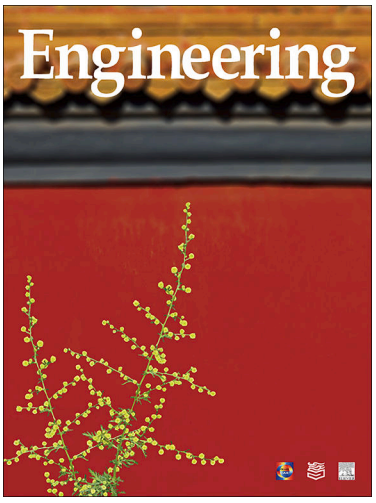
Journal Pre-proofs



Article

Deep Reinforcement Learning for Scheduling of a Steel Plant in the Electricity Spot Market

Margi Shah, Yue Zhou, Jianzhong Wu, Max Mowbray

PII: S2095-8099(26)00070-6
DOI: https://doi.org/10.1016/j.eng.2025.12.038
Reference: ENG 2247

To appear in: *Engineering*

Received Date: 31 March 2025
Revised Date: 23 September 2025
Accepted Date: 23 December 2025

Please cite this article as: M. Shah, Y. Zhou, J. Wu, M. Mowbray, Deep Reinforcement Learning for Scheduling of a Steel Plant in the Electricity Spot Market, *Engineering* (2026), doi: https://doi.org/10.1016/j.eng.2025.12.038

Next Gen-AI Enabled New Type of Power System—Article

Deep Reinforcement Learning for Scheduling of a Steel Plant in the Electricity Spot Market

Margi Shah [a], Yue Zhou [a,*], Jianzhong Wu [a], Max Mowbray [b,*]

[a] *School of Engineering, Cardiff University, Cardiff CF24 3AA, UK*

[b] *Imperial College London, London SW7 2AZ, UK*

* Corresponding authors.

E-mail addresses: zhouy68@cardiff.ac.uk (Y. Zhou), m.mowbray@imperial.ac.uk (M. Mowbray).

ARTICLE INFO

Article history:

Received

Revised

Accepted

Available online

Keywords

Demand response

Production process

Steel plant

Reinforcement learning

Optimization

ABSTRACT

The steel industry, characterized by its substantial energy consumption, is grappling with rising energy costs and the imperative to decarbonize. However, the scheduling of a steel plant is challenged by the complexity and interdependency of its processes with various uncertainties. This study introduces a deep reinforcement learning (DRL) methodology specifically designed to optimize scheduling in the presence of the exogenous uncertainties brought by electricity prices and on-site renewable generation. The scheduling problem is formulated as a partially observable Markov decision process (POMDP), which enables decision-making despite the state not being fully observable. The attention mechanism is utilized to abstract a representation of a window of observations upon which decisions are conditioned. The control space is defined by domain knowledge-informed heuristic rules, and evolutionary search is utilized for the purpose of policy optimization. The case study considers an electric arc furnace (EAF)-based steel plant with various problem sizes and processing times for steelmaking tasks. The performance of the proposed method is compared with a traditional mixed integer linear programming (MILP) approach and the policy gradient method, proximal policy optimization (PPO). The proposed method is evaluated under uncertainty conditions arising from market prices and on-site renewable energy sources. Case study results reveal that the proposed DRL strategy effectively integrates uncertainties into real-time decision-making, achieving a desirable performance level with minimal online computational cost.

1. Introduction

1.1. Steel industry decarbonization and control

Steel is a building block of the modern world and an indispensable element for the energy transition. It accounts for 8% of global energy demand, which is projected to surge by over a third in the period to 2050 [1]. The steel industry contributes 7% to the energy sector's CO_2 emissions. According to an International Energy Agency (IEA) 2020 report, these emissions must decrease by at least 50% by 2050 to align with global energy and climate objectives [1]. However, the industry's strong reliance on fossil fuels poses immense challenges to this. The steelmaking production method of the electric arc furnace (EAF), which utilizes scrap metal, eliminates the need for carbon-intensive ironmaking, resulting in lower production-based emissions. Nevertheless, it is an electricity-intensive process that incurs high electricity costs. Fortunately, EAF are batch operations, with potential electricity cost savings available through utilizing the associated process flexibility [2]. With the advancement of intelligent energy management, demand response (DR) is acknowledged as an effective way to encourage lower electricity consumption during periods of high electricity prices or when the grid reliability is at risk [3]. This capability is particularly important for integrating renewable energy sources (RESs), as DR facilitates load balancing by adjusting demand to match the variable supply from renewables, thus enhancing grid stability and optimizing the use of clean energy.

In the realm of industrial energy management for DR, control strategies for production scheduling typically employ rule-based controllers (RBCs) or model predictive control (MPC). RBC is underpinned by hard-coded rules that generate controls based on historical or plant data. However, they exhibit limited performance, relying solely on expert experience, proving inadequate in adapting to evolving objectives, and come with no certificate of mathematical optimality. Furthermore, these controllers neglect the incorporation of forecasts on electricity prices or available onsite RESs [4].

By contrast, MPC has significantly contributed to the energy management in DR programs [5]. By exploiting approximate systems models, MPC has the ability to optimize specific control objectives while adhering to environmental and operational constraints [6]. Models are typically subject to uncertain parameters, and it is difficult to account for these uncertainties in a state-feedback fashion due to the time scale required to solve the underlying model. As a result, when applied in a receding horizon framework, nominal descriptions of uncertain parameters are typically used instead or a precomputed offline solution is used instead.

The decision problems descriptive of steelmaking processes is typically formulated as mixed-integer programming (MIP), which has several drawbacks. First, they typically provide some approximation to the complexity of the steelmaking process, particularly when energy balances are accounted for, as these involve nonlinear equations that add additional challenge to solution methods. In Ref. [7], the authors propose a metaheuristic approach to preserve all the features of the industrial process. The authors chose particle swarm optimization (PSO) for its ability to handle nonlinear problems with high computational efficiency. Although metaheuristic algorithm can incorporate nonlinear dependencies without compromising computational feasibility, their main disadvantage is that they do not guarantee finding the global optimum and may struggle with effectively representing complex constraints in scheduling problems [8]. Second, a significant challenge faced by scheduling steel plants is the uncertainties that need to be considered, such as those in electricity prices and available renewable energy over time. Scheduling problems in steel plants are often large, making MIP methods challenging to deploy online. Stochastic and robust MIP methods can be used to explicitly treat these uncertainties, but both have limitations. Stochastic MIP describe uncertainty through probability distributions but to ensure a finite dimensional constraint system typically rely on sample approximations. This means the underlying constraint system observes a combinatorial explosion in the size of the model and renders exact solution intractable online and even offline in some cases [9]. Meanwhile robust optimization describes uncertain variables as taking a set of values, and is generally conservative when applied within receding horizon frameworks. Despite these challenges, the existing literature on energy-aware scheduling for steelmaking demonstrates the use of MPC for minimizing electricity costs [10,11], providing ancillary services [12], peak load management [13,14], and prespecified energy curve tracking [15].

1.2. Opportunity for reinforcement learning (RL)

Faced with the issues of RBC and MPC, RL can be leveraged for DR from steel plants. RL is an advanced control technique for solving sequential decision-making problems under uncertainty [16] and has received substantial research attention in various DR programs [17]. The major aim of RL is to identify function approximations of the optimal control policy through simulation of the plant model. In doing so, one shifts the computational work for identifying a state-feedback policy offline and minimizes approximations to the propagation of model uncertainty through the Monte Carlo (MC) method. Once identified, the policy function identifies online scheduling decisions conditional to available state observations (which may be partial in nature). These decisions are identified in a short time frame via a function evaluation, which has computational benefit over solving a MIP problem online, as required by MPC. Policy function approximation is often provided by incorporating deep learning techniques, giving rise to deep RL (DRL) and the use of artificial neural networks (ANNs) [18].

In Ref. [19], a model-free DRL algorithm was developed for smart facilities energy management using a practical experimental setup. The system's diversified facilities are classified into three types—critical, deferrable, and regulatable—based on priority and characteristics. The energy management problem is formulated as a Markov decision process (MDP), as the next time-slot energy consumption depends only on the current consumption, electricity price, and control signal, satisfying the Markov property. Long short-term memory (LSTM) units are adopted to extract discriminative features from past 24 h electricity price sequences, which are invariant to control signals but influence facility dynamics. These features, combined with measured energy and time-related information, are fed into fully connected multilayer perceptrons (MLPs),

forming a deep Q-network to approximate the optimal policy. The proposed DRL approach achieves 41.05 GBP, only 1.31% abov[Journal Pre-proofs]blem is modelled as an MDP with an LSTM primarily used to capture past prices, representing a partially observable MDP (POMDP) using LSTM can be challenging, as they rely on a compressed hidden state that can lose long-range information and struggle to weigh all past observations effectively which is the case for large-scale industrial scheduling problem. The authors in Ref. [20] proposed a multiagent DRL-based DR scheme for industrial energy management, wherein the manufacturing system was initially formulated as a partially observable Markov game, and then a multiagent deep deterministic policy gradient (DDPG) algorithm was adopted to obtain the optimal schedule for different facilities. However, the study does not show how to explicitly address process constraints, and while it considers partial observability, it does not leverage mechanisms to extract informative features from historical observations. The reliance on a policy-gradient method like DDPG makes the approach susceptible to convergence to poor local optima, particularly in high-dimensional scheduling problems. Additionally, its exploration is limited, so it can struggle with sparse rewards, complex landscapes, and highly stochastic environments. Ref. [21] proposes a scheduling method based on DRL to tackle the multi-objective steelmaking-continuous casting (SCC) scheduling problem under time-of-use (TOU) electricity tariffs. The method aims to minimize both total sojourn time and electricity cost by modeling the problem as an MDP. Action selection is guided by a branching dueling double deep Q-network (BD3QN) where the action space is defined as the heuristic rules. While the authors benchmark the approach against other algorithms, there is no direct comparison with mixed integer linear programming (MILP) formulations, leaving its performance relative to traditional optimization unclear. To enforce process constraints such as continuous casting (CC) sequences with setup and waiting times, a backward scheduling strategy is used, which pre-arranges charges and reverses start times. In this strategy, the casting end time determined by the BD3QN algorithm is taken as the anchor, and earlier stage timings are recursively computed by subtracting processing, waiting, and transportation times at each step. Charges are first allocated to machines, their feasible end times are derived from the next stage's start time, and their start times are then obtained by back-calculating from these end times. By propagating timings backward, the strategy ensures that each stage finishes exactly in time for the next, thereby satisfying CC, setup, and waiting constraints without infeasibility. In Ref. [22], an actor-critic 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. The study employs an MDP formulation, which assumes full state observability and may fail to capture the partially observable and history-dependent dynamics inherent in complex industrial scheduling. It relies on an actor-critic DRL algorithm, which can be prone to convergence to poor local optima in high-dimensional policy spaces and may struggle with exploration. Additionally, the study does not demonstrate how process constraints are handled, which are critical for practical industrial applications.

