combination of two actions.

In order to depict and understand how actions affect the microgrid, and for the simplicity of the presentation, assuming a deterministic environment where the resulting state depends only on the current state and the action performed by the agent, the succession of states may be presented as in Figure 5. In the state where dSOC = 2, dtank = 3 and dpb = 4, if the actions 'C' and 'O' are performed, the agent will move to the state where dSOC = 3, dtank = 4 and dpb = 4. The battery will be charged and simultaneously the desalination unit will add potable water into the tank (surplus of power). If the actions 'D' and 'S' are performed, the agent will remain at the same state. The dynamic load is supplied by the PV power (surplus of the power), the desalination unit is not operating, thus no water is added into the tank, and the battery is not discharged as the dynamic load is supplied directly by the PV source.

Of course, in our case the environment is not that deterministic given that the load and water demands may change unpredictably and the energy produced by the PV may also be affected by external factors (e.g. the weather): These issues have been taken into account in our experimental study.

5.3. Reward

The reward depends on three factors, the SOC of the battery, the amount of the water into the tank and the coverage of the power demand.

The factor of the reward regarding the SOC (r_{SOC}) of the battery in the resulting state is as follows:

$$r_{SOC} = 3.5 \cdot \text{soc\_dt}, \tag{3}$$

where 'soc_dt' is the difference in the rate of the SOC in two consecutive time instants, that is, the resulting and the current one. The value of 3.5 is used to scale r_{SOC} in the range of [−1,1] according to the maximum and minimum value of the soc_dt.

The factor of the reward regarding the percentage of the water into the tank (r_{wat}) in the resulting state is calculated as follows:

$$r_{wat} = 5.5 \cdot \text{pwat\_dt}, \tag{4}$$

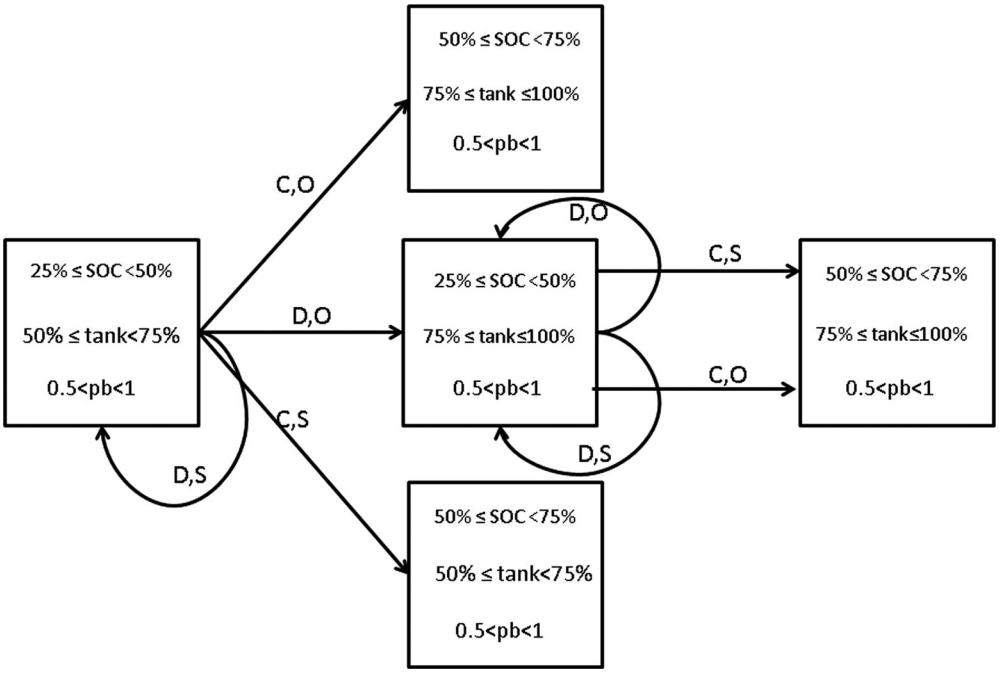

Figure 5. Success of states.

where 'pwat_dt' is the difference in the percentage of the water into the tank between two consecutive time instants, that is, the resulting and the current one. The value of 5.5 is used for scaling the variable r_{wat} in the range of [−1,1].

The reward for the coverage of the energy demand (r_{cov}) by the load is calculated by:

$$r_{cov} = 2 \cdot (1 - l\_per) - 1. \tag{5}$$

The value of *l*_per is the difference between the demanded and the supplied power.

$$l\_per = \frac{(l\_dem - l\_pow)}{l\_dem} \tag{6}$$

where '*l*_dem' is the power demand by the consumers and '*l*_pow' is the power supplied to the consumers. The '*l*_per' is in the range of [0,1] and the transformation of Equation (5) is used in order the 'r_{cov}' to be in the range of [−1,1].

These three quantities, arising from Equations (3), (4) and (5), compose the input vector of the FS that calculates the reward. For each input three MFs are used (Figure 6). The quantization of the inputs in three areas provides enough details in order to cover the range of each input. In addition to that, the rule base remains small which makes the implementation to a microcontroller reachable. The N, Z, P denote for Negative, Zero and Positive.

