IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 5, 2020, accepted November 12, 2020, date of publication November 18, 2020, date of current version December 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3038863

MARLA-SG: Multi-Agent Reinforcement Learning Algorithm for Efficient Demand Response in Smart Grid

SALLY ALADDIN[1], SAMAH EL-TANTAWY[2], MOSTAFA M. FOUDA[1,3], (Senior Member, IEEE), AND ADLY S. TAG ELDIEN[1]

[1]Department of Electrical Engineering, Faculty of Engineering at Shoubra, Benha University, Cairo 11629, Egypt
[2]Department of Engineering Mathematics and Physics, Faculty of Engineering, Cairo University, Giza 12613, Egypt
[3]Department of Electrical and Computer Engineering, College of Science and Engineering, Idaho State University, Pocatello, ID 83209, USA

Corresponding author: Sally Aladdin (sallyaladdin@gmail.com)

ABSTRACT The population is sharply growing in the last decade, resulting in non-potential power requests in dense urban areas, especially with the traditional power grid where the system is not compatible with the infrequent changes. Smart grids have shown strong potential to effectively mitigate and smooth power consumption curves to avoid shortages by adjusting and forecasting the cost function in real-time in response to consumption fluctuations to achieve the desired objectives. The main challenge for the smart grid designers is to reduce the cost and Peak to Average Ratio (PAR) while maintaining the desired satisfaction level. This article presents the development and evaluation of a Multi-Agent Reinforcement Learning Algorithm for efficient demand response in Smart Grid (MARLA-SG). Also, it shows a simple and flexible way of choosing state elements to reduce the possible number of states, regardless of the device type, range of operation, and maximum allowable delay. It also produces a simple way to represent the reward function regardless of the used cost function. SARSA (State-Action-Reward-State-Action) and Q-learning schemes are used and attained PAR reduction of 9.6%, 12.16%, and an average cost reduction of 10.2%, 7.8%, respectively.

INDEX TERMS Smart grid, demand response, reinforcement learning, Q-learning, SARSA (State Action Reward State Action).

I. INTRODUCTION

Traditional power grids are no longer able to deal with the enormous increase in the number of users and the massive load of modern devices, which results in either a power shortage or extensive raise in power prices to force the reduction of users' consumption [1]. Smart grids are mainly used to overcome the common problems of old power grids either by directly controlling the demand from the retailer company, or by changing the cost tariffs to encourage users to rearrange their requests and reduce the consumption during peak hours, or by using both methods together as the smart grid provides bi-directional communication between the retailer and the smart meter. Changing the frequent consumption patterns of users, also called demand response [2], is done either by rearranging devices operating hours, which will of course affect the satisfaction level of the customers [3], [4], or by giving incentives for using chargeable devices during off-peak hours to compensate the heavy load on the grid during peak hours [5], or using different price-based programs to redistribute the consumption during peak and off-peak hours and reduce the gap between them, which results in a more smooth consumption curve [6]–[8]. Price forecasting techniques were introduced in [9]–[12] to efficiently manage the demand, while load and renewable energy forecasting are used to maximize the profit as in [13], [14].

Power utilities offer different demand response programs to the customers so that the customers can either choose to join these programs or not, depending on their needs. Direct load control programs for example provide direct access to the appliances, in which the utility can turn on/off the controllable devices according to the status of the grid [15], while curtailable load programs offer incentives consequence to the reduction in power consumption during peak hours [16]. On the other hand, price-based programs indirectly encourage users to redistribute their consumption as a result of changing

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

prices over the day [17]–[19]. Demand response programs can be achieved either by a predefined model including dynamic programming as in [20], linear programming as in [21], or by using learning methods. A predefined model can produce a good solution in a shorter time as in [22] which achieved a superlinear convergence rate. The drawbacks of the predefined model is that it needs expensive calculations, also, it assumes perfect environment model [23]. The power consumption in households has a stochastic nature which gives the motive to cast aside the predefined models and start thinking of a solution that matches these stochastic fluctuations in demand. Different learning algorithms have been introduced to provide a suitable and effective way to control and improve the performance of the grid [24]–[26].

The learning agent (e.g., smart meter) responds to the different changes in the environment (e.g., power demand). At the start of each time slot, the agent observes the state parameters (e.g., devices' requested range, remaining slots to the end of the job, current delay and cost level, etc.) of the environment, and takes an action (e.g., switch on/off the device) depending on simple calculations or complex or even stochastic calculations, accordingly. It then receives a feedback reward as a measure of the state-action pair (e.g., a satisfaction level or cost or a function of both) for the actions taken, and adjusts its policy until it converges to the optimal mapping from states to optimal actions that maximize the cumulative reward on the long run (e.g., maximizes satisfaction or minimizes cost on the long run depending on the user's needs). This process is done through exploring the reward resulting from different actions (learning process), then exploiting the action that gives a maximum reward in the long run (decision making). Different learning methods have been introduced in [23], including SARSA (State-Action-Reward-State-Action) and Q-learning by which our system has been tested.

Our contribution in this article is twofold. We propose Multi-Agent Reinforcement Learning Algorithm for efficient demand response in Smart Grid (MARLA-SG) which represents the reinforcement learning elements in a manner that decreases the storage capacity required for learning. First, MARLA-SG modifies the state elements to minimize the possible number of states characterizing the environment independent of the device parameters such as the duration or maximum delay, unlike [24] where the device parameter affects directly the number of states, and with a flexible duration for each device throughout the simulation time, without assuming fixed run time as in [25]. Second, MARLA-SG simplifies the calculations, by developing a simple equation to represent the reward function, where desired satisfaction level, cost, and Peak to Average Ratio (PAR) reduction are guaranteed through a comparison between the MARLA-SG based on online ''SARSA'', and offline Q-learning methods versus the system without MARLA-SG.

The rest of the paper is structured as follows. Section II presents relevant research works on smart grid demand side management either by using learning methods or predefined models. Section III contains a detailed description of the problem statement. Our proposed algorithm for multi-agent reinforcement learning in smart grid is proposed in Section IV. The performance evaluation of MARLA-SG is validated in Section V. Finally, the paper concludes in Section VI.

II. RELATED WORK

An online-learning method based on actor-critic is used in [24] to put scheduling for each user to minimize the long-term cost, smart devices are connected with Energy Consumption Controller (ECC), which is responsible for controlling the operation of the devices. The state of each device consists of three terms; remaining slots to complete a certain task of this device, the maximum number of interruptions allowed, and the number of slots since the last call to operate this device. The action was chosen to be whether the device will be operated or delayed in the next slot. The reward is a measure of the minimization in the cost and the Peak to Average Ratio (PAR), resulting in a cost reduction of 28% and 13% reduction in PAR.

The authors in [26], introduced a scheduling plan for devices operation using Q-learning for a single household. The Energy Management System (EMS) is responsible for receiving both requests and target times from the user, and grid signals (e.g., prices at certain times of the day) from the utility company, then the EMS introduces a scheduling plan for devices operation. The technique used in this study considers the following features: the user can cancel a request of an uncompleted task, the time is divided into episodes and not fixed slots, each episode terminates when the requested operation is finished or canceled. The state is chosen to consist of a device index, request time, target time, and priority of the device. The action contains only two cases; the device is ON or OFF at the next state, and the reward contains both the cost of electricity consumption and user satisfaction. User satisfaction is evaluated using the completed/canceled value that is sent by the user to the EMS at the end of the day. One drawback of [26] is that the simulation is done for a single device only, which makes the performance of the system when the number of users/devices increases and its ability to manage the unfixed slot time questionable, also, the simulation did not take into account the problem of minimizing the peak period, so it did not account for the possibility of peak shift if the number of users increases.

At the traditional distributed optimization, a control center at the utility company is responsible for sending the total consumption of all users to one of the smart meters, which sends its load scheduling back to the control center, this process is done consecutively to all users until the system reaches the optimal case as in [27] where each agent tries to redistribute the consumption according to the received prices from the retailer and regardless of the other agents' consumption, while in [22], centralized coordination at each smart meter is responsible for its optimization problem independently. The work considered in [22], does not depend on reinforcement

learning; however, it depends on game theory concepts that can accommodate multiple users as players in a game where each user needs to minimize the cost while maintaining the allocated amount of consumption determined by the utility using a quadratic cost function. A comparison between a traditional distributed optimization that needs centralized coordination and a proposed Newton method was done and resulted in a superlinear convergence rate. The drawback of the scheduled policy in [22], is that the cost is high, as it needs to calculate matrix inversion at each iteration, which results in increasing the cost when the number of users increases.

Joint online learning and pricing algorithm is presented in [28], where the utility sends out the pricing signal to the customers and receives their change in scheduling as a response to these prices. A quadratic cost function that is not known to the utility was used, so the utility needs to make an optimization problem to get the parameters of this quadratic cost function. At the same time, the utility needs to predict an optimum price to be sent to the users at the beginning of each time slot to maximize its profit. The model used the Monte Carlo learning method, and resulted in a logarithmic regret "difference between the cost of the produced algorithm and the optimal offline solution".

Deep reinforcement learning is used in [29] to reach the policy that provides the commerce of energy between microgrids and attained 12.7% reduction in power plant scheduled for the proposed system model. The state is chosen to describe the current battery level, the predicted production of renewable energy, and the forecasted demand. The main drawback is the centralized control of the system.

In the next section, the problem of effectively scheduling power distribution in smart grid is described in detail.

