Distributed learning of energy contracts negotiation strategies with collaborative reinforcement learning

Tiago Pinto[1], Zita Vale[2]

[1] GECAD Research Group, Polytechnic of Porto (ISEP/IPP), Porto, Portugal
tcp}@isep.ipp.pt
[2] Polytechnic of Porto (ISEP/IPP), Porto, Portugal
zav@isep.ipp.pt

Abstract — **The evolution of electricity markets towards local energy trading models, including peer-to-peer transactions, is bringing by multiple challenges for the involved players. In particular, small consumers, prosumers and generators, with no experience on participating in competitive energy markets, are not prepared for facing such an environment. This paper addresses this problem by proposing a decision support solution for small players negotiations in local transactions. The collaborative reinforcement learning concept is applied to combine different learning processes and reached an enhanced final decision for players actions in bilateral negotiations. The reinforcement learning process is based on the application of the Q-Learning algorithm; and the continuous combination of the different learning results applies and compares several collaborative learning algorithms, namely BEST-Q, Average (AVE)-Q; Particle Swarm Optimization (PSO)-Q, and Weighted Strategy Sharing (WSS)-Q and uses a model to aggregate these results. Results show that the collaborative learning process enables players' to correctly identify the negotiation strategy to apply in each moment, context and against each opponent.**

Index Terms— **Collaborative reinforcement learning, Electricity Markets, Energy Contracts, Negotiation Strategies, Q-Learning**

I. Introduction

Local electricity markets are quickly evolving into a reality [1]. This brings multiple challenges for market players, especially small consumers and generators, which are unexperienced in active market participation. Hence, reaching good agreements is an arduous task. In order to overcome this problem, it is necessary to provide players with decision support in order to help players applying negotiation strategies and tactics; and enable their adaptive behavior depending on the contexts and opponents [2]. Multiple alternative negotiation strategies can be found in the literature, such as the application of game theoretical models [3], assessing risk management [4], and using machine learning models to forecast market prices and optimize the proposed bids[5]. However, current models are not capable of adapting to different market circumstances and negotiating contexts, as they are limited to specific market scenarios and are not integrated in actual market simulation or decision support systems. Thereby current approaches are not able to provide market players with the means to change their behaviour in a real market environment, and therefore pursuit the achievement of the best possible outcomes.

This paper contributes to overcoming these problems by proposing a solution for decision support of market players in bilateral energy contracts negotiations. The collaborative reinforcement learning concept is used to combine the learning process of different negotiation strategies. The diverse learning sources are the learning processes of several agents, which learn the same problem under different perspectives (using different utility or results assessment functions). By combining the different independent learning processes, it is possible to gather the diverse knowledge and reach a final decision on the most suitable negotiation strategy to be applied. The reinforcement learning process is based on the application of the Q-Learning algorithm [6]; and the continuous combination of the different learning results applies and compares several collaborative learning algorithms, namely BEST-Q, Average (AVE)-Q; Particle Swarm Optimization (PSO)-Q, and Weighted Strategy Sharing (WSS)-Q [7]. Additionally a model that uses the aggregated results of these algorithms is presented.

Results show that the collaborative learning process enables players' to correctly identify the best (a-priori identified) negotiation strategy to apply in each moment, context and against each opponent. Moreover, the different algorithms enable the adaptation according to needs of each learning process, i.e. faster, yet not so solid, convergence; or slower convergence, but with higher guarantees of success.

The remaining of the paper is organized as follows. Section II presents the proposed methodology, section III presents the case study that validates the proposed model. Finally, section V presents the conclusions of the work.

II. Proposed Methodology

The approach proposed in this paper concerns the combination of the learning process of different agents through collaborative learning. The different agents learn the same problem under different perspectives, using different utility or results assessment functions, which result from their own perspective when analysing the problem and the corresponding context. Despite the independent learning processes, all agents use Q-Learning as the reinforcement learning algorithm for this

This work has received funding from the European Union's Horizon 2020 research and innovation programme under project DOMINOES (grant agreement No 771066) and from FEDER Funds through COMPETE program and from National Funds through FCT under the project UID/EEA/00760/2019

problem. The combination of the different learning process is then applied through several collaborative learning algorithms, namely BEST-Q, AVE-Q; PSO-Q, and WSS-Q [7]. Also, a strategy that aggregates the learning process from these algorithms is also presented.

