Deep Reinforcement Learning for Demand Response of a Steel Plant in Energy and Spinning Reserve Markets

Margi Shah[*], Yue Zhou[*], Jianzhong Wu[*], Max Mowbray[†]
[*] School of Engineering, Cardiff University, Cardiff, UK
[†] Department of Chemical Engineering, Imperial College London, London, UK
{ shahmr5, ZhouY68, WuJ5 } @cardiff.ac.uk, m.mowbray@imperial.ac.uk

***Abstract*—Industrial demand response has potential to enhance power systems' operational flexibility amid the operational challenges posed by massive proliferation of distributed energy resources. As an energy intensive industry, steel manufacturing has the potential to participate in demand response via responding to time-varying electricity price and providing spinning reserve service at the same time, leading to reduced electricity costs while supporting the power systems. However, this potential is hindered due to the complex and intertwined processes involved in steelmaking and the uncertainties of electricity prices and onsite renewable power generation. In this paper, we present a novel deep reinforcement learning based demand response scheme to address these challenges, which optimizes the schedules of steelmaking processes for maximizing the benefits in both the energy and spinning reserve markets.**

Keywords*—Demand response, optimization, production process, reinforcement learning, spinning reserve, steel plant

I. Introduction

Steel production accounts for 8% of global energy demand and contributes 7% to the energy sector CO2 emissions. The emissions must decrease by at least 50% by 2050 to align with global energy and climate objectives [1]. With the advancement of intelligent energy management, demand response (DR) offers a valuable solution for providing stability and flexibility to power systems while supporting the increased penetration of renewable resources. Optimizing production scheduling in an industrial plant plays a crucial role in enabling effective DR. Recent advancements in scheduling models and solution techniques for industrial plants have revolutionized production scheduling, with the state-of-the-art underpinned by mathematical programming with heuristic decision-making algorithms also widely used [2]. In general, the formulation is a Mixed-Integer Linear Program (MILP). MILP solvers are widely known to be effective considering the size and structure of production scheduling models [2].

Scheduling problems in industrial plants are often large, making MILP methods for handling uncertainties essentially intractable online. For example, stochastic MILP, which models uncertainty using probability distributions, accounts for probabilistic descriptions of uncertain variables and typically optimizes the expected utility of decisions. However, it relies on sample approximations. This approach leads to a combinatorial explosion in the size of the model and renders exact solution intractable online and in some cases offline, especially as the number of uncertain variables and scenarios increases. Heuristics or approximate methods are often required online, but even these can be costly online due to the sample inefficiency of Monte Carlo methods and stochastic search algorithms [3]. Robust optimization focuses on the worst-case scenarios, often leading to overly cautious solutions that sacrifice the expected performance to protect against the worst case.

With the advancement of machine learning, reinforcement learning (RL) has become a promising method to address the above-mentioned challenges in a tractable way online while accounting for uncertainty. In general, RL learns to make sequential decisions by interacting with the underlying stochastic process, explicitly learning the plant's dynamics and exogenous uncertainties. RL can also be used as means to parameterize a function to approximate the optimal policy by utilizing a simulation model for performing offline policy optimization. The parameterized policy function can then be used online to identify controls through prediction, offering a much cheaper alternative to online MILP, which requires repeatedly solving a MILP in real time. For example, in [4] a model free deep RL (DRL) algorithm for smart facilities was developed to minimize electricity costs and it achieved 8.29% lower electricity cost than model predictive control (MPC). In [5], a multi-agent DRL system was developed for energy management in a lithium-ion battery assembly manufacturing system. It effectively minimized electricity costs and maintained production tasks compared to a benchmark without DR. Finally, in [6], an actor-critic based DRL method was applied to optimize energy management in steel powder manufacturing, yielding a notable 24.12% reduction in total energy costs compared to the absence of DR implementation.