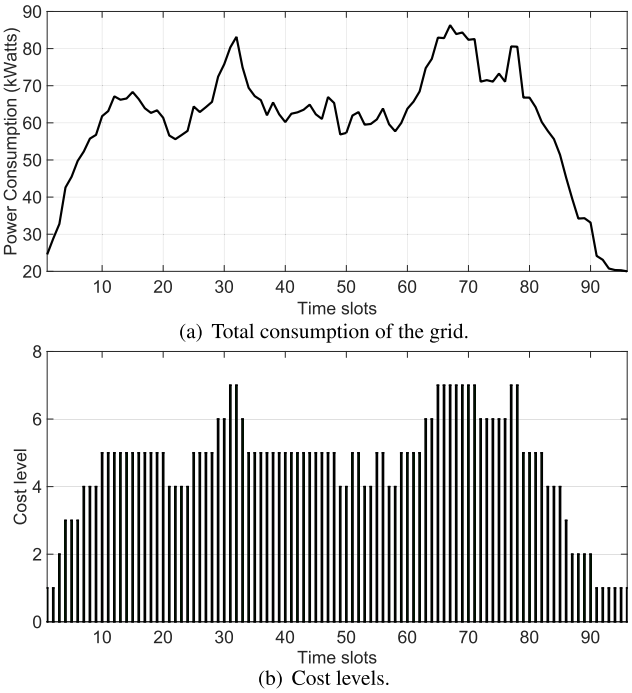
III. PROBLEM FORMULATION

In our system model, the Energy Management System (EMS), connected to the smart meters, is responsible for generating a scheduling technique to be followed by the devices. At the beginning of each time slot, the utility sends power prices to the smart meters inside each household. The utility uses real-time pricing with seven levels, these levels are determined according to the users' total consumption PT at the previous time slot $PT(t-1)$ versus the grid capacity G_c, where t represents the current time slot. Fig. 1 describes the quantization of the total grid consumption into seven cost levels (Cl) using Eq. 1.

$$Cl(t) = \frac{PT(t-1)}{G_c} * 100\%, \tag{1}$$

To decide whether to turn on or off a device, there is some information that should be known, and restrictions to be followed. The information to be included in the state, that describes the current condition of the device is as follows.

1) Requested slots $Rq(t)$: The duration of operating the device. This is received from different devices at the beginning of each time slot.



(a) Total consumption of the grid.
(b) Cost levels.

FIGURE 1. Real time pricing.

2) Remaining slots $Rm(t)$: The number of remaining time slots to complete the requested task.
   - *Set to zero if there is no request.*
   - *Set to the same value of the requested range if the task is requested in the current slot.*
   - *Remains the same if the task was previously requested and the device was delayed in the previous slot.*
   - *Decreased by one if the task was previously requested and the device was ON in the previous slot.*
3) The maximum allowable delay Mdl.
4) The current delay $dl(t)$: The number of delayed slots from the requested task.
   - *Set to zero if there is no request or if the task is requested in the current slot.*
   - *Increased by one if the task was previously requested and the device was delayed in the previous slot.*
   - *Remains the same if the task was previously requested and the device was ON in the previous slot.*
5) The current cost level $Cl(t)$: number between 0 to 7 identified by the utility.
   - *Set to zero if there is no request.*
   - *Calculated as a function of the total grid consumption in the previous slot relative to the grid capacity.*

Choosing the state to include all these data will be very hard and expensive (the curse of dimensionality which grows exponentially, for example, a device "D_{8*10}" with a duration of 8 slots and a maximum delay of 10 slots, the possible number of states will be:

1) First conventional state element Sc^1 = remaining slots. Sc^1 (for D_{8*10}): (from 0 'for no request' to 8 'completed task') = 9 possible states.
2) Second conventional state element Sc^2 = current delay. Sc^2 (for D_{8*10}): (from 0 'no delay' to 10 'maximum delay') = 11 possible states.
3) Third conventional state element Sc^3 = current cost level.
   Sc^3 (for D_{8*10}): from 0 'no request' to 7 'maximum cost') = 8 possible states.

Therefore, the number of states for D_{8*10} = 9*11*8 = 792 states for this single device. Accordingly, there should be a way to reduce the *conventional state elements* ($Sc^{1,2,3}$) by using equations to describe the state uniquely, taking into account the case when there is no request or no delay. In MARLA-SG, the *new state elements* ($Sn^{1,2,3}$) are chosen to be independent of the device parameters. In other words, the number of possible states for each device is fixed regardless of the duration and maximum delay using Eqs. 2, 3, and 4, where n is the household's ID, and d is the device's ID.

$$Sn^1_{n,d}(t) = \begin{cases} \dfrac{Rm_{n,d}(t)}{Rq_{n,d}(t)}, & \text{if } Rq_{n,d}(t) \neq 0. \\ 0, & \text{if } Rq_{n,d}(t) = 0. \end{cases} \tag{2}$$

$$Sn^2_{n,d}(t) = \begin{cases} \dfrac{dl_{n,d}(t)}{Mdl_{n,d}}, & \text{if } Mdl_{n,d} \neq 0. \\ 0, & \text{if } Mdl_{n,d} = 0. \end{cases} \tag{3}$$

$$Sn^3_{n,d}(t) = \begin{cases} Cl(t), & \text{if } Rq_{n,d}(t) \neq 0. \\ 0, & \text{if } Rq_{n,d}(t) = 0. \end{cases} \tag{4}$$

The proposed state elements $\{Sn^1$ and $Sn^2\}$ take real numbers from 0 to 1 are quantized to get discrete values *quantized new state elements* Sqn^1 and Sqn^2 as in Eqs. 5 and 6. While Sn^3 which represents the current cost level is modified to $\{Sqn^3\}$ to get only 3 possible values; (*peak, off-peak, and no request*) as in Eq. 7. As the quantization levels (QL) increase, the states become more accurate but more expensive calculations are required. Our work includes finding the optimal quantization level that decreases the calculation without affecting the performance.

$$Sqn^1_{n,d}(t) = \begin{cases} 0, \text{ if } Rq_{n,d}(t) = 0. \\ \text{QL} - 1, \\ \quad \text{if } (Rq_{n,d}(t) \neq 0) \& \ (Rm_{n,d}(t) = Rq_{n,d}(t)). \\ \text{ceil}((\text{QL} - 2) * \dfrac{Rm_{n,d}(t)}{Rq_{n,d}(t)}), \text{ otherwise.} \end{cases} \tag{5}$$

$$Sqn^2_{n,d}(t) = \begin{cases} 0, \text{ if } Mdl_{n,d} = 0. \\ \text{QL} - 1, \\ \quad \text{if}(Mdl_{n,d} \neq 0) \& (dl_{n,d}(t) = Mdl_{n,d}). \\ \text{ceil}((\text{QL} - 2) * \dfrac{dl_{n,d}(t)}{Mdl_{n,d}}), \text{ otherwise.} \end{cases} \tag{6}$$

$$Sqn^3_{n,d}(t) = \begin{cases} 0, & \text{if } Rq_{n,d}(t) = 0. \\ 1, & \text{if } Cl(t) \leq 4. \\ 2, & \text{if } Cl(t) > 4. \end{cases} \tag{7}$$

After setting the state elements and before taking the action, some restrictions might force the device to behave in a certain manner. To understand these restrictions, first, we have to identify devices-types (DT). The devices under study are divided into three categories: **Must-run**: device operated once it is requested, it can not be delayed. **Non-interruptible**: a device that could be delayed as long as the current delay is less than the maximum allowable delay, but once it is operated, it can not be interrupted. **Interruptible**: a device that could be delayed as long as the current delay is less than the maximum allowable delay. To decide the current action $a_{n,d}(t)$ to be taken for each device (turn ON or OFF, or put in sleep mode), there are some restrictions to be followed:

1) If the device is not requested, it would be illogical to turn it on, in other words:
   If $Rq_{n,d}(t) = 0, \rightarrow a_{n,d}(t) = 0$.
2) If the device did not complete the ordered task, and the current delay equals the maximum allowable delay, the device has to be turned on at the current slot.
   If $Rq_{n,d}(t) > 0 \& dl_{n,d}(t) = Mdl_{n,d}, \rightarrow a_{n,d}(t) = 1$.
3) If the device is non-interruptible, the requested range for it is not zero, and this device has been activated on the previous slot, then the device will be on until it finishes its task. Which means the device is on at the current slot.
   If $Rq_{n,d}(t) > 0 \& DT_{n,d} = 1 \& a_{n,d}(t-1) = 1, \rightarrow a_{n,d}(t) = 1$.
4) If the device is a must run device, then the action follows the requests all time.
   If $Rq_{n,d}(t) > 0 \& DT_{n,d} = 0, \rightarrow a_{n,d}(t) = 1$.
   If $Rq_{n,d}(t) = 0 \& DT_{n,d} = 0, \rightarrow a_{n,d}(t) = 0$.

