Multi-Agent Reinforcement Learning for the Energy Optimization of Cyber-Physical Production Systems

Jupiter Bakakeu*, Schirin Baer, Hans-Henning Klos, Joern Peschke, Matthias Brossog and Joerg Franke

Abstract This chapter proposes an artificial intelligence based solution for the efficient operation of a heterogeneous cluster of flexible manufacturing machines with energy generation and storage capabilities in an electricity micro-grid featuring a high volatility of electricity prices. The problem of finding the optimal control policy is first formulated as a game theoretic sequential decision making problem under uncertainty, where at every time step the uncertainty is characterized by future weather dependent energy prices, high demand fluctuation, as well as random unexpected disturbances on the factory floor. Because of the parallel interaction of the machines with the grid, the local viewpoints of an agent are non-stationary and non-Markovian. Therefore, traditional methods such as standard reinforcement learning approaches that learn a specialized policy for a single machine are not applicable. To address this problem, we propose a multi-agent actor-critic method that takes into account the policies of other participants to achieve explicit coordination between a large numbers of actors. We show the strength of our approach in mixed cooperative and competitive scenarios where different production machines were able to discover different coordination strategies in order to increase the energy efficiency of the whole factory floor.

1 Introduction

Today the cost-efficient and stable supply of energy for production processes is a topic of increasing relevance. Changes in the structure of power generation towards

Jupiter Bakakeu, Matthias Brossog and Joerg Franke
Friedrich-Alexander-Universität Erlangen-Nürnberg, Egerlandstr. 7-9, 91058 Erlangen, Germany, e-mail: jupiter.bakakeu@faps.fau.de, matthias.brossog@faps.fau.de, joerg.franke@faps.fau.de

Schirin Baer, Hans-Henning Klos and Joern Peschke
Siemens AG, Gleiwitzer Str. 555, 90475 Nuremberg, Germany, e-mail: schirin.baer@siemens.com, hans-henning.klos@siemens.com, joern.peschke@siemens.com

decentralized systems and the growing share of renewable energy sources such as photovoltaics and wind power lead to a volatile situation in the supply of electrical energy. The concept in which electricity generation solely follows a demand based on statistical forecasts is no longer applicable. As a consequence, new concepts that harness the flexibility of energy consumption on the consumer side are becoming increasingly important, resulting in new developments in energy markets where a flexible consumer can get incentives for providing flexibility and participating in specific programs.

In this context, cyber-physical production systems (CPPS) with multiple possible plant topologies and the ability to produce huge number of product variants can make a significant contribution to a transition to a flexible energy demand. However, ensuring such flexibility in the industry 4.0 context requires a sophisticated control strategy to intelligently supervise the power consumption of the underlying components at every given time so that the total energy consumption follows given load profile boundaries. Moreover, the control strategy has to ensure a high productivity while coping with the uncertainties that can arise from the high fluctuations of weather-dependent energy prices, fast market changes as well as random unexpected disturbances in the factory floor.

Traditionally, analytical and simulation-based approaches are used to compute the optimal control strategy for the connected loads, distributed generation and storage components. However, these approaches often require complex models with precise descriptions of system features and functions. The engineering efforts for these systems are often a crucial aspect, which restricts implementations in the large scale. This problem becomes even more important in the industry 4.0 context, where fast market changes can lead to frequent changes in product and process configurations and therefore to the system models.

For these reasons, model-free control solutions and especially new artificial intelligence algorithms are considered as valuable alternatives or supplements [1]. One of the most popular model-free control paradigms that has gained in importance in recent years is Reinforcement Learning (RL) [2]: a model-free algorithm that does not require system identification steps or apriori knowledge. While being very popular in single agent settings (see [3], [4], [5], [6]), a naive application of RL in a multi-agent environment (as it is the case for CPPS) is likely to perform poorly. One issue is the non-Markovian nature of the resulting environment: since the action of one agent can affect the outcome of the action of other agents, the single perspective of an agent, which mainly consists of partial observations, becomes non-stationary in the sense that changes that cannot only be explained by the inherent stationary dynamics of the environment can occur. It is therefore important to coordinate the choices of actions of all agents in order to achieve the intended effects. Furthermore, an agent trained in a single environment setting is inherently bounded by the task description. A slight modification of the task or an evolution of the environment creates new pressures for retraining.

In order to solve these problems, many have worked on unsupervised exploration and skill acquisition methods such as *intrinsic motivation* and *self-play*. Rather than training the agents to solve a specific task in a predefined environment, the agent are

conditioned in a autocurricula environment whereby the agents have to continuously coordinate their strategies (via competition or cooperation) in order to solve an ever evolving task. As a result, the agents learn an ensemble of emergent skills and coordination strategies rather than a specific complex policy making them more robust to environment changes.

There has been much success in leveraging multi-agent autocurricula to solve multi-agent problems such as [7] and [8] both in discrete and continuous real-time domains. In both cases, self-play approaches provided the agents with a perfectly tuned curriculum. The success in these settings inspires confidence that inducing autocurricula in physically grounded and open-ended environments such industrial micro-grids could eventually enable the agents controlling the grid components to acquire an unbounded number of robust and emergent skills that can lead to a global resource optimization. In this chapter, we investigate whether the idea of autocurricula multi-agent reinforcement learning environments can yield fruit in solving the problem of the energy optimization of an heterogeneous cluster of cyber-physical production systems with energy generation and storage capabilities in an electricity micro-grid featuring a high volatility of electricity prices.

In more detail, we first formulate the energy optimization problem as an observable Markov game which we then use to design an autocurricula environment that simulates a micro-grid of flexible manufacturing machines with energy generation and energy storage capabilities. We then introduce several multi-agent tasks with comparative goals, where the different agents would need to learn not only highly developed skills but also coordinate the learned strategies in order to achieve the intended global effect. We then train the agents using a distributed implementation of a recent policy gradient algorithm, Proximal Policy Optimization (PPO) [6] in a framework of centralized training with decentralized execution. By adding a simple exploration curriculum to aid exploration in the environment, we found that in both cooperative and competitive scenarios different production machines were able to discover different coordination strategies in other to increase the energy efficiency of the whole factory floor.

The remainder of this chapter is structured as follows. In section 2, backgrounds and related work on load management systems as well as multi-agent reinforcement learning algorithms are presented. Then, we formulate the energy optimization problem as a sequential decision making problem under uncertainty in section 3. Section 4 gives a detailed description of our training algorithm and the autocurricula environment. Experimental results demonstrating the strength of our approach are presented in section 5. Finally, we summarize the paper and discuss some directions for future work in section 6.