The FS is zero-order type Sugeno and the outputs of the rules are constant values (Figure 7). The NB, NM, Z, PM and PB denote for Negative Big, Negative Medium, Zero, Positive Medium and Positive Big respectively. These values are selected in the range of [−1,1] which is the desirable range of the reward. Thus, the PB corresponds to '1' where

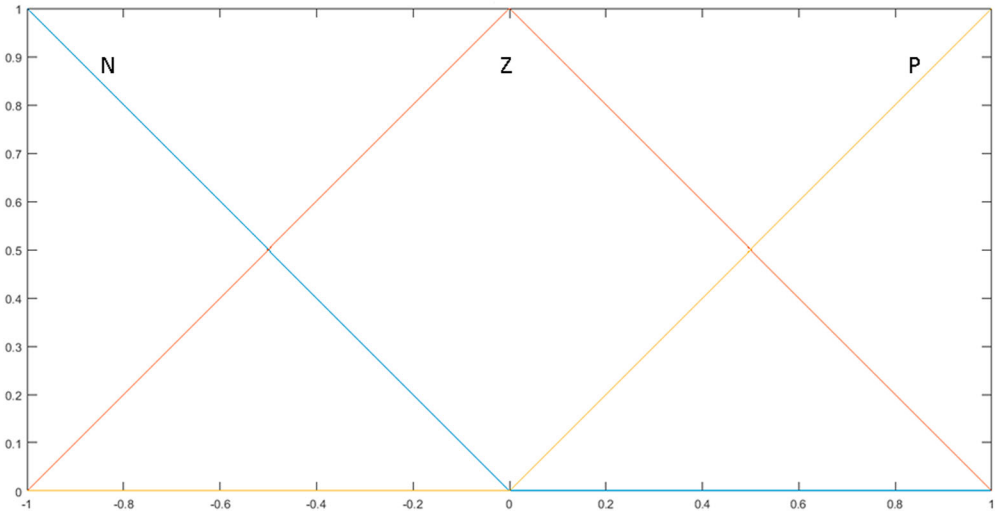

Figure 6. Input MFs.

the reward is maximum, the NB corresponds to '−1' where the reward is minimum and there are three additional values between these limits with equal distances. Five fuzzy singleton sets in the output and three fuzzy sets in each input can provide enough detail to determine the value of the reward signal by using the trial and error method. Additionally, the number of input/output fuzzy sets' parameters that may be optimized remain low. More fuzzy sets increase the number of these parameters. For example, for every triangular membership function the parameters that need to be optimized are three, two parameters for locating the base of the triangle and one parameter for locating the peak. Consequently, for every membership function that is added, three parameters are added to the optimization algorithm.

The main idea is to keep the SOC of the battery and the percentage of the water into the tank (p_wat) at their maximum values and simultaneously, cover the power demand. In case where the SOC of the battery and the percentage of the water into the tank are not at their maximum values, the goal is to cover the power demand and simultaneously to increase the SOC of the battery and the volume of the water into the tank. If this is not possible, the goal is to cover the energy demand. The rule base of the FS consists of 27 rules which are shown in Table 1. The number of the rules is defined by combining the MFs of the inputs, in our case three inputs with three MFs each one. The rule base developed takes into consideration the aforementioned strategy for defining the consequent of each rule. For example the 27th rule has as consequence a reward equal to PB as the load demand is covered and simultaneously the SOC and the volume of the water are rising. The 18th rule has as consequence a reward equal to PM as the load demand is covered, the water volume raises but the SOC of the battery remains at the same level.

The defuzzifier of the system is the weighted average, thus the output of a zero-order FS type Sugeno with 3 inputs and 27 rules can be expressed as:

$$y(x) = \frac{\sum_{l=1}^{27} C^l \prod_{l=1}^{3} \mu_i^l(x_i)}{\sum_{l=1}^{27} \prod_{l=1}^{3} \mu_i^l(x_i)}, \tag{7}$$