The advantages of applying DRL to address DR challenges have already been demonstrated by the studies referenced above, as well as by many other extensive publications. However, current studies have certain limitations: ① There is a significant lack of studies on using DRL for scheduling the whole EAF-based steelmaking process for DR; ② despite the above studies, existing research offers limited insight into how DRL methods account for specific process constraints; ③ most of the studies use gradient-based policy learning methods, but stochastic search methods, such as evolutionary algorithms, are overlooked; ④ there is a lack of formalism for partial observability and how it is handled algorithmically.

1.3. Contributions of this work

In this work, we propose a novel DRL scheduling framework for the DR of a steelmaking plant equipped with onsite RES in the electricity spot market. Our approach aims to minimize the electricity cost while accounting for critical production constraints (PCs) and is validated by performing ablation experiments with varying problem sizes and processing time data for a steel plant. The main contributions of this work are as follows.

(1) We formulate steel plant scheduling as a POMDP with state-specific restrictions on control sets, where the true system state is the full constructed production schedule (i.e., the entire decision history), which evolves over time, as well as forecasts of exogenous uncertain variables such as electricity prices and on-site renewable generation. We assume only a fixed finite-dimensional representation of this state is available, making the system partially observable. Unlike standard MDPs, which assume full state knowledge, this formulation enables tractable policy learning in a complex, scheduling environment.

(2) We propose a DRL algorithm via evolutionary search to ensure local and global search of the policy space, mitigating the tendency of policy-gradient methods to converge to poor local optima in such complex policy spaces.

(3) We integrate an attention mechanism into the policy network to selectively extract informative features from histories of partial observations, thereby improving the policy's ability to act effectively under partial observability.

(4) We define the control space through domain-inspired heuristic rules, with the aim of significantly reducing the search complexity. The approach is benchmarked against directly identifying task-unit assignment decisions to evaluate the performance of the scheduling policy in terms of its accuracy, computational cost, and robustness.

2. Problem formulation

The steelmaking process considered comprises four consecutive production stages: EAF, argon oxygen decarburization (AOD), ladle furnace (LF), and CC. The primary raw material is constituted by 80%–90% of recycled scrap alongside virgin materials in the form of various ferroalloys and is initially melted within the EAFs. Subsequently impurities present in the

molten steel, such as carbon elements, are extracted, and additional alloys are introduced through the AOD units. The molten meta[Journal Pre-proofs]rves to solidify the refined molten metal into slabs of varying shapes. These resulting slabs are distinguished by their grade, width, and thickness. Each specific grade is defined by distinct chemical composition, processing duration, temperature parameters, and energy consumption across the various production stages. The initial three production stages function in a batch mode, wherein equipment (e.g., furnaces) processes a specific quantity of metal per instance, and this metal quantity remains consistent throughout the various stages. Each such metal quantity is referred to as a heat. Notably, since EAFs operate in a batch mode and a significant portion of electric energy is consumed during the melting stage, the timing of these batches to be produced can be adjusted by responding to time-varying prices and on-site RESs for providing operational flexibility. For simplicity, the casting stage is not considered in this work.

Fig. 1 provides description of RL for scheduling price-based DR within a steel plant. In the manufacturing execution systems (MES), the digital twin or systems model of the steel plant is constructed to mirror the real production process. This digital twin is continuously updated with three main data streams: electricity prices from the smart meter, forecasts of onsite renewable generation, and real-time operation parameters from distributed control systems (DCS).

During the offline training stage, the DRL agent interacts with the detailed systems model in a closed-loop manner and is trained to obtain a near-optimal policy. In the deployment stage, the trained DRL policy is transferred online to the real steel plant (i.e. on the detailed systems model in MES). Here, the MES executes the learned scheduling policy in real-time, generating control signals (start/stop times of equipment) and sending them to the DCS layer.

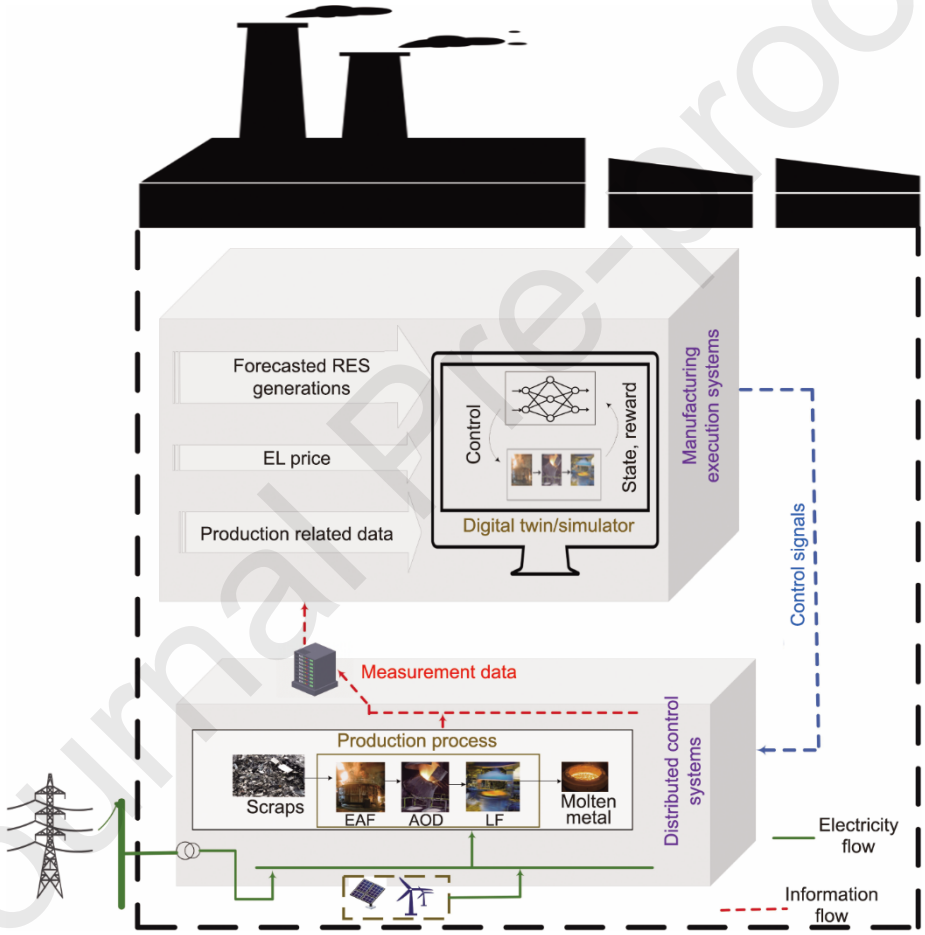


Fig. 1. Intuition of RL for DR in a steel plant. EL: electricity consumption.

2.1. MILP formulation

The production scheduling of steelmaking can be formulated as a MILP problem. The discrete-time formulation is considered here, where the optimization aims to minimize the electricity cost while satisfying critical PCs.

The resource-task network (RTN) modeling framework is able to represent complex chemical processes systematically [14]. Using RTN, the steelmaking process is represented by resources and tasks through a network structure. The resources include the process-related entities like production units, intermediate and final products, as well as electricity consumption (EL). The set of resources considered in the model is shown in Eq. (1).

$$R = \{\mathrm{EAFs},\mathrm{AODs},\mathrm{LFs}\} \cup \left\{\mathrm{EA}_h^{\mathrm{b}},\mathrm{EA}_h^{\mathrm{a}},\mathrm{AL}_h^{\mathrm{b}},\mathrm{AL}_h^{\mathrm{a}}\right\} \cup \{H_h | h \in H\} \cup \{\mathrm{EL}\} \quad (1)$$

where EAFs, AODs, and LFs represent production units in three stages; $\left\{\mathrm{EA}_h^{\mathrm{b}},\mathrm{EA}_h^{\mathrm{a}},\mathrm{AL}_h^{\mathrm{b}},\mathrm{AL}_h^{\mathrm{a}}\right\}$ represents the set of intermediate products differentiated by before and after transfer to the next process (with the superscript b and a, respectively). EA are

between stages EAF and AOD; AL$_h$ are between stages AOD and LF; EA$_h^{\mathrm{b}}$ denotes the intermediate product of heat h, which just finishes the processing at the EAF and is waiting to be transferred to the AOD; while AL$_h^{\mathrm{a}}$ denotes the same intermediate product after it is transferred to the AOD to be further processed. H_h represents the final product of heat h and H represents a set of all the heats to be produced. The tasks represent the operations performed across the three stages, as well as the transfer tasks between the stages. The set of tasks is shown in Eq. (2).

$$N = \{\mathrm{EAF}_h, \mathrm{AOD}_h, \mathrm{LF}_h, \mathrm{EA}_h, \mathrm{AL}_h | h \in H\} \quad (2)$$

where EAF$_h$,AOD$_h$, and LF$_h$ represents the processing tasks across three stages, respectively; EA$_h$ and AL$_h$ represents the two transfer tasks between the corresponding two stages.

The discrete time representation is considered here, consisting of T slots with a specific length δ (min) across 24 h. The interaction between tasks and resources is associated with an interaction parameter $\mu_{r,i,\theta}$ for each task $i \in N$. It represents how much of resource $r \in$ REL (REL is the set of all resources, excluding resource EL) is consumed or generated by task i at the relative time slot θ after the start of the task i. More detail about the illustration of the parameters can be found in Ref. [14].