The reward ($R_{n,d}(t)$) at time (t) is the sum of satisfaction level $Rs_{n,d}(t)$, incentives $Rc_{n,d}(t)$ and total power level $Rp_{n,d}(t)$ as in Eq. 8. The main objective is to minimize the cost while maintaining both satisfaction and power at a reasonable level, without the need to develop a complex equation that describes the reward.

$$R_{n,d}(t) = \sum_{n=1}^{N} \sum_{d=1}^{D} Rs_{n,d}(t) + Rc_{n,d}(t) + Rp_{n,d}(t) \tag{8}$$

Satisfaction level ($SF_{n,d}(t)$) of a user (n) and device (d) at a time slot (t) is a measure of the device's current state. $SF_{n,d}(t)$ is a function of both the remaining slots and the current delay as in Eq. 9, where, at any time slot (t), $(Rm_{n,d}(t) \leq Rq_{n,d}(t)) \& (dl_{n,d}(t) \leq Mdl_{n,d})$.
Accordingly, $0 \leq (Rm_{n,d}(t)/Rq_{n,d}(t)) * (dl_{n,d}(t)/Mdl_{n,d}) \leq 1$

$$SF_{n,d}(t) = \begin{cases} 1, \text{ if } Mdl_{n,d} = 0 \text{ or } Rq_{n,d}(t) = 0. \\ 1 - \dfrac{Rm_{n,d}(t)}{Rq_{n,d}(t)} * \dfrac{dl_{n,d}(t)}{Mdl_{n,d}}, \text{ otherwise.} \end{cases} \tag{9}$$

The best satisfaction level ($SF_{n,d}(t) = 1$) occurs when either ($Rm_{n,d}(t)/Rq_{n,d}(t)$) or ($dl_{n,d}(t)/Mdl_{n,d}$) or both equals zero, which happens in the following cases:

1) $Rq_{n,d}(t) = 0$, if no request ordered for the current device.
2) $Rm_{n,d}(t)/Rq_{n,d}(t) = 0$, if the requested task is completed "$(Rm_{n,d}(t) = 0)\&(Rq_{n,d}(t) \neq 0)$"
3) $Mdl_{n,d} = 0$, if the device is a must-run device.
4) $(dl_{n,d}(t)/Mdl_{n,d}) = 0$ if the device has not been delayed since its request "$(dl_{n,d}(t) = 0)\&(Mdl_{n,d} \neq 0)$"

The worst satisfaction level ($SF_{n,d}(t) = 0$) occurs when both ($Rm_{n,d}(t)/Rq_{n,d}(t)$) and ($dl_{n,d}(t)/Mdl_{n,d}$) equal 1, which happens in the following cases:

- $(Rm_{n,d}(t)/Rq_{n,d}(t)) = 1$ if the requested task for the device under study has not started yet.
- $(dl_{n,d}(t)/Mdl_{n,d}) = 1$ if the requested task is delayed until it reached its maximum delay limit.

Otherwise, the satisfaction level ($SF_{n,d}(t)$) takes values between 0 and 1, which happens in the following cases:

- The requested task for the device under study has begun but not yet finished ($Rm_{n,d}(t) < Rq_{n,d}(t)$).
- The requested task has been delayed but did not reach its maximum limit yet ($dl_{n,d}(t) < Mdl_{n,d}$).

Since not requesting the device is better than reaching its maximum delay without completing the operation; an offset $-W_d/2$ is added to the reward's satisfaction term after scaling it to ($W_d * SF_{n,d}(t)$) to get positive and negative values representing the current state as in Eq. 10, where W_d is a weighing function calculated according to Eq. 11, and $CLVL$ is the maximum number of price levels.

$$Rs_{n,d}(t) = \begin{cases} 1, \text{ if } Mdl_{n,d} \text{ or } Rq_{n,d}(t) = 0. \\ W_d * SF_{n,d}(t) - \dfrac{W_d}{2}, \text{ otherwise.} \end{cases} \tag{10}$$

$$W_d = min(Mdl_{n,d}, CLVL), \tag{11}$$

The second term affecting the reward value is the incentive $Rc_{n,d}(t)$ which is a function of the current cost level $Cl(t)$ and the current action $a_{n,d}(t)$ as in Eq. 12, where $\sim a_{n,d}(t)$ is the ones' complement of $a_{n,d}(t)$.

$$Rc_{n,d}(t) = \begin{cases} a_{n,d}(t)[Rc_{n,d}(t-1)+1] \\ +\sim a_{n,d}(t)[Rc_{n,d}(t-1)-1], & \text{if } Cl(t) < 4. \\ \sim a_{n,d}(t)[Rc_{n,d}(t-1)+1] \\ +a_{n,d}(t)[Rc_{n,d}(t-1)-1], & \text{if } Cl(t) > 4. \\ Rc_{n,d}(t-1), & \text{if } Cl(t) = 4. \end{cases} \tag{12}$$

In order to avoid creating new peaks, the total power level $Rp_{n,d}(t)$ at each time slot is chosen to be the final term affecting the reward value as in Eq. 13. If at any time slot t, the total power of the active devices ($PT(t)$) exceeded a predefined limit *(Gird Capacity G_c)*, the total power level $Rp_{n,d}(t)$ will be set to a large negative value in order to avoid the re-occurrence of the current state. $PT(t)$ is calculated using Eq. 14, where $P_{n,d}(t)$ is the power of device d requested by customer n at time slot t.

$$Rp_{n,d}(t) = \begin{cases} 0, & \text{if } Rq_{n,d}(t) = 0 \text{ or } PT(t) < G_c. \\ -CLVL, & \text{otherwise.} \end{cases} \tag{13}$$

$$PT(t) = \sum_{n=1}^{N} \sum_{d=1}^{D} P_{n,d}(t) * a_{n,d}(t). \tag{14}$$

IV. PROPOSED ALGORITHM

Reinforcement learning (RL) is a goal-based problem, where an agent learns how to control an environment using rewards. Both SARSA and Q-learning are popular reinforcement learning methods. SARSA and Q-learning do not need prior knowledge of the environment. They depend on mapping situations to actions to achieve the maximum reward. This is done throughout exploration and exploitation; in exploration, the reinforcement learning agent tries different actions to determine which action will cause the maximum reward, while in exploitation the agent chooses the action which maximizes the reward [23]. To balance between exploration and exploitation either ϵ-greedy or softmax are used. In ϵ-greedy, the agent chooses randomly between different actions regardless of the estimated reward values, while softmax gives a high probability to the actions with greater estimated reward. In MARLA-SG, a combination between ϵ-greedy and softmax is used to get the best performance (ϵ-softmax) [30]. At exploration time, the next action is chosen randomly like ϵ-greedy but with probability calculated using Q-value as in Eq. 15. Time is divided into steps called time slots. At each time slot (t), the agent receives the current state "$Sqn(t) \in S$", where S represents all possible states that describe the environment, and takes an action "$a(t) \in A_s(t)$", where $A_s(t)$ represents all actions that can be taken in state $Sqn(t)$. At next state $Sqn(t+1)$, the agent receives a numerical reward $R(t+1)$. Note that the reward $R(t+1)$ results from choosing action $a(t)$ at state $Sqn(t)$, and it is independent of the action chosen at state $Sqn(t+1)$. The agent's policy π associates each state $Sqn(t)$ with one action $a(t)$.

$$Prob(a_i(t)) = \frac{Q(Sqn(t), a_i(t))}{\sum_{i=1}^{M} Q(Sqn(t), a_i(t))} * 100\%, \tag{15}$$

where, M represents the number of possible actions at state $Sqn(t)$.

The proposed learning approach in MARLA-SG is either based on SARSA as in Algorithm 1, or Q-learning as in Algorithm 2. The description of the proposed steps is as follows:

- At the beginning of each time slot t, the Energy Management System (EMS) agent receives two operating signals; the first is sent by the devices to identify their requests $Rq_{n,d}(t)$, and the second is sent by the utility to determine the power price level $Cl(t)$ at the current time slot. Using the predefined devices' parameters (i.e. maximum allowable delay and devices' power), the agent

constructs the state matrix $Sqn(t)$ that characterizes the current status.
- The agent chooses a current action $a(t)$ using ϵ-softmax [30].
- The agent calculates the reward $R(t+1)$.
- Initialize the state-value function $Q(Sqn(t), a(t))$ at the first visit of only the state we pass through.
- The agent determines the next state $Sqn(t+1)$.
- The agent chooses the next action a_{t+1} based on "explore-Versus-exploit" probability. The exploration probability (ϵ) is a function of the number of state visits $V_{Sqn(t)}$ as in Eq. 16.
- To update the state-value function $Q(Sqn(t), a(t))$, we use either SARSA as in Eq. 17, or Q-learning as in Eq. 18. The step size parameter $\alpha(t)$ depends on the number of state-action pair visits $V_{Sqn(t),a(t)}$ as in Eq. 19.

Note: The number of state visits $V_{Sqn(t)}$ is initially set to zero for all states, and incremented every time the agent visits a certain state. As the number of visits increases, the agent will have a pre-knowledge of the best action to follow at the current state. So the probability of exploration will decrease and the probability of exploitation will increase. Similarly, the number of state-action pair visits $V_{Sqn(t),a(t)}$ is initially set to zero, and incremented each time the agent passes through a state and chooses a certain action.

$$\epsilon(t) = exp^{(\frac{-(V_{Sqn(t)})^2}{300})} \tag{16}$$

$$Q_{new}(Sqn(t), a(t)) = (1-\alpha)Q_{old}(Sqn(t), a(t)) + \alpha[R(t+1) + \gamma(Q(Sqn(t+1), a(t+1))], \tag{17}$$

$$Q_{new}(Sqn(t), a(t)) = (1-\alpha)Q_{old}(Sqn(t), a(t)) + \alpha[R(t+1) + \gamma max_a(Q(Sqn(t+1), a)], \tag{18}$$

$$\alpha(t) = exp^{(\frac{-(V_{Sqn(t),a(t)})^2}{3})}. \tag{19}$$

V. PERFORMANCE EVALUATION

The section starts by introducing the system model used in our experiments. Then we find the optimum number of quantization levels for single-agent reinforcement learning algorithm. After that, we evaluate the performance of the proposed Multi-Agent Reinforcement Learning Algorithm for efficient demand response in Smart Grid (MARLA-SG) using SARSA or Q-learning. Finally, we introduce a comparison between single-agent learning and various multi-agent learning schemes.