Most of the studies using (D)RL in industrial plants for energy management using DR have primarily focused on minimizing electricity cost and make-span. We introduce a novel DRL based DR scheme to optimize the scheduling of steelmaking and maximize the profits by participating in energy and spinning reserve markets. The contributions of this work are: 1) the use of an offline-online training and control framework that shifts computational expense offline, and identifies scheduling decisions online through prediction; 2) the use of a stochastic search method, i.e. Separable Natural Evolutionary Strategy (SNES) for policy optimization; 3) use of the attention mechanism to focus on the most relevant information within a window of observations; and 4) a benchmark comparison with mathematical programming. Specifically, we identify a policy that selects between heuristic decision rules and is able to handle the uncertainty from electricity prices. To the best of our knowledge, this

This work was supported by EPSRC through the projects EP/T021969/1(MC2), SUSTAIN Feasibility project EP/S018107/1(SFSC2-203) and by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 101096946 (FlexCHESS).

study is the first of its kind in this domain. The remaining of the paper is organized as follows: Section II outlines the problem formulation; Section III introduces the algorithm; and Section IV presents the case study of the steelmaking plant, upon which conclusions are drawn in Section V.

II. Problem Formulation

The steelmaking process considered in this paper involves four main stages: electric arc furnace (EAF), argon oxygen decarburization (AOD), ladle furnace (LF), and continuous casting (CC). Solid metal scrap is melted in EAF, impurities are removed in AOD, refined in LF, and solidified into slabs in CC. The molten metal batch is called a 'heat' [7]. For simplicity, the casting stage is not considered in this work. Fig. 1 illustrates the RL framework for scheduling steel plant. The daily production target as well as the hourly energy prices (EL), spinning reserve prices (SP) and forecasts of on-site renewable energy sources (RESs) is known ahead of time. This detailed system model within the simulator is then used for training to obtain a near optimal policy.

We formulate the DR problem in a steelmaking plant as a finite-horizon partially observable Markov decision process (POMDP). This POMDP definition is a tuple $\langle \mathbb{X}, \mathbb{U}, \Omega, \mathcal{P}, O, R, T \rangle$, where $\mathbb{X}$ is a state space, $\mathbb{U}$ the action space and Ω the observation space. At each discrete time index $t \in \{0,1,2,\ldots,T\}$, the state of the plant is represented by $\boldsymbol{x}_t \in \mathbb{X} \subseteq \mathbb{R}^{n_x}$ .When control action $\boldsymbol{u}_t \in \mathbb{U}$ is executed, the state changes according to transition density function, $X_{t+1} \sim \mathcal{P}(\boldsymbol{x}_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{u}_t)$. Subsequently the agent receives a partial observation $\boldsymbol{o}_{t+1} \in \Omega \subseteq R^{n_x}$ according to the density function $O_{t+1} \sim O(\boldsymbol{o}_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{u}_t)$, on which it can condition control selection, and also a cost $R_{t+1} \sim R(\boldsymbol{r}_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{u}_t)$.

The agent acts according to its policy $\pi(\boldsymbol{u}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{u}_{<t})$ which returns the probability of taking action $\boldsymbol{u}_t$ at time t, and $\boldsymbol{o}_{\leq t} = (\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_t)$ and $\boldsymbol{u}_{<t} = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_{t-1})$ are the observation and action histories respectively. The agent's goal is to learn a policy π that minimises

$$J = \mathbb{E}_{\tau \sim p(\tau)} \left[\sum_{t=1}^{T} \gamma^{t-1} R_t \right] \tag{1}$$

over trajectories $\tau = (\boldsymbol{x}_0, \boldsymbol{o}_0, \boldsymbol{u}_0, \ldots, \boldsymbol{u}_{T-1}, \boldsymbol{o}_T, \boldsymbol{x}_T)$ induced by its policy, where $0 \leq \gamma < 1$ is the discount factor. In theory, a POMDP agent must condition its actions on the entire history $(\boldsymbol{o}_{\leq t}, \boldsymbol{u}_{<t})$ which grows exponentially in t. In practice, near-optimal actions can be approximated using methods like recurrent neural networks and Long Short-Term Memory networks. However, recent advances in large language modeling suggest that attention mechanisms could be beneficial. We propose leveraging the attention mechanism to address this challenge.

