Training an Energy Management Simulation with Multi-Agent Reinforcement Learning

Alexander Haemmerle [*], Kapil Deshpande [*], Philipp Moehl [*], Georg Weichhart [*]
[*]Robotics and Automation Systems
PROFACTOR GmbH, Steyr, Austria
https://www.profactor.at/
emails: {alexander.haemmerle,kapil.deshpande,philipp.moehl,georg.weichhart}@profactor.at

***Abstract*—In this paper, we report on the application of multi-agent reinforcement learning to the development of a microgrid energy management simulation. The simulation is made up of energy producers and consumers as well as storage devices. We regard these components as agents that are trained in a shared environment with reinforcement learning. A significant share of energy production in the microgrid is provided by renewable energy sources with stochastic characteristics, e.g., photo-voltaic installations. The stochastic nature of such producers, as well as of consumers, is captured in energy consumption/production profiles that are used for training the respective agents. For our results, the agents have been trained with an actor-critic algorithm, using real-world energy profile data for photo-voltaic installations and industrial consumers in Austria. A centralised critic addresses the multi-agent nature of the energy management problem. Running what-if analyses is an application scenario for the trained simulation. In such analyses the effects of different microgrid configurations on energy management performance can be investigated. The presented work has been conducted in the context of the projects RESINET and Zer0p.**

***Index Terms*—Energy Management; Multi-Agent Reinforcement Learning; Photo-Voltaic; Battery Storage; Microgrid.**

I. Introduction

Green Industry for a sustainable and economically prosperous future is becoming a reality [1]. One of the pillars for Green Industry is the introduction of renewable energy sources into the energy supply for industrial companies. In many cases, the installment of a renewable energy source will be local, e.g., photo-voltaic (PV) panels on a plant roof. Because of the stochastic nature of prominent renewable energy sources like PV panels and wind turbines, such installments have to be accompanied with adequate storage systems to absorb excess renewable energy, and to provide energy in times of low energy production. However, companies that are willing to invest into local renewable energy sources are facing a decision problem, as they have to decide upon the right size of the installation, i.e., storage capacity and power output. The ”right size” does not only depend on the company's energy consumption profiles, but also on local (meteorological) conditions that impact renewable energy production. An energy consumption/production profile is a time series of power values. In our work, we use real-world data from Austria, containing industrial consumption profiles as well as energy production profiles from PV installations.

The above decision problem motivated the following research question: Is it possible to develop an energy management simulation for a microgrid, leveraging historic data for renewable energy production profiles and consumption profiles? Such a simulation would then allow to run what-if analyses with different sizes of local renewable energy installations.

The research question contains two important aspects: leverage data, and energy management, which is a sequential decision-making problem in nature. These aspects motivated the usage of Deep Reinforcement Learning (DRL) to tackle the research question. Moreover, an energy management system contains multiple agents with potentially selfish agendas. However, in order to meet system-wide objectives (e.g., load balancing) the individual agents have to cooperate. To also meet the multi-agent character of energy management, we employ Multi-Agent Reinforcement Learning (MARL).

The remainder of the paper is structured as follows. In section II we present related work and argue our contribution, the proposed approach is described in section III, results are reported in section IV, and finally section V presents conclusions and further research steps.

II. Related Work

In recent years, the application of Reinforcement Learning (RL) to Energy Management (EM) problems in microgrids attracted some attention. The literature can be classified into single-agent and multi-agent approaches, which is reflected in the following subsections.

A. Single-Agent Reinforcement Learning

The work of Qin et al. [2] proposes a novel privacy preserving load control scheme for a residential microgrid, where a central operator controls a number of smart homes. In fact, preservation of privacy could be achieved with a multi-agent RL approach, where each smart home is represented by an agent. The observation of such an agent would then include privacy information, which is not visible in other agents‘ observations. However, the authors choose a single-agent reinforcement learning approach, where the microgrid operator is represented as an agent, and the agent is trained with an actor-critic algorithm. The authors argue that the integration of a recurrent neural network with the deep learning model solves the privacy issue.

Muriithi and Chowdhury [3] deal with EM in a microgrid consisting of a PV installation, a Battery Energy Storage System (BESS) and local loads. The microgrid can exchange energy with the main grid. The EM objective function is the minimisation of energy cost, where a battery degradation cost model is taken into account. Single-agent RL with Q-learning is applied to solve the EM problem, where the actions to be learned by the agent are discrete charging/discharging actions for the BESS.

The microgrid architecture in Ji et al. [4] consists of distributed generators, a BESS, renewable energy production from a PV installation and a wind turbine, and some local consumption loads. The microgrid is connected to the main grid. A deep Q-network algorithm is used to train the microgrid controller agent. The EM objective is to find cost-efficient energy generation schedules for the distributed generators, thus the action space for the agent does not contain demand response actions.