Using the RTN method, we present the MILP model of the scheduling of a steelmaking plant. The model aims to minimize the electricity cost by considering critical steelmaking constraints. The objective is represented in Eq. (3). Here, it is assumed that surplus onsite RES power cannot be sold back to the power grid to earn additional revenue.

$$\min \sum_{t\in\{0,\ldots,T\}} \lambda_{\mathrm{EL},t} \cdot \max[0, (\vartheta_{\mathrm{EL},t} - E_{\mathrm{res},t})] \quad (3)$$

where $\lambda_{\mathrm{EL},t}$ is the electricity price at t, $\vartheta_{\mathrm{EL},t}$ is the total energy consumption of all three stages from $t-1$ to t in MW·min, and $E_{\mathrm{res},t}$ is the electric energy generated from onsite RESs in MW·min at t.

The critical steelmaking constraints are listed below:

Resource balance constraints. The resource balance constraints manage the resource evolution over the time horizon. It is represented in Eq. (4).

$$R_{r,t} = R_{r,t-1} + \sum_{i\in N} \sum_{\theta=0}^{\tau_i} \mu_{r,i,\theta} I_{i,t-\theta} \quad \forall t \in T, r \in \mathrm{REL} \quad (4)$$

in which the value $R_{r,t}$ of resource r at the time slot t is equal to $R_{r,t-1}$ adjusted by the quantity produced or consumed by all tasks; $\mu_{r,i,\theta}$ represents the interaction quantity between resource r and task i at the θth time slot since the start of the task i; $I_{i,t-\theta}$ is the binary variable indicating whether task i starts at time slot $t-\theta$; τ_i is the length of task i (in time slots). The resource balance constraints for the electricity resource are formulated as

$$\vartheta_{\mathrm{EL},t} = \sum_{i\in N} \sum_{\theta=0}^{\tau_i} \mu_{\mathrm{EL},i,\theta} I_{i,t-\theta} \quad \forall t \in T \quad (5)$$

where $\vartheta_{\mathrm{EL},t}$ is the total energy consumption of the entire plant during time slot t in MW·min.

Task execution constraints. These constraints state that each heat should be processed only once within the scheduling horizon, as shown in Eq. (6). Similarly, the intermediate products should be transferred only once between the stages as defined by Eq. (7).

$$\sum_{t\in T} I_{i_{o_h},t} = 1 \quad \forall h \in H, o \in \{E,A,L\} \quad (6)$$

$$\sum_{t\in T} I_{i_{\mathrm{tr}_h},t} = 1 \quad \forall h \in H, \mathrm{tr} \in \{\mathrm{EA},\mathrm{AL}\} \quad (7)$$

where $I_{i_{o_h},t}$ is the binary variable indicating whether the arbitrary processing task of heat h in stage AOD starts at time slot t and $I_{i_{\mathrm{tr}_h},t}$ is the binary variable indicating whether the transfer task of heat h between stages EAF and AOD starts at time slot t; E, A, and L are the stages EAF, AOD, and LF, respectively;.

Transfer time constraints. Intermediate products before transfer are set to 0 because the transfer task is forced to be executed immediately. It is defined by Eq. (8).

$$R_{\mathrm{ip}_h^{\mathrm{b}},t} = 0 \quad \forall h \in H, \mathrm{ip} \in \{\mathrm{EA},\mathrm{AL}\} \quad (8)$$

where $R_{\mathrm{ip}_h^{\mathrm{b}},t}$ is the resource intermediate product (ip) of heat h before transfer at time t.

In the steel manufacturing process, low temperature can compromise the product quality, and hence it is necessary to make sure that the cooling effect during transportation does not adversely affect the processing of each heat in the next stage. It is enforced by constraints defined in Eq. (9).

$$\delta \sum_{t\in T} R_{\mathrm{ip}_h^{\mathrm{a}},t} + w_{\mathrm{ip}} \le W_{\mathrm{ip}} \quad \forall h \in H, \mathrm{ip} \in \{\mathrm{EA},\mathrm{AL}\} \quad (9)$$

where $\sum_{t\in T} R_{\mathrm{ip}_h^{\mathrm{a}},t}$ represent the total number of time slots that the ip wait before entering the next processing stage, w_{ip} represent the transfer time of the intermediate product, W_{ip} denote the maximum allowable transfer time.

Product delivery. By the end of the time horizon, the final products must be delivered. It is enforced by Eq. (10).

$$R_{H_h,T} = 1 \quad \forall h \in H \quad (10)$$

In the context of online scheduling, the MILP formulation, especially when there are uncertainties, presents significant limitations, as presented in Section 1. Therefore, we propose modeling this decision-making problem as a POMDP. The first step involves representing steel plant scheduling as a stochastic process within the POMDP framework. This approach explicitly accounts for uncertainties, such as fluctuations in electricity prices and variations in on-site renewable power, within the process dynamics. Next, we leverage RL to derive a function approximation to the optimal decision policy function for

this environment through offline model simulations. Once trained, the policy function can be deployed online to provide sche[Journal Pre-proofs]uling decisions to be made in a very short time frame (i.e., on the order of microseconds to milliseconds).

2.2. POMDP formulation

We formulate the price-based DR problem in Section 2.1 as a POMDP with a discrete finite time horizon. The steel production environment is controlled by a single agent, and the goal for the agent is to reduce the electricity cost under DR while satisfying all the critical PCs. A POMDP within a discrete time horizon (of length T) is a tuple $\langle X,U,\Theta,P,O,\phi\rangle$, where X is a state space, U is the action space, Θ is the observation space, P is the transition density function, O is the density function, and ϕ is the electricity cost. At each discrete time index $t \in \{0,1,2,...,T\}$, the state of the plant is represented by $x_t \in X$. When control action $u_t \in U$ is executed, the state changes according to the transition density function, $X_{t+1} = P(x_{t+1}|x_t,u_t)$. Subsequently, the agent receives a noisy observation $o_{t+1} \in \Theta$ according to the density function $O_{t+1}O(o_{t+1}|x_{t+1})$, and also a cost $\phi_{t+1} \sim p(\phi_{t+1}|x_t,u_t)$.

An agent acts according to its policy $\pi(u_t|o_{\leq t},u_t)$ which returns the probability of taking action u_t at time t, and $o_{\leq t} = (o_1,o_2,..., o_t)$ and $u_t = (u_1,u_2,...,u_{t-1})$ are the observation and action histories, respectively. The agent's goal is to learn a policy π that minimizes, over trajectories $\omega = (x_0,o_0,u_0,...,u_{T-1},x_T,o_T)$ induced by its policy, where $0 \leq \gamma < 1$ is the discount factor.

$$J = E_{\omega\sim p(\omega)}\left[\sum_{t=1}^{T} \gamma^{t-1}\phi_t\right] \quad (11)$$

where J is the return and $E_{\omega\sim p(\omega)}$ is expected trajectories of state–observation–action described according to probability density function $p(\omega)$.

2.2.1. System states and observations

As discussed in Ref. [23], the state x_t in the case of steel plant scheduling at a given time index $t \in \{0,...,T\}$ within the discrete finite time horizon may be highly dimensional and may also vary in its dimension 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, therefore, a significant challenge. Hence, we formulate the problem as POMDP, where a sequence of state observations collected from previous time intervals are considered for taking the next control. We denote this window of observation history as $\boldsymbol{\Omega}_t \in R^{n_w \times \mathcal{N}}$, with the window constituting those time indices $\bar{t} \in \{t - n_w,...,t\}$. The constant n_w determines the length of the window, which is treated as a hyperparameter.

The mth row vector of the observation matrix, which is denoted as $\boldsymbol{\Omega}_t$ at a given time index t within the discrete finite time horizon (of length T), is equivalent to the mth observation within the window,

$$o_m = [o_{m,1},...,o_{m,\mathcal{N}}]^{\mathrm{T}} \quad (12)$$

where $o_{m,n}$ is the nth element of mth row vector of observation matrix $\boldsymbol{\Omega}_t$, and $\mathcal{N}$ is the length of observation vector.

Each element within the observation vector, $n \in \backslash\{1,...,\mathcal{N}\backslash\}$, represents information from the steel production environment. The columns ($1 \leq n \leq 5$) represent the task information for stage 1 (EAF): heat number, processing time, start and end times, and the unit used (from two parallel units). Similarly, $6 \leq n \leq 10$ represents the same information for stage 2 (i.e., the AOD stage), and $11 \leq n \leq 15$ represents the information for stage 3 (i.e., the LF stage). The final three columns, $16 \leq n \leq 17$, represent the exogenous information of electricity price and power generation from on-site RESs. At time t, we receive information about prices a time step ahead (i.e., the prices at $t + 1$). This structure allows the observation to describe the future energy market as allowed for by market mechanisms, which typically fix the price a short period ahead of time.

2.2.2. Actions

In this work, the control decisions are discrete. They represent heuristic decision rules at a given time slot. At a given time, indexed by $t \in \{0,...,T\}$, the agent selects the control action u_t^j (u_t^j represents the available heuristic decision rules that can be adopted for the n_u units in each stage, $j \in \{E,A,L\}$ and $s \in \{1,...,n_u\}$; Eq. (13)). For stage 1 (at EAFs; i.e., $u_t^E, u_{s,t} \in Z_9$ represents nine scheduling rules for stage 1), as listed in Table 1. Here Z_k indicates the set of integers from 1 to $k \in Z$.

$$\begin{cases} u_t = \left[u_t^{\mathrm{EAF}},u_t^{\mathrm{AOD}},u_t^{\mathrm{LF}}\right]^{\mathrm{T}} \\ u_t^j = [u_{1,t},...,u_{s,t},...,u_{n_u,t}]^{\mathrm{T}} \end{cases} \quad (13)$$

Table 1
Scheduling rules in stage 1 (at EAFs).

Scheduling rule	Description (prioritize the process of)
1	The heat with the shortest processing time (SPT)

	The heat with the longest processing time (LPT)
2	The heat with the lowest total energy consumption (LEC)
3	The heat with the highest total energy consumption (HEC)
4	The heat with the shortest remaining machine time (SRM)
5	The heat with the longest remaining machine time (LRM)
6	The heat with nearest due time (NDD)
7	The heat with farthest due time (FDD)
8	Nothing—don't do anything
9	

For stage 2 (at AODs; i.e., u_t^A, $u_{s,t} \in Z_{11}$ represents 11 scheduling rules for stage 2) as listed in Table 2.