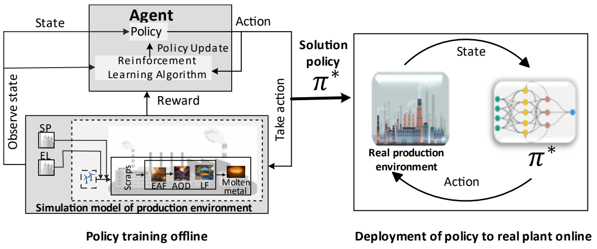

Fig. 1. RL scheme for decision making in steelmaking

1) System observation formulation: The state $\boldsymbol{x}_t$ in the case of steel plant scheduling can be very large at any given time index $t \in \{0,\ldots,T\}$ within the discrete finite time horizon. As discussed in [8] its dimension also varies over the time horizon. Defining a state in a scheduling problem that captures all relevant decision-making information while maintaining a fixed and finite dimensionality is a significant challenge for RL. Hence, we formulate the problem as POMDP where a sequence of states from previous time intervals are considered for taking the next action. We denote this window of observation history as a matrix $\Omega_t \in \mathbb{R}^{n \times N}$, with the window constituting those time indices $\bar{t} \in \{t - n, \ldots, t\}$. The constant n determines the length of the window which is treated as a hyperparameter. The observation matrix, denoted as Ω_t at a given time index t within the discrete finite time horizon (of length T), has a row vector equivalent to an i^{th} observation within the window,

$$\boldsymbol{o}_i = \left[o_{i,1}, \ldots, o_{i,N} \right]^T. \tag{2}$$

Each element within the observation vector, $n \in \{1,\ldots,N\}$ represents information from the production environment. The elements associated with column index, $1 \leq n \leq 5$ represents the information of the tasks for stage I, corresponding to heat number, the processing time of the heat, the start and end time of the corresponding heat and the unit being used for processing the heat from the two parallel units in a stage. Similarly, $6 \leq n \leq 10$ represents the same information for stage II, and $11 \leq n \leq 15$ represents the information of the tasks for stage III. The final three columns represent the exogenous information of electricity price, spinning reserve prices and power generation from on-site RESs. At t, within the observation Ω_t, information about prices a time step ahead are available, i.e. $t+1$. This structure allows the observation to describe the future energy market as allowed by market mechanisms, which typically fix the price a short period ahead of time. This is particularly useful for managing uncertainties in electricity prices and wind power.

The attention in our proposal is the single-head attention (SHA) utilized in the transformer model, known as "Scaled Dot-Product Attention" [9]. Leveraging the attention mechanism can help focus on the most relevant information of the observation matrix Ω_t allowing the policy to learn long-range dependencies more effectively. The attention mechanism is described by Eq. 3,

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{3}$$

where Q is matrix of queries, K is matrix of keys of dimension d_k and V is matrix of values packed simultaneously. The matrices Q, K and V are functions of the observation matrix Ω_t. This allows our model to jointly attend to information from different observation subspaces, which reflect different aspects of steel plant scheduling characteristics.

2) System action formulation: At a given time, index $t \in \{0,\ldots,T\}$, the agent selects the control action $\boldsymbol{u}_t$ which represents the available heuristic decision rules that can be used to schedule the n_u units in each stage,

$$\boldsymbol{u}_t = [\boldsymbol{u}_t^{\boldsymbol{EAF}}, \boldsymbol{u}_t^{\boldsymbol{AOD}}, \boldsymbol{u}_t^{\boldsymbol{LF}}]^T \tag{4}$$
$$\boldsymbol{u}_t^j = \left[u_{1,t}, \ldots, u_{n_u,t} \right]^T.$$

Here $u_{i,t} \in \mathbb{Z}_7$ where $\mathbb{Z}_7 = \{ x \in \mathbb{Z} \,|\, 1 \leq x \leq 7\}$ represents seven scheduling rules as listed in Table I.