B. Multi-Agent Reinforcement Learning

In Samadi et al. [5], they deal with a microgrid composed of renewable energy sources (wind and PV), an electrical energy storage system, and combined heat and power producers as well as a diesel generator as controllable energy producers. Additionally there are several electrical and thermal energy consumers, respectively. The microgrid is connected to the main grid. The paper proposes a multi-agent, market-based approach to EM, i.e., each energy producer/consumer is represented as an agent that sells/buys energy in the microgrid EM market. The EM goal is then to minimise energy cost for consumers in the microgrid, and Q-learning is used to train the agents towards the EM goal. Agents representing renewable energy sources actually do have no action choices, they just submit the produced energy per time step to the energy market, and thus these agents are not part of training.

The microgrid in Foruzan et al. [6] is composed of energy sellers (PV and wind generators as well as diesel generators) and energy buyers, a storage system and a connection to the main grid. In an auction-based market approach each microgrid component is represented as an agent, and the EM goal of an agent is to maximise its profit. Q-learning is applied to learn optimal agent policies.

In Fang et al. [7], a residential microgrid is modeled as an auction-based marketplace for renewable energy production agents, an agent repesenting a set of residential loads, and an agent representing a fleet of electric vehicles that can serve as storage system for the microgrid. With Q-learning the agents learn to maximise their individual profits, and the overall EM goal is to reach a Nash-equilibrium for the microgrid.

The work of Fang et al. [8] considers a regional microgrid with PV installations, wind turbines and micro turbines as producers, distributed batteries as storage, and industrial and residential loads as consumers. The microgrid is connected with the main grid. The EM problem is modeled as a double auction market, with seller agents (producers), buyer agents (consumers) and management agents being responsible for inter-agent communication and auction clearing. A deep Q-network algorithm is applied to train the non-management agents, where each agent has an individual Q-network, learning Q values for the agent's individual observation and the agents' joint actions as input. An equilibrium selection process in the training process ensures convergence towards a Nash equilibrium with respect to the agents' Q values.

The work of Xu et al. [9] targets a residential microgrid connected to the main grid, with a PV installation and various residential loads, e.g., electric vehicle and air conditioning. The multi-agent EM system consists of four agents, corresponding to non-shiftable loads, time-shiftable loads, power-shiftable loads and electric vehicle load, thus only demand side actions are considered in the EM system. For agent training, a Q-learning algorithm is integrated with a neural network model predicting energy price and PV generation. The reward scheme considers energy cost and so-called dissatisfaction terms penalising load shifting.

C. Discussion

The related work shows that algorithms based on Q-learning are predominantly used when it comes to solve EM problems with RL. The application of policy gradient methods, and here especially actor-critic methods, deserves more attention, as these methods show excellent performance in applications like mastering the games of Go [10] and Dota 2 [11], and solving Rubik's cube with a robot hand [12].

In many cases in the literature the reward schemes are rather simple, as they are used in market-based models for EM with cost minimisation as EM objective. In multi-objective EM the reward scheme has to reflect these objectives, and the impact of such complex reward schemes on RL peformance has to be investigated.

For training an energy management simulation including components with stochastic energy production/consumption characteristics it is essential that the simulation model is trained with a rich variety in data patterns, captured in energy production/consumption profiles. In the majority of contributions in the related work, energy profiles or predictors for renewable energy production and/or consumption are used. However, these profiles/predictors do rather show a limited variety in data patterns. On the one hand the time resolution is rather coarse (1h time steps), and on the other hand the profiles do look smoothened out, not exhibiting the stochastic variations found in raw energy profiles. Based on the above findings, our contribution can be summarised as follows.

1) A multidimensional reward scheme encodes the following EM objectives: a) follow a given energy profile as close as possible (for profile-driven agents), b) load balancing, i.e., matching energy production with consumption, c) minimise energy production from non-renewable sources, and d) correct charging behaviour of BESS: charge if excess renewable energy is available, discharge otherwise.

2) For agent training, we use an extension of Reinforcement Learning library (RLlib)'s implementation of a Proximal Policy Optimisation (PPO) algorithm, which is an actor-critic RL algorithm. The extension consists of a centralised critic approach, where a critic model (implemented as a deep neural network) processes all agents' observations and actions, and the agents share this critic model.
3) For training profile-driven agents, we use energy production and consumption profiles with a 15 min resolution, and due to the real-world character of these profiles, they show a considerable variety in data patterns being observed by the agents.

III. Proposed Approach

In this section, we describe the energy management system to be simulated, followed by a brief introduction into DRL and MARL. The main part of this section is then dedicated to the description of the training environment and the agents that make up the energy management system.