A. SYSTEM MODEL

Despite the fact that the proposed algorithm does not need a prior knowledge of the environment, as neither the offline Q-learning nor the online SARSA schemes need transition matrix to provide a scheduling technique, it was a must to develop a model in order to test the performance of the proposed scheme.

Our system model is shown in Fig. 2. In our system model, each user has a group of different appliances; controllable devices, divided into interruptible (e.g., Air Conditioner, Water Heater, and Electric Vehicle (EV) Battery), non-interruptible devices (e.g., Washing Machine, Dish Washer, Microwave), and must-run devices (e.g., Television (TV)). The users' requests follow a sum of multiple normal distributions with different means and standard deviations.

Algorithm 1 MARLA-SG Using SARSA

while time $t \leq T$ **do**
  initialize: $Cl(t)$
  while customer $n \leq N$ **do**
    while device $d \leq D$ **do**
      initialize: $Rq_{n,d}(t)$, $Mdl_{n,d}$, $P_{n,d}(t)$
      if time $t = 1$ **then**
        $Rm_{n,d}(t) = Rq_{n,d}(t)$
        $dl_{n,d}(t) = 0$
        if $Rq_{n,d}(t) > 0$ **then**
          $a_{n,d}(t) = 1$
        else
          $a_{n,d}(t) = 0$
        endif
      else
        observe: $Rm_{n,d}(t)$, $dl_{n,d}(t)$
      endif
    endwhile
  endwhile
  set: $Sqn^1(t)$, $Sqn^2(t)$, $Sqn^3(t) \leftarrow$ Eq. 5, 6, 7.
  update: state visits $V_{Sqn(t)}$
  $\epsilon = exp^{(\frac{-(V_{Sqn(t)})^2}{300})}$
  update state-action pair visits $V_{Sqn(t),a(t)}$
  $\alpha = exp^{(\frac{-(V_{Sqn(t),a(t)})^2}{3})}$
  calculate satisfaction term: $Rs(t+1) \leftarrow$ Eq. 10
  calculate incentives term: $Rc(t+1) \leftarrow$ Eq. 12
  calculate total power: $PT(t) \leftarrow$ Eq. 14
  calculate power level term: $Rp(t+1) \leftarrow$ Eq. 13
  calculate reward: $R(t+1) \leftarrow$ Eq. 8
  if visits $(V_{Sqn(t),a(t)}) = 1$ **then**
    initialize $Q(Sqn(t), a(t)) = 0$
  endif
  observe: $Sqn(t+1)$
  choose: $a(t+1)$ using ϵ-softmax method.
  $Q_{new}(Sqn(t), a(t)) \leftarrow (1-\alpha)Q_{old}(Sqn(t), a(t)) + \alpha[R(t+1) + \gamma Q(Sqn(t+1), a(t+1))]$
  update SARSA-table
  $Sqn(t) \leftarrow Sqn(t+1)$
  $a(t) \leftarrow a(t+1)$
endwhile

Algorithm 2 MARLA-SG Using Q-Learning

while time $t \leq T$ **do**
  initialize: $Cl(t)$
  while customer $n \leq N$ **do**
    while device $d \leq D$ **do**
      initialize: $Rq_{n,d}(t)$, $Mdl_{n,d}$, $P_{n,d}(t)$
      if time $t = 1$ **then**
        $Rm_{n,d}(t) = Rq_{n,d}(t)$
        $dl_{n,d}(t) = 0$
        if $Rq_{n,d}(t) > 0$ **then**
          $a_{n,d}(t) = 1$
        else
          $a_{n,d}(t) = 0$
        endif
      else
        observe: $Rm_{n,d}(t)$, $dl_{n,d}(t)$
      endif
    endwhile
  endwhile
  set: $Sqn^1(t)$, $Sqn^2(t)$, $Sqn^3(t) \leftarrow$ Eq. 5, 6, 7.
  choose: $a(t)$ using ϵ-softmax method.
  update: state visits $V_{Sqn(t)}$
  $\epsilon = exp^{(\frac{-(V_{Sqn(t)})^2}{300})}$
  update state-action pair visits $V_{Sqn(t),a(t)}$
  $\alpha = exp^{(\frac{-(V_{Sqn(t),a(t)})^2}{3})}$
  calculate satisfaction term: $Rs(t+1) \leftarrow$ Eq. 10
  calculate incentives term: $Rc(t+1) \leftarrow$ Eq. 12
  calculate total power: $PT(t) \leftarrow$ Eq. 14
  calculate power level term: $Rp(t+1) \leftarrow$ Eq. 13
  calculate reward: $R(t+1) \leftarrow$ Eq. 8
  if visits $(V_{Sqn(t),a(t)}) = 1$ **then**
    initialize $Q(Sqn(t), a(t)) = 0$
  endif
  observe: $Sqn(t+1)$
  choose: $a(t+1)$ using ϵ-softmax method.
  $Q_{new}(Sqn(t), a(t)) \leftarrow (1-\alpha)Q_{old}(Sqn(t), a(t)) + \alpha[R(t+1) + \gamma max_a Q(Sqn(t+1), a)]$
  update Q-table
  $Sqn(t) \leftarrow Sqn(t+1)$
endwhile

The user is not aware of his future consumption as well as the price of energy [24]. Negotiation is not allowed in the performance which means that the utility sends out the cost at the beginning of each time slot and commits to this price till the next time slot [28]. A survey from 100 Egyptian users was collected by us. The users were asked to indicate the most likely hours when they turn on some of their home appliances, and the duration of it, plus the maximum number of operations per day for those devices, and the maximum allowable delay for each controllable device. The collected data is used to develop a realistic case study model to test the

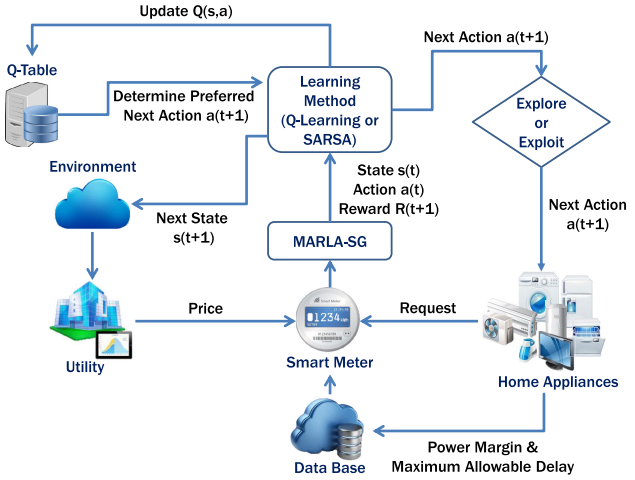


FIGURE 2. System model.

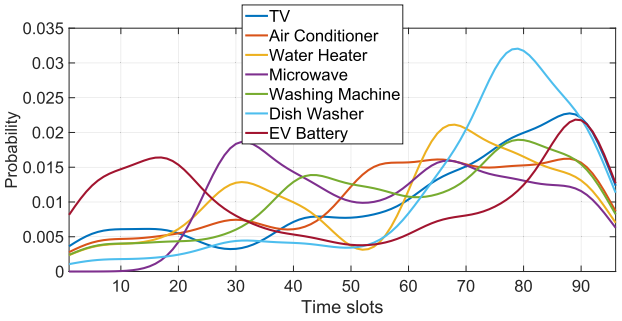


FIGURE 3. Request probability for the scheduled appliances.

performance of the proposed scheme, and it only provides the probability of using the device throughout the day. However, the performance will not be affected by the change of the used model, As the number of states does not depend on the device parameters. In our simulation, the day is divided into 96-time slots. Each time slot represents 15 minutes starting with 12 AM (time slot 1). For simplicity, the survey is taken as if the day is divided into only eight periods of time. Each period is represented as a normal probability distribution curve using predefined mean and standard deviation multiplied by the percentage of user's requests at that time, as in Table 1. Each mean represents the center of the current period in time slots, while the standard deviation represents a quarter the width of this margin in time slots to make sure that 95% of the requests are within this margin. Fig. 3 shows the request probability of the appliances.

Table 2 contains the minimum and maximum limit for the power consumed by each device in watts per hour according to the Ministry of Electricity & Renewable Energy in Egypt [31]. The maximum operating times per day, minimum duration, and maximum duration are set according to the collected survey data.

Table 3(a), illustrates the traditional prices in Egyptian pounds (EGP) taken according to the Ministry of Electricity & Renewable Energy in Egypt [31] in March 2020, where the prices are set regardless of peak hours. The prices only

TABLE 1. The Percent of Usage According to the Collected Data.

	Period 1	Period 2	Period 3	Period 4	Period 5	Period 6	Period 7	Period 8
Starts at	00:00	03:00	06:00	09:00	12:00	15:00	18:00	21:00
Ends at	03:00	06:00	09:00	12:00	15:00	18:00	21:00	00:00
μ (slot Number)	7	19	31	43	55	67	79	91
σ (in time slots)	3	3	3	3	3	3	3	3
Request Probability (%)								
TV	20.7	20.7	6.9	27.6	24.1	44.8	62.1	82.8
Air Conditioner	15.8	15.8	26.3	15.8	52.6	52.6	47.4	57.9
Water Heater	7.5	7.5	26.1	17.4	0	43.5	30.4	26.1
Microwave	0	0	62.5	37.5	25	50	37.5	37.5
Washing Machine	8.6	8.6	10.3	31	24.1	20.7	41.4	34.5
Dishwasher	2.1	2.1	5.4	4.6	3.2	15.8	40.2	26.1
EV Battery	24	30	12	8.5	5.4	13.2	15.6	44.6

TABLE 2. System Model Elements.