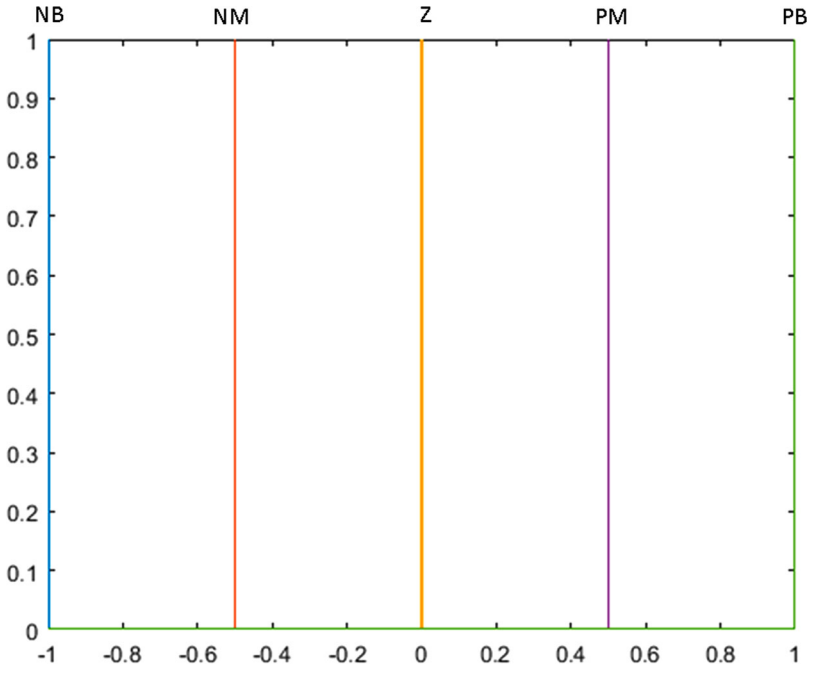

Figure 7. Output membership functions.

Table 1. FS rule base.

Rule	r_{SOC}	r_{wat}	r_{cov}	Reward
1	N	N	N	NB
2	N	N	Z	NB
3	N	N	P	NM
4	N	Z	N	NB
5	N	Z	Z	NB
6	N	Z	P	NM
7	N	P	N	NB
8	N	P	Z	NB
9	N	P	P	Z
10	Z	N	N	NB
11	Z	N	Z	NB
12	Z	N	P	NM
13	Z	Z	N	NB
14	Z	Z	Z	NB
15	Z	Z	P	Z
16	Z	P	N	NB
17	Z	P	Z	NB
18	Z	P	P	PM
19	P	N	N	NB
20	P	N	Z	NB
21	P	N	P	Z
22	P	Z	N	NB
23	P	Z	Z	NB
24	P	Z	P	PM
25	P	P	N	NB
26	P	P	Z	NB
27	P	P	P	PB

where $y(x)$ is the crisp output of the FS (the total reward in our case), $\mu(x_i)$ is the membership degree of the input x_i to fuzzy set A_i^l and C^l is the value of the reward defined in the consequence of each rule.

5.4. Exploration/exploitation strategy

In the beginning, the agent does not know which the best action for each state is. In order to perform high exploration in the beginning of the simulation the algorithm explores for a certain number of rounds based on the action list size. The number of exploration rounds is specified $e_{\text{num}} = a_{\text{num}} * m$, where a_{num} is the number of defined actions and m is a multiplier defined heuristically. In our case the a_{num} equals to 4 and the m is set to 40. This allows to randomly explore all possible actions for a predetermined number of times. After that the controller greedily chooses the action with the best state-action value.

The overall agent cycle is presented in Figure 8. The agent perceives the state of the environment by reading the corresponding sensors and specifies the current state of the microgrid by performing the discretization specified above. According to the value of the e_{num} it decides whether it will explore or exploit. Thus, it decides the action to be performed and perceives the resulting state by acquiring sensors' measurements. It then calculates the reward and finally updates the Q-table according to Equation (2). The learning process continues to the whole extent of the simulation as the Q-Table is

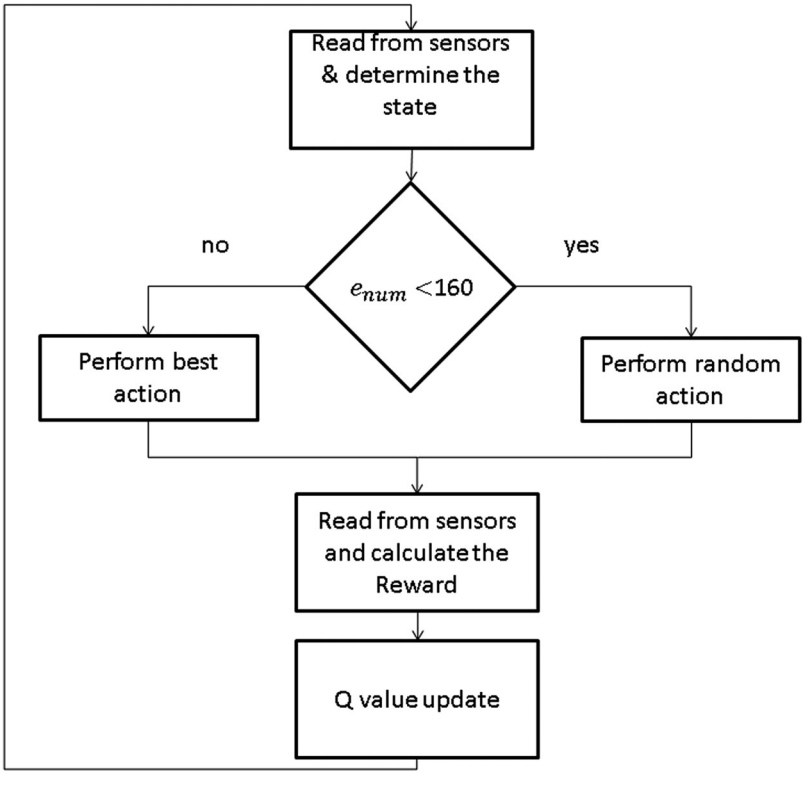

Figure 8. Flowchart of Q-learning agent.

updated both in the exploration and the exploitation stage. This exploration approach was preferred for its simple implementation. New and more sophisticated approaches will be tested in our future work.