A. Q-Learning

Q-Learning is a very popular reinforcement learning method. It is an algorithm that allows the autonomous establishment of an interactive action policy. It is demonstrated that the Q-Learning algorithm converges to the optimal proceeding when the learning state-action pairs Q is represented in a table containing the full information of each pair value [8]. The basic concept behind Q-Learning is that the learning algorithm is able to learn a function of optimal evaluation over the whole space of state-actio- pairs s x a. This evaluation thus defines the confidence value Q that each action a can represent the state s. The Q function performs the mapping as in (1).

$$Q: s\ x\ a\ \rightarrow U \tag{1}$$

where U is the expected utility value when selecting action a in state s. As long as the state does not omit relevant information, nor introduce new information, once the optimal function Q is learned, the decision method will know precisely which action results on the higher future reward under each state. The reward r is attributed to each pair *action-state* in each iteration, representing the quality of this pair, and allows the confidence value Q to be updated after each observation. r is defined as in (2).

$$r_{a,s,t} = 1 - norm\left|RP_{a,s,t,o,p} - EP_{a,s,t,o,p}\right| \tag{2}$$

where $RP_{a,s,t,o,p}$ represents the real price that has been established in a contract with an opponent o, in state s, in time t, referring to an amount of power p; and $EP_{\ a,s,t,o,p}$ is the estimation price of scenario that corresponds to the same player, amount of power and state in time t. All r values are normalized in a scale from 0 to 1, in order to allow the $Q(s, a)$ function to remain under these values, so that the confidence values Q can be easily assumed as probabilities of scenario occurrence under a context. $Q(s, a)$ is learned through by try an error, being updated every time a new observation (new contract establishment) becomes available, following equation (3).

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha\big[r_{s,a,t} + \gamma U_t(a_{t+1}) - Q_t(s_t, a_t)\big] \tag{3}$$

where □ is the learning rate; □ is the discount factor; and U_t (4) is the utility resulting from action a under state s, obtained using the Q function learned so far.

$$U_t(s_{t+1}) = \max_a Q(s_{t+1}, a) \tag{4}$$

The Q Learning algorithm is executed as follows:

- For each a and s, initialize $Q(s, a) = 0$;
- Observe new event;
- Repeat until the stopping criterion is satisfied:
  - Select the action that presents the higher Q for the current state;
  - Receive reward $r_{a,s,t}$;
  - Update $Q(s, a)$;
  - Observe new state s';
  - s □ s'.

As the visiting of all action-state pairs tends to infinite, the method guarantees a generation of an estimative of Q_t which converges to the value of Q. In fact, the actions policy converges to the optimal policy in a finite time, however slowly. In order to accelerate the convergence process, not only the Q value of the chosen action is updated, but also that of all scenarios, since the r regarding all alternative scenarios can be computed by comparing the estimated prices by each action and the actual values that have been verified in a new contract agreement. After each updating process, all Q values are normalized, as in equation (5), so that they are always kept in a scale from 0 to 1, thus facilitating the interpretation as the probability of each action in correctly representing the negotiation reality.

$$Q'(s, a) = \frac{Q(s, a)}{\max\left[Q(s, a)\right]} \tag{5}$$

B. Collaborative learning approaches

1) BEST-Q

The BEST-Q algorithm selects, for each state-action pair, the best value (Q-value) from all tables (Q-tables) of all agents, as in (6). Then each agent updates its individual Q-table accordingly.

$$Q_i(s,a) \leftarrow Q^{best}(s,a),\ \forall i,s,a \tag{6}$$

where i is the agent.

The disadvantage of this approach is that optimum values (Q-values) are not found because the values (Q-values) become equal after each update. However, the BEST-Q algorithm can achieve good long-term simulation policy.

The BEST-Q algorithm uses as assumption the best confidence value for each state-action pair according to all the data of the agents present in the environment. Each agent updates its Q-table by updating the pairs with the best values obtained previously.

2) AVE-Q

The AVE-Q algorithm is similar to the BEST-Q except that each agent updates its Q-values with the average of its current value and the best value (Q-value) for each state-action from the tables (Q-tables) of all agents, as presented in (7)

$$Q_i(s, a) \leftarrow \frac{Q^{best}(s, a) + Q_i(s, a)}{2}, \qquad \forall\ i, s, a \tag{7}$$

The main disadvantage of the AVE-Q algorithm is that it does not eliminate the bad values (Q-values) in the interaction stage. The AVE-Q algorithm is very similar to the BEST-Q algorithms except for updating the agent. It uses as assumption the best value of confidence for each state-action pair according to all the data of the agents present in the environment and its current value of learning, so the table of the agent is updated through the average of these two values. Each agent updates its Q-table by updating the pairs with the previous values.