2 Backgrounds and Related Work

2.1 Load Management Systems

A growing number of works on load management systems have primarily focused on industrial load control and have generally ignored the volatility of the energy prices [1, 9, 10, 11]. Most recent works [12, 13, 14] took into account this factor and showed that analytical approaches can be used to compute the optimal time to change the energetic state of a machine for energy control purposes without sacrificing the system throughput. Energy-aware scheduling was also considered to optimize the production plan using mixed integer programming [15, 16]. However, all the proposed solutions rely on fixed product and machine configurations and thus are not applicable in a dynamic context. Furthermore, these solutions require a precise estimation of the model of the production machine as well as the associated parameters. Since the acquisition of such models mostly rely on expert knowledge, the proposed solutions are not scalable and not cost effective. In this chapter, we focus on a model-free control solution that use RL and do not require an explicit system identification.

2.2 Reinforcement Learning

Reinforcement learning (RL) is one of the main classes of machine learning methods that has been successfully applied to solve sequential decision-making problem under uncertainties [17]. In such a setting, an agent learns to act using a (partial observable) Markov decision process (MDP) [2] formalism. At each time step, the agent selects an action from a finite set of possible actions. After executing the selected action, the environment is updated and the agent receives a rewards signal, which can also be interpreted as a punishment from the environment. The goal of the agent is to maximize the discounted reward in the long run. To achieve this, the agent has to compute the optimal policy based on interactive learning with the environment (see Fig. 1).
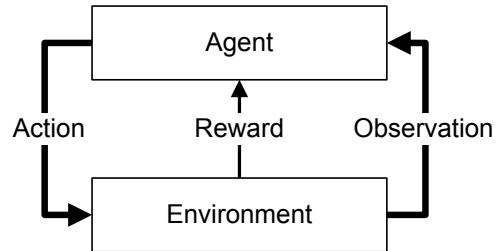

Fig. 1 The agent-environment interaction in Reinforcement Learning [2].

The literature provide many examples of RL algorithms applied to energy management problems. In 1996. Anderson et al. applied Q-learning to modulate the output of the PI controller for a heating coil [18, 19]. In the same direction, Ruelens et al.[20, 21], De Somer et al. [22] and Kazmi et al. [23] were able to apply RL methods to reduce the domestic water heating costs. Jiang et al. implemented a hierarchical multi-agent Q learning approach for dynamic demand response and distributed energy resource management in a micro-grid [24]. In recent studies [25, 26, 27], the authors implemented a multi-agent Deep Q-Network (DQN) [3] approach where each agent controlled a house. The agent could consume/store energy, request energy from or grant energy to neighbors, deny energy to neighbors or purchase it from the grid.

2.3 Multi-Agent Reinforcement Learning

In multi-agent reinforcement learning (MARL), multiple agents perform concurrently in a single environment [28, 29, 30, 31]. The simplest approach to learning in multi-agent settings is to train each agent independently. This was attempted with Q-learning in [31], but does not perform well in practice [32] since many applications assume interdependency between the agents. More recently, deep neural networks have been successfully used in RL to approximately represent policy and value functions. It is therefore not surprising that deep neural networks have also been applied to MARL settings [33, 34], allowing multi-agent learning in high-dimensional/continuous state spaces. However, applying standard RL methods to MARL problems is challenged by some limitations, such as non-stationary of the environment from the perspective of individual agents [35, 36, 37], lack of coordination/communication in cooperative settings [36, 38, 39, 40], credit assignment in cooperative settings with global rewards [37, 41, 42], and the failure to take opponent strategies into account when learning agent policies [43].

Most relevant to this work are recent approaches [39, 37] that propose an actor-critic framework consisting of centralized training with decentralized execution as well as some approaches that utilize attention in a fully centralized multi-agent setting [44, 45]. The proposed method uses separate centralized critics for each agent, which take all other agent's actions and observations as input during the training phase, and actors whose policies are conditioned only on local information. This practice reduces the non-stationary of multi-agent environments, as considering the actions of other agents to be part of the environment makes the state transition dynamics stable from the perspective of one agent. In practice, these ideas greatly stabilize learning because of the reduced variance in the value function estimates.

Our proposed solution follows these recent developments and propose a centralized critic for PPO applied in a multi-agent setting. It is more flexible than the aforementioned approaches in the sense that it is able train policies in environments with any reward setup, observation and action spaces making it broadly applicable to different environment types including industrial micro-grid environments.

3 System Model and Problem Formulation

The chapter addresses the problem of optimally controlling an industrial micro-grid featuring a large share of renewable energy and a high volatility of electricity prices. We consider a micro-grid as a localized group of energy sources, loads and storage components that can operate in two distinct modes: grid-connected mode and isolated mode. In grid-connected mode, the micro-grid system has the possibility to buy/sell energy from/to the macro-grid in order to meet the load demand. The challenge in connected mode is to reduce the total energy cost. In isolated mode, the micro-grid can function autonomously in the sense that enough energy can be generated in the grid to supply the loads. In this case the challenge is to meet the load demand and to balance the power flow between the components of the grid. In our setting, the grid is executed in mixed mode. In other words, the industrial micro-grid, we are considering, can switch from one mode to another.

3.1 System Model

As illustrated in Fig.2, the micro-grid system consists of the following components: renewable energy sources such as PV-Arrays, electricity loads representing production machines, regenerative energy consumers, energy storages systems such as batteries and auxiliary process loads for air compressors and cooling systems. The components share the same energy bus, enabling a transfer of electrical energy between the components. The proposed system model is equally applicable to ac and
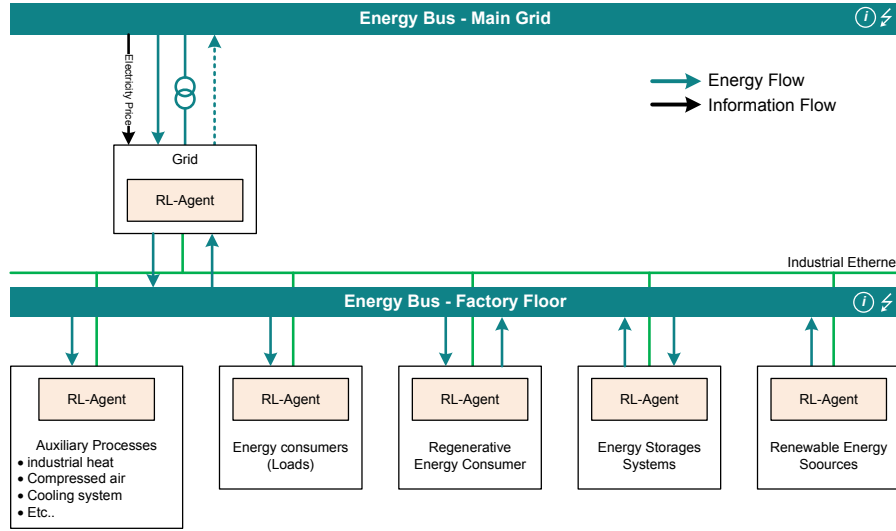