Device	Min Power (Watt)	Max Power (Watt)	Max. Operating Times **"MOT"**	Min. Duration **"MND"** (time slots)	Max. Duration **"MXD"** (time slots)	Max allowable Delay **"Mdl"** (time slots)	Device Type **"DT"** "Must-run (0) Non-interruptible(1) interruptible(2)"
TV	200	325	8	2	12	0	0
Air Conditioner	1100	2400	6	4	12	3	2
Water Heater	1000	5000	3	1	12	4	2
Microwave	2000	4000	3	1	2	5	1
Washing Machine	700	700	3	2	12	10	1
Dishwasher	1200	1500	3	2	6	14	1
EV Battery	2000	2000	3	4	12	18	2

depend on the total consumption for each user per month, while in MARLA-SG, the prices are set according to the total consumption of the grid at previous time slot $PT(t-1)$ versus the grid capacity G_c as in Table 3(b).

B. OPTIMAL NUMBER OF QUANTIZATION LEVELS FOR SINGLE-AGENT REINFORCEMENT LEARNING ALGORITHM

To balance between minimizing the number of states and efficiently characterizing the environment, and to identify the best quantization level for the state elements, the simulation compares between the *Single-Agent*'s performance using the analog state elements $\{Sn^1, Sn^2$ and $Sn^3\}$ introduced in Eqs. 2, 3, and 4 versus the *Single-Agent*'s performance using the modified state elements $\{Sqn^1, Sqn^2$ and $Sqn^3\}$ introduced in Eqs. 5, 6, and 7 with three different quantization levels (QL= 4, 6, and 8). **Note:** *To avoid scaling problem for a large number of states in case of high quantization levels, the simulation is done for Air-conditioners of 6 households*. Fig. 4(a) shows that the *Single-Agent*'s simulation, using Q-learning with different quantization levels, convergences to the same value "delay = 5-time slots" with zero Root Mean Square Error (RMSE), only with more calculations and slower performance as "QL" increases. The simulation with the original analog state elements $\{Sn^1, Sn^2$, and $Sn^3\}$ $\rightarrow$ "QL=0" converges at time slot=30 after a series of large fluctuations, which is the slowest performance. While the system using modified state elements $\{Sqn^1, Sqn^2$ and $Sqn^3\}$ and QL=4 converges to the same delay value, at time slot = 21, with smoother fluctuations. Both QL=6 and QL=8 converges to the same delay faster than the analog state elements and slower than QL=4. which means that the modified state elements $\{Sqn^1, Sqn^2$ and $Sqn^3\}$ with QL=4, are more than enough to uniquely represent each state in the environment. Similarly Fig. 4(b), shows that the *Single-Agent*'s simulation using SARSA with the modified state elements $\{Sqn^1, Sqn^2$ and $Sqn^3\}$ and QL=4 convergences to the same value "delay = 2 time slots" as the simulation with the original analog state elements $\{Sn^1, Sn^2$ and $Sn^3\}$ $\rightarrow$ "QL=0", but much faster. As the number of quantization levels (QL=8) increases, the fluctuations at the start of the convergence curve increase, because the number of states representing the environment increases. So the agent takes longer time to explore because the probability to explore depends on the number of state-visits (i.e., in exploration, the agent tries different non-greedy actions, so the reward might not be the best required in this case, but it is the only way to identify the best action for each state on the long run).

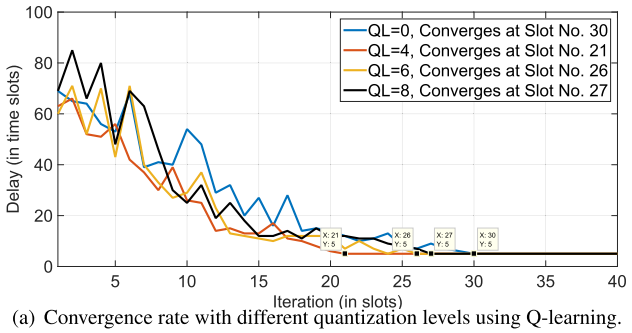
TABLE 3. Traditional Pricing Versus Proposed Pricing Scheme.

(a) Prices according to the Ministry of Electricity and Renewable Energy in Egypt, March, 2020.

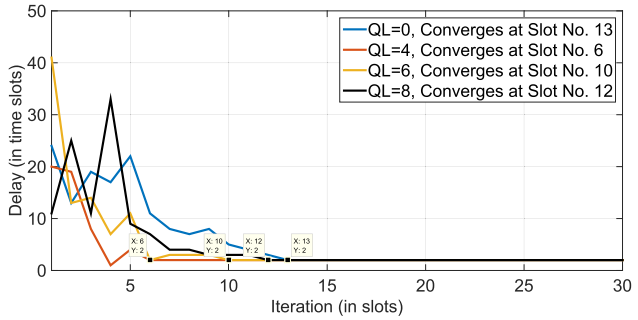
Price in (Egyptian Pounds)	Consumption (Kwh/month) From	To
0.22	0	50
0.3	51	100
0.36	101	200
0.7	201	350
0.9	351	650
1.35	651	1000
1.45	More than 1001	
Ministry of Electricity Pricing Scheme		

(b) Prices according to MARLA-SG.

Cl(t)= 100*PT(t-1)/G_c (%) From	To	Price Level	Price in (Egyptian Pounds)
0	15	1	0.22
15	30	2	0.3
30	45	3	0.36
45	60	4	0.7
60	75	5	0.9
75	90	6	1.35
90	100	7	1.45
MARLA-SG Pricing Scheme			



(a) Convergence rate with different quantization levels using Q-learning.



(b) Convergence rate with different quantization levels using SARSA.

FIGURE 4. Convergence rates using different learning methods.

Fig. 5(a) and Fig. 5(b) shows the PAR reduction by 28.2% and 28.14% of a single user's consumption for a random day using Q-learning and SARSA, respectively. The proposed Single-Agent Reinforcement Learning Algorithm using both learning methods results in smoothing the consumption curve. It is obvious that *Single-Agent*'s simulation, using SARSA is faster than using Q-learning but at the expense of the PAR reduction, and this will be manifested as the number of customers increase in the proposed MARLA-SG. **Note:** The consumption peak is not affected by changing the number of quantization levels, because the number of quantization levels (QL) only affects the possible number of states representing the environment, which in turn changes the time required to converge, but eventually, the system with different quantization levels converge to the same performance as in Figs. 4() and 4(b), only for longer time with high quantization levels.

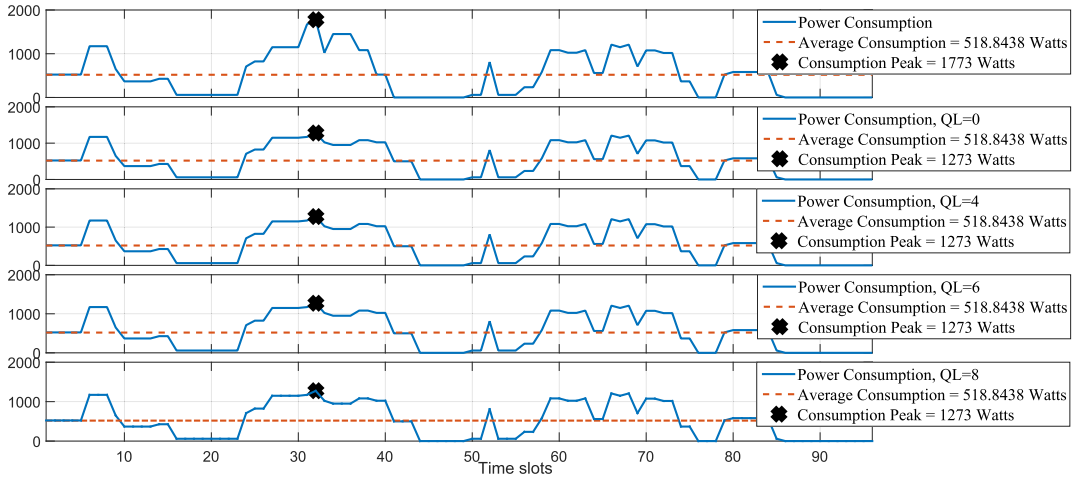
C. MULTI-AGENT REINFORCEMENT LEARNING ALGORITHM

The simulation is done for a grid consisting of 100 users (i.e., $N = 100$), with time slot $=$ 15 minutes, the number of appliances D for each user is 7 (6 controllable devices plus a single must-run device), Maximum cost level $CLVL = 7$, and discount rate $\gamma = 0.2$. Both SARSA and Q-learning were used to test the performance of MARLA-SG for a learning simulation time $T = 10$ months for each learning method.

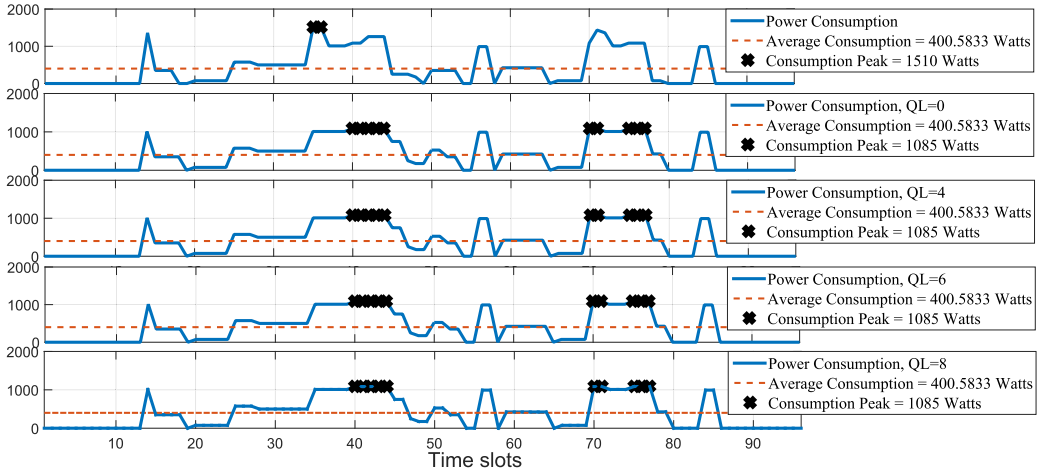
Fig. 6 shows that the convergence rate of MARLA-SG using SARSA and Q-learning methods reaches its stable level before completing the second month. The convergence rate of MARLA-SG using SARSA in Fig. 6(b) is faster than that using Q-learning in Fig. 6(a) with almost 3 days. Fig. 7 indicates the PAR (Peak to Average Ratio) before and after MARLA-SG for a simulation time of 10 months.