A. Energy Management in a Microgrid

Our work is based on a microgrid with five components: 1) PV energy producer, 2) fully controllable energy producer (e.g., diesel generator), 3) profile-driven energy consumer, 4) BESS, and 5) free acting energy consumer. In contrast to 3), the free consumer is not profile-driven, but it can freely absorb excess production that cannot be absorbed by 3) and 4). Energy management in the described microgrid is a sequential decision-making process. At any time step, the components adjust their loads in pursuing the following goals: a) load balancing, i.e., energy production should match consumption at any time, b) profile following, i.e., components 1) and 3) should follow given energy profiles as close as possible, c) the BESS component should charge if there is more renewable energy available than the profile-driven consumer needs, and BESS should discharge when not enough renewable energy is available to satisfy the profile-driven consumer's demand, d) the fully controllable producer should only produce energy if the profile-driven consumer's demand can not be satisfied by components 1) and 4). The components are configured with max/min State Of Charge (SOC) (for the BESS), max/min load (for all consumers and producers) and max increase/decrease in load from one time step to the next one (for all components).

B. Deep Reinforcement Learning

RL is learning to make decisions from interactions with an environment. Interactions are episodic, leading to a sequential decision-making process. The environment defines an observation space $\mathcal{S}$ and an action space $\mathcal{A}$. In every time step t, the RL agent receives an observation s_t and a reward r_t from the environment and chooses an action a_t, following a policy function $\pi(a_t|s_t)$. The learning goal is to maximize the expected cumulative reward,

$$R_t = \sum_{k=1}^{\infty} \gamma^k r_{t+k+1}, \quad \gamma \in (0, 1]. \tag{1}$$

For a state s, the value of a policy π is defined as: $v_\pi(s) = \mathbb{E}_\pi(R_t|s_t = s)$. Maximizing for the value function also leads to a maximization in the goal sense. We call this policy optimal. Taking action a in a state s leads to the action-value function of a policy π: $q_\pi(s, a) = \mathbb{E}_\pi(R_t|s_t = s, a_t = a)$.

With the latest achievements in Deep Learning (DL), new possibilities in many areas of machine learning arose. Especially the combination of DL and RL, DRL, achieved new impressive results in various fields, e.g., superhuman performance in video games. In DRL deep neural networks are used as function approximators for value and policy functions. Function approximation is crucial for larger spaces of states and/or actions, where a tabular representation is not feasible. It also enables the policy to be optimised directly, by searching in the policy space $\{\pi_\theta(a_t|s_t), \theta\}$ for optimal parameters θ of such a function approximation.

For neural networks, θ are the weights and biases and we can use the gradient ascent method (Baird and Moore [13]) to optimise, leading to a class of algorithms called policy gradient methods. The gradient of an objective function is representing an estimate, to update the parameters. A commonly used objective function for policy gradient methods is (cf. Schulman et al. [14]):

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t[\log \pi_\theta(a_t|s_t)\hat{A}_t] \tag{2}$$

where $\hat{A}_t$ is an estimator of the advantage function, describing the extra reward that could be obtained by taking action a_t.

Combining policy gradient methods with action-value functions leads to actor-critic methods. The actor approximates the policy, and the critic approximates the action-value function, thus criticising the actions taken by the policy.

C. Multi-Agent Reinforcement Learning

A generalisation of RL into multi-agent systems is MARL, where we study how multiple agents learn within a shared environment. A key challenge in MARL is the fact that other agents are part of the training environment, and they are modifying the environment with their actions. The observation that an agent receives does not only reflect the agent's action, but also the actions taken by other agents. In other words: in MARL multiple agents are interacting indirectly through their actions in the training environment. In the energy management case, load balancing requires coordinated actions from all agents. To be able to train coordinated actions, we used RLlib's implementation of a PPO algorithm and extended it with a centralised critic. The usage of a centralised critic approach is inspired by Yu et al. [15].

PPO is a new family of actor-critic methods, proposed by Schulman et al. [14]. With an adaptation of (2), an idea to stabilise training was introduced. The new objective constrains large policy changes, leading to smaller steps and enabling for multiple epochs of mini-batch updates. With the ratio between new and old policy $r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{old}}(a_t|s_t)$, the new objective is defined as:

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t] \tag{3}$$

where $clip(r_t(\theta), 1-\epsilon, 1+\epsilon)$ clips the ratio to the interval $[1-\epsilon, 1+\epsilon]$.

D. Training Environment and Agents

This subsection describes the development of a multi-agent compatible RL training environment, where the five components described in section III-A are represented as agents. The training objective is then to learn optimal policies to achieve the energy management goals described in section III-A.

1) Training Environment: For the development of the training environment RLlib's [16] multi-agent environment has been bootstrapped, which makes it compatible with OpenAI gym environments. The environment uses Box observation spaces and discrete action spaces for the agents. The decision to use discrete action spaces with PPO algorithm has been inspired by Tang and Agrawal [17], who state, "the discrete policy achieves significant performance gains with state-of-the-art on-policy optimization algorithms PPO". Tang and Agrawal [17] also give an optimum number of discrete sampling of a continuous action space which is (7-15) and as per our experiments 11 discrete actions gave the best results.