6. Simulation results

The simulation time is set to one year with simulation step time of 5 s. The data concerning the pp are data acquired from a 25 kW PV park located in Attica Greece with sample time of 300 s. The pc data set is collected from a four household building with 50 s sampling time. For the water demand a constant consumption of 120 lt/h is assumed, with a variation of 40 lt/h for 12 h/day (during daytime) and 0 lt/h with a positive variation of 20 lt/h for 12 h/day (during night). During day, the variation follows a uniform distribution ranging from [−40, 40] lt/h while during night the variation follows a uniform distribution ranging from [0, 40] lt/h. Figure 9 shows the water demand for a day (24 h).

Figure 10 shows the reward with respect to time. As it is shown, in the beginning of the exploration process the reward has big negative values. After the exploration the reward has more positive values as the electrical demand is covered and in many cases the water in the tank and the SOC are simultaneously rising. The negative values of the reward arise from discharges of the battery in order to cover the power demand and from the high water demand that decreases the water level. Figure 11 shows the difference between the consumed power and the power demand. High positive values of this quantity mean that the agent does not satisfy the power demand. On the other hand, values close to zero show that the agent performs well, and the power demand is covered. In the beginning, there are many cases where the agent does not succeed in covering the power demand. As the agent is in the exploitation stage the majority of the power demand is covered and at the end of the year only 74.77 kWh have not been covered which correspond to only 0.8% of the total power demand. Figure 12 depicts the SOC

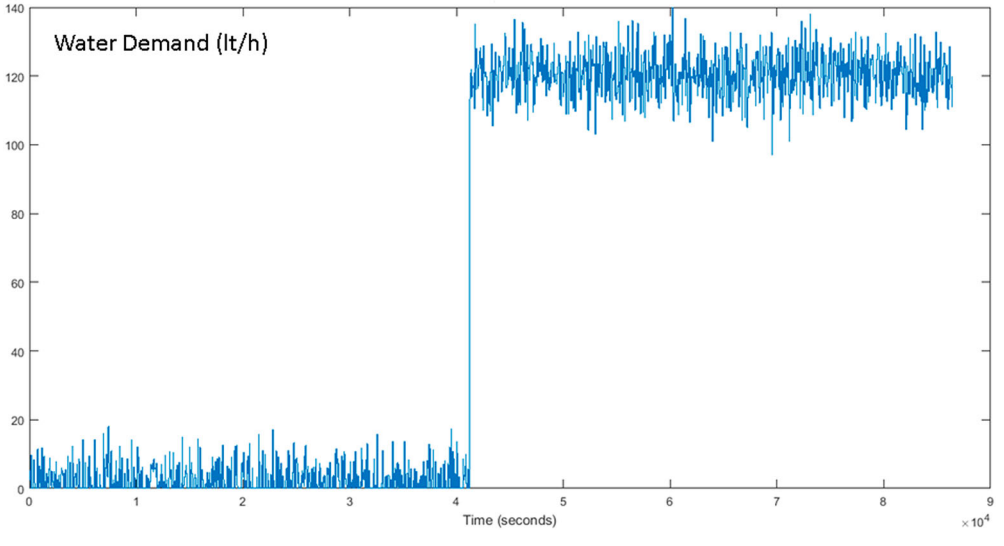

Figure 9. Water demand.