In Fig. 7(a), MARLA-SG using Q-learning reduced the peak consumption from 93.6 to 82.2 kWatts, while maintaining the average consumption of almost 41.5 kWatts, resulting in PAR reduction of 12.16%. In Fig. 7(b), MARLA-SG using SARSA reduced the peak consumption from 88.2 to 79.7 kWatts, while maintaining the average consumption of almost 39 kWatts, resulting in PAR reduction of 9.6%.

The power consumption cost $Cp_n(t)$ for user n is calculated using Eq. 20, and Fig. 8 shows the change in the average cost per month in Egyptian Pounds "EGP" with MARLA-SG for a simulation time of 10 months. In MARLA-SG using Q-learning, Fig. 8(a) shows that the average cost of users through the simulation time is reduced



(a) Power consumption curve with different quantization levels using Q-learning.



(b) Power consumption curve with different quantization levels using SARSA.

FIGURE 5. **Power reduction using different learning methods.**

from 226.5 to 208.8 EGP, achieving a cost reduction of about 7.8%, while in MARLA-SG with SARSA, Fig. 8(b) shows that the average cost is reduced from 190.3 to 170.9 EGP, achieving a cost reduction of about 10.2%.

$$Cp_n(t) = Tr(Cl(t)) * \sum_{d=1}^{D} P_{n,d}(t), \tag{20}$$

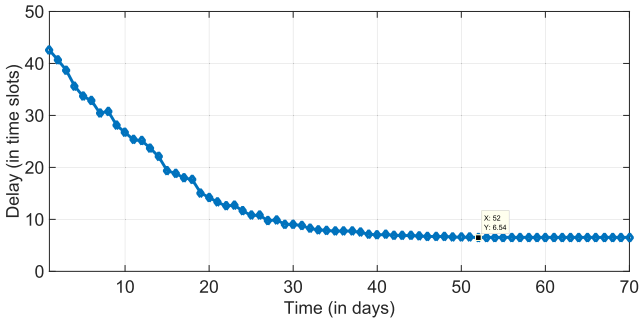
where Tr is the price tariff represented in a discrete vector = [0.22 0.3 0.36 0.7 0.9 1.35 1.45], as in Table 3(b).

D. A COMPARISON BETWEEN DIFFERENT LEARNING SCHEMES

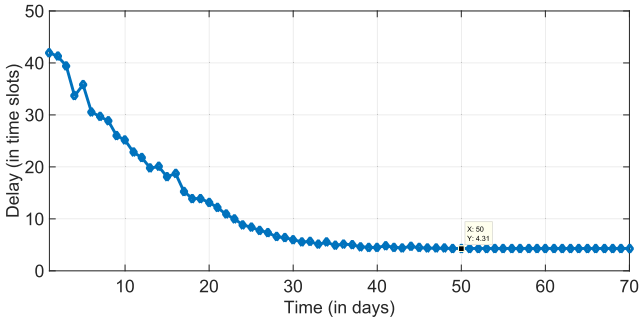
In this section, we introduce the performance of single-agent learning, partially coordinated multi-agent learning, totally independent multi-agent learning, and our proposed clustered multi-agent learning schemes. Table 4 shows the difference between the aforementioned algorithms. The convergence rate, cost, and PAR reduction are shown in Figs. 9, 10, and 11, respectively.

For simplicity, the simulation was performed for 6 households with 5 devices for each household using offline Q-learning. The totally independent multi-agent learning is competitive learning, where each user n is interested only in maximizing his profit and does not know the total grid consumption. For a user, $n \in N$, the state of the totally independent learning is formed using Eq. 21, while the reward is formed using Eq. 22. Offline Q-learning is used to get the optimum policy using Eqs. 23 and 24. The totally independent multi-agent learning is the fastest between the tested algorithms, but this fast convergence resulted in the worst behavior, as the PAR ratio is decreased only by 6.8% and the average cost is reduced only by 2.1%.

$$\begin{aligned} Sqn_n^1(t) &\leftarrow Sqn_{n,\text{ for all } d \in D}^1 \\ Sqn_n^2(t) &\leftarrow Sqn_{n,\text{ for all } d \in D}^2, \\ Sqn_n^3(t) &\leftarrow Sqn_{n,\text{ for all } d \in D}^3, \end{aligned} \tag{21}$$

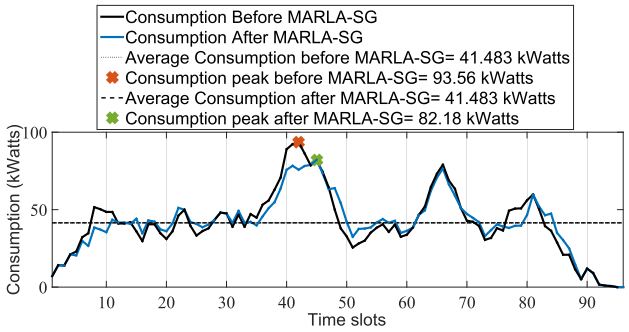


(a) Convergence rate of MARLA-SG using Q-learning.

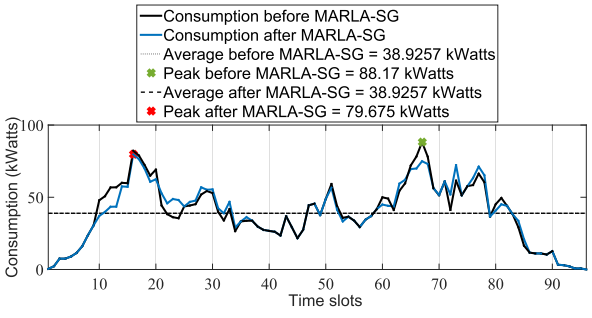


(b) Convergence rate of MARLA-SG using SARSA.

FIGURE 6. Convergence rates of MARLA-SG using different learning methods.



(a) PAR Reduction of MARLA-SG using Q-learning.

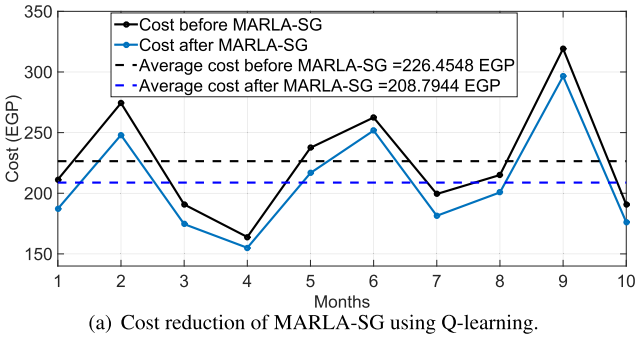


(b) PAR reduction of MARLA-SG using SARSA.

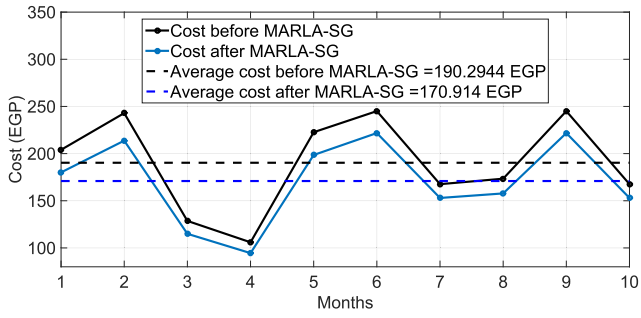
FIGURE 7. PAR reduction of MARLA-SG using different learning methods.

$$R_n(t) = \sum_{d=1}^{D} Rs_{n,d}(t) + Rc_{n,d}(t), \tag{22}$$

$$\begin{aligned} &Q_{new}(Sqn_n(t), a_n(t)) \\ &\quad = (1-\alpha_n)Q_{old}(Sqn_n(t), a_n(t)) \\ &\qquad + \alpha[R_n(t+1) + \gamma + max_a(Q(Sqn_n(t+1), a_n)], \end{aligned} \tag{23}$$



(a) Cost reduction of MARLA-SG using Q-learning.



(b) Cost reduction of MARLA-SG using SARSA.

FIGURE 8. Cost reduction of MARLA-SG using different learning methods.