3) System cost formulation: Given the production activities and their power profile and the energy and spinning reserve prices, the steel plant optimizes the production schedule to minimize its net cost – the cost of electric energy minus the revenue from spinning reserve provision. Here it is assumed that surplus onsite RES power cannot be sold back to the power grid to earn additional revenue,

$$\boldsymbol{r}_t = (EL_{price_t} * \max[0, (\vartheta_{EL,t} - E_{wind,t})] - (SP_{price_t} * \vartheta_{SP,t})) \tag{5}$$

where $\vartheta_{\mathrm{EL},t}$ is the total energy consumption from $t-1$ to t. , $\vartheta_{\mathrm{SP},t}$ is the available reserve capacity in MW and $\vartheta_{\mathrm{SP},t} \in [0, \overline{\vartheta_{\mathrm{SP},t}}]$ where $\overline{\vartheta_{\mathrm{SP},t}}$ is the upper bound of available amount of reserve capacity. $E_{\mathrm{wind},t}$ is the onsite RES power output at t.

III. Methodology

The organization of methodology is designed to highlight key properties of our algorithm. First, we explain how discrete control decisions are identified, followed by the algorithm.

A. Identifying discrete control decisions

The restriction on the control space arises from production constraints (PCs) within steelmaking process. We hypothesise that are time index t, the constraint set is $\widehat{\mathbb{U}}_t \subseteq \mathbb{U}$. Traditionally, assignment and sequencing decisions are represented as binary variables in MILP models. Let $N_{P,h,t}$ denote the start of the processing of heat h is represented at t. Here $P \in \{E, A, L\}$ where E, A, L represents processing in stage EAF, AOD and LF respectively. Similarly, the start of the transportation of the material between stages is indicated by $N_{IP,h,t}$. Here $IP \in \{EA, AL\}$ where EA and AL represent transfer between EAF-AOD and AD-LF stage respectively. We utilise logical operations to define these binary variables at time t via discrete control decisions $\boldsymbol{u}_t \in \mathbb{U}$, which represent heuristic decision rules. At each discrete time step $t \in \{0, \dots, T\}$, we hypothesise that a set of controls, $\boldsymbol{u}_t \in \overline{\mathbb{U}}(\boldsymbol{x}_t)$, may be identified that adhere to the logic provided by the PCs, based on the current state of the plant, $\boldsymbol{x}_t \in \mathbb{X}_t$. This functional transformation is denoted, $f_{PC} : \mathbb{U} \times \mathbb{X} \to \overline{\mathbb{U}}$, where $\overline{\mathbb{U}} \subseteq \mathbb{U}$, and is assumed non-smooth. The set $\overline{\mathbb{U}}(\boldsymbol{x}_t)$ is then used to implement an action-masking scheme [10] , which enables a SoftMax policy to ensure that invalid controls have zero probability of selection and therefore the constraints are satisfied. However we don't have state constraints in this problem. Based on the decision rule selected, we then map $\boldsymbol{u}_t$ to the binary variables to make assignment decisions. In the case one is unable to identify (and satisfy) all constraints via f_{PC}, such that $\overline{\mathbb{U}}_t \subset \widehat{\mathbb{U}}_t$, we penalize the violation of those constraints that cannot be handled, and incorporate a penalty function for the constraint violation, $\phi_{t+1} \sim \varphi(\phi_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{u}_t)$. For example, by the end of the horizon all the heats should be processed. We define the expected penalised return as:

$$J^{\varphi} = \mathbb{E}_{\pi} \left[\sum_{t=1}^{T} \gamma^{t-1} \phi_t \right] \tag{6}$$