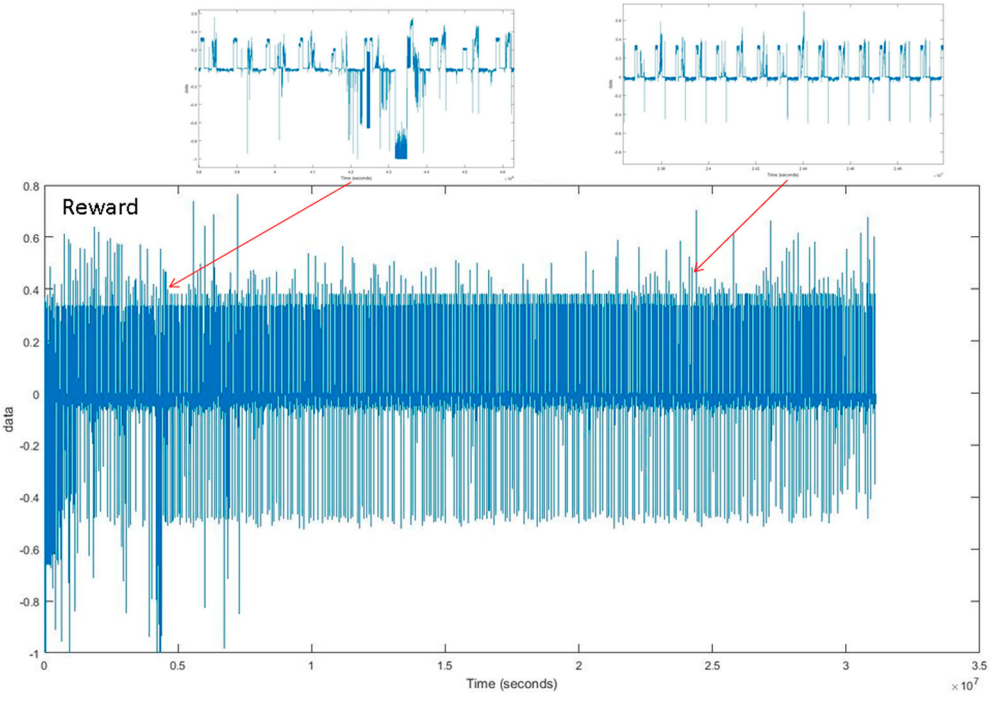

Figure 10. Reward.

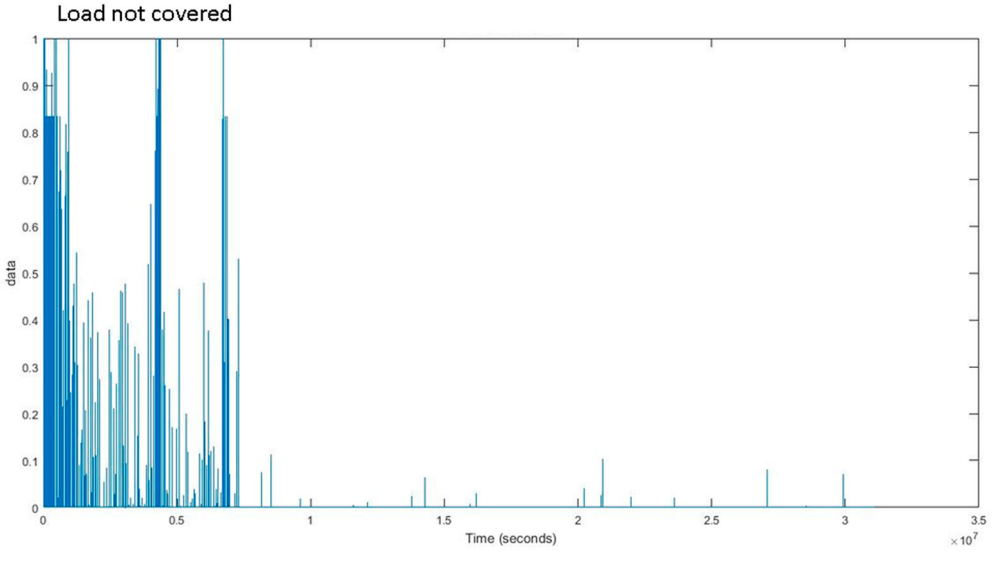

Figure 11. Load coverage.

of the battery. The battery remains charged during the whole year with little fluctuations. It generally remains over 50% charged and in the cases where the SOC is dropped under 50% is due to the high values of the electrical demand and/or the lack of solar power.

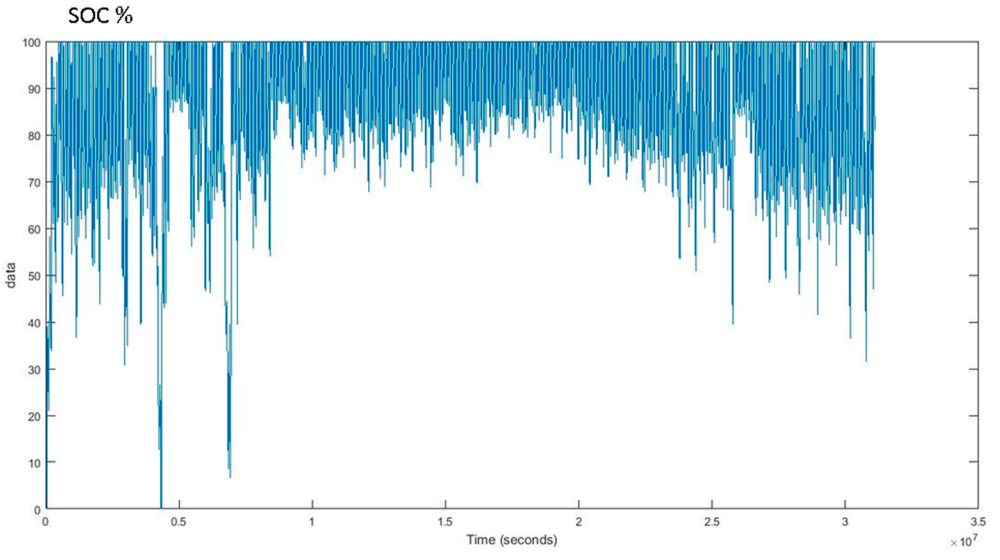

Figure 12. SOC of the battery.