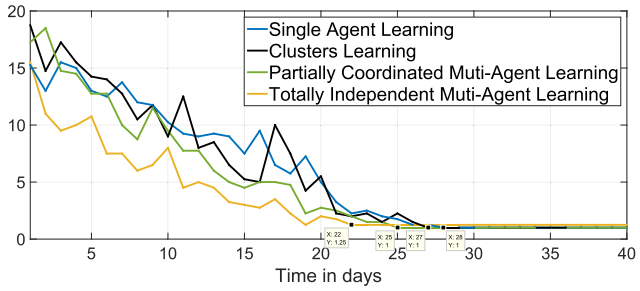


FIGURE 9. Convergence rate of different learning algorithms.

$$\alpha_n(t) = exp^{\left(\frac{-(V_{Sqn_n(t),a_n(t)})^2}{3}\right)}. \tag{24}$$

Single-agent learning is centralized learning and is interested in maximizing the gain of the total grid. The state of the single-agent learning is a matrix containing the state of each device $d \in D$ for each user $n \in N$ formed using Eq. 25, while the reward is the sum of each device's reward using Eq. 26. Offline q-learning is used to get the optimum policy using Eqs. 27 and 28. Single-agent learning is the slowest algorithm, yet it reached the optimum solution that achieves the best performance. Single-agent learning resulted in a cost reduction of 4.8% and 10.1% PAR reduction.

$$\begin{aligned} Sqn^1(t) &\leftarrow Sqn^1_{\text{for all } n\in N, d\in D} \\ Sqn^2(t) &\leftarrow Sqn^2_{\text{for all } n\in N, d\in D} \\ Sqn^3(t) &\leftarrow Sqn^3_{\text{for all } n\in N, d\in D}, \end{aligned} \tag{25}$$

$$R(t) = \sum_{d=1}^{D}\sum_{n=1}^{N} Rs_{n,d}(t) + Rc_{n,d}(t) + Rp_{n,d}(t), \tag{26}$$

TABLE 4. Difference Between Single-Agent Learning And Multi-Agent Learning Algorithms.

	Agent	Q-Table	State	Action	Reward	Learning Goal
Single-Agent Learning	Single-agent at the utility.	Single Q-table.	Matrix describing total grid.	Action matrix of all devices in the grid.	Single reward for all households in the grid.	Maximizing average gain of all households in the grid.
Clusters	Grid is divided to groups called clusters of equal number of household.	The number of Q-tables equals the number of clusters (N/Nc).	Matrix describing the current cluster's state.	Action matrix of all devices in the cluster.	Reward of current cluster plus the power level of neighbouring clusters.	Maximizing average gain of all clusters in the grid.
Partially Coordinated	Each household is an agent.	The number of Q-tables equals the number of households (N).	Matrix describing the current household's state.	Action matrix of all devices in the household.	Reward of current household plus the power level of neighbouring households.	Maximizing average gain of current households while taking into account the total reward of the grid.
Totally Independent	Each household is an agent.	The number of Q-tables equals the number of households (N).	Matrix describing the current household's state.	Action matrix of all devices in the household.	Reward of the current household.	Maximizing average gain of current households regardless the total reward of the grid.

$$Q_{new}(Sqn(t), a(t)) = (1-\alpha)Q_{old}(Sqn(t), a(t)) + \alpha[R(t+1)+\gamma max_a(Q(Sqn(t+1), a)], \quad (27)$$

$$\alpha(t) = exp^{(\frac{-(V_{Sqn(t),a(t)})^2}{3})}. \quad (28)$$

Since the total grid consumption affects directly power prices, in partially coordinated learning used in [24], each user has complete knowledge of the total energy consumption of the grid, but each user aims to minimize his monthly cost regardless of the other users. The state of the partially coordinated learning for a user n is a matrix containing the state of each device $d \in D$ formed using Eq. 29, while the reward is the sum of each device's reward using Eq.30. The learning is done using Eqs. 31 and 32. Partially coordinated learning attained cost reduction of 2.6% and 8.1% PAR reduction.

$$Sqn_n^1(t) \leftarrow Sqn_{n,\text{ for all } d\in D}^1$$
$$Sqn_n^2(t) \leftarrow Sqn_{n,\text{ for all } d\in D}^2$$
$$Sqn_n^3(t) \leftarrow Sqn_{n,\text{ for all } d\in D}^3, \quad (29)$$

$$R_n(t) = \sum_{d=1}^{D} Rs_{n,d}(t) + Rc_{n,d}(t) + Rp_{n,d}(t), \quad (30)$$

$$Q_{new}(Sqn_n(t), a_n(t)) = (1-\alpha_n)Q_{old}(Sqn_n(t), a_n(t)) + \alpha[R_n(t+1)+\gamma max_a(Q(Sqn_n(t+1), a_n)], \quad (31)$$

$$\alpha_n(t) = exp^{(\frac{-(V_{Sqn_n(t),a_n(t)})^2}{3})}. \quad (32)$$

MARLA-SG is utilizing the clustered multi-agent learning, Eqs. 35, and 36, it is a combination between the single-agent learning and partially coordinated multi-agent learning. It was introduced to avoid the centralized control used in single-agent learning, large memory required, in addition to the slow performance to reach the optimal policy [32], while maintaining the advantages of achieving the best cost and PAR reduction. The main idea is to divide the total grid into groups of equally numbered households each is called a cluster c. Each cluster is performing as a single-agent learning, while different clusters are partially coordinated. The state of the clustered learning for a user n in a cluster c is a matrix containing the state of each device $d \in D$, for user $n \in Nc$, is formed using Eq. 33, where (Nc): is the number of households in each cluster c. The reward is the sum of each device's reward in the cluster using Eq. 34. Clustered learning managed to reduce the average cost by 5.1% and the PAR by 10.1%.

$$Sqn_c^1(t) \leftarrow Sqn^1_{\text{for all } d\in D, n\in Nc}$$
$$Sqn_c^2(t) \leftarrow Sqn^2_{\text{for all } d\in D, n\in Nc}$$
$$Sqn_c^3(t) \leftarrow Sqn^3_{\text{for all } d\in D, n\in Nc}, \quad (33)$$

$$R_c(t) = \sum_{d=1}^{D}\sum_{n=1}^{Nc} Rs_{n,d}(t) + Rc_{n,d}(t) + Rp_{n,d}(t), \quad (34)$$

$$Q_{new}(Sqn_c(t), a_c(t)) = (1-\alpha_c)Q_{old}(Sqn_c(t), a_c(t)) + \alpha[R_c(t+1) + \gamma max_a(Q(Sqn_c(t+1), a_c)], \quad (35)$$

$$\alpha_c(t) = exp^{(\frac{-(V_{Sqn_c(t),a_c(t)})^2}{3})}. \quad (36)$$

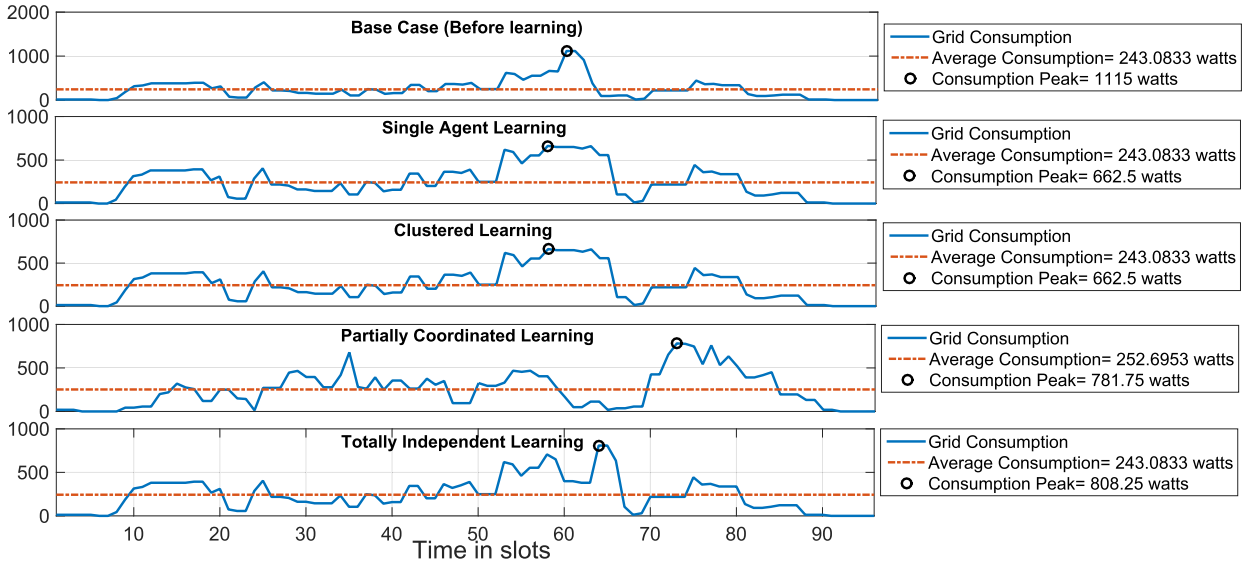


FIGURE 10. PAR reduction of different learning algorithms.

FIGURE 11. Cost reduction of different learning algorithms.

VI. CONCLUSION

In this article, a Multi-Agent Reinforcement Learning Algorithm for efficient demand response in Smart Grid (MARLA-SG) is proposed to improve the performance of the demand response in smart grid by not only reducing the cost without breaking the satisfaction limits identified by the user but also by reducing the PAR in a manner that avoids creating new peaks in lower prices hours if the number of users increases, without the need for complex calculations to represent the reward function or large memory area to save a huge number of possible states. Neither changing the price tariffs by the government at any time nor modifying the devices' parameters, will affect the performance of the proposed system.

REFERENCES

[1] R. Deng, Z. Yang, M.-Y. Chow, and J. Chen, "A survey on demand response in smart grids: Mathematical models and approaches," *IEEE Trans. Ind. Informat.*, vol. 11, no. 3, pp. 570–582, Jun. 2015.

[2] D. Picault, O. Cottet, and T. Ruez, "Demand response: A solution to manage loads in the smart grid," in *Proc. IEEE 15th Int. Conf. Environ. Electr. Eng. (EEEIC)*, Jun. 2015, pp. 352–356.