We can then identify a solution policy π^*as follows:

$$\min_{\pi} J^{\varphi} \tag{7}$$

TABLE I. Heuristic Decision Rules

Decision rule	Description
1	Process the heat with shortest processing time (SPT)
2	Process the heat with longest processing time (LPT)
3	Process the heat with shortest remaining machine time (SRM)
4	Process the heat with longest remaining machine time (LRM)
5	Process the heat longest in waiting area (MFT)
6	Process the most recent heat in waiting area (MRT)
7	Don't process anything

B. Stochastic search policy optimisation

In this work we adopt Evolution strategy which is a class of black box optimization algorithm with heuristic search procedures inspired by natural evolution for finding a near optimal policy. We employ Separable Natural Evolutionary Strategy (SNES). The primary reason for selecting this category of algorithm are its scalability to high dimensions and parallelizability, which are highly beneficial for our problem. The algorithm is as shown below.

```
Algorithm: Separable Natural Evolution Strategy (SNES)
Input: Neural network (Policy function), $\pi(\cdot)$, a number of samples to evaluate each policy $n_S$, a population initialisation strategy $f_{init}(\cdot)$, SNES algorithm $f_{SNES}(\cdot)$, population size P, number of optimisation iterations I, $J_{\pi^*} = \infty$
I    Generate initial policy parameters, $\Theta_1 = f_{init}(\hat{\theta}, P)$
II   for i = $1,2, \dots \dots, I$ do
       1    Construct policy population, $\Pi_i = \{ \pi_{i,k}(\theta) \, \forall \, \theta \in \Theta_i \}$ and allocate memory buffer, $\mathcal{B}$
       2    for $\pi_{i,k} \in \Pi_i$ do
              (i)    for episode = $1,2,3 \dots \dots, n_S$ do
                       (a)    Receive observation of initial state $\boldsymbol{o}_0$
                       (b)    for t = $0,1,2 \dots \dots, T-1$ do
                                (b.1)    Generate control $\boldsymbol{u}_t$
                                (b.2)    Recieve next observation $\boldsymbol{o}_{t+1}$
                                (b.3)    Observe cost $\phi_{t+1}$
                       (c)    Calculate penalised return $J^{\varphi}$ (see Eq. 6)
              (ii)   Calculate sample average approximation $J_{i,k}$ (see Eq. 7) and append $(\pi_{i,k}, J_{i,k})$ to $\mathcal{B}$
              (iii)  If $J_{i,k} < J_{\pi^*}$, then update the best-known policy $(\pi^*, J_{\pi^*})$
       3    Generate new parameters $\Theta_{i+1} = f_{SNES}(\mathcal{B})$
III  Best ranked policy, $\pi^*$.
```

IV. Case study

In this section, we present the study of the daily scheduling for the typical EAF steel plant to demonstrate the effectiveness of the DRL model.

A. Test system description

The hourly electric energy prices in the Fig. 2 are based on the NORD POOL UK wholesale electricity prices. The reserve prices are assumed lower than energy prices, as shown in Fig. 2. Wind power generation data is taken from [11]. Parameters for the steel plant are drawn from [7] but not detailed due to page limitations. The plant layout comprises three stages, each with two parallel equipment's: two EAFs, two AODs, and two LFs. The sustaining power needed for spinning reserve provision is assumed to be 60% of the nominal EAF melting power. Sustaining power is the power

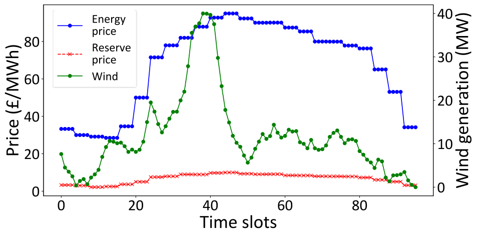

Fig. 2. Electric energy prices, reserve prices and wind power generation

required for melting the heat, i.e., the EAF power cannot be below this level once it is turned ON for guaranteeing the steel quality. The steel plant produces 24 heats daily and is equipped with a local wind power system. The scheduling horizon spans a 24-hour production cycle, divided into 96 time slots of 15 minutes each.