2) Agents: In the following agent configuration characteristics are described. For a profile-driven agent, the energy profile and the profile-corridor, introducing some tolerance for deviating from the profile, are important characteristics, while for BESS their initial SOC and minimum SOC play an important role. For all the agents, their max-load-diff denotes the maximum load difference between consecutive time steps, thus max-load-diff determines the agent's maximum speed of reaction. Another important configuration parameter for all agents is the load-balancing tolerance. If the absolute difference between total production and total consumption is smaller than the tolerance, load-balancing is achieved. The observation of an agent is composed of four time series with five time steps each, see Figure 1.

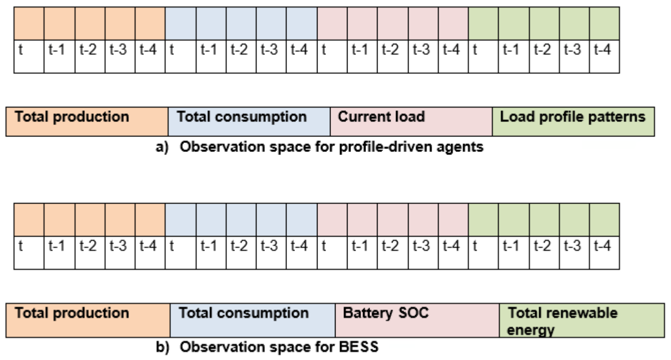


Fig. 1: Observation space for agents.

In the following the agents are specified in more detail.

- PV producer agent:
  This agent is a profile-driven agent that follows the energy production profiles of a PV panel, with a small tolerance with which the agent is allowed to deviate from its specific profile. The agent has a 20-dimensional observation space represented by Figure 1 (a), and a discrete action space of 11 non-negative numbers, with (0-4): decrease production load, 5: do nothing and (6-10): increase production load. The effective increase/decrease of production load is then (0.2, 0.4, 0.6, 0.8, 1.0) * max-load-diff.
- Profile-driven consumer agent:
  This agent is a profile-driven agent that follows power consumption profiles. The profile following tolerance, observation space and action space are the same as for the PV agent.
- BESS agent:
  This agent simulates the behaviour of a battery storage, where the main aim is to charge/discharge appropriately. It has a 20-dimensional observation space represented by Figure 1 (b), an action space of 11 non-negative numbers, with (0-4): battery discharges, 5: battery does nothing and (6-10): battery charges. In the BESS agent's context, max-load-diff is the maximum charging/discharging rate of a battery. The effective charging/discharging rate is called the battery magnitude and is calculated as (0.2, 0.4, 0.6, 0.8, 1.0) * max-load-diff. The BESS agent is further specified with initial battery SOC, minimum SOC and maximum SOC. The setting of maximum SOC value is decided based upon total renewable energy available after consumption has been satisfied. The setting of initial SOC is based upon the energy required for consumption at the initial part of the episode. Minimum SOC is chosen randomly, but it can be chosen considering safe operation of the battery.
- Fully controllable producer agent, free acting consumer agent:
  The observation space and the action space are the same as for the PV producer agent. An important configuration parameter for the fully controllable producer is the maximum power output that can be delivered into the microgrid. Profile-following is not of concern for these agents.

3) Deep Reinforcement Learning Model: The current implementation adapts the PPO implementation of RLlib, such that the agents share a centralised critic model. Figure 2 shows the DRL model used for each agent. The actor model has three layers, with the action logits in the final layer. The centralised critic model for each agent has three input layers. To understand the input layers let us assume that we have n agents. The first input layer corresponds to the agent's own observation with shape $(, 20)$, the second input layer processes the opponent agents' observations with shape $(, 20 * (n-1))$, and the third layer processes the opponent agents' actions with shape $(, 11 * (n-1))$. The three input layers are concatenated, followed by two dense hidden layers. The final layer outputs a single value, indicating how good the input is in terms of cumulative rewards over an episode. For all layers the first dimension in $(, \text{size})$ is not specified, as it depends on the mini-batch size used by the PPO algorithm.

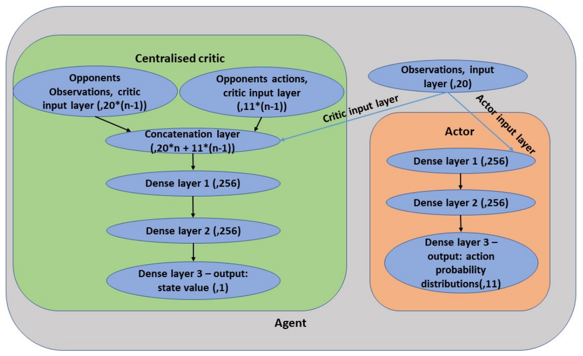

Centralised critic
Opponents Observations, critic input layer (,20*(n-1))
Opponents actions, critic input layer (,11*(n-1))
Concatenation layer (,20*n + 11*(n-1))
Dense layer 1 (,256)
Dense layer 2 (,256)
Dense layer 3 – output: state value (,1)
Observations, input layer (,20)
Critic input layer
Actor input layer
Actor
Dense layer 1 (,256)
Dense layer 2 (,256)
Dense layer 3 – output: action probability distributions(,11)
Agent

Fig. 2: DRL model.