The battery is deeply discharged (under 20%) for only four times during the year. Finally, Figure 13 presents the percentage of the potable water into the tank; it is obvious that the water in the tank remains in high levels during the year (over 70%). The water demand has not been covered for only 1674 lt that corresponds to 0.3% of the total demand.

Table 2 presents efficiency indicators regarding our approach and two others comparative approaches. One with extended exploration phase where the random actions per state has been raised to 400 (the same proposed approach but with extensive

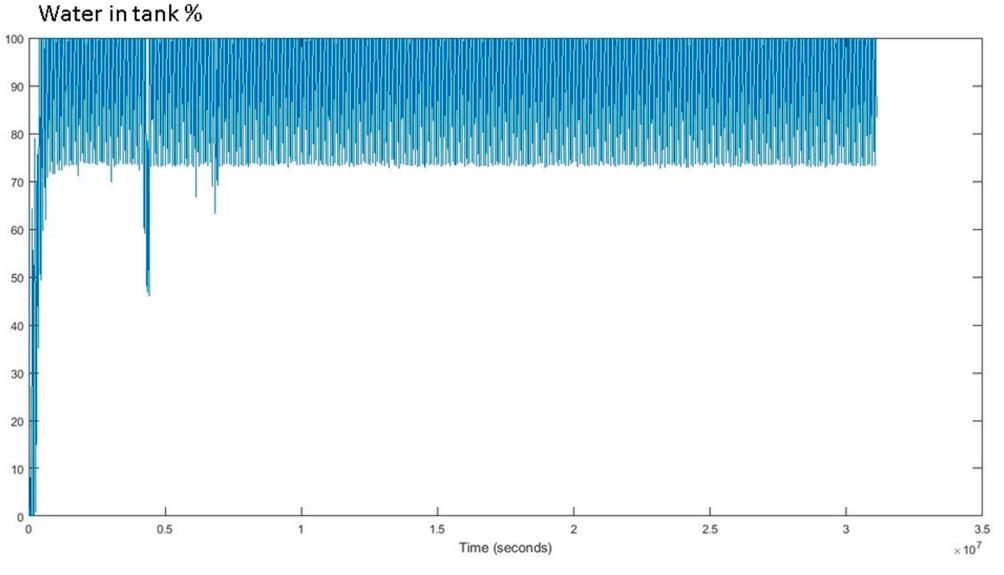

Figure 13. Percentage of water into the tank.

Table 2. Efficiency indicators.

Reinforcement learning approach	Number of not covered (kWh)	Number of litres not covered	Number of deep discharges
Reinforcement learning with mean reward (160 exploration rounds/state)	79.24 (0.8% of the total demand)	2178 (0.4% of the total demand)	5
Reinforcement learning with fuzzy reward (the proposed approach with extensive exploration of 400 rounds/state)	107.4 (1.1% of the total demand)	2854 (0.5% of the total demand)	4
Reinforcement learning with fuzzy reward (160 exploration rounds/state)	74.77 (0.8% of the total demand)	1674 (0.3% of the total demand)	4

exploration) and one with different reward signal where the immediate reward is calculated as follows:

$$\text{reward} = \frac{r_{SOC} + r_{wat} + r_{cov}}{3}. \tag{8}$$

Figures 14, 15 and 16 show the way that the efficiency indicators are changing through time for each case. The simulation runs for the duration of one year. In the beginning of the simulation the values of the indicators are raising quickly as more exploration is performed. It is obvious, that the proposed approach performs better than the other two approaches as the values of the indicators are stabilized more quickly and in lower levels.