Table 2
Scheduling rules in stages 2 (at AODs).

Scheduling rule	Description (prioritize the process of)
	The heat with SPT
1	The heat with LPT
2	The heat with LEC
3	The heat with HEC
4	The heat with SRM
5	The heat with LRM
6	The most former heat in the transfer area (MFT)
7	The most recent heat in the transfer area (MRT)
8	The heat with NDD
9	The heat with FDD
10	Nothing—don't do anything
11	

For stage 3 (at LFs; i.e., u_t^L, $u_{s,t} \in Z_9$ represents nine scheduling rules for stage 3) as listed in Table 3.

Table 3
Scheduling rules in stages 3 (at LFs).

Scheduling rule	Description (prioritize the process of)
	The heat with SPT
1	The heat with LPT
2	The heat with LEC
3	The heat with HEC
4	MFT in the transfer area
5	MRT in the transfer area
6	The heat with NDD
7	The heat with FDD
8	Nothing—don't do anything
9	

The heuristic rule SPT selects the heat with the shortest processing time in that stage, while LPT selects the heat with the longest. LEC chooses the heat with the lowest total energy consumption over its production time, whereas HEC selects the

highest. In stage 1, SRM calculates the total processing time of the remaining two stages and selects the heat with the shortest rem… …ects the heat with the longest remaining processing time. For NDD and FDD, each heat is assigned a due time in $t \in \{0,1,2,...,T\}$; NDD selects the nearest due time, while FDD selects the farthest. MFT and MRT consider heats that have completed transfer and are waiting for the next stage, which is why these rules don't apply to stage 1. MFT selects the heat with the earliest transfer end time, whereas MRT selects the most recent.

2.2.3. Electricity cost

The electricity cost ϕ_t of the plant at time slot t is defined as below. The aim is to minimize the sum of ϕ_t over the time horizon in expectation.

$\phi_t = (\lambda_{\mathrm{EL},t} \cdot \max[0,\ (\vartheta_{\mathrm{EL},t} - E_{\mathrm{res},t})]$ (14)

3. Methodology

3.1. Constraint handling

The scheduling of the steelmaking process is often recognized as one of the most intricate industrial scheduling problems as it is a large-scale, multistage, multiproduct batch process encompassing parallel equipment and critical production-related constraints [24]. In this work, starting times of the tasks are moved in time to provide the operational flexibility. The restriction on the control space arises from PCs within the steelmaking process, as discussed in Section 2.1. We hypothesize that the constraint set for the time index t is $\grave{A}_t^j \subseteq H$, where $j \in \{E,A,L\}$. Note that the parallel units within each stage will have a separate constraint set. For example, $\grave{A}_{s,t}^E \subseteq H$ denotes the constraint set for sth unit of stage EAF at time slot t. PCs can generally be expressed logically. We assume that these logical operations can be expressed as a point to set mapping, f_{PC}:$X \rightarrow P(H)$, where $P(H)$ indicates the power set of set of heats. We will detail each constraint individually, explaining the logic employed in our DRL model.

Consider the constraint in Eq. (4), in our DRL framework, this constraint can be interpreted in Boolean terms. Consider an equipment resource (e.g., EAF). If no tasks are using EAF at time t, the constraint evaluates as True, indicating that EAF is available to process a new task. If a task is currently using EAF, the constraint evaluates as False, preventing scheduling conflicts. This logical perspective allows the DRL agent to learn feasible actions while respecting resource capacities.

The total electricity consumption at time t is the sum of all tasks (i_{o_h}) currently being processed as shown in Eq. (5). In the DRL framework, for each task (i_{o_h}), the total energy requirement is estimated from its processing time and the equipment's nominal power and then distributed over the task's duration (τ_i) to form a time-dependent energy profile. Summing the contributions of all tasks active at time t gives the total electricity consumption at that time. Thus, the electricity cost is computed according to the actual processing time of each task, following the same approach as in the MILP formulation while resource blocking may be rounded to discrete time steps to align with the scheduling grid.

In the DRL framework, the task execution constraints (Eqs. (6) and (7)), are again handled via Boolean terms. Each processing (i_{o_h}) and transfer task (i_{tr_h}) is associated with a Boolean flag indicating whether it has been completed (True) or is still available (False). At each discrete time step t, the agent can only select tasks that are still False, forming the set of feasible actions that belong to the constraint set $\grave{A}_t^j$. This ensures that each heat is processed and each intermediate transfer is executed exactly once, as in the MILP model. By updating the flags after each action, the DRL agent is prevented from selecting tasks that would violate these constraints.

For the transfer constraint in Eq. (8), a transfer task (i_{tr_h}) at time step t can only be scheduled if it is still False (i.e., unprocessed) and its preceding processing task has just completed, ensuring that no intermediate product accumulates before transportation. Similarly, to satisfy the condition in Eq. (9), a processing task (i_{o_h}) in stage 2 at discrete time t can only be processed if it is associated with False Boolean and its preceding transfer task has not violated w_{EA} and W_{EA} limits.

At time t, we select heats from the available control sets through discrete control decisions $u_t^j \in U$, which represent heuristic decision rules. These heuristic decision rules are defined with respect to the constraint set $\grave{A}_t^j$ and directly assign tasks available in $\grave{A}_t^j$ at time t. These constraint sets are identified in a state dependent fashion via $f_{\mathrm{PC}}(x)$. It is worth noting that the current state of the plant is assumed available, but not compatible with the prediction mechanisms of function approximations; hence the use of a POMDP formulation. If we are able to define and satisfy all constraints via f_{PC} then the assignment decision at time t will satisfy constraints imposed by construction. In the case one is unable to identify (and satisfy) all constraints via f_{PC}, we penalize the violation of those constraints that cannot be handled and incorporate a penalty function for the constraint

violation. $\phi_t^p = \varphi(\phi_t^p, x_t, u_t)$. For example, in our problem, the constraint detailed by Eq. (10) is enforced through this penalty method. We define the expected penalized return as

$$J^{\varphi} = E_{\pi}\left[\sum_{t=1}^{T} \gamma^{t-1} \phi_t^p\right] \quad (15)$$

where φ is the penalty function.

We can then identify a solution policy π as follows:

$$\min_{\pi} J^{\varphi} \quad (16)$$

Alternatively to the penalty method, one may leverage the flexibility of evolutionary search to maintain only feasible policies. While feasibility cannot be guaranteed theoretically, in practice these mechanisms ensure constraints are effectively respected.

3.2. Policy function design

We propose to utilize the attention mechanism, which has become an integral part of sequence modelling in various tasks [25]. In this context, the attention mechanism could construct a representation of the most relevant information within the observation matrix $\boldsymbol{\Omega}_t$ and help agents to learn long-range dependencies between observations within the historical window. The attention in our proposal is the single-head attention (SHA) utilized in the transformer model, known as scaled dot-product attention [25]. The attention mechanism is described by Eq. (17),

$$\text{Attention}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \text{SoftMax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\text{T}}}{\sqrt{d_{\text{k}}}}\right)\boldsymbol{V} \quad (17)$$

where $\boldsymbol{Q}$ is a matrix of queries, $\boldsymbol{K}$ is a matrix of keys of dimension d_{k}, and $\boldsymbol{V}$ is a matrix of values. The matrices $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ are functions of the input observation matrix $\boldsymbol{\Omega}_t$. They are stage-specific; for stage 1, we denote them Q_{E}, K_{E}, and V_{E} each of dimension $R^{n_{\text{w}}\times 3}$. This allows the model to jointly attend to information from different representation subspaces, which reflect different aspects of steel plant scheduling characteristics. Fig. 2 represents the attention-based policy structure. The observation matrix $\boldsymbol{\Omega}_t$ is partitioned and fed forward through different modules within the policy structure. Specifically, $[\boldsymbol{\Omega}_t]_{\text{EAF}}$ corresponds to elements with column index $1 \le n \le 5$ as defined in Section 2.2.1, $[\boldsymbol{\Omega}_t]_{\text{AOD}}$ to $6 \le n \le 10$ and $[\boldsymbol{\Omega}_t]_{\text{LF}}$ to $11 \le n \le 15$. $[\boldsymbol{\Omega}_{t+1:t}]_{\text{price}}$ represents electricity price information $(n = 16)$ and $[\boldsymbol{\Omega}_{t+1:t}]_{\text{gen}}$ represents power generation from on-site RESs $(n = 17)$.

As shown in Fig. 2, $[\boldsymbol{\Omega}_t]_{\text{EAF}}$, $[\boldsymbol{\Omega}_t]_{\text{AOD}}$, and $[\boldsymbol{\Omega}_t]_{\text{LF}}$ are passed through a QKV layer, followed by an attention operation and a subsequent linear transformation. In contrast, the electricity price and on-site power generation matrices bypass these operations. The attention layer produces an output matrix of dimension $R^{n_{\text{w}}\times 3}$ for each stage. After the linear transformation, the resulting matrices are concatenated to form a final matrix $\boldsymbol{\Omega}_t^{\text{a}}$ of dimension $R^{n_{\text{w}}\times 11}$. Further details of these operations are provided in the Appendix A. The entire policy function design for identifying actions (i.e., the heuristic decision rules to be adopted) is shown in Fig. 3. $\boldsymbol{\Omega}_t^{\text{a}}$ is processed through a feedforward neural (FFN) network, and then the resultant matrix is passed through distinct linear layers corresponding to each stage. Note that in Fig. 3, only three linear layers are shown corresponding to three stages, but there are six because of two parallel units in each stage. The outputs of these linear layers provide logits used to define a SoftMax policy, which is used to find the decision rule with the highest probability for each stage. As a result, each unit within each of the stages requires output dimensionality equivalent to the number of heuristic decision rules. Based on the selected decision rule, we then use the constraint set $\grave{A}_t^j$ for scheduling heat h at time t as described in Section 3.1.