Fig. 2 System Overview. Every grid-component is controlled by a reinforcement learning agent.

dc micro-grids and a simple formal description of the components can be described as follow:

- **Renewable energy sources** are renewable energy generators such as PV-Arrays or wind turbines. The power generated depends on environmental and weather factors such as solar radiation profile and wind speed. It is therefore not predictable. Energy generators are characterized by two operating states: on and off. The generators are considered to be in the off-state when the electrical power generated by the energy sources is negligible for example because of weather conditions such as cloudy weather or nighttime. We assume that if the generators are not in off state, then the power generated is constant and equal to the maximum electricity power that can be produced by the source. Therefore

$$P_G = \begin{cases} 0 & \text{if } state = off \\ P_G max & \text{otherwise} \end{cases} \tag{1}$$

- **Energy storage systems**. In order to take full advantage of renewable energy sources, it is vital to have energy storage systems capable of handling variations in energy production. In our environment, we consider batteries or super-capacitor systems that consume a constant energy when charging and produce a constant energy when discharging. In addition, the operation of energy storage systems has to be carefully designed and controlled to protect them from damages that are: overcharging and overdischarging. Therefore, each storage system is also characterized by a maximum and minimum state-of-charge (SoC). The energy storage systems have three required states: $charging$, $discharging$ and $idle$. The electrical power consumed or released by the component highly depends on the current operational state and the component's state-of-charges.
- **Energy loads** represent production machines that consume energy in order to execute a production task. A production machine can have different operation states: $poweredoff$, $executing$, $standby$, etc... Depending on the production task and the operation state, a production machine can follow a predefined or a characteristic load profile. For a fixed set of production tasks, the load profile of a machine can be measured and approximated. In our setting, we consider 3 basic load profile as illustrated in Fig. 3. Any machine's load profile can be seen as a linear combination of this basic load profiles. In isolated mode, the micro-grid cannot guarantee continuous power supply to the load because it is often influenced by the unpredictable power generation. When the generated power is not able to drive the loads, the noncritical loads must shut down.
- **Regenerative energy consumers** are energy consumers with the exception that these consumers can recuperate energy for a small period of time (less than a minute). In this case we consider the load to be constant and negative. Fig. 4 shows a typical load profile.
- **Auxiliary processes** are energy consumer processes in the factory floor that do not execute a manufacturing task directly, but are required by the production task. These are for example compressors or cooling systems. Auxiliary processes have two required states: off and on.
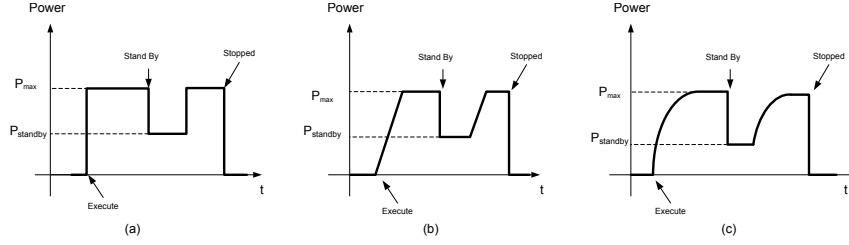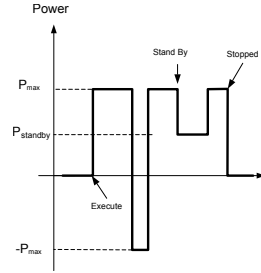
Fig. 3 Basic load profiles of the energy loads. In *execute*-state, the electrical power consumed P_i can be constant and equal to P_{max} (a), increase linearly (b) or exponentially (c) to P_{max}.

Fig. 4 Basic load profiles of regenerative energy consumers. In *execute*-state, the electrical power consumed P_i can be negative for a short period of time.

- The **grid component** represents the main grid and is responsible for buying/selling electricity from/to the main grid. It has three state: *buying*, *selling*, *idle*. If the grid agent is in *idle*-state or in *off*-state, the industrial micro-grid can be considered to be executed in isolated mode because there are no interaction with the main grid.

3.2 Problem Formulation

The main objective is to minimize the total cost the energy bought from the main grid to achieve a high productivity while considering future energy prices and weather dependent renewable energy generators. Let M denotes the set of components of the micro-grid (or controllable machine components), such that $M_i \in M, \forall i \in [0, M-1]$ with $M \in \mathbb{N}$. We assume that each component M_i can take an energy state $s_{i,t}$ (i.e. "stopped", "running", "aborted", "standby", etc.) at the time step $t \in \mathbb{N}$ and the dynamic power consumption P_i^t of M_i solely depends on the current state $s_{i,t}$ of the component and the current time step t: $P_i^t = P_i(t, s_{i,t})$. The total energy requested/sold from/to the main grid E_i^t at time interval $\triangle t$ over the complete time horizon $T \in \mathbb{N}$ is therefore the sum of the power consumed/generated of all components during the time horizon. Please notice that P_i^t can also be negative in the case of renewable energy sources or regenerative energy consumers for example.

$$E_i = \sum_{t=0}^{T-1} P_i^t \cdot \triangle t, \forall i \in [0, M-1] \tag{2}$$

If $\lambda_t^- \in \mathbb{R}$ is the actual energy price and $\lambda_t^{\sim} \in \mathbb{R}$ the forecasted energy price at a time step t, the optimization problem can be formulated for the time horizon T as follow:

$$minimize \sum_{k=0}^{t} \lambda_t^- \cdot (\sum_{i=0}^{M-1} P_i(k, s_{i,k}) \cdot \triangle t) + \sum_{l=t+1}^{T-1} \lambda_t^{\sim} \cdot (\sum_{i=0}^{M-1} P_i(l, s_{i,l}) \cdot \triangle t) \tag{3}$$

If $a_l(s_{i,l-1}, s_{i,l})$ denotes the action of changing the state of the component M_i at time l from the state $s(i, l-1)$ to the state $s(i, l)$, the Eq. 3 can be rewritten as follow:

$$minimize \sum_{k=0}^{t} \lambda_t^- \cdot (\sum_{i=0}^{M-1} P_i(k, s_{i,k}) \cdot \triangle t) + \sum_{l=t+1}^{T-1} \lambda_t^{\sim} \cdot (\sum_{i=0}^{M-1} P_i(l, a_l(s_{i,l-1}, s_{i,l})) \cdot \triangle t)$$
$$\tag{4}$$

Several constraints should be taken into consideration. One of them is the power balance between the energy demand of the micro-grid and the energy supply from the main grid:

$$\sum_{i=0}^{M-1} E_i \le E_{max}^{\sim}, \forall i \in [0, M-1] \tag{5}$$

where E_i denotes the total energy demand of the micro-grid component M_i over the time horizon T and $E_{max}^{\sim}$ the total available energy from the main grid.