3) *PSO-Q*

Multi-agent optimization known as Particle Swarm Optimization (PSO), is part of the swarm intelligence methodologies and techniques. This algorithm was inspired by the rules of alignment and cohesion of the flocks of birds, and its particularity is represented by the transmission and sharing of information [9].

Each agent is initialized with a set of possible random solutions and the optimal solution is searched for in each generation. The movement of each agent is influenced by the global optimum and personal memory, with each agent having the ability to adapt its speed that directs its movement and remembers the best position found to date [10]. This movement follows the following four rules:

- Separation: there must be a separation between each agent, to avoid collisions.
- Alignment: it is necessary that each agent follows the same direction of neighboring particles.
- Cohesion: it is necessary that each agent follows the same position of neighboring particles.
- Deviation: in the encounter of an obstacle, it is necessary that the agent is able to deviate.

The PSO-Q algorithm uses PSO to find the near-optimal solution. PSO is an optimization method that repeatedly improves the candidate solution accordingly to with the qualitative measure. PSO solves decision problems that have multiple decision variables. In the PSO-Q algorithm the best values (Q-values) of each agent and the best global values (Q-values) of all agents are used by each agent to update its Q-table, as in (8) according to a velocity function V_i (9) that determines the movement of the particles involved in the search process.

$$Q_i(s,a) \leftarrow Q_i(s,a) + V_i(s,a), \qquad \forall\, i,s,a \tag{8}$$

$$\begin{aligned} V_i(s,a) = WV_i(s,a) + C_1R_1[P_i(s,a) - Q_i(s,a)] \\ + C_2R_2[G(s,a) - Q_i(s,a)] \end{aligned} \tag{9}$$

where W is the inertia component, which defines the degree in which the movement will stay closer to the previous position; $P_i(s,a)$ is the best Q-value of agent i for the pair s x a, $G(s,a)$ is the best global solution for the s x a pair, C_1 and C_2 are weight components that determine the degree in which the new position will tend to the personal and global best, respectively; and R_1 and R_2 are random values ranging [0, 1].

In the PSO-Q, the reinforcement learning problem is modeled as an optimization problem in which the candidate solutions are the values (Q-values of the table), and the qualitative measure is the Q-function. In the PSO-Q algorithm, the best values (Q-values) of each agent and the best overall value of all agents are used for each agent to update its Q-table.

4) *WSS-Q*

In the WSS (Weighted Strategy-Sharing) method, it is assumed that homogeneous Q-Learning agents learn in some distinct environments, so their actions do not alter the environment of other agents and no hidden state is produced.

Agents learn in two ways: individual learning mode and cooperative learning mode. First, all agents are in the individual learning mode. The agent performs several learning attempts. Each learning attempt starts from a random state and ends when the agent reaches the goal. After a specified number of individual attempts, all agents switch to cooperative learning mode. In the collaborative mode, each agent delegates a weight to the other agents according to their expertise (trust value). Then, each agent updates through a weighted average with the values of the other tables resulting in a new table.

Using the WSS-Q algorithm, each agent assumes a weight for the tables of the other agents based on the relative skill of each agent. Subsequently, each agent uses the weighted average of all values of tables (Q-tables) to update its own table (10).

$$Q_i(s,a) \leftarrow \sum_{j=1}^{n} \left[W_{i,j} Q_j(s,a)\right] \quad , \qquad \forall\, i,s,a \tag{10}$$

where W_{ij} is the weight that agent i takes on the skill of agent j.

5) *Algorithms aggregation*

The collaborative Q-learning algorithms BEST-Q, AVE-Q, PSO-Q and WSS demonstrate different behaviors based on their sharing strategy. These observations motivated the combination of the sharing strategies in a single strategy to reduce the variability in performance for different problems. The Q-learning algorithm that uses this strategy is referred to as average aggregation Q-learning. In average aggregation Q-learning, each agent updates its Q-values by averaging the Q-values of each of the BEST-Q, AVE-Q, PSO-Q and WSS algorithms for each state action, according to (11), where i is the agent identifier and 4 is the number of strategies.

$$Q_i(s,a) = \frac{Q^{BEST-Q}(s,a) + Q^{AVE-Q}(s,a) + Q^{PSO-Q}(s,a) + Q^{WSS}(s,a)}{4} \tag{11}$$

III. CASE STUDY

A. *Specifications*

This case study considers 4 independent agents, which learn the same problem from different perspectives. In summary, each agent needs to learn which, from 10 distinct actions, is the best one; in which each action refers to the choice on a negotiation strategy to be applied against an opponent in a bilateral negotiation. Table I shows the a-priori defined best actions from each agent's perspective.