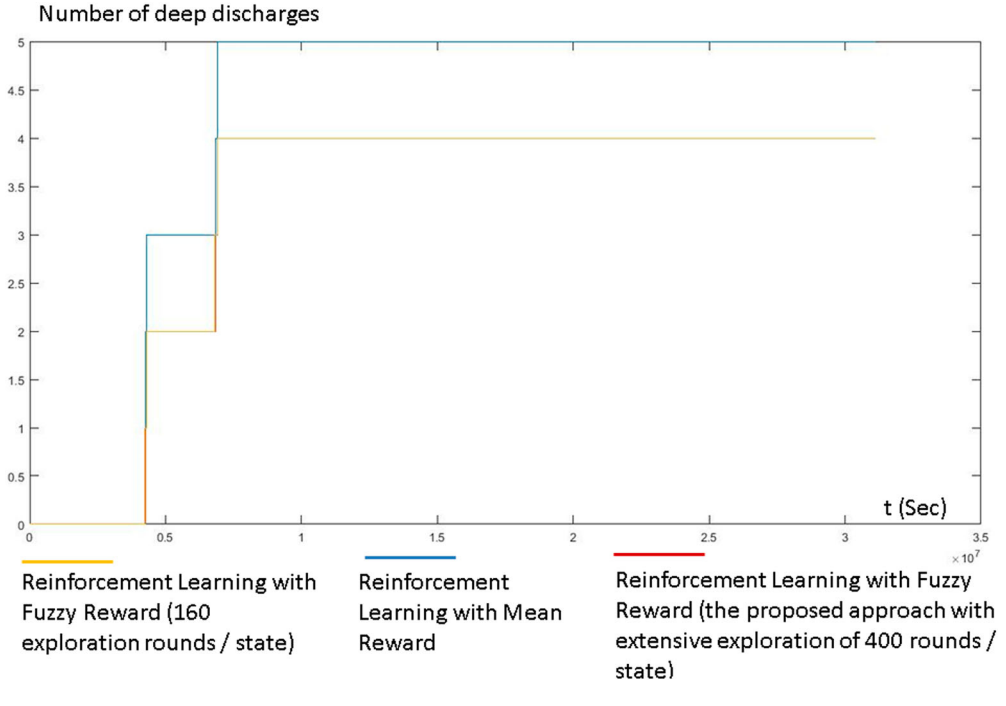

Figure 14. Number of deep discharges.

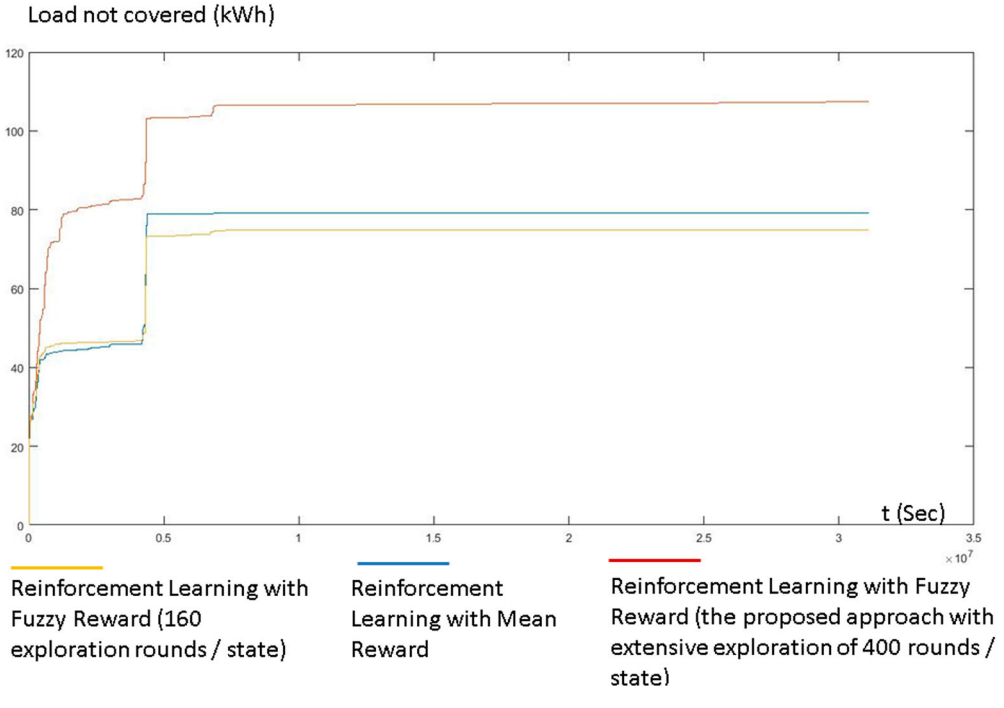

Figure 15. Load not covered.

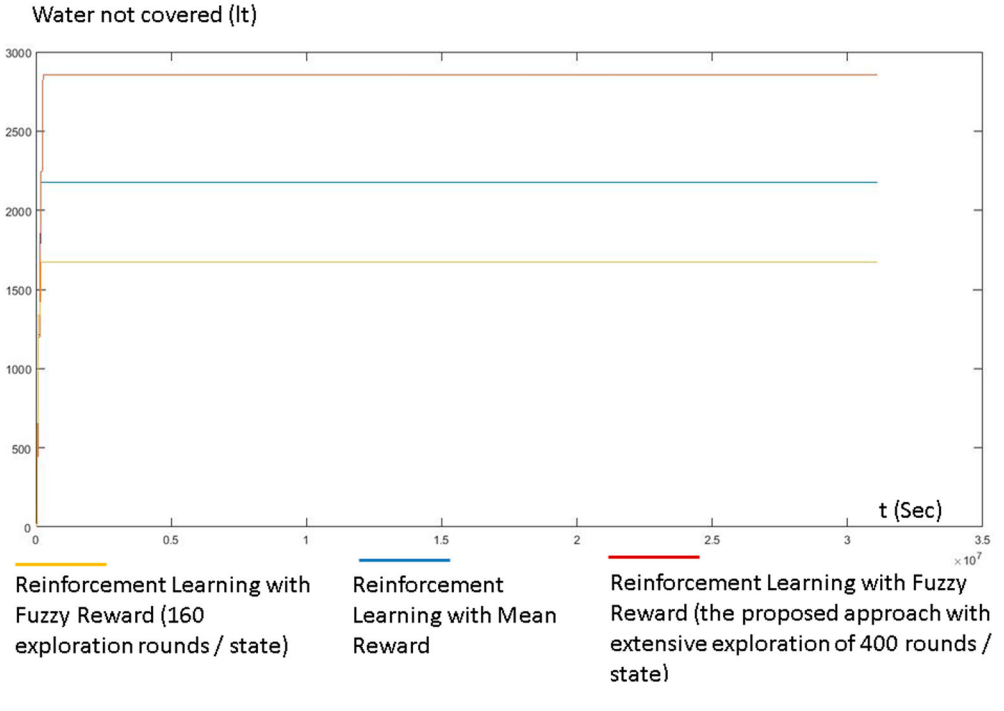

Figure 16. Water not covered.