[3] H. Bilil, G. Aniba, and H. Gharavi, "Dynamic appliances scheduling in collaborative MicroGrids system," *IEEE Trans. Power Syst.*, vol. 32, no. 3, pp. 2276–2287, May 2017.

[4] M. Safdar, M. Ahmad, A. Hussain, and M. Lehtonen, "Optimized residential load scheduling under user defined constraints in a real-time tariff paradigm," in *Proc. 17th Int. Sci. Conf. Electr. Power Eng. (EPE)*, May 2016, pp. 1–6.

[5] Z. Wang, C. Gu, F. Li, P. Bale, and H. Sun, "Active demand response using shared energy storage for household energy management," *IEEE Trans. Smart Grid*, vol. 4, no. 4, pp. 1888–1897, Dec. 2013.

[6] S. Datchanamoorthy, S. Kumar, Y. Ozturk, and G. Lee, "Optimal time-of-use pricing for residential load control," in *Proc. IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Oct. 2011, pp. 375–380.

[7] M. Yu and S. Ho Hong, "A real-time demand-response algorithm for smart grids: A stackelberg game approach," *IEEE Trans. Smart Grid*, vol. 7, no. 2, pp. 879–888, Mar. 2016.

[8] F.-L. Meng and X.-J. Zeng, "A profit maximization approach to demand response management with customers behavior learning in smart grid," *IEEE Trans. Smart Grid*, vol. 7, no. 3, pp. 1516–1529, May 2016.

[9] A. Arif, "Electricity load and price forecasting using machine learning algorithms in smart grid: A survey," in *Proc. Workshops Int. Conf. Adv. Inf. Netw. Appl.*, Cham, Switzerland: Springer, 2020, pp. 471–483, doi: 10.1007/978-3-030-44038-1_43.

[10] I. P. Panapakidis and A. S. Dagoumas, "Day-ahead electricity price forecasting via the application of artificial neural network based models," *Appl. Energy*, vol. 172, pp. 132–151, Jun. 2016.

[11] D. Keles, J. Scelle, F. Paraschiv, and W. Fichtner, "Extended forecast methods for day-ahead electricity spot prices applying artificial neural networks," *Appl. Energy*, vol. 162, pp. 218–230, Jan. 2016.

[12] D. Wang, H. Luo, O. Grunder, Y. Lin, and H. Guo, "Multi-step ahead electricity price forecasting using a hybrid model based on two-layer decomposition technique and BP neural network optimized by firefly algorithm," *Appl. Energy*, vol. 190, pp. 390–407, Mar. 2017.

[13] Q. Zhou, W. Guan, and W. Sun, "Impact of demand response contracts on load forecasting in a smart grid environment," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2012, pp. 1–4, doi: 10.1109/PESGM.2012.6345079.

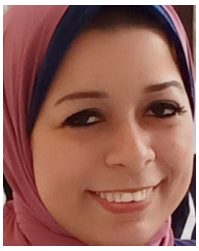
[14] D. Li and S. K. Jayaweera, "Reinforcement learning aided smart-home decision-making in an interactive smart grid," in *Proc. IEEE Green Energy Syst. Conf. (IGESC)*, Nov. 2014, pp. 1–6.

[15] M. M. Fouda, Z. M. Fadlullah, N. Kato, A. Takeuchi, and Y. Nozaki, "A novel demand control policy for improving quality of power usage in smart grid," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2012, pp. 5154–5159.

[16] D. W. Caves, J. A. Herriges, P. Hanser, and R. J. Windle, "Load impact of interruptible and curtailable rate programs: Evidence from ten utilities (tariff incentives)," *IEEE Trans. Power Syst.*, vol. 3, no. 4, pp. 1757–1763, Nov. 1988.

[17] Q. Duan, "A price-based demand response scheduling model in day-ahead electricity market," in *Proc. IEEE Power Energy Soc. Gen. Meeting (PESGM)*, Jul. 2016, pp. 1–5.
[18] A. E. Clements, A. S. Hurn, and Z. Li, "Forecasting day-ahead electricity load using a multiple equation time series approach," *Eur. J. Oper. Res.*, vol. 251, no. 2, pp. 522–530, Jun. 2016.
[19] R. Lu and S. H. Hong, "Incentive-based demand response for smart grid with reinforcement learning and deep neural network," *Appl. Energy*, vol. 236, pp. 937–949, Feb. 2019.
[20] Y. J. Zhang, C. Zhao, W. Tang, and S. H. Low, "Profit-maximizing planning and control of battery energy storage systems for primary frequency control," *IEEE Trans. Smart Grid*, vol. 9, no. 2, pp. 712–723, Mar. 2018.
[21] A. Gholian, H. Mohsenian-Rad, and Y. Hua, "Optimal industrial load control in smart grid," *IEEE Trans. Smart Grid*, vol. 7, no. 5, pp. 2305–2316, Sep. 2016.
[22] C. Li, X. Yu, W. Yu, G. Chen, and J. Wang, "Efficient computation for sparse load shifting in demand side management," *IEEE Trans. Smart Grid*, vol. 8, no. 1, pp. 250–261, Jan. 2017.
[23] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 1998.
[24] S. Bahrami, V. W. S. Wong, and J. Huang, "An online learning algorithm for demand response in smart grid," *IEEE Trans. Smart Grid*, vol. 9, no. 5, pp. 4712–4725, Sep. 2018.
[25] N. Chauhan, N. Choudhary, and K. George, "A comparison of reinforcement learning based approaches to appliance scheduling," in *Proc. 2nd Int. Conf. Contemp. Comput. Informat. (IC31)*, Dec. 2016, pp. 253–258.
[26] Z. Wen, D. O'Neill, and H. Maei, "Optimal demand response using device-based reinforcement learning," *IEEE Trans. Smart Grid*, vol. 6, no. 5, pp. 2312–2324, Sep. 2015.
[27] K. Dehghanpour, M. H. Nehrir, J. W. Sheppard, and N. C. Kelly, "Agent-based modeling of retail electrical energy markets with demand response," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3465–3475, Jul. 2018.
[28] P. Li, H. Wang, and B. Zhang, "A distributed online pricing strategy for demand response programs," *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 350–360, Jan. 2019.
[29] X. Lu, X. Xiao, L. Xiao, C. Dai, M. Peng, and H. V. Poor, "Reinforcement learning-based microgrid energy trading with a reduced power plant schedule," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10728–10737, Dec. 2019.
[30] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown toronto," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1140–1150, Sep. 2013.
[31] *Power Prices According to the Ministry of Electricity and Renewable Energy in Egypt*. Accessed: Mar. 1, 2020. [Online]. Available: http://egyptera.org/ar/Tarrif5.aspx
[32] L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," in *Innovations in Multi-Agent Systems and Applications—1, Studies in Computational Intelligence*, vol. 310. Berlin, Germany: Springer, 2010, pp. 183–221, doi: 10.1007/978-3-642-14435-6_7.

SALLY ALADDIN received the B.Sc. degree in electronics and communications from Cairo University, Cairo, Egypt, in 2009. She is currently pursuing the M.Sc. degree with the Department of Electrical Engineering, Faculty of Engineering at Shoubra, Benha University, Egypt. Her research interests include deep learning and smart grid communications.



SAMAH EL-TANTAWY received the Ph.D. degree in ITS from University of Toronto, in 2012. During her Ph.D. research, she developed a coordinated traffic signal control system using game theory concepts and multi-agent reinforcement learning approaches (MARLIN). She is currently an Assistant Professor with the Engineering Mathematics and Physics Department and also the Co-Director of the Technical Center for Career Development (TCCD), Faculty of Engineering, Cairo University. She has been published as a U.S. provincial and PCT international patents.

Dr. El-Tantawy received a number of prestigious international awards/funding/recognitions, including: the 2013 IEEE-ITSS Best PhD Dissertation Award, First place, 2013 INFORMS George B. Dantzing Dissertation Award, the 2014 University of Toronto Inventor of the Year, the 2015 ITS Canada New Canadian Commercial/Industry/Academic ITS Technology/Innovation/R&D Award, the 2017 State Encouragement Award for Women from the Academy of Scientific Research and Technology, and the 2017–2018 Second Place and Silver Medal for ICT R&D Category Award in the 5th Cairo Innovates Conference from the Academy of Scientific Research and Technology, for the project of "Development of an Adaptive Personalized Platform for Effective and Advanced Learning." Since January 2017, she has been the PI of a funded project by ASRT developing an adaptive and personalized platform for effective and advanced learning for school students.

MOSTAFA M. FOUDA (Senior Member, IEEE) received the Ph.D. degree in information sciences from the Graduate School of Information Sciences (GSIS), Tohoku University, Japan, in 2011. He has served as an Assistant Professor for Tohoku University, Japan. He was a Postdoctoral Research Associate with Tennessee Technological University, USA. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, Idaho State University, ID, USA. He is also an Associate Professor with Benha University, Egypt. He has published more than 30 articles in IEEE conference proceedings and journals. His research interests include cyber security, machine learning, blockchain, the IoT, 5G networks, smart healthcare, and smart grid communications. He has served on the technical committees for several IEEE conferences. He is also a Reviewer in several the IEEE TRANSACTIONS and Magazines and an Associate Editor of IEEE ACCESS.

ADLY S. TAG ELDIEN received the B.Sc., M.Sc., and Ph.D. degrees from Benha University, Egypt, in 1984, 1989, and 1993, respectively. He was the Ex-Head of the Network and Information Center, Benha University. He is currently serving as a Professor with the Department of Electrical Engineering, Benha University. His research interests include robotics, networks, and mobile communications.