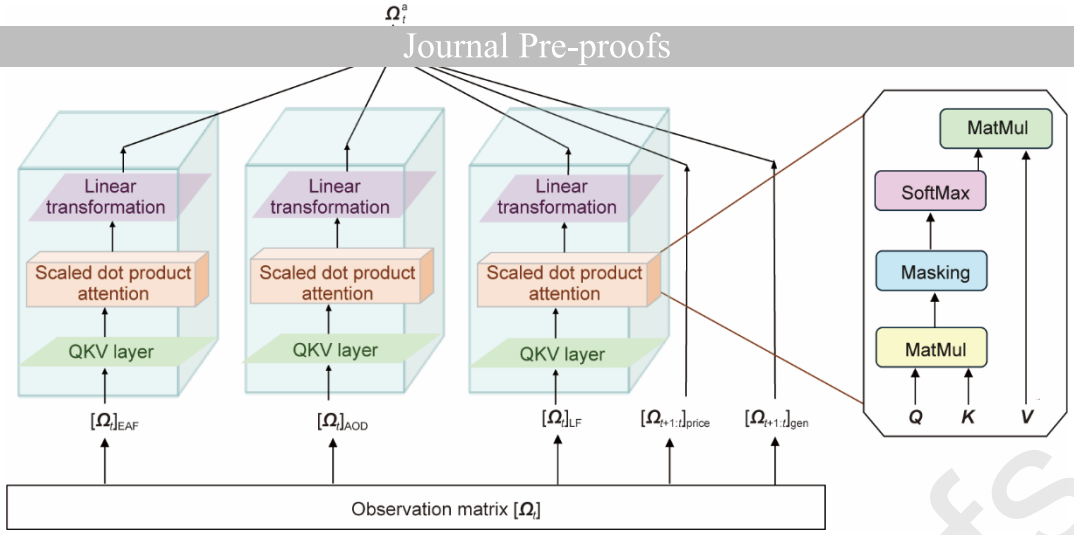


Fig. 2. Attention-based state formulation.

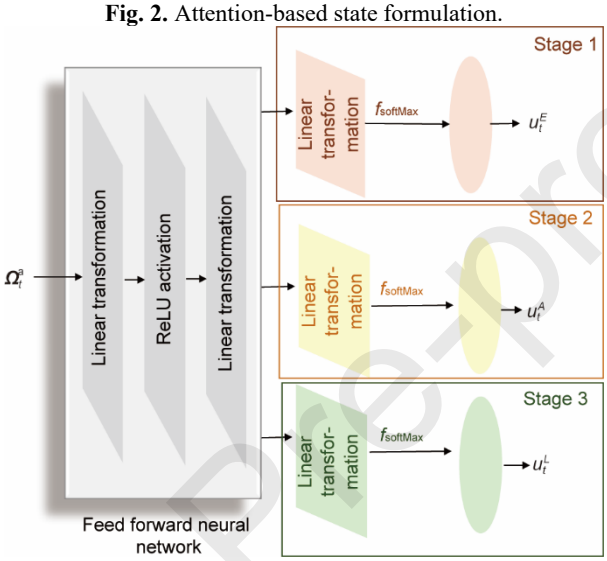


Fig. 3. Heuristic decision rules policy function design. f_{softMax}: the SoftMax policy.

3.3. Stochastic search policy optimization algorithm

In this work, we adopt an evolutionary strategy, which is a class of black box optimization algorithms with heuristic search procedures inspired by natural evolution to find a near optimal policy. The primary reason for selecting this category of algorithm is its scalability to high dimensions and parallelizability, which are highly beneficial for our problem. A high-level description of the algorithm used in this work is provided by Algorithm 1.

Algorithm 1. A generalized population-based policy optimization framework.

Input: Neural network (policy function), $\pi(\grave{\theta}, \cdot)$ parametrized by $\grave{\theta}$; a number of samples to evaluate each neural network, n_S; a general population-based optimization algorithm, f_{DRL}; number of optimization iterations, I; population size of candidate policy functions, $\mathcal{P}$; upper, θ_{UB}, and lower, θ_{LB}, bounds on the search space; population initialization method, $f_{\text{init}}(\grave{\theta},\mathcal{P},\theta_{\text{LB}},\theta_{\text{UB}})$; a memory buffer, B; $J_\pi = \infty$

I Generate initial policy parameters $\Theta_1 = f_{\text{init}}(\grave{\theta},\mathcal{P},\theta_{\text{LB}},\theta_{\text{UB}})$ where $\Theta_1 = \{\theta_1,...,\theta_{\mathcal{P}}\}$;

II **for** $i = 1,2,...,I$ **do**

1 Construct policy population $\Pi_i = \{\pi_{i,k}(\theta) \quad \forall\theta \in \Theta_i\}$

2 for $\pi_{i,k} \in \Pi_i$, **do**

(i) Receive observation of initial state o_0

(ii) **for** $t = 0,1,2...,T-1$ **do**

(ii.1) Generate control u_t^j

(ii.2) Receive next observation o_{t+1}

(ii.3) Observe electricity cost ϕ_{t+1}

(iii) Observe the penalized return J^{φ} (Eq. (15))

B Calculate sample average approximation $J_{i,k}$ (Eq. (16)) and append $(\pi_{i,k}, J_{i,k})$ to B

C If $J_{i,k} < J_{\pi}$, then update the best-known policy (π, J_{π})

3 Generate new parameters $\Theta_{i+1} = f_{\mathrm{DRL}}(\mathrm{B})$

III Best ranked policy, π

The first step is to generate a population of parameters for a policy (neural network). Then, in step II, the following steps are performed iteratively: ① a population of neural networks, are constructed from the population of parameters; ② A and B calculate the average return of all the samples obtained after performing steps (i)–(iii); C is the current best policy is tracked and stored in memory (if a criterion is satisfied); ③ the population parameters are then perturbed according to a stochastic search optimization algorithm. After a number of optimization iterations, the best policy identified is output. The number of optimization iterations is defined according to the available computational budget or according to a threshold on the rate of improvement. The stochastic search optimization algorithm, f_{DRL}, which is used in this work is separable natural evolution strategy (SNES). For further information on SNES, please see the Appendix A. We also provide flowchart of the scheduling process within a discrete time-grid for each sample. It is shown in Fig. 4.

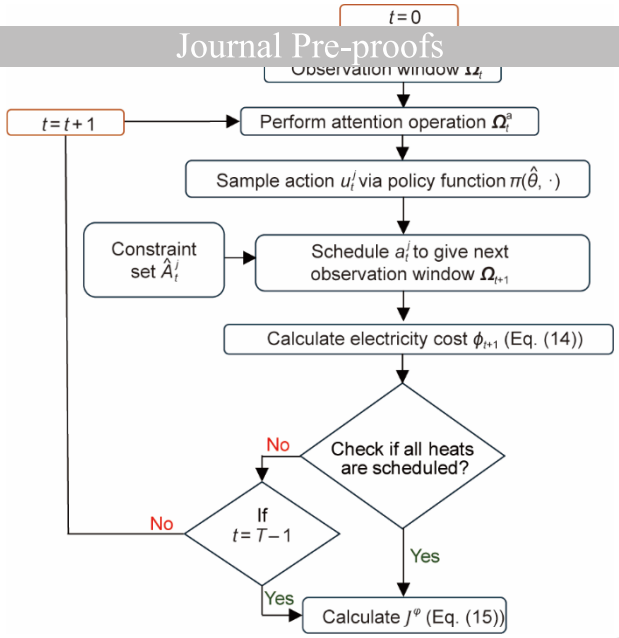


Fig. 4. Flowchart of the scheduling process within a discrete time-grid.

4. Case study

In this section, the description of the test system and associated cases are presented.

4.1. Case study design

The steel plant comprises three stages, each with two parallel devices: two EAFs, two AODs, and two LFs. The casting stage is not considered in this work. Parameters corresponding to the plant are drawn from Ref. [26]. The plant is equipped with a local wind power system, and the wind power data is from Ref. [27]. The hourly electricity prices adopted are based on the NORD POOL (UK) wholesale electricity prices [28]. They are shown in Fig. 5. Details of the nominal power consumption of the units, minimum and maximum transfer times, and processing times of heats are detailed in the Appendix A. The scheduling horizon spanned a 24 h production cycle divided into 96 time slots, with each being 15 min. The objective is to minimize the electricity cost of the steel plant. The problem is modeled as a POMDP with a discrete time formulation.

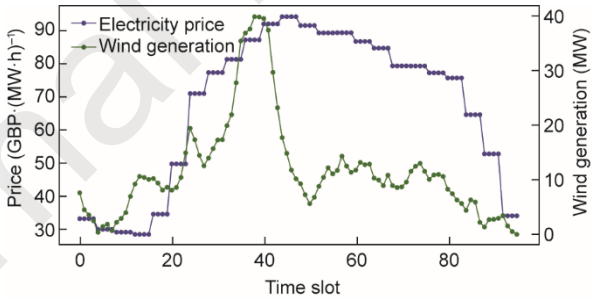


Fig. 5. Day-ahead wholesale electricity prices and wind power generation.

Our method employs a heuristic decision policy combined with an evolutionary search for policy optimization (ES-HP). The first part of the results (Section 4.2) evaluates the ES-HP's performance under various conditions, discussed in detail below. The second part (Section 4.3) presents an ablation study, comparing the contributions of the method with other approaches, such as validating attention-based state formulation, comparing the heuristic decision policy with a direct assignment policy, comparing evolutionary search optimization with proximal policy optimization (PPO) and the impact of varying hyperparameters, such as the population size of candidate policy functions.

We would like to validate the method through various case studies, as detailed in Table 4. We consider two different distinct problem instances to assess its scalability and ability to maintain a similar optimality gap with larger problems. The first instance (P1) involves a smaller problem size, focusing on scheduling 12 heats per day. The second instance (P2) examines the method's performance with larger problems, handling 24 heats. We also evaluate the method using two sets of processing time data for the heats: the first set (PT1) exhibits more uniform processing times, while the second set (PT2) shows greater variance. This analysis aims to assess the performance of ES-HP, given its decision rules are influenced by the heat's processing times. The table is, therefore, characterized by case study 1 (CS1), case study 2 (CS2), and case study 3 (CS3). In section 4.2, for all the studies, a benchmark is provided wherein the optimization problem is formulated under a discrete time MILP framework. Firstly, we evaluate the performance of the method under uncertainty scenarios, as detailed in Table 5. This is discussed in more detail in section 4.2.1. Secondly, in section 4.2.2, we evaluate the method using two distinct problem instances. P1 corresponds to 8 640 constraints, 11 871 integer variables, and 288 continuous variables in the benchmark MILP formulation. P2 includes 16 800 constraints, 23 451

integer variables, and 288 continuous variables. Thirdly, in section 4.2.3, we evaluate the method using two sets of processing time data. Finally, in Section 4.2.4, we compare our proposed method with other classical approach. In practice, unit scheduling is typically carried out according to a specific rule, often referred to as a state-of-the-art (SOTA) heuristic. To reflect this, we adopt one heuristic decision rule (among those employed in ES-HP) at a time and evaluate its performance against our method and the MILP benchmark.