In addition, the total energy consumption of the micro-grid at each time interval t should satisfy the upper and lower bound given by the overall load profile and a tolerance interval. This constraint can be expressed as follows:

$$L_{target}^{\sim} - \triangle L^{\sim} \le \sum_{i=0}^{M-1} P_i(t, s_{i,t}) \le L_{target}^{\sim} + \triangle L^{\sim} \tag{6}$$

with $L_{target}^{\sim}$ being the target load at time step t, $\triangle L^{\sim}$ a symmetric tolerance around $L_{target}^{\sim}$ and $i \in [0, M-1]$

As expressed above, solving the optimization problem is equivalent to compute at each time step the optimal state (by choosing the action $a_l(s_{i,l-1}, s_{i,l})$) so that the resulting total energy cost of the micro-grid for a complete time horizon is globally minimized. If every component of the micro-grid is controlled by an agent, then the optimization problem is equivalent to finding a coordinated strategy for all the agents. In this case, the main challenge lies in the volatility of future energy prices and the direct dependence of future energy consumption from actual decisions. Furthermore, other constraints such as the the throughput, production time, the product quality, the production cost, etc. have to be considered.

3.3 Markov Game Formulation

The formulated optimization problem can be transformed into a reinforcement learning task, where each agent controlling a component of the grid has to learn a policy that maximizes a cumulative reward signal derived from the objective function and is conditioned by the constraints formulated in Eq. 5 and Eq. 6.

If we consider the global state of the environment as the aggregation of the observations of all agents, then the observed state of the environment solely depends on the actions of all the agents and the previously observed global state of the environment. In other words, no external factors besides the actions of all the agents will affect the dynamics of the environment. Under this assumption, the multi-agent reinforcement-learning task satisfies the Markov property and can be therefore formulated as a Markov decision process (MDP).

In this work, we consider a multi-agent extension of Markov decision processes called observable Markov games [28]. A Markov game for N agents is defined by a set of states S describing the possible configurations of all agents, a set of actions $A_1, A_2, \ldots, A_N$ and a set of observations $O_1, O_2, \ldots, O_N$ for each agent. To choose actions, each agent $i(i = 1, 2, \ldots, N)$ uses a stochastic policy $\pi_{\theta_i} : O_i \times A_i \mapsto [0; 1]$, where θ_i are the parameters of the policy. The policy produces the next state according to the state transition function $T : S \times A_1 \times \ldots \times A_N \mapsto S$. Each agent i obtains rewards as a function of the state and agent's action $r_i : S \times A_i \mapsto \mathbb{R}$, and receives a private observation correlated with the environment state $o_i : S \mapsto O_i$. The initial states are determined by a distribution $: S \mapsto [0; 1]$. Each agent i aims to maximize its own total expected return $R_i = \sum_{t=0}^{T} \gamma^t \cdot r_i^t$ where γ^t is a discount factor at time step t and T is the time horizon. The *action-value function* is defined as $Q^{\pi_i}(s_{i,t}, a_t^i) = \mathbb{E}[R_i^t | s_{i,t}, a_t^i]$, while the *state-value function* is defined as $V^{\pi_i}(s_{i,t}) = \mathbb{E}[R_i | s_{i,t}]$. The *advantage function* $A^{\pi_i}(s_{i,t}, a_t^i) = Q^{\pi_i}(s_{i,t}, a_t^i) - V^{\pi_i}(s_{i,t})$ describes whether taking action a_t^i is better or worse for agent i when in state $s_{i,t}$ than the average action of policy π_{θ_i}.

For our micro-grid, we provide a uniform representation of the operational state of the grid components which is illustrated in Fig. 5. The state space of every agent is represented by the operation state of the grid component it controls aggregated with the target load of the grid, the current time step within the production shift, the amount of production tasks to execute as well as the current energy price. Notice that we do not assume inter-dependencies between the production tasks. The action space of each agent is represented by all the actions that can change the operational state of a grid component (See Fig. 5). The reward of each agent depends on the type of the grid component it controls and therefore is use case dependent. See Eq. 7 in section 4.2 for more details.
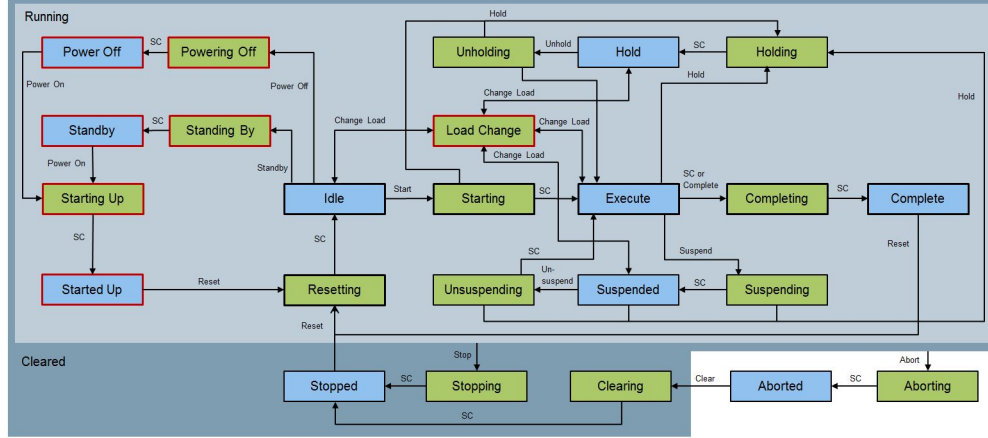
Fig. 5 Uniform state representation of a grid component. Every component can be stopped, halted, suspended, powered up or aborted. Once the component reach the $execute$-state then it begins to execute a production task, to generate energy or to store/release energy.