To assess the performance of the proposed DRL methodology, a benchmark without learning is investigated, wherein the optimization problem is formulated under a MILP framework [7] and solved by a commercial Gurobi v9.1.2 solver. The proposed method utilized the PyTorch v1.9.0 python package, Anaconda v4.10.3 and Evotorch v0.5.1. The 24 heats to be produced correspond to 16800 constraints and 11520 binary decision variables in case of MILP formulation.

B. *Results and discussion*

We demonstrate the performance of the DRL method in two cases: 1) with no uncertainty, and 2) with uncertainty in electricity prices. The policy is generated by SNES. In learning, all candidate policies were evaluated over $n_s = 100$ samples ($n_s = 1$ when there is no uncertainty), with population size of $P = 1000$ and maximum optimization iterations of $I = 100$. Hyperparameter tuning was conducted via grid search. The link to the GitHub repository[1] can be found in footnote of this page.

1) Nominal case: Demonstration of the training profiles are provided by Fig. 3. We don't use ground truth values of MILP for training and as outlined in Algorithm in Section III-B, we roll out the policy in simulation and use SNES to optimse the policy.The benchmark can be identified as the ideal strategy as it is assumed that it has full information about steel plant dynamics and utilizes the accurate model to minimize the net cost which leads to a mathematically optimal result. In contrast, the DRL method utilizes its learning capability to explore the policy space to minimize the net cost. It is seen that, at the initial stage, DRL strategy does not work well but as learning progresses, it learns the steel plant dynamics by optimizing the policy. This highlights the main advantage of DRL being an effective form of simulation based optimisation. Hence, it is reasonable to suggest the application of DRL complex industrial DR problems. The best known policy satisfies the constraints imposed on the problem, indicating the efficacy of framework proposed. The performances reported here are from assessing the policy on the state distribution observed in training. It is likely the uncertain variables will be misspecified in practice, and the policy robustness should be further interrogated through out-of-distribution analysis. These kind of analysis are explored in [3] as an example, but are beyond the scope of this paper.

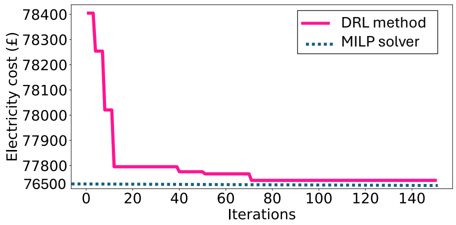

Fig. 3. Net cost in the DRL training process compared with that of MILP

Figs. 4a and 4b provide comparative plots of the day-ahead production schedule of the steel plant via MILP and DRL methods respectively. Importantly, Fig. 4b highlights the ability of the DRL method proposed to account for critical steelmaking related constraints as all heats are scheduled in proper order. The highlighted part from timestep $t = 52$ to $t = 72$ shows that there is no production in EAF1 by both approaches except heat 22 is scheduled by MILP. The corresponding spinning reserve provision schedule is displayed in Fig. 5. To illustrate the results more clearly, the aggregated energy consumption of all equipment under the proposed DRL method is plotted in Fig. 6. It is evident that the units use more electricity when the price is low and then decrease their demand when the price is high, thereby avoiding energy consumption during peak times. As detailed from Table II, the policy obtained from the DRL method proposed achieves very close objective value compared to the benchmark MILP method with the optimality gap of 1.75%.

TABLE II. NET COST OBTAINED BY MILP AND DRL

Method	Net Cost (k£)	EL Cost (k£)	SP Revenue (k£)
MILP	76.50	105.60	29.10
DRL	77.74	110.86	33.12

This could be attributed to the complex and non-smooth mapping in the model space as well as parameter space.

2) Case with uncertainity in electricity price: In this section, we turn our attention to demonstrate the ability of the framework to handle uncertainty from electricity price. We consider the electricity price shown in Fig. 2 as the day-ahead forecast price. We further assume that the forecast price is subject to uncertainty, which follows a normal distribution with the mean $\mu = 0$ and standard deviation $\sigma = 10\%$.