4) Reward Scheme for Agents: The reward scheme for the agents in our energy management simulation is as follows:

- Profile-driven producer and consumer:
  For profile-driven agents there are only two penalties or rewards:
  - Profile deviation penalty

```
$L$ Current Agent Load
$PL$ Current Agent Profile Load
$CORR$ profile corridor tolerance
**if** abs($L$ - $PL$) <CORR **then**
  No Penalty
**else**
  Penalty = abs($L$ - $PL$) * const-penalty
**end if**
```

  - Load balancing reward

```
$TP$ Total Production
$TC$ Total Consumption
**if** abs($TP$ -$TC$) <balance-tolerance **then**
  Reward = constant
**else**
  No Reward
**end if**
```

- Fully controllable producer and free acting consumer:
  For the fully controllable producer and free acting consumer agents there are three penalties and rewards and they are as follows:
  - Load balancing reward
  - Excess-energy penalty and appropriate-energy reward

```
$R$ Renewable energy
$C$ Consumption, only profile-driven consumer
$BM$ Battery Magnitude charging/discharging
$BM$ >0.0 {charging}, $BM$ <0.0 {discharging}

$FCP$ Load of fully controllable producer
$FCC$ Load of free acting consumer
$EE$ = $R$ - $C$ - $BM$ Remainder energy after battery charges/discharges
**if** $R$ - $C$ >0.0 **then** {Excess renewable energy is available}
  **if** $FCP$ >0.0 **then** {$FCP$ should not get greater than zero}
    penalty $\propto$ $FCP$
  **end if**
  **if** abs($FCC$ - $EE$) >balance-tolerance **then** {$FCC$ should equate to remainder energy $EE$}
    penalty = constant
  **else**
    reward = constant
  **end if**
**else** {No excess renewable energy is available}
  **if** $FCC$ >0.0 **then** {$FCC$ should not get greater than zero}
    penalty $\propto$ $FCC$
  **end if**
  **if** abs($FCP$ -$EE$) >balance-tolerance **then** {$FCP$ production equates to remainder energy $EE$}
    penalty = constant
  **else**
    reward = constant
  **end if**
**end if**
```

- BESS:
  For the BESS agent there are three penalties and rewards and they are as follows:
  - Load balancing reward
  - Correct charging/discharging behaviour reward and incorrect behaviour penalty
    This reward scheme, allows to select a specific action from the action sets (0,10). The battery magnitude for discharging actions (0,4) is (-0.2,-0.4,-0.6,-0.8,-1.0) * max-load-diff, and for charging actions (6,10) it is (0.2, 0.4, 0.6, 0.8, 1.0) * max-load-diff, at action 5 the battery does nothing.

```
**if** consumption >PV production **then**
  action(0,4) is rewarded. The reward favours the action with the best effect on load balancing.
**else if** PV production >consumption **then**
  action (6,10) is rewarded. The reward favours the action with the best effect on load balancing.
**else**
  action 5 is rewarded
**end if**
```

IV. Results

This section describes results for training the EM simulation with the proposed reward scheme. The results have been achieved in the context of the projects RESINET and Zer0p.

A. Experiment Setup

The experimental setup is as follows. The PV producer agent and the profile-driven consumer agent are trained each with a single profile. The profiles are taken from real-world data for large-scale PV installations and industrial consumers in Austria, with maximum consumption loads at 1700 kW, and maximum production loads at 600 kW. For our experiments, we scaled down the power values of the producer by 10^5 and consumption by $6 * 10^5$. The decision to scale down the consumption further by a factor of 6 was taken to create a scenario where charging of the battery is feasible. Each profile has data of 96 time steps covering one day, which is then the episode length for RL training. The profile-corridor parameter for the profile-driven agents has been kept really small (0.0001), so that the agents are forced to follow their respective profile as close as possible. The load-balancing tolerance is set up as 0.5. For the BESS agent, the maximum SOC is 60, and both the initial battery SOC and the minimum battery SOC are at 2% of the maximum value. The max-load-diff parameters are set as follows. BESS: 2.5, PV producer: 0.4, profile-driven consumer: 0.25, fully controllable producer and free acting consumer: 0.5. With the above configuration, the agents are trained for 20,000 episodes. Figure 3 illustrates the training progress with different reward curves, x-axis units are time steps.