4 Approach

4.1 Multi-Agent Proximal Policy Optimization

Since the naive application of standard reinforcement learning algorithms performs poorly in multi-agent settings, our goal is to derive an algorithm that can operate under the following constraints:

- The learned policies can only use local information (i.e. own observations) at execution time.
- There is no particular communication method between the agents, since it can restrict the scalability of the solution.
- We do not assume a differential model of the environment dynamics.

Fulfilling the above desiderata would provide a general-purpose multi-agent learning algorithm that could be applied not just to cooperative games with explicit communication channels, but also to competitive games and games involving only physical interactions between agents.

We accomplish our goal by adopting the framework of centralized training with decentralized execution in which agent policies are composed of two separate networks with different parameters: a policy network (actor) that generates an action distribution and a critic network (critic) that predicts discounted future returns. The critic uses extra information to ease training process whilst actors take actions based solely on their own local observations. This is still a valid approach as long as the information used during the training of the agents are not used at test/execution time.

As illustrated in Fig. 6, our centralized action-value function (critic) for each agent takes as input the actions and observations of all agents and predicts the discounted
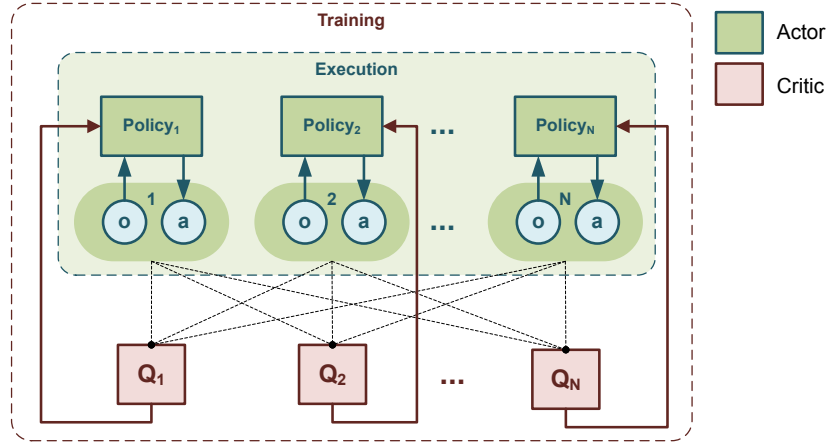
Fig. 6 Overview of our multi-agent decentralized actor, centralized critic approach. The centralized action-value function (critic) for each agent takes as input the actions and observations of all agents and predicts the discounted future returns (Q-values) for the agent while the actor produces an action distribution based solely on the agent's observations and actions.

future returns for the agent. Additional information can also be included in the inputs. Since each action-value function is learned separately, each agent can have arbitrary rewards including conflicting rewards in competitive scenarios.

Furthermore, we extend the idea of [6] by adopting the proximal policy optimization algorithm (PPO) as the main policy optimization algorithm instead of deep deterministic policy gradient (DDPG) as presented in the paper. PPO is a family of policy optimization methods that use multiple epochs of stochastic gradient ascent to perform each policy update. It performs comparably to or better than DDPG while being much simpler to implement and tune.

4.2 Cooperation and Competition

As we seek the emergence of coordination strategies between the components of the micro-grid, the agents controlling the mico-grid components are trained using *self-play* in an *autocuricula* configuration. This acts as a natural curriculum as these agents are always in a mixed cooperative and competitive setting. For example, the energy loads (i.e. production machines) of the micro-grid are always in competitive settings. If there is not enough electrical energy available in the grids, then some production machines have to turn themselves off so that the production can be carried on by the other machines. In the same direction, cooperation can take place if, for example, the energy storage components need to be switched to the discharging state in order to increase the electrical energy available on the energy bus and thus enable the production machines to perform additional production tasks.

Adopting such a configuration also offers many advantages. If a new successful strategy is adopted by an agent, it implicitly changes the task distribution for the competing agents which are then forced to develop a better strategy. These evolutionary arms races create an implicit autocurriculum in which competing agents constantly create new tasks for each other and thus discover new powerful and robust strategies and counter-strategies. In addition, the multi-agent autocurriculum is open-ended. That means that the training will only stabilizes once equilibrium between the different agents is founded which can only be reached if a suitable coordination strategy between the agents has emerged. Due to the exploration factors, many coordination strategies can also emerge at different stages of the training.

In order to create a competitive setting, competing agents are rewarded with competitive rewards. If an agent wins the game, it receives a positive reward and all other competing agents are penalized by an opposite negative reward. If nobody wins the game, then all competing agent are penalized. In a cooperative setting, agents are given a team based reward; meaning that all members of the team receive a positive reward, if a team member wins the game. To confine agent behavior to a reasonable space, we also introduced an environmental-based penalization.

The success of agents in the autocuricula setting requires the agents to occasionally solve the task (win the game) by random actions. The probability of this happening in most games is minuscule as they require as a prerequisite some fundamental skills like the ability to properly control the operational state machine of the grid component. For example the agent must first quickly learn how to bring a production machine in the *execute*-state in order to start a production task or how to turn on/off the storage component without suspending/halting the production. To overcome this problem, we use simple dense rewards at each step so that the agents can first learn basic motor skills that increase the likelihood that random actions of the agent will result in a relatively small positive reward. We refer to this reward as the *exploration reward*. The exploration reward is then gradually annealed to zero, in favor of the *competition/cooperation reward*, to allow the agents to learn coordination strategies for the remaining training time. This is achieved using a linear annealing factor α. We also introduce an *environmental reward* which aims to penalize action sequences leading to dangerous outcomes (for example system configurations violating the constraints formulated in Eq. 5 and Eq. 6) and a *productivity reward* which encourage production machines to execute production tasks more frequently.

So, at time-step t, if the exploration reward is r_{explo}, the competition reward is r_{comp}, the environmental reward r_{env}, the the productivity reward r_{prod} and T is the termination time-step, then the total agent reward is:

$$r_t = \alpha \cdot r_{explo} + (1 - \alpha) \cdot (r_{comp} + r_{env} + r_{prod}) \qquad (7)$$

With this formulation, the agents are trained initially on the dense reward for only about 10-20% of the training epochs in order to gain some basic skills first.

5 Experiments

Our aim is to show that the agents controlling an industrial micro-grid can learn complex behaviors and coordination strategies that can lead to an optimization of the global energy consumption. Therefore, we train the agents in different competitive and cooperative scenarios. In each scenario, the environment consists of multiples agents with discrete action space, discrete time space and mixed observation spaces. Every agent controls a grid component and may take physical actions in the environment by changing the operational state of the corresponding grid component. We do not assume that all agents have identical action and observation spaces, or act according to the same policy π_θ. The environment of each experiment were implemented as an extension of OpenAI environment Gym [46]. We found this framework useful for creating environments involving complex interactions between agents, while keeping the control and perception problems simple, as we are primarily interested in addressing agent strategies.