7. Conclusions and future work

This paper demonstrates the potential of a Q-learning algorithm to control the energy flow between the elements of the solar microgrid. The agent learns by trial and error and demonstrates a good performance. The simulation results highlight the performance of the agent as the water tank remains full and only in very few cases the level of the water drops under 70%, the SOC of the battery remains over 20% with only four deep discharges in a year. The load coverage is in high levels as only the 0.8% of the power demand has not been covered.

In future, it is our aim to perform comparative experiments with learning methods for continuous state spaces. MAS learning methods will also be studied for their effectiveness and efficiency in relation to a single agent solution. Furthermore, we plan to apply these techniques to real-world settings where a microgrid contains more elements, such as a wind turbine, fuel cell, hybrid electrical vehicles and the electrical grid. In a larger scale microgrid, the architecture of fuzzy MAS will be tested in order to increase the performance and the reliability of the microgrid under the uncertainties that consumers and production units introduce.

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

Anastasios I. Dounis http://orcid.org/0000-0002-2204-3955

References

Angelidakis, A., & Chalkiadakis, G. (2015). *Factored MDPs for optimal prosumer decision-making in continuous state spaces*. Paper presented at proceedings of EUMAS15, Athens, Greece.

Barto, A., Sutton, R., & Anderson, C. W. (1983). Neuro-like adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, *13*(5), 834–846.

Berenji, H. R., & Khedkar P. (1992). Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Transactions on Neural Networks*, *3*, 724–740.

Chung, I.-Y., Yoo, C.-H., & Oh, S.-J. (2013). Distributed intelligent microgrid control using multi-agent systems. *Engineering*, *5*, 1–6.

Dounis A. I. (2010). Artificial intelligence for energy conservation in buildings. *Advances in Building Energy Research*, *4*, 267–299.

Foo, Y. S. E., Gooi, H. B., & Chen, S. X. (2014). Multi-agent system for distributed management of microgrids. *IEEE Transactions on Power Systems*, *30*, 24–34.

Huang, T., & Liu, D. (2013). A self-learning scheme for residential energy system control and management. *Neural Computing and Applications*, *22*, 259–269.

Kim, H.-M., Lim, Y., & Kinoshita, T. (2012). An intelligent multiagent system for autonomous microgrid operation. *Energies*, *5*, 3347–3362.

Kofinas, P., Vouros, G., & Dounis, A. I. (2016). *Energy management in solar microgrid via reinforcement learning*. Paper presented at proceedings of the 9th hellenic conference on artificial intelligence SETN '16, Thessaloniki, Greece.

Kurano, M., Yasuda, M., Nakagami, J. I., & Yoshida, Y. (2003). Markov decision processes with fuzzy rewards. *Journal of Nonlinear and Convex Analysis*, *4*(1), 105–116.

Kuznetsova, E., Li, Y.-F., Ruiz, C., Zio, E., Ault, G., & Bell, K. (2013). Reinforcement learning for microgrid management. *Energy*, *59*, 133–146.

Leo, R., Milton, R. S., & Sibi, S. (2014). *Reinforcement learning for optimal energy management of a solar microgrid*. Paper presented at proceedings of IEEE Global Humanitarian Technology conference, Trivandrum, India, pp. 181–186.

Mamdani, E. H., & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, *7*(1), 1–13.

Raju, L., Sankar, S., & Milton, R. S. (2015). Distributed optimization of solar microgrid using multi agent reinforcement learning. *Procedia Computer Science*, *46*, 231–239.

Russel, S., & Norving, P. (1995). *Artificial intelligence: A modern approach*. Upper Saddle River, NJ: Prentice Hall.

Sechilariu, M., Wang, B. C., Locment, F., & Jouglet, A. (2014). DC microgrid power flow optimization by multi-layer supervision control. Design and experimental validation. *Energy Conversion and Management*, *82*, 1–10.

Smith, M. (July 2012). *DOE microgrid initiative overview*. Paper presented at conference on 2012 DOE microgrid workshop, Chicago, Illinois.

Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, *15*, 116–132.

Watkins, C. J. C. H. (1989). *Learning from delayed reinforcement signals* (PhD thesis). University of Cambridge, England.

Zhai, Z., Chen, W., Li, X., & Guo, J. (2009). *A modified average reward reinforcement learning based on fuzzy reward function*. Paper presented at proceedings of the international multiconference of engineers and computer scientists, Vol I, Hong Kong.