- rew_charging_mean: reward curve for correct charging/discharging behaviour of the BESS agent
- rew_dev_profile_mean: reward curve for profile following
- rew_total_load_mean and rew_total_out_mean : reward curve for load balancing
- rew_exp_energy_mean: reward curve for fully controllable producer
- rew_episode_reward_mean: total reward curve

B. Evaluation

Figures 4a and 4b illustrate the performance of the trained energy management simulation, x-axis units are time steps, y-axis units are arbitrary power units resulting from scaling down the real-world data. Figure 4b shows that load balancing is achieved quite accurately, plotting total energy production vs. total energy consumption. In Figure 4a the load curves of the individual agents are plotted, as well as the profiles for the profile-driven agents. The figure shows that profile-following is achieved quite accurately, both for the profile-driven consumer and the PV producer. The BESS agent shows proper charging behaviour: the agent does nothing in the initial part of the episode when there is no renewable power available, in the middle part it is charging when renewable power is available, and it discharges in the last part when there is only very little output from the PV producer. The fully controllable producer produces sufficient energy in the initial and last part of the episode to fulfill consumption demand, and it outputs near zero energy when renewable energy is available. The free acting consumer only consumes energy in the middle of the episode, where there is remainder energy from the PV producer that cannot be consumed by the profile-driven consumer and the BESS.

C. Scalability

In multi-agent systems scalability issues are of interest, where it is investigated how the number of agents impacts system performance. So far, our experiments with respect to scalability have been limited. For these experiments we used three agent types: PV producer, profile-driven consumer and fully controllable producer. We investigated different system configurations (I, J, K) where I denotes the number of PV producer agents, J denotes the number of profile-driven consumers and K denotes the number of fully controllable producers. We performed experiments with the following configurations: (1,1,1), (2,2,2), (5,5,3) and (5,5,5). We found that all configurations resulted in proper profile following and load balancing. Table I provides an overview of the conducted scalability experiments in terms of number of episodes used for training and training time on a state-of-the-art desktop PC.

TABLE I: SCALABILITY RESULTS

Configuration	Number of agents	Number of episodes	Training time
(1,1,1)	3	8000	40mins
(2,2,2)	6	16000	1hr 20mins
(5,5,3)	13	60000	4hr 30mins
(5,5,5)	15	60000	5hr 15mins

V. CONCLUSION

The results show that it is possible to train an energy management simulation for a microgrid, leveraging data for renewable energy production profiles and consumption profiles in the training process. A crucial point was the development of an appropriate reward scheme, enabling to learn the key desired agent behaviours ”profile following” and ”load balancing” in an energy management case with five agent types: profile-driven producer/consumer, fully controllable producer, free acting consumer and storage. In the following we outline our further research steps.

1) Quantitative analysis of simulation performance: for evaluation of the trained simulation we currently rely on a visual analysis of the results of simulation runs. For a quantitative analysis a metric has to be developed, measuring the performance of the multi-agent system when running a simulation. We envisage the application of simple statistical methods such as *Mean Absolute Error* and *Root Mean Square Error*, calculated over all time steps of a simulation run with respect to the desired behaviours ”profile following” and ”load balancing”.
2) Multi-profile training: so far, we have only considered single-profile training, i.e. the renewable energy producer and the profile-driven consumer, respectively, have been trained with one profile each. With multi-profile training, the goal is to achieve flexible agent behaviour, i.e., the trained behaviours should be able to cope with various situations. A situation is characterised by a