Unless otherwise specified, the actor and the critic of each agent is parameterized by a two-layer ReLU MLP with respectively 256 units per layer for the actor and 512 units per layer for the critic. In all of our experiments, we use the Adam optimizer with a learning rate of 0.0001 and the discount factor γ is set to be 0.9. The size of the replay buffer is 10240 and the sampling time horizon is 1024. We update the network parameters after every 1024 sampling steps and we use a batch size of 512. We train with 10 random seeds for environments with stark success/fail conditions (*experiment 2* and *experiment 3*) and with only 2 seeds for *experiment 1*. For the PPO-algorithm, we set the clipping parameter ϵ to 0.1, the entropy coefficient to 0.005 and the exploration noise to 0.2.

This section describes in detail the designed experiments as well as the specific training parameters and analyzes various aspects of the learned behaviors. Source codes for the environments as well as learned policy parameters of the agents in all environments are open-sourced and available at [47].

5.1 Experiment 1: Autocurricula and Emergent Behavior

In this scenario, we consider a micro-grid of 4 identical production machines (energy loads) executing the same production task. The grid is operated in connected mode. However, the main grid can only provide enough energy for 3 production machines to operate. Therefore, the production machines are in full competition. The exploration reward of the agents is set so that the agent is penalized if the production machine is shut down or suspended. The competition reward is based on the performance of the machines meaning that the agent will receive a positive incentive every time a production task is accomplished. In addition, all agents are penalized if the total energy demand is greater than the energy provided by the main grid. The exploration annealing factor α is set to 0.2 so that the agent will first learn how to execute a production task during the initial 20% of the training time. During the training each

agent observed the full environment including the observations and the actions of the other agents. During the execution, the agents only observe the amount of energy available as well as the current operational state of the machine it controls. We train the agents with 2 random seeds for 500 sampling horizons.

Fig. 7 shows a typical progression of the emergent strategies learned by all agents during the course of the training measured in this case by the respective average cumulative rewards. Clearly, training the agents against each other creates many different strategies, each of which introduces a previously non-existent pressure on other agents to do better in order to move to the next stage. Note that there are no direct incentives for agents to explore, but rather the emergent strategies are solely a result of the auto-curriculum induced by multi-agent competition.

As illustrated in Fig. 7, the agents initially learn simultaneously how to control the corresponding production machine so that it can reach the *execute*-state as often as possible. After approximately 30 episodes, all agent have mastered this task and the energy demand of the whole grid begin to surpass the energy provided by the main grid. Consequently, the agents are then frequently penalized leading to a significant reduction of the production performance and the accumulated reward. At this point some agents begin to take advantage of the observations and actions of other agents and intentionally modify their policies accordingly. We can see, for example, some agents wait in the *idle*-state and then switch to the *execute*-state, when other agents
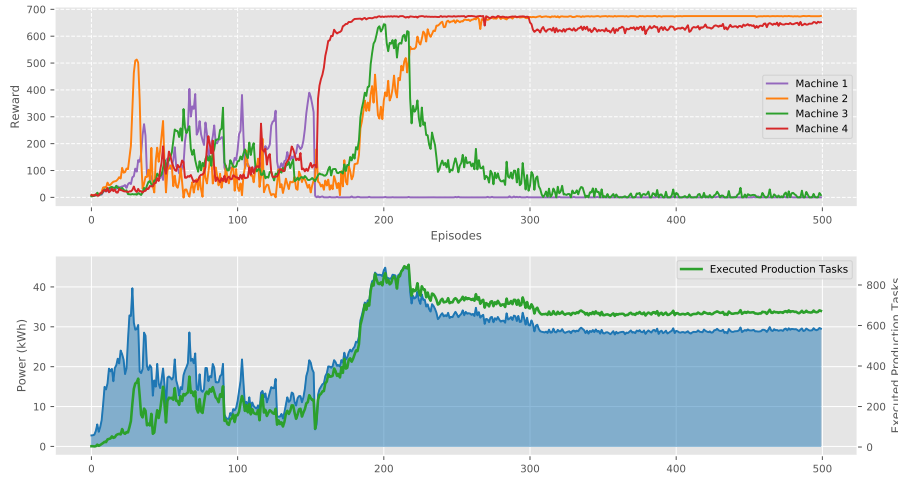

Fig. 7 Emergent Skill Progression From Multi-Agent Autocurricula. Through the reward signal agents go through different distinct stages of emergence. (a) Episode 1-30: the different agents learn basic skills (how to execute a production task) and the productivity rise. (b) Episode 30-160: strategy exploitation and competition take place between the agents and stabilize only when *machine 1* adopts a very bad strategy. (c) Episode 160-220: the first coordination strategy emerge. *Machine 2*, *Machine 3* and *Machine 4* start to produce more frequently while *Machine 1* remains in the *idle*-state more often. (d) Episode 220-500: a second coordination strategy emerge. *Machine 2*, *Machine 4* reached their highest productivity while *Machine 3* is frequently halted.

are stopped or halted. However, the more the new strategy becomes efficient, the more it becomes predictable and the other agents begin to defend themselves against this strategy. After another 5-20 episodes, all agents have now learned to exploit the strategy of other agents and the energy demand of the production machines again surpass the energy supply and the agents are penalized again more frequently. At this point a new adaptation/exploitation cycle restarts. Because of random exploration factors, the best performing agent during this phase changes from cycle to cycle. After approximately 3-6 cycles (which corresponds to episode 30 - 160), the training stabilized when at least one agent adopts a very bad policy e.g. only switching between $stopped$- and $idle$- state. The first acceptable coordination strategy emerges from episode 160 to approximately episode 220: *Machine 2*, *Machine 3* and *Machine 4* start to produce more frequently while *Machine 1* remains in the $idle$-state oftenly. It is a valid strategy since the energy required for the production can be supplied by the main grid and the overall productivity is relatively high. The second and final coordination strategy emerge at episode 220: while *Machine 2* and *Machine 4* reached their highest productivity, *Machine 3* is frequently halted.

In this experiment, we found that the emergent auto-curriculum is fairly robust, as similar observations can be made with different initialization strategies. Out of 10 experimental runs, more than 7 runs were able to converge to an equilibrium. However, many design choices can drastically reduce the convergence time and the sample complexity such as the surrogate loss and the entropy loss of the underlying PPO-algorithm. We also experimented with different configurations (for example by including energy storage and generator components) and found that these configurations also lead to emergent behaviors, providing further evidence that multi-agent interaction is a promising path towards self-supervised energy management.