Table 1 Best a-priori actions for each agent

Agent id	1	2	3	4
Best actions #	10	10, 2	8, 2	8

From Table 1 it is visible that the best overall actions accordingly to the perspective of the 4 agents are actions 2, 8 and 10. The number of Q-Learning episodes to perform has the value 200 being that each episode is composed of 1000 repetitions of the Q-Learning steps. The sharing of information between agents in done at every 10 episodes. All agents initially start in episode 1. The parameterization for Q-Learning is: the discount factor is 0.9 for a slower exploration and a learning rate of 0.01.

B. *Results*

Figure 1, 2, 3, 4 and 5 present the evolution of the Q-values of each action, from each agent's perspective, throughout all the episodes, when using the BEST-Q, AVE-Q, PSO-Q, WSS-Q and average aggregation Q-learning algorithms, respectively.

From Figure 1 is can be seen that the agents present partially identical graphs because they use the best values of the other agents. The BEST-Q algorithm reaches a relative convergence at around 360 iterations. From Figure 2 one can see that the AVE-Q algorithm in the first iterations presents a marked increase in values for the actions with greater reinforcement. The algorithm reaches a balance from the 160 iterations. It is concluded that AVE-Q reaches a quicker convergence that BEST-Q on the best actions.

From Figure 3 it is visible that PSO-Q in the first iterations presents a marked increase in values for the actions with greater reinforcement. The algorithm reaches a balance from the 160 iterations. Although with the increase in the number of iterations another action stands out; *i.e.* the algorithm allows to explore other possibilities and make a management of learning with exploration and experience. In comparison with the previous algorithms this algorithm achieves a fast equilibrium allowing for the search of new emergent good actions. From Figure 4, one can see that the WSS algorithm presents variations along the number of iterations. This algorithm limits the choice in only 3 actions for the proposed problem (2, 8 and 10 as a-priori identified). In comparison with the previous algorithms this one identifies the best actions, but it does not demonstrate a clear convergence, like the other algorithms. From Figure 5 it is visible that the average aggregation enables identifying the best actions, and converging quite quickly, benefiting from the results of the algorithms that present a fast convergence. However, the algorithm still enables the refinement of the learning process by incorporating the stochasticity of WSS, and thus keep adapting to any new occurrences throughout the iterations.

IV. Conclusion

This paper has presented the application of five collaborative reinforcement learning algorithms (BEST-Q, AVE-Q, PSO-Q and WSS-Q and average aggregation) to the problem of identifying the best action (negotiation strategy) that is learned independently by several different agents, with different perspectives.

Results show that with BEST-Q all agents converge to the same Q-Tables, which prevents them from adding their independent perspective on the problem; nevertheless, the best actions are identified, among others that also present good potential. AVE-Q converges quickly to the best actions. PSO-Q also converges quickly, but enables for the future identification of other emerging good actions, due to the stochastic nature. WSS-Q presents a great variation throughout the entire set of episodes, but it is the only one that enables identifying the exact 3 a-priori best actions, while the 3 other algorithms identify these 3 but also add some other relatively good actions into the mix. Average aggregation has enabled a fast convergence but with still some room for exploration, by incorporating the perspectives of the different algorithms.