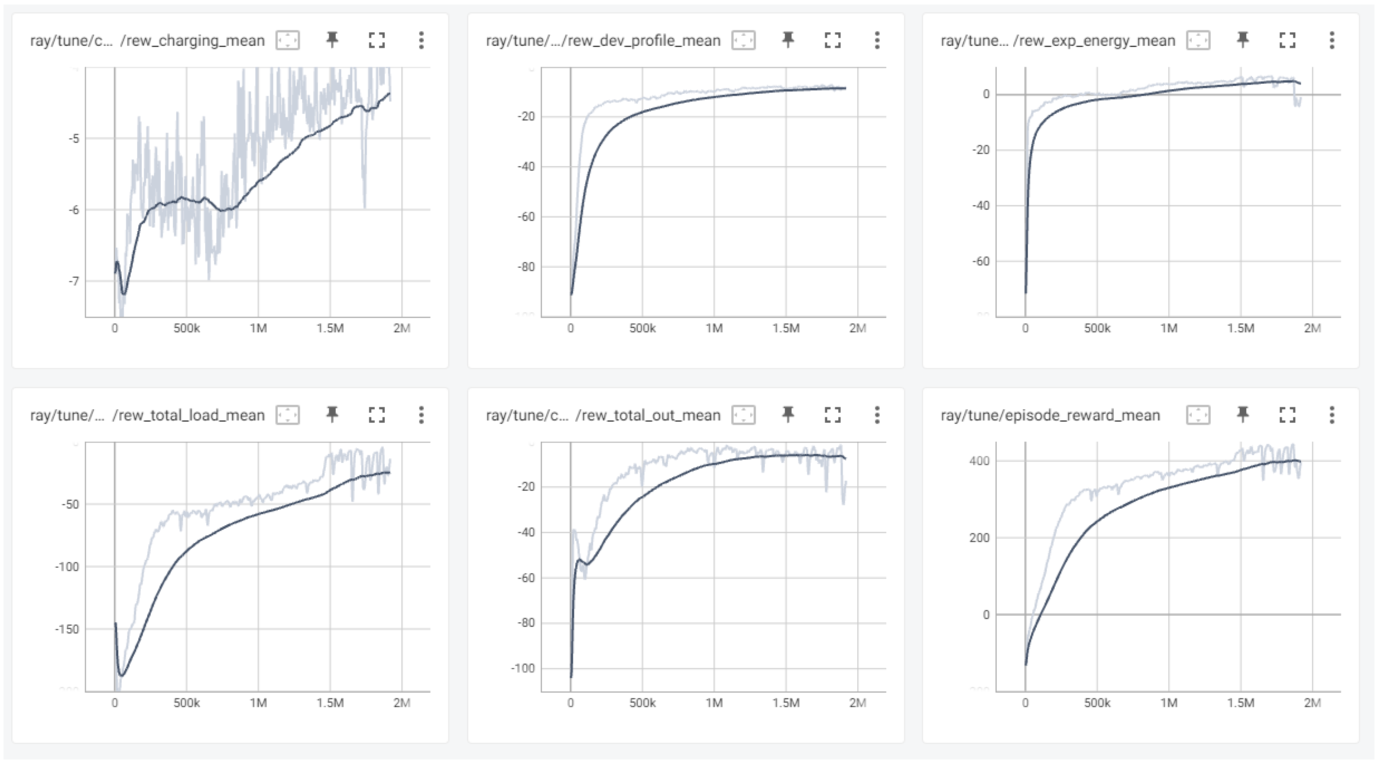
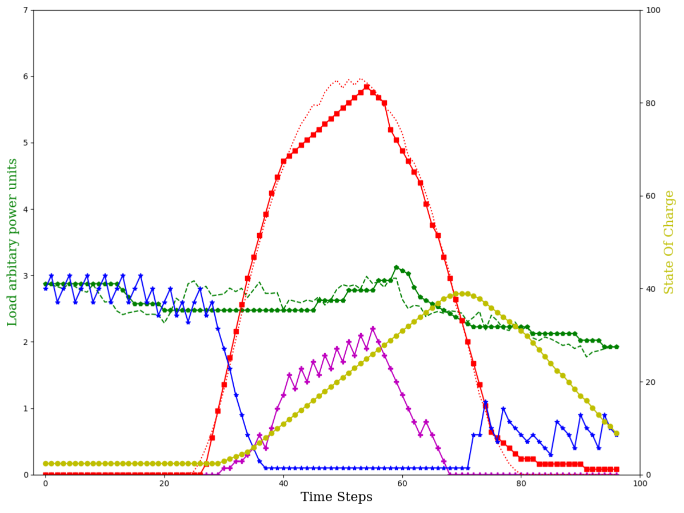
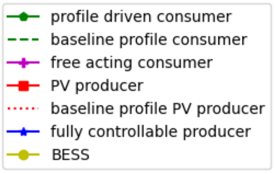
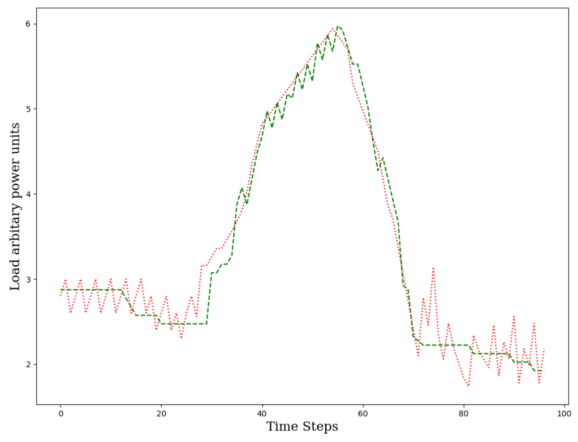


Fig. 3: Training progress.



(a) Agent load curves and profiles.





(b) Load balancing.

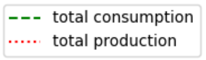


Fig. 4: Simulation results.

concrete pair of energy profiles, one for the renewable energy producer, the other one for the profile-driven consumer. To what extent such flexibility can be achieved is a future research question.

3) Up-scaling the number of agents: in future experiments we will intensify our research with respect to up-scaling the number of agents in the energy management system, thus increasing system size. We will investigate the question, how system size does influence training and simulation performance.