5.2 Experiment 2: Fully Cooperative Production

In this scenario, the micro-grid is operated in isolated mode and consists of two production machines (a CNC-machine and a welding machine), an energy generator (a wind turbine) and an energy storage system. The energy generated by the wind turbine is lower than the energy needed by the production machines. Therefore, the energy storage system has to be turned on in order to execute a production task. All the components should therefore cooperate. The exploration rewards is similar to the previous experiment and the team-based cooperation reward of the agents is based on the performance of the production machines. We train the agents for 300 episodes. The agents also used the same policy network architecture as in experiment 1 (see section 5.1).

As we can observe in Fig. 8, the components rapidly learn the required basic skills and a coordination strategy emerge around the episode 100. While the energy generator is constantly switched on, the energy storage component learns to charge when the production machines are standing still and to discharge when the production machines perform a production task and thus supports production. Once the energy
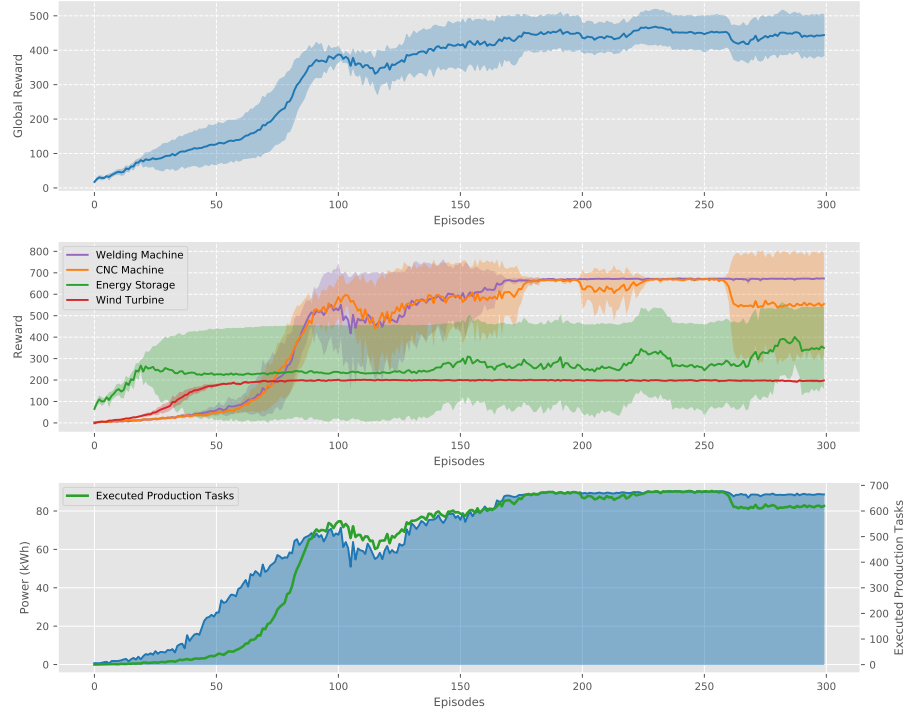
Fig. 8 Average rewards on cooperative production. A coordination strategy emerges rapidly after 100 episodes. While the energy generators rapidly learn how to generate energy, the energy storage component adopted to discharge when the production machines in the *execute*-state and thus allowing the production machine to increase the productivity.

generator and the energy storage component have adopted a strategy, the production machines rapidly learn to increase their productivity which reach its highest level around episode 250.

To determine the seed dependence of the skill emergence, we ran this environment with 10 random seeds and looked at the variability of the onset of behavioral shifts across runs. We find that the behavioral shifts happen at very similar times around episode 100 in training. Rewards and environment statistics are very consistent in early as well as in later times in training.

6 Conclusion and Future Work

This chapter presents an artificial intelligence algorithm especially a multi-agent reinforcement algorithm for the energy optimization of an heterogeneous cluster of flexible manufacturing machines with energy generation and energy storage capa-

bilities in an electricity micro grid featuring a high volatility of electricity prices. It demonstrated that simple game rules, multi-agent competition, and standard reinforcement learning algorithms at scale can induce agents to learn complex coordination strategies and skills that can lead to an optimization of the global energy consumption while ensuring a high productivity.

We analyze the performance of the proposed approach with respect to the number of agents and the micro-grid configuration in mixed cooperative and competitive scenarios and found that different production machines were able to discover different coordination strategies in other to increase the energy efficiency of the whole factory floor. These empirical results are promising and we intend to extend the solution to highly complicated and dynamic environments.

The presented results should be viewed as a proof of concept showing that multi-agent autocurricula can lead to physically grounded coordination strategies. We acknowledge that the strategy space in the proposed environments are inherently bounded and do not reflect the reality of the factory floor since many factors such as product quality, the production time and the unexpected human interventions in the production process have to be considered. However, because the algorithm was built on the functional abstraction of cyber-physical production systems, it is therefore applicable to real production systems if an invocation interface is connected to the agents. Our solution is therefore physically grounded and very extensible. In order to support further research, we have open-sourced our environment code [47].

References

1. A. Gholian, H. Mohsenian-Rad, and Y. Hua, "Optimal industrial load control in smart grid," *IEEE Transactions on Smart Grid*, vol. 7, pp. 2305–2316, Sep. 2016.
2. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning series)*. A Bradford Book, 2018.
3. V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing atari with deep reinforcement learning," *CoRR*, vol. abs/1312.5602, 2013.
4. OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang, "Solving rubik's cube with a robot hand," 2019.
5. M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," 2015.
6. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.
7. R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," 2017.
8. T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, and I. Mordatch, "Emergent complexity via multi-agent competition," 2017.

9. R. Chen, H. Sun, Q. Guo, H. Jin, W. Wu, and B. Zhang, "Profit-seeking energy-intensive enterprises participating in power system scheduling: Model and mechanism," *Applied Energy*, vol. 158, pp. 263 – 274, 2015.

10. Z. Wang, F. Gao, Q. Zhai, X. Guan, K. Liu, and D. Zhou, "An integrated optimization model for generation and batch production load scheduling in energy intensive enterprise," in *2012 IEEE Power and Energy Society General Meeting*, pp. 1–8, July 2012.

11. A. Middelberg, J. Zhang, and X. Xia, "An optimal control model for load shifting – with application in the energy management of a colliery," *Applied Energy*, vol. 86, no. 7, pp. 1266 – 1273, 2009.