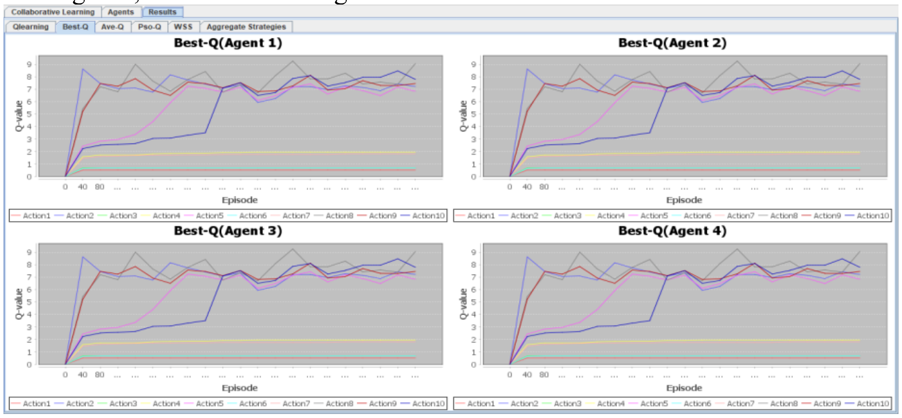

Figure 1. Evolution Q-Values for BEST-Q

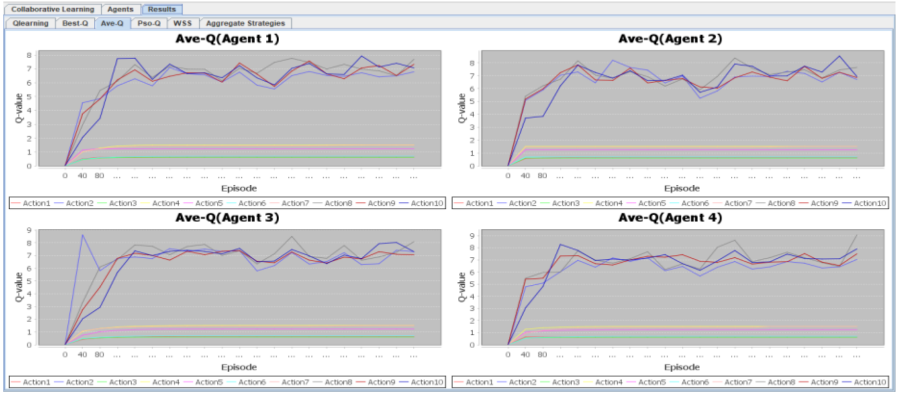

Figure 2. Evolution Q-Values for AVE-Q

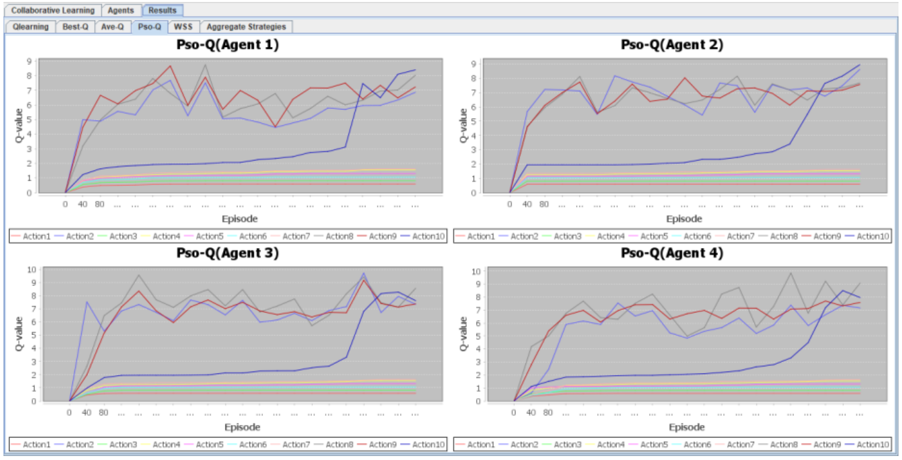

Figure 3. Evolution Q-Values for PSO-Q

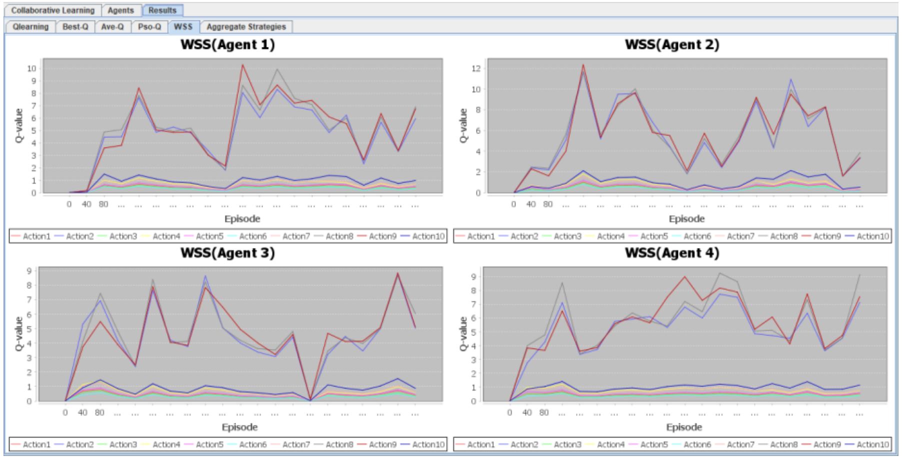

Figure 4. Evolution Q-Values for WSS-Q

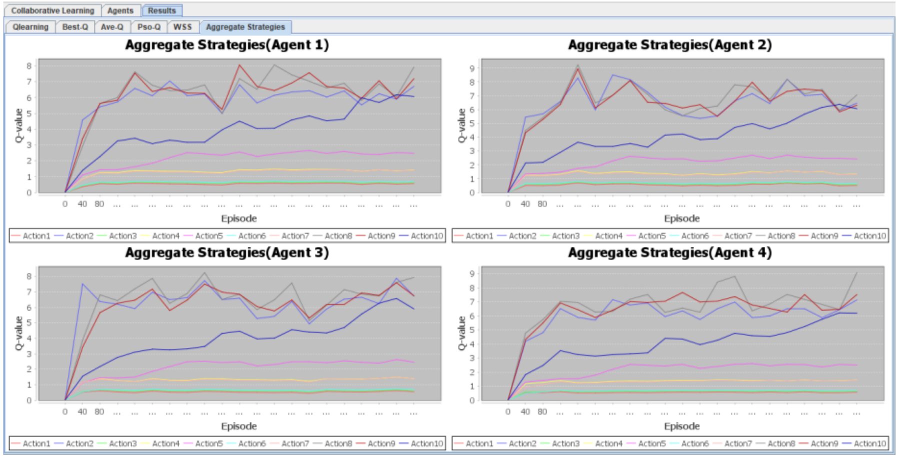

Figure 5. Evolution Q-Values for average aggregation Q-learning

REFERENCES

[1] M. Ampatzis, P. H. Nguyen, and W. Kling, “Local electricity market design for the coordination of distributed energy resources at district level,” in *IEEE PES Innovative Smart Grid Technologies, Europe*, 2014, pp. 1–6.

[2] T. Pinto, Z. Vale, T. M. Sousa, I. Praça, G. Santos, and H. Morais, “Adaptive Learning in Agents Behaviour: A Framework for Electricity Markets Simulation,” *Integr. Comput. Eng.*, vol. 21, no. 4, pp. 399–415, 2014.

[3] M. N. Faqiry, R. Kundu, R. Mukherjee, S. Das, and A. Pahwa, “Game theoretic model of energy trading strategies at equilibrium in microgrids,” in *2014 North American Power Symposium, NAPS 2014*, 2014.