Table 4
Case study design.

Problem size	Processing time data (PT1)	Processing time data (PT2)
P1	CS1	CS3
P2	CS2	—

Table 5
Uncertainty conditions.

Scenario	σ = 10%		σ = 20%	
	Electricity price uncertainty	Wind power uncertainty	Electricity price uncertainty	Wind power uncertainty
S1	×	×	×	×
S2	✓	×	✓	×
S3	×	✓	×	✓
S4	✓	✓	✓	✓

σ: standard deviation.

4.2. Results

4.2.1. Performance of the proposed method under uncertainty scenarios

In this section, we demonstrate the ability of ES-HP to handle uncertainty in electricity prices and uncertainty in wind power generation. In a scenario without uncertainty (S1), we compare the day-ahead production schedule generated by ES-HP and MILP. The MILP has full information about steel plant dynamics and utilizes the accurate model to minimize the electricity cost, which leads to a mathematically optimal result. In contrast, ES-HP utilizes its learning capability to explore the action space to maximize the reward. In the remaining scenarios (S2, S3, and S4) where uncertainty is considered, we assume that the day-ahead forecast electricity price and wind power are subject to forecast error, which follows a normal distribution with the mean $\mu = 0$ and the standard deviation σ = 10% and 20%. MILP optimizes the production schedule solely based on the day-ahead forecasts. By contrast, the ES-HP optimizes 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 and wind power of the current hour, and only the schedule of the current hour is executed. CS1 is considered here. In learning, the policy was evaluated over n_S= 200 samples (n_s = 1 when there is no uncertainty), with a population size of $\mathcal{P} = 1000$ and maximum optimization iterations of $I = 100$. The performance indicators used to evaluate the respective methods include the expected performance of the scheduling policy, which is the net electricity cost, $\mu_z = E[Z]$, the standard deviation of the performance $\sigma_z = \sum_\pi[Z]$ and the conditional-value-at-risk [9], $\bar{\mu}_\beta$ with β = 0.1 and 0.25 (i.e., $\bar{\mu}_\beta$ represents the expected value of the worst-case policy performance observed with probability less than or equal to 0.1 and 0.25). MILP is solved by the 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.

Fig. 6 shows the cumulative electricity cost for S1-CS1 under ES-HP and MILP. The cumulative costs for ES-HP and MILP are 29.14 and 27.52 kGBP, respectively. It is seen that the optimality gap is 5.75% in comparison to MILP. The difference in the optimality gap can be attributed to non-smooth mapping between state and optimal control. To illustrate the results more clearly, the aggregated energy consumption of all equipment is plotted in Fig. 7. As it is seen, in the case of ES-HP, the production continues until time step 56. But the net power consumption from time step 45 to 56 is null. This is because the wind generation is higher during those time slots, which makes $\vartheta_{\mathrm{EL},t} - E_{\mathrm{res},t}$ as negative. Hence, it incurs no electricity cost. We also present the production schedules generated by ES-HP and MILP in Fig. 8, demonstrating that they comply with the key PCs.

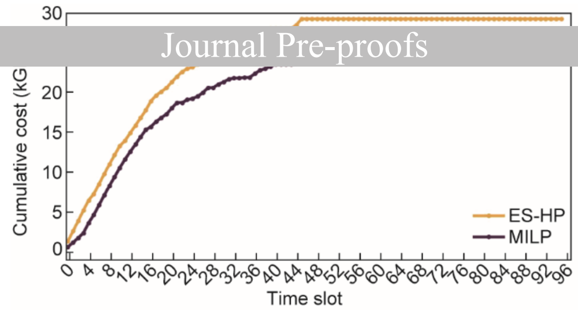


Fig. 6. Cumulative electricity cost for S1-CS1 under ES-HP and MILP.

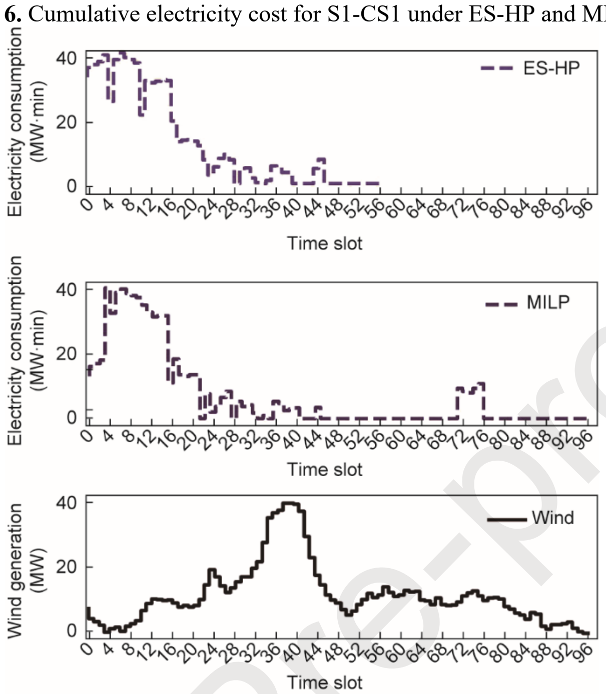


Fig. 7. Aggregated electricity consumption for S1-CS1 under ES-HP and MILP.

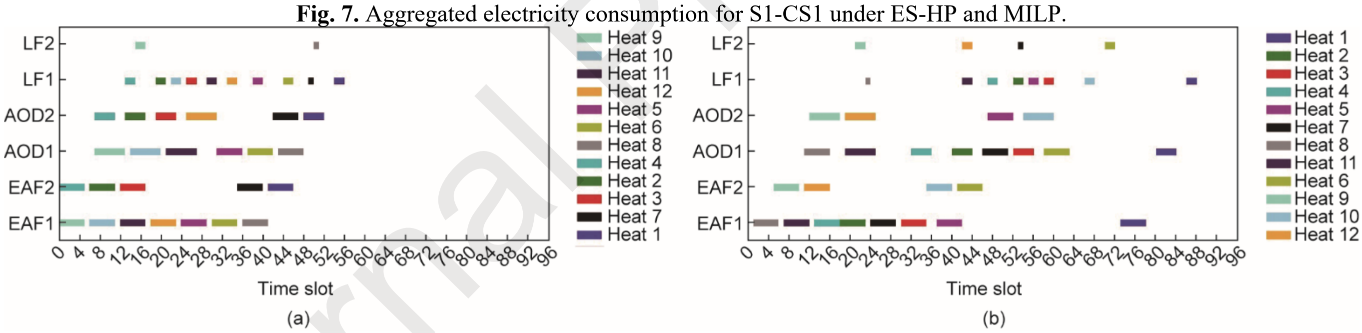


Fig. 8. (a) production schedule generated by ES-HP and (b) production schedule generated by MILP.

Next, we turn our attention to the handling of electricity price and wind power uncertainty. This corresponds to scenarios S2, S3, and S4. MC simulation is used to generate 500 scenarios of the true electricity price and wind power throughout the day via sampling from the normal distribution of the forecast error. As ES-HP aims to optimize the production schedule in a rolling manner, for the 500 scenarios, the ES-HP will generate 500 different production schedules. Then, the actual net electricity costs of the production schedules in each scenario, calculated given the true electricity price and wind power values, are calculated for the schedules generated by both the MILP and ES-HP approaches. The average net electricity costs of the 500 scenarios are presented in Table 6 for experiments in S2, S3, and S4.

In both the cases (σ = 10% and 20%) , it is observed that μ_z of the policies by ES-HP is competitive with the MILP approach. Generally, the standard deviation σ_z of μ_z across all the experiments is competitive with MILP while in some cases even surpasses it (i.e., S4 when σ = 10% and S3 when σ = 20%). In terms of conditional value at risk (CvaR) $\bar{\mu}_\beta$, in all the cases, the performance is competitive with MILP. Further, in all of the experiments, constraints are respected absolutely across all 500 MC simulations by both the methods.

Table 6
Results for scenarios considering uncertainty.

Scenario	ES-HP (σ = 10%)				MILP (σ = 10%)				ES-HP (σ = 20%)				MILP (σ = 20%)			
	μ_z (kGBP)	σ_z (kGBP)	$\bar{\mu}_\beta$ (10%, kGBP)	$\bar{\mu}_\beta$ (25%, kGBP)	μ_z (kGBP)	σ_z (kGBP)	$\bar{\mu}_\beta$ (10%, kGBP)	$\bar{\mu}_\beta$ (25%, kGBP)	μ_z (kGBP)	σ_z (kGBP)	$\bar{\mu}_\beta$ (10%, kGBP)	$\bar{\mu}_\beta$ (25%, kGBP)	μ_z (kGBP)	σ_z (kGBP)	$\bar{\mu}_\beta$ (10%, kGBP)	$\bar{\mu}_\beta$ (25%, kGBP)

S2									30.08	1.10	31.12	30.05	27.77	0.92	29.43	28.99
S3	29.80	0.85	31.49	31.02	28.33	0.83	29.82	29.41	31.56	1.42	32.42	32.42	29.63	1.49	31.61	31.61
S4	30.25	0.94	33.36	32.23	28.39	0.96	30.15	29.65	31.72	1.92	34.53	34.53	29.65	1.85	32.01	32.01

4.2.2. Impact of problem size

In this section, we turn our attention to validate our method in case of a larger problem instance. Table 7 presents the results for both problem sizes (i.e., S4-CS1 (of P1) and S4-CS2 (of P2)), under uncertainty level $\sigma = 10\%$ for ES-HP and the MILP benchmark.