12. Z. Sun and L. Li, "Opportunity estimation for real-time energy control of sustainable manufacturing systems," *IEEE Transactions on Automation Science and Engineering*, vol. 10, pp. 38–44, Jan 2013.

13. J. R. Duflou, J. W. Sutherland, D. Dornfeld, C. Herrmann, J. Jeswiet, S. Kara, M. Hauschild, and K. Kellens, "Towards energy and resource efficient manufacturing: A processes and systems approach," *CIRP Annals*, vol. 61, no. 2, pp. 587 – 609, 2012.

14. A. Pechmann and I. Schöler, "Optimizing energy costs by intelligent production scheduling," in *Glocalized Solutions for Sustainability in Manufacturing* (J. Hesselbach and C. Herrmann, eds.), (Berlin, Heidelberg), pp. 293–298, Springer Berlin Heidelberg, 2011.

15. A. Bruzzone, D. Anghinolfi, M. Paolucci, and F. Tonelli, "Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops," *CIRP Annals*, vol. 61, no. 1, pp. 459 – 462, 2012.

16. K. Fang, N. Uhan, F. Zhao, and J. W. Sutherland, "A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction," *Journal of Manufacturing Systems*, vol. 30, no. 4, pp. 234 – 240, 2011. Selected Papers of 39th North American Manufacturing Research Conference.

17. C. Liu, X. Xu, and D. Hu, "Multiobjective reinforcement learning: A comprehensive overview," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, pp. 385–398, March 2015.

18. C. W. Anderson, D. C. Hittle, A. D. Katz, and R. Kretchmar, "Synthesis of reinforcement learning, neural networks and pi control applied to a simulated heating coil," *Artificial Intelligence in Engineering*, vol. 11, no. 4, pp. 421 – 429, 1997. Applications of Neural Networks in Process Engineering.

19. L. Li and Z. Sun, "Dynamic energy control for energy efficiency improvement of sustainable manufacturing systems using markov decision process," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 43, pp. 1195–1205, Sep. 2013.

20. F. Ruelens, B. J. Claessens, S. Vandael, S. Iacovella, P. Vingerhoets, and R. Belmans, "Demand response of a heterogeneous cluster of electric water heaters using batch reinforcement learning," in *2014 Power Systems Computation Conference*, pp. 1–7, Aug 2014.

21. F. Ruelens, B. J. Claessens, S. Quaiyum, B. De Schutter, R. Babuška, and R. Belmans, "Reinforcement learning applied to an electric water heater: From theory to practice," *IEEE Transactions on Smart Grid*, vol. 9, pp. 3792–3800, July 2018.

22. O. De Somer, A. Soares, K. Vanthournout, F. Spiessens, T. Kuijpers, and K. Vossen, "Using reinforcement learning for demand response of domestic hot water buffers: A real-life demonstration," in *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, pp. 1–7, Sep. 2017.

23. H. Kazmi, F. Mehmood, S. Lodeweyckx, and J. Driesen, "Gigawatt-hour scale savings on a budget of zero: Deep reinforcement learning based optimal control of hot water systems," *Energy*, vol. 144, pp. 159 – 168, 2018.

24. B. Jiang and Y. Fei, "Smart home in smart microgrid: A cost-effective energy ecosystem with intelligent hierarchical agents," *IEEE Transactions on Smart Grid*, vol. 6, pp. 3–13, Jan 2015.

25. A. Anvari-Moghaddam, A. Rahimi-Kian, M. S. Mirian, and J. M. Guerrero, "A multi-agent based energy management solution for integrated buildings and microgrid system," *Applied Energy*, vol. 203, pp. 41 – 56, 2017.

26. A. Prasad and I. Dusparic, "Multi-agent deep reinforcement learning for zero energy communities," 2018.

27. R. Lu, S. H. Hong, and X. Zhang, "A dynamic pricing demand response algorithm for smart grid: Reinforcement learning approach," *Applied Energy*, vol. 220, pp. 220 – 230, 2018.
28. M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *ICML*, p. 157–163, 1994.
29. J. Hu and M. P. Wellman, "Multiagent reinforcement learning: Theoretical framework and an algorithm," in *ICML*, 1998.
30. L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, pp. 156–172, March 2008.
31. M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *In Proceedings of the Tenth International Conference on Machine Learning*, pp. 330–337, Morgan Kaufmann, 1993.
32. L. Matignon, G. J. Laurent, and N. L. Fort-Piat, "Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems," *Knowledge Eng. Review*, vol. 27, pp. 1–31, 2012.
33. A. Tampuu, T. Matiisen, D. Kodelja, I. Kuzovkin, K. Korjus, J. Aru, J. Aru, and R. Vicente, "Multiagent cooperation and competition with deep reinforcement learning," 2015.
34. J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Autonomous Agents and Multiagent Systems* (G. Sukthankar and J. A. Rodriguez-Aguilar, eds.), (Cham), pp. 66–83, Springer International Publishing, 2017.
35. J. Foerster, N. Nardelli, G. Farquhar, T. Afouras, P. H. S. Torr, P. Kohli, and S. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," 2017.
36. R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *ArXiv*, vol. abs/1706.02275, 2017.
37. J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *AAAI*, 2017.
38. S. Sukhbaatar, a. szlam, and R. Fergus, "Learning multiagent communication with backpropagation," in *Advances in Neural Information Processing Systems 29* (D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, eds.), pp. 2244–2252, Curran Associates, Inc., 2016.
39. I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," *CoRR*, vol. abs/1703.04908, 2017.
40. J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," *CoRR*, vol. abs/1605.06676, 2016.
41. T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. N. Foerster, and S. Whiteson, "QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning," *CoRR*, vol. abs/1803.11485, 2018.
42. P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and et al., "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, (Richland, SC), p. 2085–2087, International Foundation for Autonomous Agents and Multiagent Systems, 2018.
43. H. He, J. Boyd-Graber, K. Kwok, and H. Daumé, "Opponent modeling in deep reinforcement learning," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, p. 1804–1813, JMLR.org, 2016.
44. J. Choi, B.-J. Lee, and B.-T. Zhang, "Multi-focus attention network for efficient deep reinforcement learning," *ArXiv*, vol. abs/1712.04603, 2017.
45. J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, (Red Hook, NY, USA), p. 7265–7275, Curran Associates Inc., 2018.
46. G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.
47. J. Bakakeu, "Multi-agent-rl-energy-management." https://github.com/jupiterbak/Multi-Agent-RL-Energy-Management, 2020.