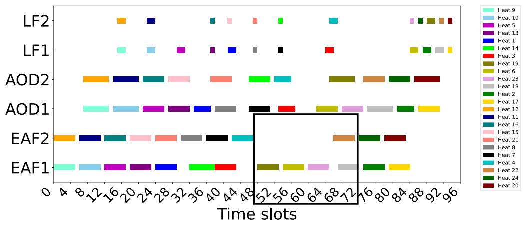

Fig. 4a. Production schedule obtained from MILP

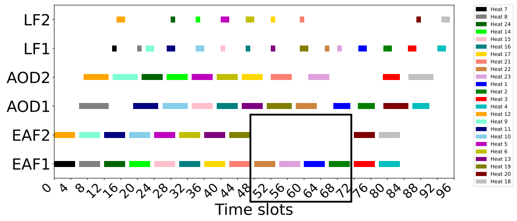

Fig. 4b. Production schedule obtained from DRL

Code available at: [1] https://github.com/shahmargi/steel_scheduling.git

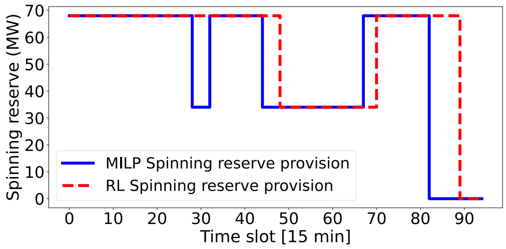

Fig. 5. Spinning reserve provision by MILP and RL method

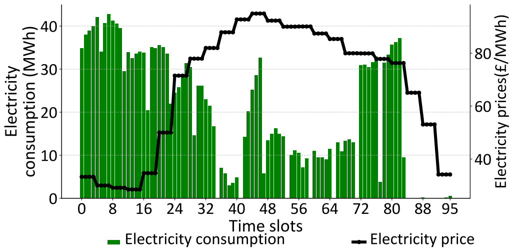

Fig. 6. Aggregated energy consumption of the steel plant under DRL

We then compare and evaluate the performance of the production schedules made by the MILP and DRL approaches. Monte Carlo (MC) simulation is used to generate 100 scenarios of the true electricity price throughout the day, via sampling from the forecast uncertainty. The MILP approach is assumed to optimize the production schedule only based on the day-ahead forecast price, so the production schedules are the same in all the 100 scenarios. By contrast, the DRL approach is assumed to optimize the production schedule in a rolling manner, that is, in each hour, the production schedule, from the current hour to the next hour is updated based on the revealed true electricity price of the current hour, and only the schedule of the current hour is actually executed. Therefore, for the 100 scenarios, the DRL approach will generate 100 different production schedules. Then the actual net electricity costs of the production schedules in each scenario, calculated given the true electricity price values, are calculated for the schedules generated by both the MILP and DRL approaches. The average net electricity costs of the 100 scenarios are presented in Table III. It is seen that the net average cost of DRL is very close to MILP, verifying its effectiveness with the electricity price uncertainty.

TABLE III. AVERAGE NET COST OBTAINED BY MILP AND DRL

Method	**Average net cost (μ_z)**
MILP	73.71
DRL	74.56

Also, we compare the computational time for both methods online to make a control decision as well as offline to train the control policy in Table IV. MILP incurs less offline time cost whereas DRL incurs higher offline time cost for training. However, DRL significantly reduces the online computational time for control decisions, unlike MILP, which doesn't update schedules in real-time. The proposed method can also be applied to other energy intensive industries like aluminum smelter and cement plant with only a requirement to create the simulator for each individual plant.

TABLE IV. COMPUTATIONAL TIME FOR OFFLINE TRAINING POLICY AND MAKING CONTROL DECISIONS ONLINE

	MILP	**DRL**
Making control decisions (online)	N/A	0.00005
Training policy (offline)	72	180

V. CONCLUSION

This paper proposes a DRL-based demand response (DR) scheme for optimally scheduling steel plants to minimize electricity costs by participating in the energy and spinning reserve markets. The production process is initially modeled as a Partially Observable Markov Decision process (POMDP). Subsequently, a Separable Natural Evolution Strategy (SNES) algorithm is used to find the near optimal policy. DRL has been used to parameterize a function to approximate the optimal policy by using a simulation model to perform the policy optimisation offline. This solution policy provides controls online which is much computationally cheaper than solving MILP online. The policy obtained by DRL method in both the cases has a performance gap of 1.75% and 1.15% to the MILP. Future work will consider the actual dispatch of spinning reserve in determining the optimal schedule of a steel plant while participating in energy and spinning reserve markets. Additionally, it will focus on bridging the gap between MILP and DRL algorithms and comparing the proposed method with other DRL methods.

REFERENCES

[1] 'Iron and Steel Technology Roadmap – Analysis - IEA'. Accessed: Dec. 16, 2024. [Online]. Available: https://www.iea.org/reports/iron-and-steel-technology-roadmap

[2] I. Harjunkoski *et al.*, 'Scope for industrial applications of production scheduling models and solution methods', *Comput. Chem. Eng.*, vol. 62, pp. 161–193, Mar. 2014, doi: 10.1016/j.compchemeng.2013.12.001.

[3] M. Mowbray, D. Zhang, and E. A. D. R. Chanona, 'Distributional Reinforcement Learning for Scheduling of Chemical Production Processes', Mar. 09, 2022, *arXiv*: arXiv:2203.00636. Accessed: Sep. 26, 2023. [Online]. Available: http://arxiv.org/abs/2203.00636

[4] R. Lu, R. Bai, Z. Luo, J. Jiang, M. Sun, and H.-T. Zhang, 'Deep Reinforcement Learning-Based Demand Response for Smart Facilities Energy Management', *IEEE Trans. Ind. Electron.*, vol. 69, no. 8, pp. 8554–8565, Aug. 2022, doi: 10.1109/TIE.2021.3104596.

[5] R. Lu, Y.-C. Li, Y. Li, J. Jiang, and Y. Ding, 'Multi-agent deep reinforcement learning based demand response for discrete manufacturing systems energy management', *Appl. Energy*, vol. 276, p. 115473, Oct. 2020, doi: 10.1016/j.apenergy.2020.115473.

[6] X. Huang, S. H. Hong, M. Yu, Y. Ding, and J. Jiang, 'Demand Response Management for Industrial Facilities: A Deep Reinforcement Learning Approach', *IEEE Access*, vol. 7, pp. 82194–82205, 2019, doi: 10.1109/ACCESS.2019.2924030.

[7] X. Zhang, G. Hug, Z. Kolter, and I. Harjunkoski, 'Industrial demand response by steel plants with spinning reserve provision', in *2015 North American Power Symposium (NAPS)*, Charlotte, NC, USA: IEEE, Oct. 2015, pp. 1–6. doi: 10.1109/NAPS.2015.7335115.

[8] V. Avadiappan and C. T. Maravelias, 'State estimation in online batch production scheduling: concepts, definitions, algorithms and optimization models', *Comput. Chem. Eng.*, vol. 146, p. 107209, Mar. 2021, doi: 10.1016/j.compchemeng.2020.107209.

[9] A. Vaswani *et al.*, 'Attention Is All You Need', 2017, *arXiv*. doi: 10.48550/ARXIV.1706.03762.

[10] S. Huang and S. Ontañón, 'A Closer Look at Invalid Action Masking in Policy Gradient Algorithms', *Int. FLAIRS Conf. Proc.*, vol. 35, May 2022, doi: 10.32473/flairs.v35i.130584.

[11] 'Elia: Belgian's Electricity System Operator', Elia. Accessed: Jun. 06, 2024. [Online]. Available: https://www.elia.be/en/