Acknowledgment

The research has been funded by the European Commission and the Government of Upper Austria through the program EFRE / IWB 2020, the priority REACT-EU and the project RESINET (RESIlienzsteigerung In energieNETzen – RESilience increase In energy NETworks; Nr.:WI-2020-701900/12). This work also received funding from the Government of Upper Austria through the program ”Programm des Landes Oö zur Stimulierung der Erschließung/Erweiterung von zukunftsweisenden Forschungsfeldern bei den Oö. außeruniversitären Forschungseinrichtungen im Zeitraum 2022-2029” and the project Zer0p (Zero Defect Manufacturing für nachhaltige Produktion; Nr.:WI-2021-299762/12-Au).

References

[1] UNIDO, “Green Industry.” [Online]. Available: https://www.unido.org/our-focus/cross-cutting-services/green-industry

[2] Z. Qin, D. Liu, H. Hua, and J. Cao, “Privacy Preserving Load Control of Residential Microgrid via Deep Reinforcement Learning,” *IEEE Transactions on Smart Grid*, vol. 12, no. 5, pp. 4079–4089, Sep. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9451164/

[3] G. Muriithi and S. Chowdhury, “Optimal Energy Management of a Grid-Tied Solar PV-Battery Microgrid: A Reinforcement Learning Approach,” *Energies*, vol. 14, no. 9, p. 2700, May 2021. [Online]. Available: https://www.mdpi.com/1996-1073/14/9/2700

[4] Y. Ji, J. Wang, J. Xu, X. Fang, and H. Zhang, “Real-Time Energy Management of a Microgrid Using Deep Reinforcement Learning,” *Energies*, vol. 12, no. 12, p. 2291, Jun. 2019. [Online]. Available: https://www.mdpi.com/1996-1073/12/12/2291

[5] E. Samadi, A. Badri, and R. Ebrahimpour, “Decentralized multi-agent based energy management of microgrid using reinforcement learning,” *International Journal of Electrical Power & Energy Systems*, vol. 122, p. 106211, Nov. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0142061520304877

[6] E. Foruzan, L.-K. Soh, and S. Asgarpoor, “Reinforcement Learning Approach for Optimal Distributed Energy Management in a Microgrid,” *IEEE Transactions on Power Systems*, vol. 33, no. 5, pp. 5749–5758, Sep. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8331897/

[7] X. Fang, J. Wang, G. Song, Y. Han, Q. Zhao, and Z. Cao, “Multi-Agent Reinforcement Learning Approach for Residential Microgrid Energy Scheduling,” *Energies*, vol. 13, no. 1, p. 123, Dec. 2019. [Online]. Available: https://www.mdpi.com/1996-1073/13/1/123

[8] X. Fang, Q. Zhao, J. Wang, Y. Han, and Y. Li, “Multi-agent Deep Reinforcement Learning for Distributed Energy Management and Strategy Optimization of Microgrid Market,” *Sustainable Cities and Society*, vol. 74, p. 103163, Nov. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2210670721004443

[9] X. Xu, Y. Jia, Y. Xu, Z. Xu, S. Chai, and C. S. Lai, “A Multi-Agent Reinforcement Learning-Based Data-Driven Method for Home Energy Management,” *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3201–3211, Jul. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8981876/

[10] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, “Mastering the game of go with deep neural networks and tree search,” *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[11] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, “Dota 2 with large scale deep reinforcement learning,” *arXiv preprint arXiv:1912.06680*, 2019.

[12] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, “Solving rubik’s cube with a robot hand,” *arXiv preprint arXiv:1910.07113*, 2019.

[13] L. Baird and A. Moore, “Gradient Descent for General Reinforcement Learning,” in *Advances in Neural Information Processing Systems*, M. Kearns, S. Solla, and D. Cohn, Eds., vol. 11. MIT Press, 1998. [Online]. Available: https://proceedings.neurips.cc/paper/1998/file/af5afd7f7c807171981d443ad4f4f648-Paper.pdf

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal Policy Optimization Algorithms,” *CoRR*, vol. abs/1707.06347, 2017, arXiv: 1707.06347. [Online]. Available: http://arxiv.org/abs/1707.06347

[15] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, “The Surprising Effectiveness of PPO in Cooperative, Multi-Agent Games,” *arXiv:2103.01955 [cs]*, Jul. 2021, arXiv: 2103.01955. [Online]. Available: http://arxiv.org/abs/2103.01955

[16] E. Liang, R. Liaw, R. Nishihara, P. Moritz, R. Fox, K. Goldberg, J. Gonzalez, M. Jordan, and I. Stoica, “RLlib: Abstractions for Distributed Reinforcement Learning,” in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, Jul. 2018, pp. 3053–3062. [Online]. Available: https://proceedings.mlr.press/v80/liang18b.html

[17] Y. Tang and S. Agrawal, “Discretizing Continuous Action Space for On-Policy Optimization,” *arXiv:1901.10500 [cs]*, Mar. 2020, arXiv: 1901.10500. [Online]. Available: http://arxiv.org/abs/1901.10500