The average electricity costs for ES-HP and MILP are 113.05 and 106.85 kGBP, respectively, for S4-CS2. The performance gap is 5.64%. It is seen that ES-HP outperforms MILP in σ_z. However, there is some discrepancy between the $\overline{\mu}_\beta$. In the case of S4-CS1, we can see a performance gap of 6.5%. These results validate that the proposed method remains effective even for larger problem instances, maintaining a comparable optimality gap.

Table 7
Performance of ES-HP and MILP under P1 and P2.

Scenario	ES-HP				MILP			
	μ_z (kGBP)	σ_z (kGBP)	$\overline{\mu}_\beta$ (10%, kGBP)	$\overline{\mu}_\beta$ (25%, kGBP)	μ_z (kGBP)	σ_z (kGBP)	$\overline{\mu}_\beta$ (10%, kGBP)	$\overline{\mu}_\beta$ (25%, kGBP)
S4-CS1	30.25	0.94	33.36	32.23	28.39	0.96	30.15	29.65
S4-CS2	113.05	1.24	127.36	121.96	106.85	1.73	109.87	109.10

4.2.3. Impact of processing time data

In this section, we turn our attention to validate our proposed method in case of a different set of processing time data. Table 8 presents the results for instances (i.e., S4-CS1 (of PT1) and S4-CS3 (of PT2)) under uncertainty level $\sigma = 10\%$ for ES-HP and the MILP benchmark.

The average electricity cost for ES-HP and MILP are 22.94 and 20.56 kGBP, respectively, under S4-CS3. Hence, it is validated that with a greater variance of processing time data, the proposed method still works well by operating within a similar optimality gap in terms of μ_z, σ_z, and $\overline{\mu}_\beta$.

Table 8
Performance of ES-HP and MILP under PT1 and PT2.

Scenario	ES-HP				MILP			
	μ_z (kGBP)	σ_z (kGBP)	$\overline{\mu}_\beta$ (10%, kGBP)	$\overline{\mu}_\beta$ (25%, kGBP)	μ_z (kGBP)	σ_z (kGBP)	$\overline{\mu}_\beta$ (10%, kGBP)	$\overline{\mu}_\beta$ (25%, kGBP)
S4-CS1	30.25	0.94	33.36	32.23	28.39	0.96	30.15	29.65
S4-CS3	22.94	0.99	24.90	24.39	20.56	0.92	22.21	21.73

4.2.4. Comparison with SOTA heuristic rules

In this section, we validate the effectiveness of our proposed method by comparing its performance with a set of classical heuristic rules. Table 9 reports the results for instance S1-CS1, showing the performance of each heuristic rule, our ES-HP method, and the MILP benchmark. For each heuristic rule, we apply a fixed scheduling policy without optimization. The rules considered include: SPT, LPT, LEC, HEC, NDD, and FDD. We compare the electricity cost (in kGBP) obtained from each approach.

As shown in Table 9, all heuristic rules perform significantly worse than both ES-HP and the MILP benchmark, with electricity costs ranging from 44.67–49.31 kGBP, compared to 29.14 kGBP for ES-HP and 27.52 kGBP for MILP. This case study highlights the limitations of relying solely on SOTA heuristics, which lead to higher operational costs. In contrast, our ES-HP method consistently achieves performance much closer to the MILP benchmark. This demonstrates that ES-HP outperforms traditional SOTA heuristics.

Table 9
Performance of heuristic rules, ES-HP, and MILP under PT1.

Method	Electricity cost (kGBP)
Heuristic rule	

	49.31
SP	
	48.10
LPT	
	45.34
LEC	
	44.67
HEC	
	47.01
NDD	
	47.06
FDD	
	29.14
ES-HP	
	27.52
MILP	

4.3. Ablation results

4.3.1. Performance of attention-based representation

In this section, we focus on validating our method both with and without the use of the attention mechanism. In the case where we don't use the attention mechanism for the policy function design, we employ a multi-layer perceptron (MLP) to predict heuristic decision rules. The observation matrix $\boldsymbol{\Omega}_t$ is flattened into a vector before being processed by the MLP. The MLP consists of an input layer composed of 68 nodes; hidden layer 1 composed of 8 feedforward nodes, hidden layer 2 composed of 8 feedforward nodes, hidden layer 3 composed of 12 feedforward nodes, and hidden layer 4 composed of 3 nodes. The rectified linear unit (ReLU) activation function was applied across all hidden layers except the hidden layer 4. The MLP consists of three output linear layers; each of those dimensions is equal to the number of heuristic decision rules as described in Fig. 3, where the predictions are made via the policy function in a continuous latent space, $w_t \in W \subseteq R^{n_u}$. Thereafter, a SoftMax policy is used to find the decision rule with the highest probability. Based on the selected decision rule, we then use the set $\bar{a}_t$ for scheduling heat. We will consider S4-CS1 with uncertainty level $\sigma = 10\%$. We performed five runs, and the average μ_z , σ_z, and $\bar{\mu}_\beta$ across the five runs are shown in Table 10.

It is observed that, using the attention mechanism, the method performs better in μ_z, σ_z, and $\bar{\mu}_\beta$. This result highlights its advantage, particularly in partially observable environments like this. The attention mechanism can help aggregate information across time-steps by emphasizing observations or features that contribute most to inferring the true underlying state, thereby improving decision-making.

Table 10
Performance of ES-HP with and without using attention mechanism.

Module	μ_z (kGBP)	σ_z (kGBP)	$\bar{\mu}_\beta$ (kGBP)	
			10%	25%
Using attention	32.21	0.22	37.26	36.23
Without using attention	33.99	0.30	39.34	38.56

4.3.2. Comparison of heuristic decision rules and direct assignment decisions

In this section, we turn our attention to demonstrate the case when we identify discrete assignment policy (DAP) instead of identifying the heuristic decision rules as the output. DAP directly represents identifying the tasks as output, which can be directly assigned to the units. For assigning direct decisions, predictions are made via the policy function in a continuous latent space,$w_t \in W \subseteq R^{n_u}$. The predictions are further normalized in the range of heats. The set $\bar{a}_t$ is then used to implement an action-masking scheme [29], which enables a SoftMax policy to satisfy PCs by construction. Briefly, this ensures that invalid controls have zero probability of selection under the SoftMax policy. We will consider the S4-CS1 with uncertainty level $\sigma = 10\%$. We performed five runs, and the average μ_z, σ_z, and $\bar{\mu}_\beta$ across the five runs for ES-HP and ES-DAP are shown in Table 11.

As it is seen from the results, ES-DAP outperforms ES-HP in terms of μ_z, σ_z, and $\bar{\mu}_\beta$. However, the computational time cost in policy identification is more in the case of ES-DAP. We validated it by comparing the number of function evaluations to reach the optimal policy for ES-HP and ES-DAP for S1-CS1. The training progress plot for the five runs for both the policies is shown in Fig. 9.

Table 11
Comparison of heuristic decision rules and direct assignment decisions.

Method	μ_z (kGBP)	σ_z (kGBP)	$\bar{\mu}_\beta$ (kGBP)

			10%	25%
	29.11	0.94	33.36	32.23
ES-HP				
	29.09	0.91	31.61	31.17
ES-DAP				

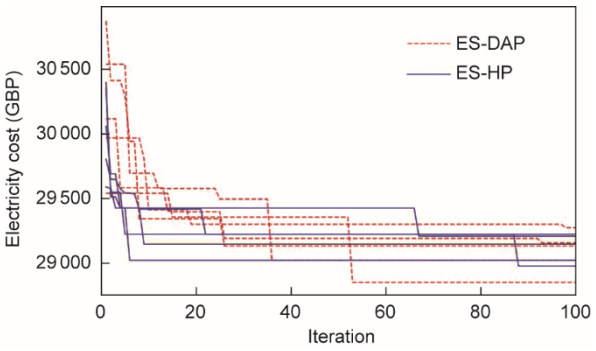


Fig. 9. Number of function evaluations to reach optimal policy for S1-CS1 under ES-HP and ES-DAP.

As seen from Fig. 9, ES-HP tends to make faster progress than ES-DAP. On average, ES-HP requires 55 800 function evaluations (calculated by multiplying iterations at which the optimal policy is observed by population size, e.g., for the first run: 15 × 1500 = 22 500, then averaging over five runs) to reach optimal policy, whereas ES-DAP requires 94 200 evaluations. However, ES-HP occasionally tends to converge at a slightly higher electricity cost than ES-DAP. This is because of the bias in the policy construction and POMDP formulation. Thereafter, we validated our method on a larger problem size (i.e., S1-CS2), with the results plotted in Fig. 10.

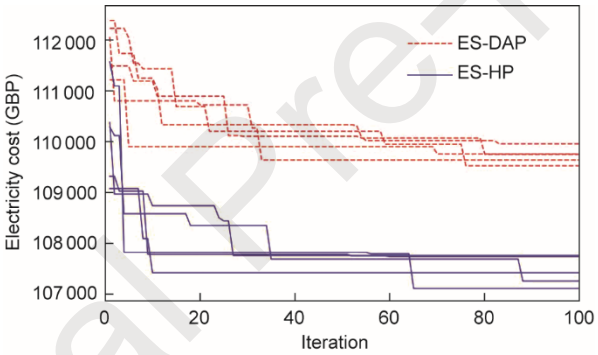


Fig. 10. Number of function evaluations to reach optimal policy for S1-CS2 under ES-HP and ES-DAP.

In this case, ES-HP requires 69 300 function evaluations, whereas ES-DAP requires 103 500 function evaluations. It is seen that with a larger problem size, the ES-HP not only tends to make faster progress than ES-DAP but also converges at better values (i.e., lower electricity cost). This validates our approach being more computational efficient. Also, when we mis-specify the uncertainty (i.e., we trained ES-DAP and ES-HP on S3-CS1 with uncertainty level $\sigma = 10\%$), but we validated its performance on S4-CS1, below are the results as shown in Table 12 as average across five runs. It can be observed that in such cases, ES-HP tends to be more robust.

Table 12
Comparison of ES-HP and ES-DAP under misspecification of uncertainty.