[4] S. S. Meghwani and M. Thakur, “Multi-criteria algorithms for portfolio optimization under practical constraints,” *Swarm Evol. Comput.*, vol. 37, pp. 104–125, 2017.

[5] J. Nowotarski and R. Weron, “Recent advances in electricity price forecasting: A review of probabilistic forecasting,” *Renew. Sustain. Energy Rev.*, vol. 81, pp. 1548–1568, 2018.

[6] M. R. Salehizadeh and S. Soltaniyan, “Application of fuzzy Q-learning for electricity market modeling by considering renewable power penetration,” *Renew. Sustain. Energy Rev.*, vol. 56, pp. 1172–1181, 2016.

[7] B. Abed-alguni, D. J. Paul, S. K. Chalup, and F. A. Henskens, “A comparison study of cooperative Q-learning algorithms for independent learners,” *Int. J. Artif. Intell.*, vol. 14, no. 1, pp. 71–93, 2016.

[8] P. Kofinas, A. I. Dounis, and G. A. Vouros, “Fuzzy Q-Learning for multi-agent decentralized energy management in microgrids,” *Appl. Energy*, vol. 219, pp. 53–67, 2018.

[9] M. S. Kiran, “Particle Swarm Optimization with a New Update Mechanism,” 2017.

[10] J. Kennedy and R. Eberhart, “Particle swarm optimization,” *Neural Networks, 1995. Proceedings., IEEE Int. Conf.*, vol. 4, pp. 1942–1948 vol.4, 1995.