Method	μ_z (kGBP)	σ_z (kGBP)	$\bar{\mu}_\beta$ (kGBP)	
			10%	25%
	31.46	0.92	34.22	33.10
ES-HP				
	33.90	0.94	35.54	34.12
ES-DAP				

4.3.3. Comparison of evolutionary search and PPO

In this section, we turn our attention to demonstrate the case where we utilize the policy gradient method instead of stochastic search. We utilize PPO. It alternates between sampling data through interaction with the environment and optimizing a surrogate objective function using stochastic gradient ascent [30]. We consider S4-CS1 with uncertainty level $\sigma = 10\%$. Table 13 presents the results for the ES-HP and PPO-HP as averages across five runs. ES-HP performs significantly better in terms of μ_z, σ_z, and $\bar{\mu}_\beta$ in comparison to PPO-HP. This can be attributed to the use of noisy directions for policy improvement, whose evaluation is known to be expensive and sometimes unreliable in policy gradient approaches. The

hyperparameters of PPO are given in Appendix A.

Table 13
Comparison of ES-HP and PPO-HP.

Method	μ_z (kGBP)	σ_z (kGBP)	$\bar{\mu}_\beta$ (kGBP)	
			10%	25%
ES-HP	30.25	0.94	33.36	32.23
PPO-HP	46.97	0.69	47.54	47.30

4.3.4. Impact of different hyperparameter settings of population size of candidate policy functions

In this section, we investigate the effect of varying the population size of candidate policy functions on the performance of our proposed method. All other parameters of SNES were kept at literature-reported values. After the policy network hyperparameters were optimized iteratively, here we focus specifically on how changes in population size influences the performance. We consider scenario S1-CS1 and report the results in terms of electricity cost (kGBP).

As observed, population sizes of 1000, 1500, and 1800 produce the lowest electricity costs, indicating that moderate to large population sizes are most effective. Very small population sizes (e.g., 300 or 500) lead to higher costs, likely due to reduced diversity in candidate policies. These results suggest that the method is robust to a range of reasonably large population sizes, while very small populations may degrade performance.

Table 14
Impact of population size.

Population size	Electricity cost (kGBP)
300	31.90
500	31.56
1000	29.14
1500	29.14
1800	29.14

5. Conclusions

In this work, we developed a DRL-based control framework for DR in an EAF-based steel plant. We presented the method that employs a heuristic decision policy combined with an ES-HP to address the production scheduling of a steel plant, which includes typical steelmaking PCs. The applicability and efficiency of the method were demonstrated with different case studies under deterministic and uncertain environments while benchmarking against a discrete-time MILP formulation. The results demonstrate that the method is able to handle constraints and exogenous uncertainties effectively. Even while increasing the problem size, the method can efficiently handle the complexity and does not compromise the optimality gap. Also, when using greater variance in processing time data, HP retains similar performance. We then presented the ablation study wherein we observed that the attention-based state formulation has an advantage over using an MLP. Further, we compared the performance of ES-HP with DAP. Although ES-DAP tends to have a better performance in terms of average, ES-HP identifies optimal policy in fewer function evaluations than ES-DAP. Also, under misspecification of uncertainty scenarios, ES-HP tends to show more robust behavior than ES-DAP. Lastly, we compared ES with the policy gradient method, validating the accuracy of ES over gradient-based optimization. The proposed method can, in principle, be applied to blast furnace-based plants or any multiproduct, multitask, multistage batch production environment, provided the underlying operations can be modelled as batch processes. Also, the proposed framework is sufficiently general to incorporate bidirectional power exchange with the bulk grid; hence, scenarios involving electricity export and associated revenue streams under various market participation or contractual arrangements can also be optimized. We attempted to assess the generalizability of the proposed method by increasing the problem size (i.e., the number of production orders) and varying processing time data. While performance generally declines as the problem size grows, numerous factors can influence each specific instance, making it largely a case-by-case situation. Consequently, we cannot make definitive general statements about performance. Intuitively, however, larger and more uncertain problems tend to be more challenging. In the future work, we shall also consider steel plants supporting power systems by providing some ancillary services such as operating reserves.

Acknowledgments

This work was supported by the European Union's Horizon 2020 Research and Innovation Programme (101096946, FlexCHESS) and United Kingdom Energy Research Centre (UKERC).

References:

[1] Iron and steel technology roadmap. Report. Paris: International Energy Agency; 2020.
[2] Devlin A, Markkanen S. Steel sector deep dive: how could demand drive low carbon innovation in the innovation in the steel industry. Report. Cambridge: Cambridge Institute for Sustainability Leadership (CISL); 2023.
[3] Benefits of demand response in electricity markets and recommendations for achieving them. Report. Washington, DC: US Department of Energy; 2006.
[4] Nagy Z, Henze G, Dey S, Arroyo J, Helsen L, Zhang X, et al. Ten questions concerning reinforcement learning for building energy management. Build Environ 2023;241:110435.
[5] Mariano-Hernández D, Hernández-Callejo L, Zorita-Lamadrid A, Duque-Pérez O, Santos García F. A review of strategies for building energy management system: model predictive control, demand side management, optimization, and fault detect & diagnosis. J Build Eng 2021;33:101692.
[6] Xie J, Ajagekar A, You F. Multi-agent attention-based deep reinforcement learning for demand response in grid-responsive buildings. Appl Energy 2023;342:121162.
[7] Rodriguez-Garcia J, Ribo-Perez D, Alvarez-Bel C, Penalvo-Lopez E. Maximizing the profit for industrial customers of providing operation services in electric power systems via a parallel particle swarm optimization algorithm. IEEE Access 2020;8:24721–33.
[8] Harjunkoski I, Maravelias CT, Bongers P, Castro PM, Engell S, Grossmann IE, et al. Scope for industrial applications of production scheduling models and solution methods. Comput Chem Eng 2014;62:161–93.
[9] Mowbray M, Zhang D, Chanona EADR. Distributional reinforcement learning for scheduling of chemical production processes. 2022. arXiv:2203.00636.
[10] Castro PM, Harjunkoski I, Grossmann IE. New continuous-time scheduling formulation for continuous plants under variable electricity cost. Ind Eng Chem Res 2009;48(14):6701–14.
[11] Castro PM, Dalle Ave G, Engell S, Grossmann IE, Harjunkoski I. Industrial demand side management of a steel plant considering alternative power modes and electrode replacement. Ind Eng Chem Res 2020;59(30):13642–56.
[12] Zhang X, Hug G, Kolter Z, Harjunkoski I. Industrial demand response by steel plants with spinning reserve provision. In: Proceedings of 2015 North American Power Symposium (NAPS); 2015 Oct 4–6; Charlotte, NC, USA. IEEE; 2015. p. 1–6.
[13] Ashok S. Peak-load management in steel plants. Appl Energy 2006;83(5):413–24.
[14] Zhang X, Hug G, Harjunkoski I. Cost-effective scheduling of steel plants with flexible EAFs. IEEE Trans Smart Grid 2017;8(1):239–49.
[15] Nolde K, Morari M. Electrical load tracking scheduling of a steel plant. Comput Chem Eng 2010;34(11):1899–903.
[16] Vázquez-Canteli JR, Nagy Z. Reinforcement learning for demand response: a review of algorithms and modeling techniques. Appl Energy 2019;235:1072–89.
[17] Oh S, Kong J, Yang Y, Jung J, Lee CH. A multi-use framework of energy storage systems using reinforcement learning for both price-based and incentive-based demand response programs. Int J Electr Power Energy Syst 2023;144:108519.
[18] Margi S, Zhou Y, Wu J, Mowbray M. A review of reinforcement learning based approaches for industrial demand response. In: Proceedings of 15th International Conference on Applied Energy (ICAE2023); 2023 Dec 3–7; Doha, Qatar. Energy Proceedings; 2024.
[19] Lu R, Bai R, Luo Z, Jiang J, Sun M, Zhang HT. Deep reinforcement learning-based demand response for smart facilities energy management. IEEE Trans Ind Electron 2022;69(8):8554–65.
[20] Lu R, Li YC, Li Y, Jiang J, Ding Y. Multi-agent deep reinforcement learning based demand response for discrete manufacturing systems energy management. Appl Energy 2020;276:115473.
[21] Pan R, Wang Q, Cao J, Zhou C. Deep reinforcement learning for solving steelmaking-continuous casting scheduling problems under time-of-use tariffs. Int J Prod Res 2024;62(1–2):404–20.
[22] Huang X, Hong SH, Yu M, Ding Y, Jiang J. Demand response management for industrial facilities: a deep reinforcement learning approach. IEEE Access 2019;7:82194–205.
[23] Avadiappan V, Maravelias CT. State estimation in online batch production scheduling: concepts, definitions, algorithms and optimization models. Comput Chem Eng 2021;146:107209.
[24] Harjunkoski I, Grossmann IE. A decomposition approach for the scheduling of a steel plant production. Comput Chem Eng 2001;25(11–12):1647–60.
[25] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. 2017. arXiv:1706.03762.
[26] Castro PM, Sun L, Harjunkoski I. Resource–task network formulations for industrial demand side management of a steel plant. Ind Eng Chem Res 2013;52(36):13046–58.
[27] elia.be [Internet]. Belgian's electricity system operator; [cited 2024 Jun 18]. Available from: https://www.elia.be/en/.
[28] nordpoolgroup.com [Internet]. Market data; [cited 2024 Jun 18]. Available from: https://www.nordpoolgroup.com/.
[29] Huang S, Ontañón S. A closer look at invalid action masking in policy gradient algorithms. 2020. arXiv:2006.14171.
[30] Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal policy optimization algorithms. 2017. arXiv:1707.06347.
[31] Wierstra D, Schaul T, Glasmachers T, Sun Y, Peters J, Schmidhuber J. Natural evolution strategies. J Machine Learning Res 2014;15:940–80.
[32] Schaul T, Glasmachers T, Schmidhuber J. High dimensions and heavy tails for natural evolution strategies. In: Proceedings of Genetic and Evolutionary Computation Conference; 2011 Jul 12–16; Dublin, Ireland. New York: Association for Computing Machinery; 2011. p. 845–52.

Declaration of Interest Statement

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The author is an Editorial Board Member/Editor-in-Chief/Associate Editor/Guest Editor for this journal and was not involved in the editorial review or the decision to publish this article.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: