Panzer, Marcel; Gronau, Norbert

Article — Published Version

Designing an adaptive and deep learning based control framework for modular production systems

Journal of Intelligent Manufacturing

Suggested Citation: Panzer, Marcel; Gronau, Norbert (2023) : Designing an adaptive and deep learning based control framework for modular production systems, Journal of Intelligent Manufacturing, ISSN 1572-8145, Springer US, New York, Vol. 35, Iss. 8, pp. 4113-4136, https://doi.org/10.1007/s10845-023-02249-3

Journal of Intelligent Manufacturing (2024) 35:4113–4136
https://doi.org/10.1007/s10845-023-02249-3

Designing an adaptive and deep learning based control framework for modular production systems

Marcel Panzer[1] · **Norbert Gronau**[1]

Received: 12 May 2023 / Accepted: 12 October 2023 / Published online: 20 November 2023

Abstract
In today's rapidly changing production landscape with increasingly complex manufacturing processes and shortening product life cycles, a company's competitiveness depends on its ability to design flexible and resilient production processes. On the shop-floor, in particular, the production control plays a crucial role in coping with disruptions and maintaining system stability and resilience. To address challenges arising from volatile sales markets or other factors, deep learning algorithms have been increasingly applied in production to facilitate fast-paced operations. In particular deep reinforcement learning frequently surpassed conventional and intelligent approaches in terms of performance and computational efficiency and revealed high levels of control adaptability. However, existing approaches were often limited in scope and scenario-specific, which hinders a seamless transition to other control optimization problems. In this paper, we propose a flexible framework that integrates a deep learning based hyper-heuristic into modular production to optimize pre-defined performance indicators. The framework deploys a module recognition and agent experience sharing, enabling a fast initiation of multi-level production systems as well as resilient control strategies. To minimize computational and re-training efforts, a stack of trained policies is utilized to facilitate an efficient reuse of previously trained agents. Benchmark results reveal that our approach outperforms conventional rules in terms of multi-objective optimization. The simulation framework further encourages research in deep-learning-based control approaches to leverage explainability.

Keywords Modular production · Production control · Deep learning · Reinforcement Learning · Simulation framework · Explainability

Introduction

Nowadays, companies must respond quickly to both, internal and external disruptions and adapt their processes to remain competitive and maintain operational profitability. In this context, the trend towards mass customization and shortening development cycles pose significant challenges for today's production systems. By the same measure, they must be capable of operating in highly uncertain market conditions while satisfying many (conflicting) customer- and process-related objectives, in the shortest possible time (Schmidt & Nyhuis 2021). In this regard, the use of advanced Industry 4.0 technologies, including the *Internet of Things* and artificial intelligence, is crucial to enable a data-driven process optimization and to cope with the increasingly complex requirements (Kang et al. 2020; Parente et al. 2020; Kapoor et al. 2021).

In recent years, simulation-based and combined hardware-in-the-loop approaches were implemented to facilitate a seamless transfer of research artifacts into practice in a low-risk environment. Especially in production planning and control, single- and multi-agent approaches were implemented to manage production complexity, each with different pre-defined agent-environment interactions (Babiceanu & Chen 2006; Gronauer & Diepold 2021). Regarding the production organization, modular systems demonstrated particular benefits, as they allocate the overall optimization task to accessible and reactive groups of agents (Sallez et al. 2010; Groover & Jayaprakash 2016). Modular production systems are noted for their flexibility, scalability, and adaptability. Unlike conventional production systems, which are often lin-

✉ Marcel Panzer
marcel.panzer@wi.uni-potsdam.de

[1] University of Potsdam, August-Bebel-Str. 89, 14482 Potsdam, Brandenburg, Germany

ear and inflexible, modules can be easily inserted, removed, or re-positioned, enabling a swift response to market changes or adaptation requirements. Through task decomposition and distributing complexity across foundational modules, we can expedite the implementation of intelligent control methods, as demonstrated in (Rojas & Rauch 2019; Zhou et al. 2022a; Tao et al. 2023). The distributed modules possess predefined process capabilities, thus ensuring a high density of coordination within and between modules which increases responsiveness and robustness of the system through parallel processing and intelligent agent orchestration (Groover & Jayaprakash 2016; Herrera et al. 2020; Sallez et al. 2010; Buckhorst et al. 2022). However, the multitude of interactions and parallel operational activities in multi-agent systems still pose a significant challenge for a coordinated control of shop-floor activities. The production control must handle a constantly growing number of data sources and information flows, to make situation-specific, optimal decisions and leverage process potentials.

To handle such complex optimization problems, machine learning techniques, particularly deep learning algorithms, were increasingly applied in production research (Kang et al. 2020; Samsonov et al. 2021; Oluyisola et al. 2022). Given their ability to capture complex non-linear relationships and to process large amounts of data in real-time for multi-objective optimization, the need for complex and rigid models is prevented. This enables the targeting of both local and global process variables and facilitates a continuous improvement process by leveraging both, machine and human-related system resources (Cadavid et al. 2019; Zhang et al. 2019; Kang et al. 2020). However, despite the potential benefits, the exploitation of machine learning in production control is not yet fully addressed, as its adoption is rather concentrated on the field of Big Data or other related disciplines (Liao et al. 2017; Cadavid et al. 2019). Nevertheless, it becomes clear, notably in Weichert et al. (2019), Zhou et al. (2022b), that due to the versatility of deep learning approaches, a multitude of practical control optimization problems can be addressed, in which fast decision-making contributes significantly to maintaining process stability (Bueno et al. 2020; Zhang & Huang 1995; Garetti & Taisch 1999). However, the practical integration of a machine learning algorithm must be conducted in an objective-specific manner and requires a dedicated deployment to balance the increasing process and model complexities and to ensure appropriate decisions and a high process reliability (Weichert et al. 2019).

In recent years, in particular, deep reinforcement learning (RL) algorithms demonstrated superior efficacy against other conventional or machine learning based benchmarks (Zhou et al. 2022b). In contrast to meta-heuristics, which serve as search process optimizers, deep RL offers significantly improved real-time capabilities, performance metrics, and higher interpretability (Zhang et al. 2022; Grumbach et al. 2022; Kallestad et al. 2023). Based on collected sensor information, deep RL is capable of making online data-driven decisions and enables a responsive and adaptive control design that addresses the challenges of volatile manufacturing environments. Due to the direct agent-environment interaction, deep RL can generalize and leverage the obtained process knowledge to enhance production stability and performance (Arunraj & Ahrens 2015; Mehlig 2021). Even though the application of deep RL demonstrated outstanding performances in various production fields, multi-agent based production control approaches were less considered, especially in matrix- or modular-shaped production systems, as reviewed and analyzed in Panzer & Bender (2022) and Panzer et al. (2022). Although control approaches of Gankin et al. (2021), Mayer et al. (2021), and May et al. (2021) already indicated robust and performant multi-agent control policies, current research lacks an adaptive approach that can address various production scenarios and offers a high transferability to similar practical problems.

To harness the benefits of deep learning and a multi-agent-based production organization, this paper introduces a novel control framework that facilitates the flexible adaptation of modular production systems. By employing a hyper-heuristic control concept for varying production objectives, our approach seeks to improve production performance and adaptability. Owing to the hyper-heuristic based algorithmic approach, the deep RL based top-level decision entity focuses on selecting low-level heuristics, thereby avoiding the adoption of deficient system policies or erroneous actions. The proposed control framework is incorporated into a flexible simulation, which accommodates a wide range of production scenarios and enables the optimization of individual performance metrics. The simulation adheres to a modular principle, which decomposes the overall production task complexity into manageable fragments, resembling the production system in its modular structure. Additionally, we distinguish between manufacturing and distribution modules, that are responsible for shop-floor and intra-logistics activities, respectively. By synergistically combining the concepts of modular production and hyper-heuristics, we harness the strengths of both domains. This fusion allows us to achieve a dual-fold reduction, both systemically and algorithmically, in optimization complexity.

The embedded deep learning based decision-making process leverages a module recognition and agent experience-sharing method that facilitates the rapid creation and initiation of multi-level production systems. The framework further aspires to progressively reduce computational efforts for neural network training through the integration of a batch of pre-trained policies.

The remainder of the paper is organized as follows. In the next section, the basics of prevailing simulation frame-

works, deep RL, and multi-agent based production control are outlined and the research objective is specified. Then, the conceptual design and artifact requirements are defined and simulation results are presented and evaluated. Finally, the paper concludes with a discussion of the framework and a conclusion that synthesizes the main findings.

Related work

This section specifically focuses on the basics of discrete-event simulations (DES) and deep learning methods, which have been increasingly applied for a wide range of production planning and control tasks over recent years. A DES constitutes an essential link between the theoretical concepts of adaptive and deep learning based production control concepts and their simulated and practical implementation in modular production systems. Without a robust foundation in DES, a framework would lack to emulate the dynamics and complexity of modular production systems. Therefore, the following DES subsection provides an in-depth analysis of the simulation foundations that are essential for operationalizing our approach.

Subsequently, the key concepts of deep RL as well as hyper-heuristics are introduced, which serve as core elements of the later developed artifact. These concepts are expected to provide significant performance improvements in production optimization through their continuous learning behavior and adaptability, enabling automated and data-driven optimization of production decisions. The discussion continues with a review of the current state of research, specifically in the context of integrating production control and deep RL.

Building upon the dual research gap from DES and algorithmic perspectives, we present the problem formulation in which we state the specific problem of our research approach. Thereby, we outline the objectives of our approach for an adaptive and deep learning based modular production control framework.

Discrete-event based production simulation

A DES describes the development of a system based on pre-defined events and their chronological sequencing as discrete occurrences that affect the system state Law (2007). In DES, events are captured at discrete time points, and system variables are modified accordingly, allowing for incremental and traceable progression of the simulated system over time. By incorporating operational resources, such as machines or labor resources, system states, and process flows, a production system can be replicated, enabling the analysis of key performance metrics and identification of operational optimization potentials (Fowler et al. 2015; Jeon & Kim 2016; Mayer et al. 2021). Such analysis may include bottleneck resource evaluation, optimal machine arrangement, or work efficiency assessment for specific system resources. Notably, this approach facilitates the uncritical testing of prototype solutions, which can be further examined in an intermediate hardware-in-the-loop approach until reaching satisfying real-world results.

However, simulation techniques are often applicable only for limited periods due to their difficulty and specificity of implementation (Neto et al. 2020). Thereby, (Mourtzis 2020) further emphasizes the challenges of integrating artificial intelligence into these simulations. Although the DES approaches can be manifold, the number and type of information sources necessitate dedicated control implementation for a data-driven and optimal decision-making. To address these hurdles, the following simulation frameworks aim to bridge the gap between advanced planning and control theory and its practical application. These frameworks are also listed in Table 1.

Apart from production planning and control practices or similar production disciplines, other simulation approaches already delved into the creation of intelligent planning and control frameworks. Notable sectors and problems include vehicle routing (Nazari et al. 2018), power grid operation (Rocchetta et al. 2019), energy supply chain management (Chen et al. 2021), or computational fluid dynamics (Pawar & Maulik 2021).

In the realm of production planning and control, current research is primarily focused on simulation frameworks designed for planning purposes. A mixed-integer linear programming (MILP) framework for the scheduling of mining operations was proposed by Manriquez et al. (2020). An intelligent multi-agent *SwarmFabSim* framework was proposed by Umlauft et al. (2022), that deploys a swarm intelligence algorithm. Other DES scheduling approaches adopted quantum annealing (Venturelli et al. 2015), cuckoo search optimization (Phanden et al. 2019), or genetic algorithms (Fumagalli et al. 2018) to increase the applicability of the respective framework. A recurring feature of such approaches is potentially extended computation times, often attributed to meta-heuristic solution methods. Other established frameworks that use deep RL for production scheduling, like the *JSSEnv* (Tassel et al. 2021) or *Schlably* framework (Waubert De Puiseau et al. 2023), primarily focus on job-shop scheduling or order release and sequencing (Samsonov et al. 2022).

For dedicated production control problems, two approaches dealt with specific single-agent DES implementations, which analyze the impact of stochastic and unpredictable variables. Zhang et al. (2020) implemented the single-agent *L2D* framework by deploying a combined deep RL and disjunctive graph representation to learn priority dispatching rules in a 3x3 job-shop. Kuhnle et al. (2019a), Kuhnle et al. (2019b) implemented the deep learning based production control

Table 1 Overview of prevailing simulation frameworks

Application	Specific application	Algorithm	Author
Other applications	Vehicle routing	RL	Nazari et al. (2018)
	Modular System design	–	Farsi et al. (2019)
	Energy supply chain	Genetic algorithm	Chen et al. (2021)
	Computational fluid dynamics	Deep RL	Pawar & Maulik (2021)
	Algorithmic trading	Deep RL	Shavandi & Khedmati (2022)
	Predictive maintenance	Deep RL	Rodríguez et al. (2022)
	Predictive maintenance	Deep RL	Su et al. (2022)
Job-shop scheduling (non-RL-based)	General scheduling	Cuckoo search	Phanden et al. (2019)
	Mining scheduling	MILP	Manriquez et al. (2020)
	General scheduling	Quantum annealing	Venturelli et al. (2015)
	General scheduling	Genetic algorithms	Fumagalli et al. (2018)
	Semiconductor scheduling	Swarm intelligence	Umlauft et al. (2022)
Job-shop scheduling (RL-based)	General *JSSEnv* framwork	Deep RL	Tassel et al. (2021)
	General scheduling	Deep RL	Samsonov et al. (2022)
	General *Schlably* framework	Deep RL	Waubert De Puiseau et al. (2023)
	Multi-agent job-shop	Deep RL	Liu et al. (2022)
Job-shop control (single-agent)	*SimPyRLFab* semi-conductor dispatching	Deep RL	Kuhnle et al. (2019b)
	General *L2D* framwork	Deep RL	Zhang et al. (2020)
Job-shop control (multi-agent)	Modular dispatching	Deep RL	Our Framework

framework *SimPyRLFab*, thereby considering the prevailing semiconductor process pre-requisites. In such DES frameworks, predefined and product-specific process sequences, machine failures, or other non-deterministic events can be triggered, and their effects on production participants, such as degrading production resources, warehouse inventories, or line effects, can be investigated (Law 2007). Additionally, systemic and agent-centered relationships can be analyzed based on their organization and interaction.

Whereas Liu et al. (2022) implemented an advanced hierarchical multi-agent scheduling framework, distinguishing tasks between routing and inter-machine scheduling, other (DES) frameworks predominantly focused on plain scheduling or single-agent production control. Notably, these simulations did not account for multiple production layers and consistently operated on a singular level.

Conclusively, from the DES viewpoint, there is a need for a framework that facilitates the conceptualization and simulation of a multi-layered modular production system. Within this framework, the optimization task is governed by a semi-heterarchical framework, facilitating the attainment of both global and local objectives by multiple agents. The modular structure allows for a versatile modification and change of system properties by adding or removing modules to meet current requirements or to cope with dynamic processes (Buckhorst et al. 2022). The semi-heterarchical backbone provides a high integration capability of potential scenarios through user-defined modules within a hierarchical organization. In parallel, the system is more robust due to the structured allocation of competencies and parallel processing, as in heterarchical systems (Valckenaers et al. 1994; Groover & Jayaprakash 2016; Derigent et al. 2021).

A significant challenge is that current heuristics have limitations in optimizing multiple performance measures (Grabot & Geneste 1994) and often exhibit minimal global coordination when processing information within local entities (Uzsoy et al. 1993; Holthaus & Rajendran 1997). Yet, in operational production settings that demand rapid, potentially real-time, decision-making, approaches like meta-heuristics often underperform when compared to traditional heuristics. These methods, due to mathematical optimization techniques, may not provide real-time decisions, especially as the problem's scope expands, leading to substantial performance declines (Nasiri et al. 2017). Moreover, meta-heuristics necessitate profound expertise and pose challenges during initialization and modification (Rauf et al. 2020; Zhou et al. 2020). Given these constraints, RL methods, known for swift and interactive decision-making, have gained traction in operations and control tasks (Samsonov et al. 2021; Bahrpeyma & Reichelt 2022; Panzer et al. 2022).

Basics of (deep) reinforcement learning and hyper-heuristics

RL constitutes an interactive paradigm of machine learning, wherein a decision-making agent selects actions for execution and thereby iteratively refines its policy to develop the process logic. The leap to the widespread adoption of RL was primarily reached through its successful implementation in the Atari environment, making it attractive for complex optimization problems (Mnih et al. 2013). In particular, deep RL, with the additional integration of a deep neural network that allows it to process large state variables, was adapted to a variety of data-centric online applications. A fundamental constraint for integration is the requirement for the optimization task or problem to adhere to the Markov property and for the decision or control process to align with a Markov Decision Process (MDP). This is accompanied by the Markov assumption, which states that all future production states depend only on the current state, but do not imply any influences from the past which reflects the basic assumption of our later DES approach (Sutton & Barto 2017). The simulation employs a model-free, off-policy Q-learning algorithm, as implemented by other successful benchmarks in production control (such as Esteso et al. 2022, ?; Panzer & Bender 2022). Q-learning does not require a model of the environment and estimates the value of a Q-function (Equation (1)), which assesses a potential action of an agent based on the Bellman equation and total accumulated expected rewards G_t.

$$Q(s_t, a_t) = r(s, a) + \gamma max(Q(s, a)) \tag{1}$$

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \tag{2}$$

In this context, s represents the current state, a is the selected action, and $r(s, a)$ summarizes the obtained reward after executing action a in state s. γ defines the discount factor ($\gamma \in [0, 1]$) that determines the relative weighting of future rewards with respect to the current reward across steps. s' is the subsequent state following the execution of action a, with $max(Q(s, a))$ being the maximal Q-value across all feasible actions a' in the subsequent state s'. The primary difference between conventional deep Q-learning and its deep learning based counterpart is the latter's use of a neural network to approximate the Q-function. Therefore, the objective is to minimize the loss between the estimated Q-function and the target value. The loss can be defined as the mean squared error $L(\theta) = E[(Q(s, a; \theta) - y)^2]$ between the estimated value $Q(s, a; \theta)$ and the target value $y = r(s, a) + \gamma; max(Q(s, a; \theta))$. Minimizing this loss allows for the updating of neural network parameters θ to better approximate the Q function. This procedure is reiter-

ated until the performance converges against a defined level or a certain number of training steps is reached. Using these, the formula for the DQN can be derived to approximate the Q-values by minimizing the loss function using the Bellman equation as summarized in Formula (3). To further stabilize learning and increase performance, a target network with weights θ^- is introduced to compute $Q(s', a')$ for the next states (Mnih et al. 2013, 2015; Sutton & Barto 2017).

$$\begin{aligned} Q(s_t, a_t, \theta) \leftarrow\ & Q(s_t, a_t, \theta) \\ & + \alpha\ [r\ +\ \gamma\ max\ Q(s', a', \theta^-) \\ & -\ Q(s_t, a_t, \theta)] \end{aligned} \tag{3}$$

Deep RL based production dispatching

The limited capabilities of existing models in coping with dynamic system behavior have led to the application of various deep learning based control approaches to increase the reproduction accuracy and to minimize manual intervention, i.e. by a control strategy approximation in Bergmann & Stelzer (2011) or Bergmann et al. (2014). Luo (2020) relied on a double DQN RL to minimize total delays and avoided otherwise assumed static conditions. Similarly, Mouelhi-Chibani & Pierreval (2010) and Zhao & Zhang (2021) outperformed conventional approaches with neural network based rule selection depending on flow or job-shop parameters. The latter used a convolutional neural network that takes matrices of processing times and two Boolean matrices of pending and completed operations as input to select rules such as SPT and LPT and outperformed a GA in terms of machine utilization and waiting times. In a job-shop environment, using the production state representation as a 2-D matrix and applying transfer learning, the scheduling policy demonstrated strong performance and increased generalizability (Zheng et al. 2020). However, these and other approaches, such as that of Altenmüller et al. (2020) or Kuhnle et al. (2020), were implemented in a single-agent environment.

To facilitate decentralized decision-making, multi-agent approaches are of particular importance for the decomposition and allocation of the total optimization process to multiple agents and to maximize the exploitation of individual skill sets as listed in Table 2. Malus et al. (2020) suggested an order dispatch mechanism based on joint global rewards for autonomous mobile robots to minimize delays. Hammami et al. (2017) proposed a multi-agent system based on simultaneous learning and information sharing between agents to reduce average delays. Dittrich & Fohlmeister (2020) and Hofmann et al. (2020) applied a centralized DQN decision module for training. Waschneck et al. (2018) introduced a training strategy in a wafer fabrication facility to optimize maximum uptime as a global goal. In a recent study by Sakr et al. (2021), a DQN was utilized to minimize queue waiting and lead times in wafer production. Specifically, they compared their approach to a prevailing heuristics strategy and found significant improvements. Gros et al. (2020) minimized costs in a system to control a car buffer after painting operations. Overbeck et al. (2021), on the other hand, leverage a PPO to find the best action in an automated manufacturing system, that was designed according to the chaku-chaku principles.

However, the previous research on deep RL and multi-agent based production control primarily focused on job-shop environments. There are some approaches in matrix and modular based production systems as proposed by Hofmann et al. (2020), that provide agents with immediate rewards for selected actions and delayed rewards based on the total global cycle time. This strategy outperformed a rule-based and non-coordinated strategy by preventing the blocking of other agents and allocating global rewards. The simulated system comprised 10 workstations and several AGVs that executed multiple process steps and were fully interconnected. May et al. (2021) followed an economic bidding approach to reduce execution time and increase utilization efficiency. This involved two system configurations, each with 15 agents and 10 stations arranged in a matrix structure, with different buffer sizes. Based on a PPO, the global utilization rate after part completion and non-value added time as well as consecutive failed bids could be optimized. Gankin et al. (2021) implemented a first large-scale plant consisting of 25 machines arranged in a five-by-five layout, based on the approach of Mayer et al. (2021). In this approach, an action masking mechanism was used to reduce the decision complexity of all 20 DQN based transportation resources that were being trained in parallel. The agents used the same neural network and buffer as the decision instance for experience sharing.

In summary, Table 2 indicates that always one organizational layer was integrated into previous approaches. There is no approach, that incorporates multiple layers and a semi-heterarchical organization within a modular production system. Furthermore, the presented approaches predominantly rely on single dedicated algorithms, such as the DQN. However, there is a need for an approach that leverages the advantages of deep learning techniques with conventional methods, as discussed in (Panzer et al. 2022). Another research gap concerns the predominantly technical optimization objectives, which are rather limited in scope. Customer-centric objectives like the processing of urgent and prioritized orders, which hold particular importance in today's economic landscape, were inadequately addressed.

Table 2 Summary of deep RL based control approaches in multi-agent production systems

Application	Algorithm	Training strategy	Control strategy	Agent interaction	Objective parameter	Orga. levels	Source
Car buffer	DQN	Iterative learning	Decentral	–	Cost/ decision time	1	Gros et al. (2020)
Chaku-chaku line	PPO	Shared PPO module	Central	–	Utilization/throughput	1	Overbeck et al. (2021)
Job-shop	SA	Concurrent learning	Decentral	Agent information exchange	Mean tardiness	1	Hammami et al. (2017)
	DQN	Iterative DQN/ heuristics learning	Decentral	Global rewards	WIP/ uptime utilization	1	Waschneck et al. (2018)
	DQN	Shared DQN module	Central	Agent information exchange	Mean cycle time	1	Dittrich & Fohlmeister (2020)
	DQN	Concurrent learning	Decentral	Agent state information	Utilization, queue waiting/lead times	1	Sakr et al. (2021)
	TD3	Concurrent learning	Decentral	Order bidding mechanism	Tardiness	1	Malus et al. (2020)
Matrix production	DQN	Shared DQN module	–	Agent state information	Throughput time	1	Hofmann et al. (2020)
	DQN	Shared DQN module	Central	–	Throughput	1	Gankin et al. (2021)
	PPO	–	Decentral	Economic bidding	Execution time/utilization eff.	1	Mayer et al. (2021)
Modular production	DQN-based hyper-heuristic	Concurrent learning	Decentral	Agent and cell/ order state exchange	Process- and customer related parameters	> 1	Our framework

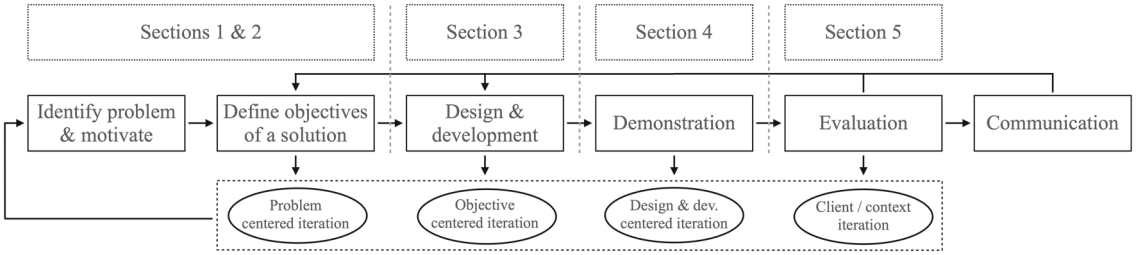


Fig. 1 Pursued *DSRM* methodology from Peffers et al. (2007)

Research highlights and key contributions

In this paper, we propose a customizable simulation framework for modular production systems that deploys multiple dispatching agents to address customer- and process-specific objectives that can be adapted to individual scenarios. As indicated in Table 2, our framework uniquely supports structuring across multiple and arbitrary organizational levels and modules. This allows for the definition and generation of module-specific control policies, depending on the process related requirements and optimization parameters within the differing production modules. The modularity should allow a specific generation of local process and control knowledge that still keeps track of multiple local and global objectives.

To control the agents, we utilize a deep learning based hyper-heuristic, that combines deep learning with heuristics, which enables a rapid scenario generation and increases key production indicators in terms of performance, resilience, and adaptability. For the deep learning based decision-making, the top-level heuristic does not have to learn intrinsic constraints but can focus on the optimization task. As evidenced in numerous studies, e.g., Liu & Dong (1996), Kashfi & Javadi (2015), Heger et al. (2015), Shiue et al. (2018), and Zhang & Roy (2019), the situation-dependent selection of dispatching rules can significantly reduce computation costs and provide an efficient tool for process optimization (Grumbach et al. 2022). The deep learning framework is the first to facilitate an automated initialization of neural networks for the distributed agents within the modular entities. To further increase learning performance and operational efficiency, we deploy a module recognition and transfer learning strategy.

Simulation design

To ensure a systematic approach for reaching the defined research objectives, we adhered to the design science research methodology as proposed by Peffers et al. (2007, referring to Fig. 1). As the problem identification and objective definition were dealt with in the previous section, we proceed with constructing the research artifact. To accommodate dynamic requirements and provide an adaptable simulation approach, it is essential to select a suitable and scalable simulation foundation. This should seamlessly incorporate the hyper-heuristic control approach, and facilitate a decentralized and parallel decision-making.

Simulation framework design

The deep learning based DES is built on a python-based simulation, developed at our chair and the *Centre for Industry 4.0*. The framework enables the rapid creation of a modular production layout with corresponding system organizations and control regulations. Incorporating the production within the DES enables the emulation of dynamic and stochastic process parameters, delineated in Table 3. Each parameter can be retrieved in its respective unit (pertaining to distance or time) or a discrete/categorical value, such as 0 or 1, i.e. indicative for the processed order type A/ B, respectively. Subsequently, these values undergo processing to be confined within predefined ranges, mitigating potential outliers detrimental to neural network efficacy. The simulation is based on the wide-spread SimPy simulation library, which is frequently applied in the field of DES in production control research, as it was demonstrated in previous studies (Kuhnle et al. 2020, 2021; Sakr et al. 2021).

For the control of the agents and the exploitation of deep learning mechanisms as well as conventional approaches, a hyper-heuristic is applied. The term hyper-heuristic was first defined by Cowling et al. (2001), and initially implemented using a machine learning algorithm to find an optimal order of a sales summit problem. In contrast to meta-heuristics, hyper-heuristics utilize a predetermined set of low-level heuristics, rather than searching through problem solution spaces, as illustrated in Fig. 2. It tries to find an optimal operational sequence of the low-level heuristics that optimally solves an optimization task within the given solution space. In recent research, especially machine and more specifically, deep learning algorithms were proposed to flexibly adapt to optimization tasks as top-level heuristics and exploit the capabilities of underlying heuristics in a case-specific manner. This allows for the automation of the design process and the utilization of the knowledge of an on- or offline machine learning algorithm as an optimizer to derive near-optimal

Table 3 Available state parameters within the simulation framework

Entity	Attribute	Value	Attribute description
Order	Time related	[min]	Order start time, due date, time in cell
	Type related	[0,1]	Order type, complexity, priority
	Process chain	[0,1,…]	Count of remaining tasks, next/ finished tasks, position
	Process related	[0,1]	Processing, locked, picked up, in input/ same cell
	Position related	[m]	Order distance n
Buffer/ storage	Type related	[n]	Count of input / output buffer slots
	Process related	[n]	Count of free slots
Machine	Tool related	[0,1,…]	Machine type, current tool setup (tool change costs time), next setup
	Process related	[min]	Remaining setup time, currently manufacturing, remaining mfg. time
	Failure related	[0,1]	Current failure, failure fixed in
Agent	Position related	[0,1,…]	Current position, next position
	Process related	[0,1]	In movement, remaining moving time, has task, locked item

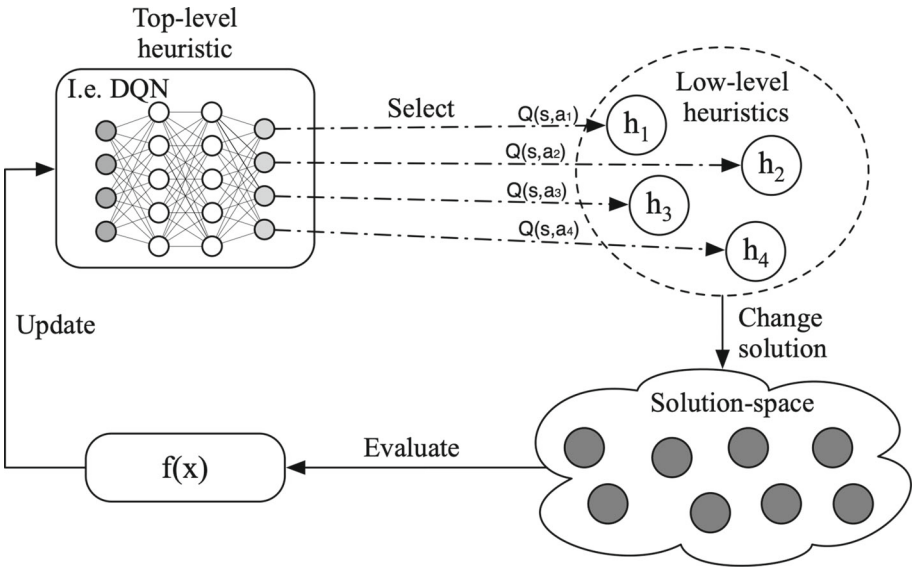


Fig. 2 Hyper-heuristics and DQN based optimization, inspired by Cowling et al. (2001); Swiercz (2017)

scheduling and dispatching policies based on the established and comprehensible low-level heuristics (Burke et al. 2010, 2019; Drake et al. 2020).

The simulation framework integrates the hyper-heuristic control concept through a multi-agent organization, where the deep learning driven agents communicate with the simulation through an order and state information exchange. Unlike a conventional single-agent approach, as illustrated on the left in Fig. 3, the agents have individual state vectors and can execute independent actions that dynamically affect the processes. Within the modules, the agents can receive information about the other agents, such as their position or order status. To prevent local optimization tendencies, global objective variables, such as the global start time, can be received through the order parameters. Furthermore, additional variables can be derived from the available data set from Table 3, e.g., allowing the tracking of WIP (work in progress) levels for the individual cells, based on storage, machines, and buffer occupancies. The WIP level can in turn be utilized for operational decision-making, particularly at the upper distribution levels.

Simulation components

The given components of the base simulation are displayed in Fig. 4, which allow for a wide range of potential optimization tasks. In this context, the individual module elements and module relationships can be flexibly defined. The number of elements within the cells and intermediate buffers can be adjusted according to the scenario specifications. Regarding the organization, the distinction between distribution and manufacturing cells is crucial, as they entail divergent intrinsic process optimization, especially in the design of the subsequent reward functions of the deep learning agents.

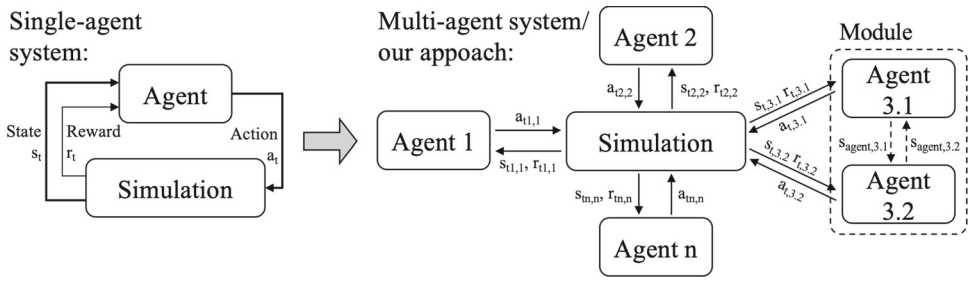


Fig. 3 Distributed decision-making and parallel processing of the agents, left side from Sutton & Barto (2017)

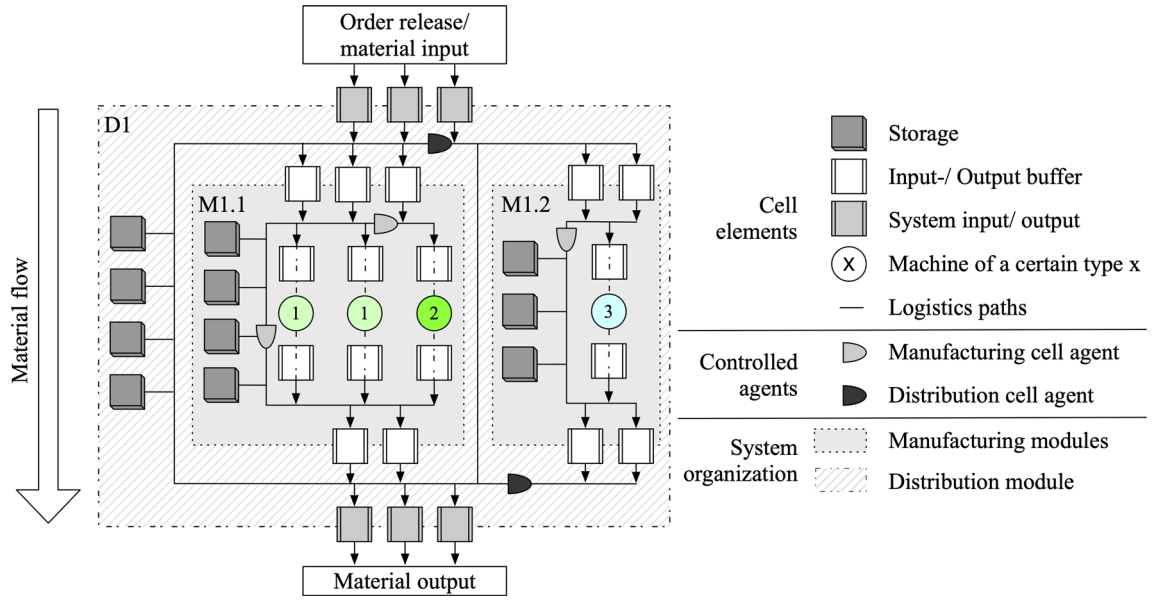


Fig. 4 Descriptive manufacturing and distribution modules within the simulation

To clearly delineate the simulation boundaries, we established foundational criteria that our approach must adhere to. To ensure a resilient modular design, each component functions as a unique entity, equipped with the autonomy to determine its operations. We also adopt fully observable states that are fed to the agents, thus facilitating decision-making via deep learning. This underpins the ongoing learning and continuous improvement process and allows for a plug-and-play simulation design. During a run, we assume that system parameters such as module layout and resource numbers remain unchanged throughout the simulated system.

However, our model does not presuppose a static system behavior. Instead, the system considers external factors such as fluctuations in order volumes, variations in system parameters due to machine malfunctions, and alterations in maintenance times. This adaptability is reflected in its capability to manage a wide array of stochastic parameters across both process-oriented and product-centric operations.

This also supports to maintain a realistic simulation design. The intention is to align the behavior of the simulated agents with real-world conditions and minimize the transfer gap. The consideration of stochastic parameters and the comprehensive set of system states, in coherence with the flexible module and deep learning based agents, contributes to a sophisticated production control simulation framework. This primarily supports the analysis of the different system parameters and can help to eliminate bottlenecks. Furthermore, it evaluates the system's resilience and its ability to respond to machine failures or other unexpected occurrences.

Hyper-heuristics based control framework

The individual modules discussed in the previous section are operated by distinct dispatching agents that make decisions based on currently received system information. For this purpose, a set of ten heuristics is provided by the base simulation that take specific parameters into account for decision-making and perform rather static and straightforward operations. Based on the simulation parameters in Table 3, other (combined) rules can be quickly designed and implemented. Nonetheless, in dynamic environments, the operational model must adapt flexibly to external conditions

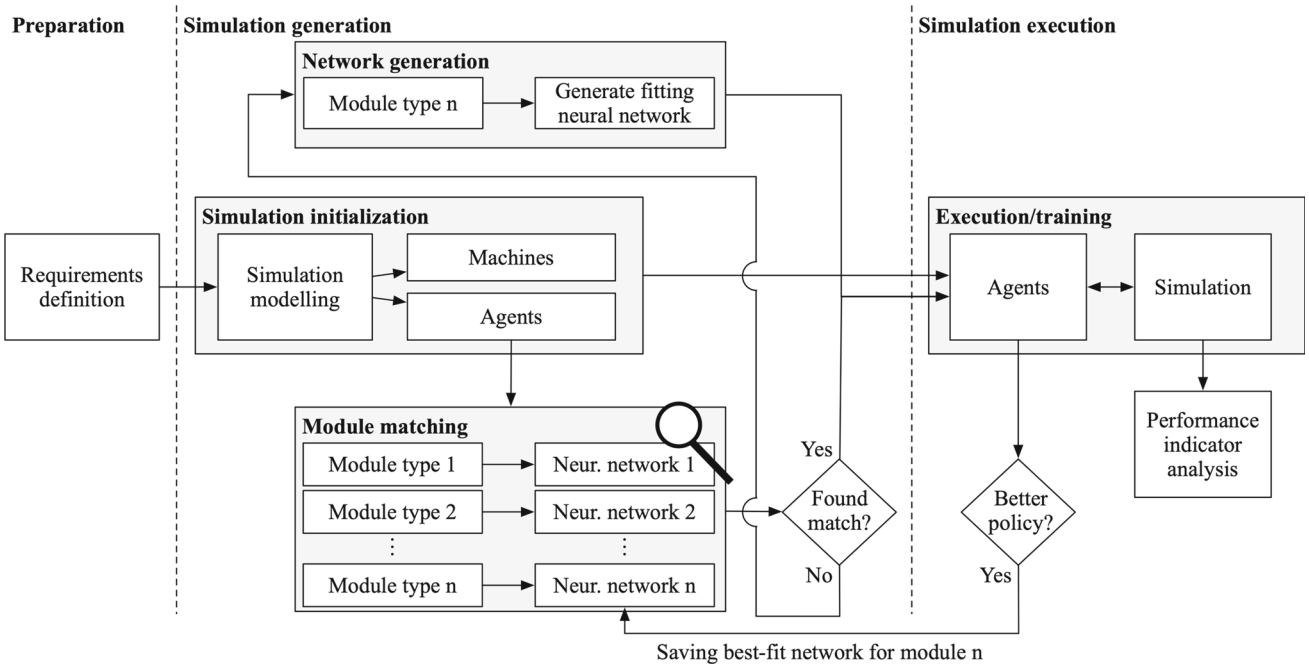


Fig. 5 Simulation framework with flexible module recognition

to enable a resilient and indicator-oriented decision-making and optimization.

Prior to the simulation and optimization process, the distributed and intelligent agents have to be initiated. Based on the defined requirements (as shown in Fig. 5, on the left), a simulation is designed that meets the scenario specifications. Using the specified manufacturing and distribution modules, an appropriate neural network is sought for an agent, using a standardized unique network identifier, which aligns with the cell's structural conditions. These identifiers' properties include the cell type (distribution, manufacturing), (intermediate) buffer and storage capacities, and the number of machine resources. When no neural network fulfills the requirements, a new one is generated and tailored to the module and its associated state vector, to match the desired properties.

For initializing the simulation and for the use of the already trained neural networks, different operating modes are available. On the one hand, during module detection, if a suitable neural network is identified, it can be embedded into the respective agent and serve as the basis for generating an optimized control policy (see case 1, Fig. 6). In contrast, the system can be completely re-trained to avoid bias. However, this increases the computational load and is only necessitated when establishing a new simulation scenario (case 2). Alternatively, a purely operational application of the neural network is possible, in which the network is not trained and adapted (case 3). This case offers a distinct advantage in rapidly identifying suitable actions, which is particularly important in real-world applications.

All trained networks are stored within a neural network stack, and, during training, the best-performing networks are compared to determine the optimal control policy for each case (see Fig. 5, bottom). In addition, after each training, commencing from a pre-determined minimum number of training steps to avoid initial instability, the moving reward average is calculated and compared to the previous best performance. This method aims to facilitate continuous tracking of training progress without encouraging an overestimation of performance due to statistical anomalies.

An additional approach for transferring pre-existing knowledge involves freezing and transferring hidden layers, that substitute the initial weights of newly generated networks during initialization. In this case, the weights of a comparable manufacturing or distribution network are taken to provide the policy to a network in a new layout with collective knowledge that was gained in past simulations. This systematic storage and retrieval mechanism, enabled by the standardized neural network stack in the center of Fig. 6, ensures that we leverage experiences from the past, effectively optimizing and scaling the simulation process.

Hyper-heuristics control mechanism

In prevailing approaches, varying layouts or problem scenarios, resulting in structural changes, are associated with the creation of a new policy. Furthermore, during the initial phase of training without action specification, wrong actions are chosen, which might be avoided through action masking. Often the actions were either an assignment of position

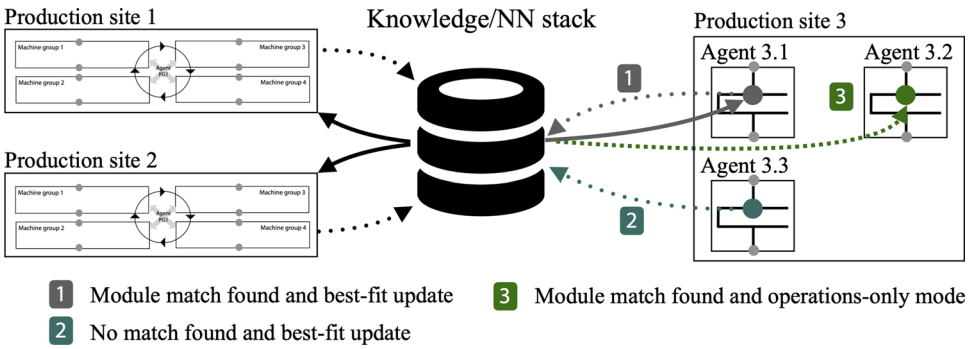


Fig. 6 Enhancing simulation scalability through a standardized neural network stack

(Gankin et al. 2021), or a combined instruction of which action is to be executed on which machine (Overbeck et al. 2021). Especially in large layouts, this quickly results in a large action space and elevated high task complexities. Conversely, the deep RL based top-level heuristic selects a low-level heuristic that is already integrating the process logic. As a result, the hyper-heuristic facilitates a complexity reduction by splitting the task into a high-level optimization and a low-level operational execution.

To implement the deep learning functions and maintain the accessibility of the *SimPy* simulation framework, *TensorFlow* was used to enable adaptive decision-making. However, prior to utilizing the *TensorFlow* framework, the deep RL control mechanism must be clarified. For this purpose, the following section quantifies the optimization objectives through a reward function and subsequently defines states and action spaces.

Reward function design

The reward function serves to capture the degree of fulfillment of the defined objectives to track the training success and to refine the neural network accordingly. Initially, the objectives have to be defined first, which are then consolidated into a reward signal. In our approach, we seek to incorporate customer-related services, particularly considering order priority and urgency as known from *Prime* and other express services. In response to current trends in prime services and evolving customer expectations, we include novel customer-centric parameters that differentiate between priority/standard and rush/non-rush orders. In addition, we incorporate technical standard variables like WIP levels, throughput times, and tardiness, consistent with several other studies (Hammami et al. 2017; Waschneck et al. 2018; Hofmann et al. 2020; Malus et al. 2020).

To account for the specifications of the individual production layers, differing total rewards are designed for the distribution and manufacturing agents. As listed in Table 4, the order distance and the global order start time are included as the individual and process-related reward R_i fraction for the distribution agents (rewards *Dist.1.1/2*). Conversely, for the manufacturing cells, the local processing start times are included as correspondence for the throughput time to avoid dissipating WIP effects (reward *Mfg.1*). In addition to R_i, the common rewards R_c aggregate general order metrics of priority, urgency, and due dates that are considered at all levels to satisfy customer-related services, in addition to minimizing overall tardiness through a time related reward $R_{due\ to}$. All individual and general rewards are constrained within a range of -200 to 200.

State and action space design

The selection of a suitable state space design is of great importance for an efficient production control and should be in accordance with the previously defined objectives. The state vector should contain all essential information, which includes order due dates and urgency, local and global processing start time, distance, and job priority. It can further contain information about the machine's operating status or other process information. It further includes buffer or storage information, such as their availability or occupancy, including all necessary job informations.

To ensure stable training gradients, faster training, and correct weight initialization, a min-max-normalization is applied to the time and distance-related values to scale the state input to the predefined and limited range between $[-1, 1]$. Furthermore, for discrete state spaces such as order priorities and urgencies, state inputs are normalized to $[0, 1]$, implying an input of $s_{i,prio} = 0$ for normal/non-rush orders, and $s_{i,prio} = 1$ for prioritized/rush orders.

In our study, the state vector encompasses information regarding local and global start times, distances, due dates, and order priorities for each individual agent. When a change occurs in the state of a module, the corresponding agent is triggered to select and execute an optimal action based on the respective metric values for all possible positions within its module. In scenarios involving multiple production and order metrics, the framework constructs the state vector by concatenating the pre-defined set of metrics.

The action space design refers to the definition of potential actions that an agent can execute in each state to deter-

Table 4 Summarized reward elements for individual and common rewards

Reward type	Formula
Individual rewards R_i	$[Mfg.1]\ R_{ltpt} = (1 - 2\frac{t_{ltpt,max} - t_{ltpt,n}}{t_{ltpt,max} - t_{ltpt,min}})^5 * R_{ltpt}$
	$[Dist.1.1]\ R_{gtpt} = (1 - 2\frac{t_{gtpt,max} - t_{gtpt,n}}{t_{gtpt,max} - t_{gtpt,min}})^5 * R_{gtpt}$
	$[Dist.1.2]\ R_{dist} = (2\frac{t_{dist,max} - t_{dist,n}}{t_{dist} - t_{dt,min}} - 1)^5 * R_{dist}$
Common rewards R_c	$[3]\ R_{due\ to} = (2\frac{t_{dt,max} - t_{dt,n}}{t_{dt,max} - t_{dt,min}} - 1)^5 * R_{dt}$
	$[4]\ R_{prio} = \begin{cases} 200 & \text{if status of order i is prioritized} \\ 0 & \text{non-priority order, no priority order available} \\ -200 & \text{non-priority order, priority order available} \end{cases}$
	$[5]\ R_{urg} = \begin{cases} 200 & \text{if order i is urgent} \\ 0 & \text{if order i is non-urgent, no urgent order available} \\ -200 & \text{if order i is non-urgent, urgent order available} \end{cases}$

mine the processing sequence. In Kanervisto et al. (2020) generic optimization approach, the action space is discretized and only necessary actions are selected, with dispatching rules as control heuristics that are linked to corresponding deep RL outputs. The selection of low-level dispatching rules is a crucial step before the training and optimization procedure and results in a representative rule set derived from benchmarks and related approaches. One advantage is, that the action space does not grow even with large layouts, as the logic is mapped intrinsically. This also prevents adapting the state space for new product types, because it only affects process-related specifications at the low-level heuristics level. However, standard and generic variables are not affected such as processing length or optimization, and customer-related parameters such as due times or order priorities. Subsequently, the highest priority first (HP), local and global first-in-first-out (FiFo), earliest due date (EDD), and lowest-distance-first (LDF) rules are applied as the low-level rule set. These widely deployed dispatching rules enable a fast order selection, which reduces the overall processing time and increases production efficiency.

Demonstration and transfer of results

For the demonstration of the results and to facilitate an iterative optimization approach of the simulation framework in accordance with the DSRM (Peffers et al. 2007), a case study for the fabrication of two product groups is presented. Subsequently, the outcomes from the training processes and operational application will enable the evaluation of performance and other indicators, such as resilience, adaptability, and explainability.

Simulated case-study

For the analysis, the specification of a case study is crucial to attain a specific benefit and to allow the deduction of product- and process-related performance indicators. For this purpose, we defined a three-stage system as presented in Fig. 7, which consists of two mid-layer distribution modules D1.1 and D1.2, each comprising two production modules.

A quality control module Q1.3 is provided in an independent additional module within the top layer D1. The modules D1.1 and D1.2 represent specifically defined production groups in which two types of goods are produced. This may include the production of two different kinds of printed circuit boards (PCBs). At the machines of type 1, the PCBs undergo exposure and etching processes to establish circuit patterns and interconnections, followed by drilling procedures to create apertures for electronic components and interconnections (machine type 2). Subsequently, electrical interconnections are developed and electronic components are soldered to the PCBs at machines of type 3. For descriptive purposes, it is assumed that two distinct PCB product groups are processed in parallel, with the first one in the D1.1, and the second one within the D1.2 module. This process serves as a demonstration and can be arbitrarily defined for other processes within the simulation framework. At this stage, we focus on the preliminary definition of the process chain to ensure the accurate adoption of fabrication procedures.

The assumed processing times are listed in Table 5. Although the described components have a seamless flow of material, the processes for manufacturing the varying PCBs are considered to be segregated operations. The incoming

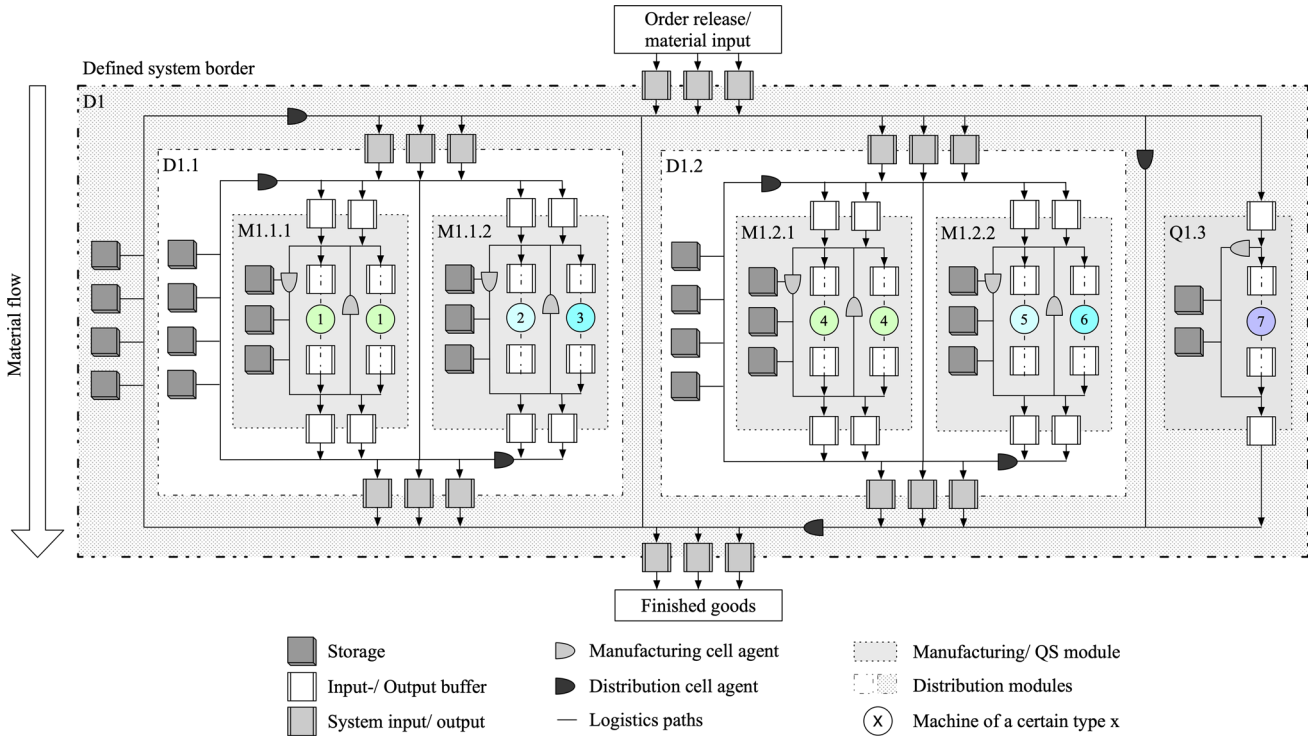


Fig. 7 Simulated 3-staged modular production system for PCB and electric drive fabrication

orders are classified as priority and/or rush orders with a 20% probability for each indicator. The orders are also subject to a 20% probability of being run through a quality assurance (QA) in module Q1.3, which takes another 2 min. The orders are released at the system input and transferred to the lower production levels by the distribution agents. Within the manufacturing cells, the orders are fed to the appropriate machines and then forwarded back to the higher levels after all scheduled operations are completed.

Table 6 outlines the configuration of the neural network model, detailing the number of neurons in both input and output layers, as well as the number and size of the hidden layers, and other additional parameters. The learning model is also specified, including the ϵ values for the beginning of the training phase, and a minimum ϵ value of 0.01, which determines the rate of random actions. A batch size of 128 was chosen to achieve a balance between learning speed and performance. The initial parameters were retrieved from established research, particularly the contributions of Mnih et al. (2015), Gankin et al. (2021), and subsequently refined through iterative optimization.

Exemplary simulation results

All of the following calculations were carried out on an Intel Core i9-12900k CPU and 64 GB RAM. If not other mentioned, three simulations with varying order sequences were conducted for each metric and analysis, to provide a representative benchmark. For the introduced operational scenarios and benchmarks, all agents (despite Q1.3) were fitted with trained neural networks through a unique cell identifier. For the system described in Figure 7, 30 neural networks (target and online) were used for a total of 15 agents respectively.

Training process analysis

In order to assess the training outcome, the progress of the different objectives of handling prioritized and urgent orders over the course of the training was evaluated with regard to the through-put times and order tardiness. As illustrated in Fig 8a and b, all processed orders were considered (dotted line), but also the combination of the different priorities and urgencies. Thereby, the considerably decreasing through-put times and tardiness rates of the higher-priority and more urgent orders, as well as their combination, becomes particularly clear. Compared to the later benchmarks, the training was conducted under a substantially elevated workload, as the focus was on maximizing learning outcomes rather than achieving a production equilibrium.

It becomes evident that throughput times and tardiness rates reach their global minimum at 1600 steps, after which the throughput times experience an upward trend again. While this increase correlates with a surge in order quantity and diversity, the consistent tardiness observed for combined

Table 5 Summarized processing times of both product groups; in [min.]

Processing step	Product group A			Product group B			Quality check
	PCB - type 1			PCB - type 2			
	Exposure	Drilling	Assembly	Exposure II	Drilling II	Assembly II	
Processing time [min.]	7	4	3	8	4	4	2
Average expected	21.4 (w.o. QA)			22.0 (w.o. QA)			
Throughput time [min.]	26.4 (with QA)			28.1 (with QA)			

Table 6 Iteratively defined deep RL and training parameter settings

Parameter	Value	Parameter	Value
Input layer size	Cell dependent (i.e. 162 for D1)	Drop out ratio	0.01
Output layer size	5 (dispatch rules)	Learning rate α	0.005
# hidden layers	2	Discount factor γ	0.99
# neurons in hidden layers	128, 128	Learning batch size	128
Target update step	5	Minimum ϵ	0.01
Activation function/ optimizer	ReLU/ Adam	ϵ-decay	0.997

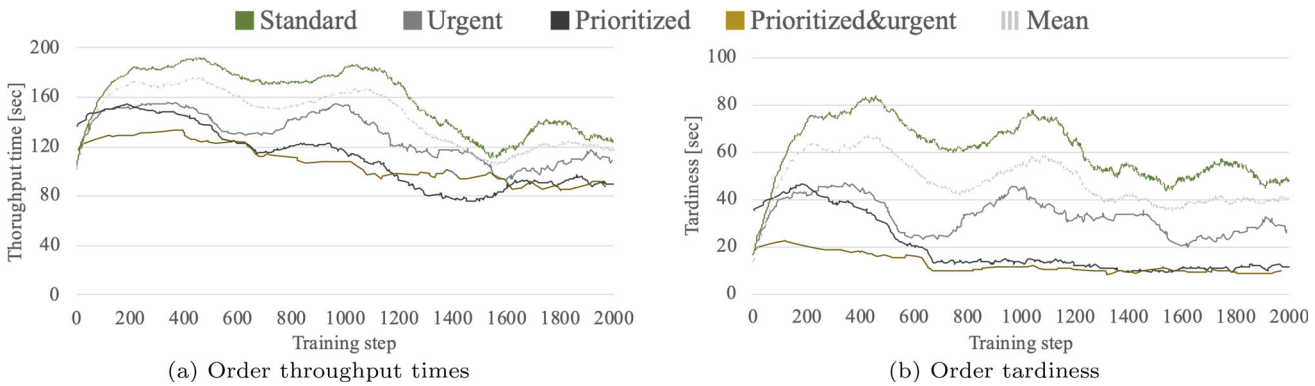
prioritized and urgent orders underscores the system's capacity to handle critical orders, relegating low-priority tasks to a waiting status.

As a result, after 2000 training steps, both priority and urgent orders exhibit a tardiness of about 10 s (Fig. 8b, right). Furthermore, it becomes evident that despite receiving the same rewards, priority orders are favored over urgent ones. This preference can be attributed to the implemented policy. If a prioritized order is selected, it is further sorted based on the due date if there are multiple orders that share the same priority. This process leads to an increased reward signal and, consequently, provokes a higher agent sensitization. Conversely, for urgent orders, no further sorting is conducted as there is often a most urgent one, resulting in no additional reward signal. This interdependence must be considered when balancing and calibrating the production objectives, which highlights the relevance of a proper reward function design and weighted rewards.

In a subsequent step, the decision-making of two agents in the D1 module was analyzed to trace the action selection back to the specific order type and to identify behavioral patterns. This enhances the optimization explainability and facilitates the comprehension of an agent's decision-making process. Figures 9 and 10 depict the chosen actions or dispatching rules as a function of the affected order throughout the first 1000 training steps. While the Fig. 9 indicate a rather balanced progression, the Fig. 10 exhibit a much more pronounced action tendency. Both have in common that in the case of prioritized orders (see Fig. 9c, d, 10c, d), the high priority first rule is noticeably and comprehensible dominant. For the second agent in Fig. 10c and 10d, it quickly displays a 100% rate of choosing the HP rule from the 400th training step onward.

In the case of urgent orders depicted in Figs. 9b and 10b, the earliest due date rule is employed in approximately 80% of taken actions. However, in Figure 9b, this is partially offset by the low distance first rule, resulting in enhanced routing efficiency, especially at the upper levels. An indifferent behavior is observed for the standard orders in Figs. 9a and 10a. In Fig. 9a, standard orders adopt a more discernible distance and due-to time-based processing, whereas, in Fig. 10a, no preferred action choice is evident for the distribution agent aside from the due-to date rule. Concurrently, a clear increase in the reward signal is observed, jumping from a moving average of 35 at the outset to 160 after 1000 training steps, suggesting a more likely increase in the achievement of the combined rewards and objectives, which contributes to the declining throughput times and tardiness as previously depicted in Fig. 8.

A further training analysis examined the different training strategies. In particular, novel modules that were integrated into a system could initially satisfy a reasonable degree of optimization requirements and exhibit a sufficient degree of stability despite the unavailability of a clear control policy. Since the transfer learning strategy is intended to accelerate the learning process, the parameters of a module similar to the D1 module were used with an increased size for the input buffers of D1.1/D1.2 and an extended storage space. In addition, different ϵ_{start} values were considered for the transfer learning-based training. Noticeably, the learning rates for the agents with $\epsilon_{start} = 0.5$ are significantly faster. Although they still encounter a 50% chance for a random action at the beginning, they exploit prior knowledge and reach higher rewards more quickly. Apparently, part of the existing control policy from the other agent could be used to make training decisions more effectively, despite the change in state inputs.



(a) Order throughput times (b) Order tardiness

Fig. 8 Development order throughput times and tardiness during the training process

Analysis of customer related indicator benchmarks

To establish a comparative benchmark against prevalent dispatching rules, the consolidated results are listed in Table 7. For these benchmarks, a cumulative duration of 7200 min with 2700 scheduled orders was simulated. Each time with varying order sets and the objective to assess the modular system's adaptability and efficacy in response to fluctuating demands. For the analysis, the combined throughput times and the tardiness are condensed as the central evaluation indicators for the satisfaction of customer-related objectives.

The application of deep RL-based agents demonstrates a measurable improvement in order tardiness and throughput times for both prioritized and rush orders relative to standard orders and conventional dispatching rules. This suggests a heightened alignment with customer-centric parameters. Specifically, there was an observed enhancement in the processing efficiency of intricate orders. Combined prioritized and urgent orders had a direct impact on the reduction of tardiness by nearly −100% and throughput times of −52%, in contrast to standard orders. When assessing standard orders within the benchmark, the hyper-heuristic exhibited increased throughput time and tardiness. Nonetheless, these factors were considered of lesser importance due to their relative insignificance.

The deep learning approach also offers the ability to optimize the allocation of resources effectively. It facilitates the development of individual control schemes for each order backlog within a module, ensuring optimal resource utilization. The inherent self-learning mechanisms of the deep RL offer two primary advantages, they support the ongoing refinement of production control and objective realization, and they enhance the performance of production processes within a dynamic environment. The adaptability of the deep RL approach underscores its potential to efficiently address diverse operational scenarios in order processing.

Evaluation of adaptability and resilience

To conduct a thorough evaluation of the framework's adaptability, we analyzed the learned control policy, first, from a structural-related perspective, and second, from an order-related perspective. The former refers to the responsiveness to a changing production environment through additional manufacturing modules, processes, and technologies or products. By adopting the hyper-heuristic approach, the deep learning component is able to strip down arising changes such as new products to a straight-forward process level that does not affect the top-level decision-making logic and process optimization. Similarly, the response to a malfunctioning machine was compensated by the rule-based decision-making process while the optimization process continued within the redefined context. Only the structural change of a system or re-scaling system components in scope, which goes along with a changed state vector, requires a re-training of the control policy, but only for the directly impacted and neighboring/upstream sub-systems. Due to the decentralized control paradigm, alterations such as the addition of a machine have a limited impact on other sub-systems. Only for the initial training, in which all agents are trained concurrently, the training takes significantly longer compared to a re-training of individual parts. For our case study, this resulted in a re-training time span of approximately 16 h against 3 h for training the M1.1.1 module. Another strategy allowed all but one of the agents in a module to be controlled by a heuristic. As a result, the training time for a single agent was notably reduced to 1.5 h, allowing for an efficient transfer of acquired knowledge to the remaining agents after completing the training process.

With respect to the structural modifications, the control design effectively stabilized the system loads and improved resilience, as detailed in Table 8. Observations from the lower section reveal that the control approach consistently

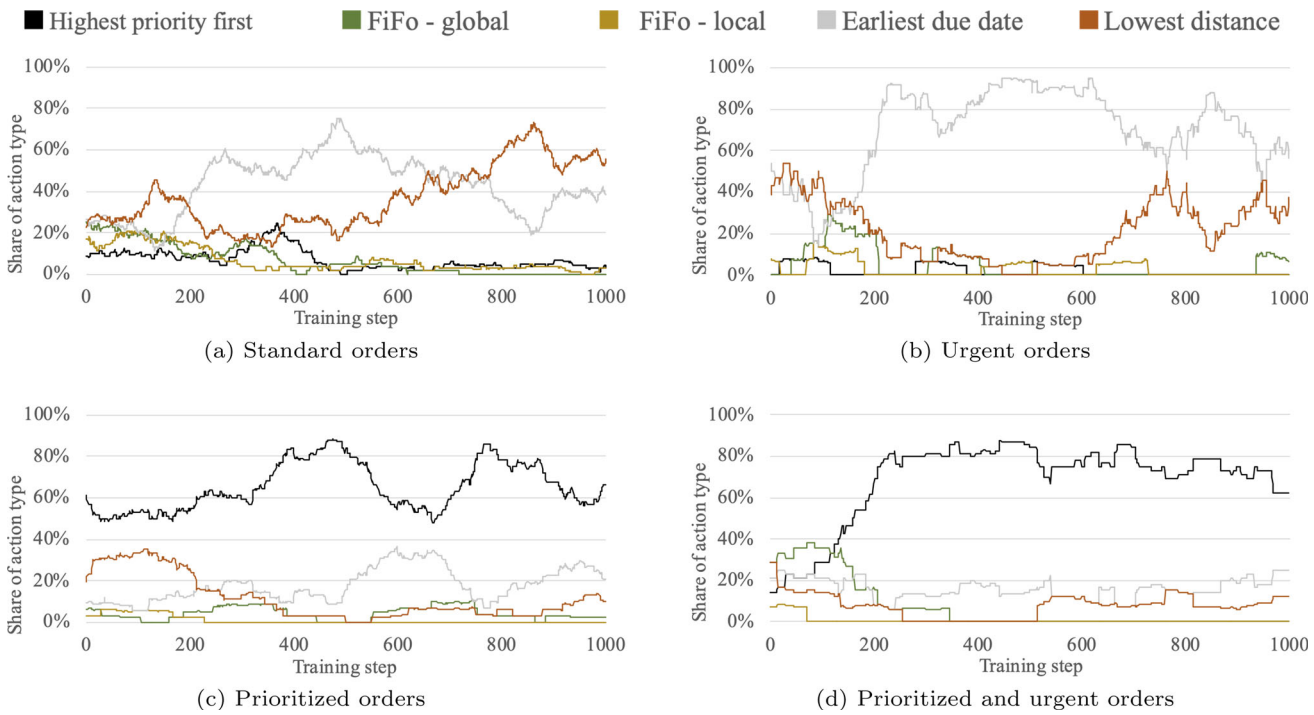


Fig. 9 Moving average of chosen actions for a D1 module agent throughout the training process

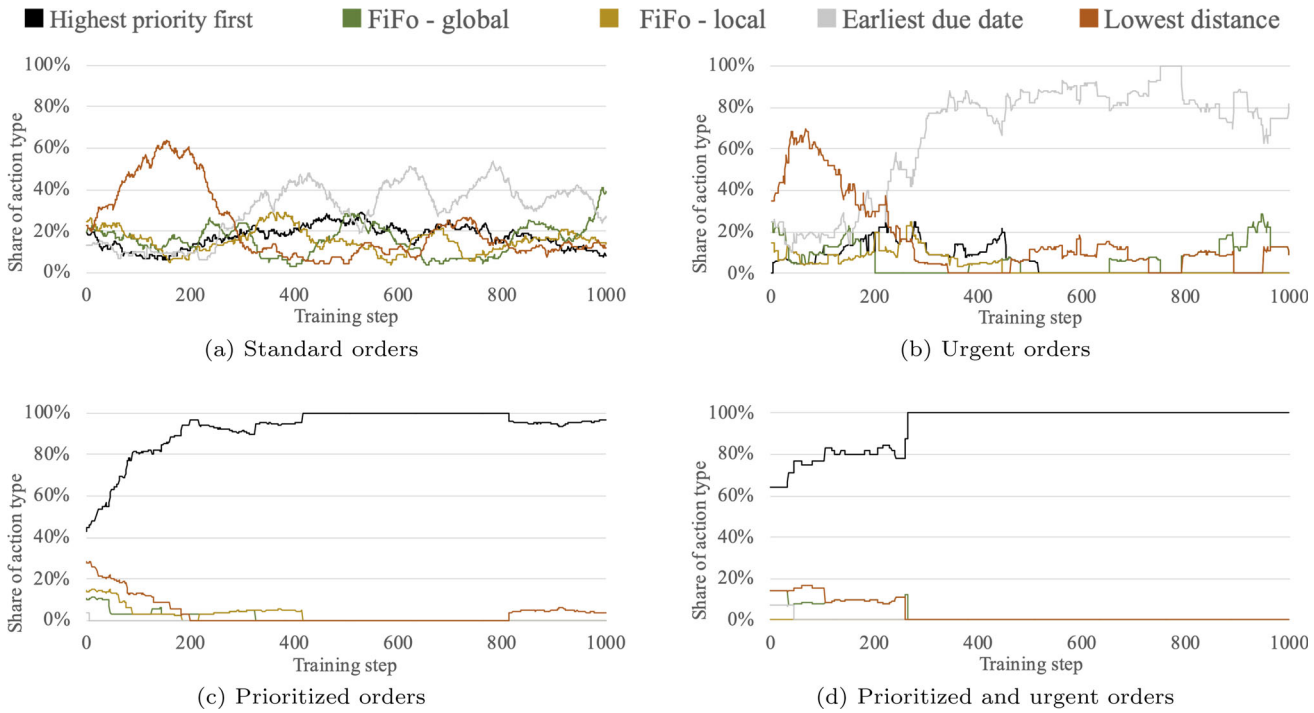


Fig. 10 Moving average of a another agent in D1, emphasizing a clear trend towards explainable action selection

Table 7 Control optimization and benchmark against conventional heuristics

Total	Hyper-heuristic		Earliest due date		FiFo global		FiFo local	
Processed orders [#]	2576		2566		2567		**2581**	
Throughput time [min]	**131.2**		131.7		132.1		133.0	
Tardiness [min]	37.8		**37.2**		38.7		37.5	
WIP [#]	40.1		**38.8**		42.0		39.0	
Throughput time order type split	Standard order	Rush order	Standard order	Rush order	Standard order	Rush order	Standard order	Rush order
Standard [min]	149.7	115.3	140.7	**98.8**	**132.6**	131.4	133.5	132.2
Priority [min]	**87.7**	**71.5**	140.3	98.6	131.6	129.1	132.4	130.3
Standard	1	−23%	1	−30%	1	−1%	1	−1%
Priority	−41%	−52%	0%	−30%	−1%	−3%	−1%	−2%
Tardiness order type split	Standard order	Rush Order	Standard order	Rush Order	Standard order	Rush Order	Standard order	Rush Order
Standard [min]	51.7	24.1	43.1	**15.8**	39.2	37.4	**38.1**	36.5
Priority [min]	**5.0**	**0.7**	42.2	16.1	38.2	37.6	36.8	36.5
Standard	1	−53%	1	−63%	1	−5%	1	−4%
Priority	−97%	−100%	−2%	−63%	−2%	−4%	−3%	−4%
WIP order type split [#]	Standard order	Rush Order	Standard order	Rush Order	Standard order	Rush Order	Standard order	Rush Order
Standard	28.1	7.4	**25.5**	**6.7**	27.7	7.3	25.8	6.8
Priority	**3.7**	**1.0**	5.2	1.4	5.6	1.5	5.2	1.4

Table 8 Assessment of System Resilience Against Fluctuating Order Loads

	High-load scenario						Mid-load scenario			Low-load scenario					
Scheduled orders	2800			2700			2600			2500			2400		
Split analysis:	WIP	Tardi	TPT	WIP	Tardi	TPT	WIP	Tardi	TPT	WIP	Tardi	TPT	WIP	Tardi	TPT
Total	45.6	43.9	144.3	37.5	33.1	124.0	31.5	27.6	103.9	18.6	9.5	73.8	11.8	0.4	34.9
Non-prio/-urgent	31.3	56.9	159.3	26.0	43.4	138.1	21.9	35.9	114.5	12.7	12.9	81.7	7.8	0.6	35.5
Non-prio, urgent	8.3	30.5	129.5	6.9	23.1	110.9	5.8	18.1	91.1	3.4	5.5	63.5	2.1	0.0	34.5
Prio, non-urgent	4.8	14.2	111.2	3.7	8.9	91.2	3.0	8.9	81.7	2.0	1.8	57.8	1.5	0.0	33.2
Prio & urgent	1.3	5.3	94.3	1.0	5.5	82.8	0.8	3.1	68.7	0.6	0.2	46.5	0.4	0.0	32.6
Relative															
Non-prio/-urgent	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Non-prio, urgent	−74%	−46%	−19%	−74%	−47%	−20%	−74%	−50%	−20%	−73%	−57%	−22%	−73%	−94%	−3%
Prio, non-urgent	−85%	−75%	−30%	−86%	−79%	−34%	−86%	−75%	−29%	−85%	−86%	−29%	−81%	−94%	−7%
Prio & urgent	−96%	−91%	−41%	−96%	−87%	−40%	−96%	−91%	−40%	−96%	−98%	−43%	−95%	−100%	−8%

enhanced the handling of prioritized orders, even with an increase in system load from 2400 to 2800 orders.

The WIP numbers in Table 8 should be contextualized with order quantities since the scheduled order entries lead to the entry of significantly fewer prioritized and urgent orders (20% for each order property). Statistically, in the case of 2800 orders with a 45.6 WIP, 1.8 combined prioritized and urgent orders should be released (4%). Yet, the data indicates a WIP of only 1.3 orders, indicating a 28% reduction. In comparison to the tardiness observed for the 2700 episodes, the tardiness decreased from 5.5 s to 5.3 s at the 2800-episode mark. While this decrease could be incidental, it also underscores the control design's efficiency in processing orders of higher importance. In essence, the analysis underscores that under conditions of high system load (e.g., with 2700/2800 scheduled orders), the increased scope of operational action enlarges the optimization range for deep learning agents, thereby compensating for the elevated WIP and resulting throughput times.

Framework discussion

In contemporary markets, characterized by fluctuating sales and supply conditions, it is essential to pursue ongoing process adaptation and optimization to maintain a competitive edge. For this purpose, simulations are increasingly recognized as instrumental to evaluating the effectiveness of (intelligent) production control strategies, including those that leverage system intelligence, and for preemptively evaluating potential real-world scenarios. In the present study, we attempted to synergize the comprehensive flexibility inherent to a simulation framework - suited for diverse production scenarios - with the resilient performance and adaptability characteristic of a deep RL-based hyper-heuristic.

Our results demonstrate that by defining a simple reward function paired with a defined action space, we were able to optimize pre-defined objectives and outperform widely applied dispatching rules. The definition of differing distribution and manufacturing levels facilitated the simulation of large-scale systems, segmenting them into modules, to decompose and manage the overall system complexity. The adoption of a decentralized agent control and the modularization of the entire production system also offer great potential for re-using trained agents, since changes in the production system do not affect the entire system, but only individual sections.

The developed framework aims to minimize the transfer gap through the automated initiation of the production system combined with the integration of various operation modes. This integration not only supports comprehensive re-training but also promotes selective and efficient utilization of individual policies, potentially resulting in faster and improved

simulation outcomes. The incorporation of the modularization concept from the foundational simulation framework into the intelligent control strategy suggests enhanced transferability to diverse practical applications.

Conclusion

In this paper, we introduced a flexible simulation framework for modular production systems that is based on a novel module and neural network recognition mechanism and a stack of trained neural networks to increase simulation efficiency and adaptability. By integrating a deep RL based top-level heuristic and process constraint mapping through low-level dispatching rules, the framework enables the optimization of various target parameters within a multi-agent production system. The hyper-heuristic control mechanism facilitated the primary utilization of deep RL for optimization purposes, thereby promoting resilient and stable processes, even during the initial training. Both, distribution and manufacturing/shopfloor levels, were implemented and optimized regarding key performance indicators by using a concurrent learning paradigm. By leveraging the synergy between the flexible simulation framework and the adaptive control concept, requirements of customer-centric production services and process target indicators can be freely defined.

In a representative evaluation, we demonstrated the multi-objective optimization performance of the control framework. Prioritized and urgent orders were processed with reduced throughput times and tardiness than standard orders, leading to accelerated response times to external disruptions, such as increased order loads. Particularly in today's demanding market environments, this contributes to maintaining a company's competitiveness. Furthermore, the framework reached a more balanced optimization, as parameters were dynamically assessed, allowing a scenario-specific emphasis on individual objectives and an assessment of explainability for the chosen action.

The presented framework leverages a structural and process-related adaptability, thus providing a flexible response to order fluctuations and facilitating the targeted processing of arbitrary order types and quantities. Further, the influence of incoming orders and machine bottlenecks on WIP can be systematically examined. Within the field of complex manufacturing, our framework employs both, a structural modularization and an algorithmic hyper-heuristic, for system complexity decomposition. The decentralized decision-making further helped to reduce optimization complexity and enabled coping with the surging information volumes and ever-increasing customer requirements. This not only streamlines operational processes but also ensures a high data management efficiency. Given its inherent adaptability, the framework remains efficient for a wide range of potential scenarios and motivates further research of intelligent control strategies in modular production systems.

Such future research endeavors might focus on examining the quantification of reward components and their weighted correlation to the attainment of desired objectives. We also foresee a focus on the practical transfer of these methodologies to real-world settings and the integration of advanced agent collaboration techniques. Additionally, weaving in a techno-economical analysis will be pivotal to ensure cost-effectiveness and operational efficiency in modular production systems

Author Contributions M. P.: Conceptualization, methodology, software, investigation, data collection, experimentation, writing. NG: Methodology, review and editing.

Funding Open Access funding enabled and organized by Projekt DEAL. This work received no funding.

Data availability The data that support the findings of this study are available from the corresponding author upon request.

Declarations

Conflict of interest The authors report no declarations of interest.

References

Altenmüller, T., Stüker, T., Waschneck, B., Kuhnle, A., & Lanza, G. (2020). Reinforcement learning for an intelligent and autonomous production control of complex job-shops under time constraints. *Production Engineering, 14*, 319–328. https://doi.org/10.1007/s11740-020-00967-8

Arunraj, N. S., & Ahrens, D. (2015). A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting. *International Journal of Production Economics, 170*, 321–335. https://doi.org/10.1016/j.ijpe.2015.09.039. https://linkinghub.elsevier.com/retrieve/pii/S0925527315003783

Babiceanu, R. F., & Chen, F. F. (2006). Development and Applications of Holonic Manufacturing Systems: A Survey. *Journal of Intelligent Manufacturing, 17*, 111–131. https://doi.org/10.1007/s10845-005-5516-y

Bahrpeyma, F., & Reichelt, D. (2022). A review of the applications of multi-agent reinforcement learning in smart factories. *Frontiers in Robotics and AI, 9*, 1027340. https://doi.org/10.3389/frobt.2022.1027340

Bergmann, S., & Stelzer, S. (2011). Approximation of Dispatching Rules in Manufacturing Control Using Artificial Neural Networks. In *2011 IEEE Workshop on Principles of Advanced and Distributed Simulation* (pp. 1–8). Nice, France: IEEE volume 12086556. https://doi.org/10.1109/PADS.2011.5936774. http://ieeexplore.ieee.org/document/5936774/

Bergmann, S., Stelzer, S., & Strassburger, S. (2014). On the use of artificial neural networks in simulation-based manufacturing control. *Journal of Simulation, 8*, 76–90. https://doi.org/10.1057/jos.2013.6

Buckhorst, A. F., Grahn, L., & Schmitt, R. H. (2022). Decentralized Holonic Control System Model for Line-less Mobile Assembly Systems. *Robotics and Computer-Integrated Manufacturing, 75*, 102301. https://doi.org/10.1016/j.rcim.2021.102301. https://linkinghub.elsevier.com/retrieve/pii/S0736584521001812

Bueno, A., Godinho Filho, M., & Frank, A.G. (2020). Smart production planning and control in the Industry 4.0 context: A systematic literature review. *Computers & Industrial Engineering*, *149*, 106774. https://doi.org/10.1016/j.cie.2020.106774. https://linkinghub.elsevier.com/retrieve/pii/S0360835220304861

Burke, E. K., Hyde, M. R., Kendall, G., Ochoa, G., Özcan, E., & Woodward, J. R. (2010). A Classification of Hyper-Heuristic Approaches. In M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of Metaheuristics* 146 (pp. 453–477). Cham: Springer International Publishing. https://doi.org/10.1007/978-1-4419-1665-5_15.

Burke, E. K., Hyde, M. R., Kendall, G., Ochoa, G., Özcan, E., & Woodward, J. R. (2019). A Classification of Hyper-Heuristic Approaches: Revisited. In M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of Metaheuristics* 272 (pp. 453–477). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-91086-4_14.

Cadavid, J. P. U., Lamouri, S., Grabot, B., & Fortin, A. (2019). Machine Learning in Production Planning and Control: A Review of Empirical Literature. *IFAC-PapersOnLine, 52*, 385–390. https://doi.org/10.1016/j.ifacol.2019.11.155. https://linkinghub.elsevier.com/retrieve/pii/S2405896319311048

Chen, S., Wang, W., & Zio, E. (2021). A Simulation-Based Multi-Objective Optimization Framework for the Production Planning in Energy Supply Chains. *Energies, 14*, 2684. https://doi.org/10.3390/en14092684. www.mdpi.com/1996-1073/14/9/2684

Cowling, P., Kendall, G., & Soubeiga, E. (2001). A Hyperheuristic Approach to Scheduling a Sales Summit. In G. Goos, J. Hartmanis, J. van Leeuwen, E. Burke, & W. Erben (Eds.), *Practice and Theory of Automated Timetabling III* (pp. 176–190). Berlin, Heidelberg: Springer Berlin Heidelberg volume 2079. https://doi.org/10.1007/3-540-44629-X_11.

Derigent, W., Cardin, O., & Trentesaux, D. (2021). Industry 4.0: contributions of holonic manufacturing control architectures and future challenges. *Journal of Intelligent Manufacturing, 32*, 1797–1818. https://doi.org/10.1007/s10845-020-01532-x

Dittrich, M.-A., & Fohlmeister, S. (2020). Cooperative multi-agent system for production control using reinforcement learning. *CIRP Annals, 69*, 389–392. https://doi.org/10.1016/j.cirp.2020.04.005

Drake, J. H., Kheiri, A., Özcan, E., & Burke, E. K. (2020). Recent advances in selection hyper-heuristics. *European Journal of Operational Research, 285*, 405–428. https://doi.org/10.1016/j.ejor.2019.07.073. https://linkinghub.elsevier.com/retrieve/pii/S0377221719306526

Esteso, A., Peidro, D., Mula, J., & Díaz-Madroñero, M. (2022). Reinforcement learning applied to production planning and control. *International Journal of Production Research, 61*, 1–18. https://doi.org/10.1080/00207543.2022.2104180

Farsi, M., Erkoyuncu, J. A., Steenstra, D., & Roy, R. (2019). A modular hybrid simulation framework for complex manufacturing system design. *Simulation Modelling Practice and Theory*, *94*, 14–30. https://doi.org/10.1016/j.simpat.2019.02.002. https://linkinghub.elsevier.com/retrieve/pii/S1569190X19300139

Fowler, J. W., Mönch, L., & Ponsignon, T. (2015). Discrete-event simulation for semiconductor wafer fabrication facilities: a tutorial. *International Journal of Industrial Engineering*, *22*. https://doi.org/10.23055/IJIETAP.2015.22.5.2276.

Fumagalli, L., Negri, E., Sottoriva, E., Polenghi, A., & Macchi, M. (2018). A novel scheduling framework: integrating genetic algorithms and discrete event simulation. *International Journal of Management and Decision Making, 17*, 371. https://doi.org/10.1504/IJMDM.2018.095738. www.inderscience.com/link.php?id=95738

Gankin, D., Mayer, S., Zinn, J., Vogel-Heuser, B., & Endisch, C. (2021). Modular Production Control with Multi-Agent Deep Q-Learning. (pp. 1–8). Vasteras, Sweden: IEEE volume 21364735. https://doi.org/10.1109/ETFA45728.2021.9613177. https://ieeexplore.ieee.org/document/9613177/

Garetti, M., & Taisch, M. (1999). Neural networks in production planning and control. *Production Planning & Control, 10*, 324–339. https://doi.org/10.1080/095372899233082

Grabot, B., & Geneste, L. (1994). Dispatching rules in scheduling Dispatching rules in scheduling: a fuzzy approach. *International Journal of Production Research, 32*, 903–915. https://doi.org/10.1080/00207549408956978

Gronauer, S., & Diepold, K. (2021). Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review, 55*, 895–943. https://doi.org/10.1007/s10462-021-09996-w

Groover, M. P., & Jayaprakash, G. (2016). *Automation, production systems, and computer-integrated manufacturing. Always learning* (4th ed.). Upper Saddle River, NJ: Pearson Prentice Hall.

Gros, T. P., Gros, J., & Wolf, V. (2020). Real-Time Decision Making for a Car Manufacturing Process Using Deep Reinforcement Learning. In *2020 Winter Simulation Conference (WSC)* (pp. 3032–3044). Orlando, FL, USA: IEEE volume 20512838. https://doi.org/10.1109/WSC48552.2020.9383884. https://ieeexplore.ieee.org/document/9383884/

Grumbach, F., Müller, A., Reusch, P., & Trojahn, S. (2022). Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-022-02069-x

Hammami, Z., Mouelhi, W., & Ben Said, L. (2017). On-line self-adaptive framework for tailoring a neural-agent learning model addressing dynamic real-time scheduling problems. *Journal of Manufacturing Systems, 45*, 97–108. https://doi.org/10.1016/j.jmsy.2017.08.003. https://linkinghub.elsevier.com/retrieve/pii/S0278612517301243

Heger, J., Hildebrandt, T., & Scholz-Reiter, B. (2015). Dispatching rule selection with Gaussian processes. *Central European Journal of Operations Research, 23*, 235–249. https://doi.org/10.1007/s10100-013-0322-7

Herrera, M., Pérez-Hernández, M., Kumar Parlikad, A., & Izquierdo, J. (2020). Multi-Agent Systems and Complex Networks: Review and Applications in Systems Engineering. *Processes, 8*, 312. https://doi.org/10.3390/pr8030312. www.mdpi.com/2227-9717/8/3/312

Hofmann, C., Krahe, C., Stricker, N., & Lanza, G. (2020). Autonomous production control for matrix production based on deep Q-learning. *Procedia CIRP, 88*, 25–30. https://doi.org/10.1016/j.procir.2020.05.005. https://linkinghub.elsevier.com/retrieve/pii/S2212827120303206

Holthaus, O., & Rajendran, C. (1997). Efficient dispatching rules for scheduling in a job shop. *International Journal of Production Economics, 48*, 87–105. https://doi.org/10.1016/S0925-5273(96)00068-0. https://linkinghub.elsevier.com/retrieve/pii/S0925527396000680

Jeon, S. M., & Kim, G. (2016). A survey of simulation modeling techniques in production planning and control (PPC). *Produc-*

tion Planning & Control, 27, 360–377. https://doi.org/10.1080/09537287.2015.1128010

Kallestad, J., Hasibi, R., Hemmati, A., & Sörensen, K. (2023). A General Deep Reinforcement Learning Hyperheuristic Framework for Solving Combinatorial Optimization Problems. *European Journal of Operational Research, 209*, 446–468. https://doi.org/10.1016/j.ejor.2023.01.017. https://linkinghub.elsevier.com/retrieve/pii/S037722172300036X

Kanervisto, A., Scheller, C., & Hautamaki, V. (2020). Action Space Shaping in Deep Reinforcement Learning. In *2020 IEEE Conference on Games (CoG)* (pp. 479–486). Osaka, Japan: IEEE. https://doi.org/10.1109/CoG47356.2020.9231687. https://ieeexplore.ieee.org/document/9231687/

Kang, Z., Catal, C., & Tekinerdogan, B. (2020). Machine learning applications in production lines: A systematic literature review. *Computers & Industrial Engineering, 149*, 106773. https://doi.org/10.1016/j.cie.2020.106773

Kapoor, K., Bigdeli, A. Z., Dwivedi, Y. K., & Raman, R. (2021). *How is COVID-19 altering the manufacturing landscape?* A literature review of imminent challenges and management interventions: Annals of Operations Research. https://doi.org/10.1007/s10479-021-04397-2

Kashfi, M. A., & Javadi, M. (2015). A model for selecting suitable dispatching rule in FMS based on fuzzy multi attribute group decision making. *Production Engineering, 9*, 237–246. https://doi.org/10.1007/s11740-015-0603-1

Kuhnle, A., Kaiser, J.-P., Theiß, F., Stricker, N., & Lanza, G. (2020). Designing an adaptive production control system using reinforcement learning. *Journal of Intelligent Manufacturing, 32*, 855–876. https://doi.org/10.1007/s10845-020-01612-y

Kuhnle, A., May, M. C., Schäfer, L., & Lanza, G. (2021). Explainable reinforcement learning in production control of job shop manufacturing system. *International Journal of Production Research, 60*, 5812–5834. https://doi.org/10.1080/00207543.2021.1972179

Kuhnle, A., Röhrig, N., & Lanza, G. (2019). Autonomous order dispatching in the semiconductor industry using reinforcement learning. *Procedia CIRP, 79*, 391–396. https://doi.org/10.1016/j.procir.2019.02.101. https://linkinghub.elsevier.com/retrieve/pii/S2212827119302185

Kuhnle, A., Schäfer, L., Stricker, N., & Lanza, G. (2019). Design, Implementation and Evaluation of Reinforcement Learning for an Adaptive Order Dispatching in Job Shop Manufacturing Systems. *Procedia CIRP, 81*, 234–239. https://doi.org/10.1016/j.procir.2019.03.041. https://linkinghub.elsevier.com/retrieve/pii/S2212827119303464

Law, A.M. (2007). *Simulation modeling and analysis*. McGraw-Hill series in industrial engineering and management science (4th ed.). Boston: McGraw-Hill. ISBN: 978-0-07-329441-4.

Liao, Y., Deschamps, F., Loures, E. D. F. R., & Ramos, L. F. P. (2017). Past, present and future of Industry 4.0 - a systematic literature review and research agenda proposal. *International Journal of Production Research, 55*, 3609–3629. https://doi.org/10.1080/00207543.2017.1308576

Liu, H., & Dong, J. J. (1996). Dispatching rule selection using artificial neural networks for dynamic planning and scheduling. *Journal of Intelligent Manufacturing, 7*, 243–250. https://doi.org/10.1007/BF00118083

Liu, R., Piplani, R., & Toro, C. (2022). Deep reinforcement learning for dynamic scheduling of a flexible job shop. *International Journal of Production Research, 60*, 4049–4069. https://doi.org/10.1080/00207543.2022.2058432

Luo, S. (2020). Dynamic scheduling for flexible job shop with new job insertions by deep reinforcement learning. *Applied Soft Computing, 91*, 106208. https://doi.org/10.1016/j.asoc.2020.106208

Malus, A., Kozjek, D., & Vrabič, R. (2020). Real-time order dispatching for a fleet of autonomous mobile robots using multi-agent reinforcement learning. *CIRP Annals, 69*, 397–400. https://doi.org/10.1016/j.cirp.2020.04.001

Manriquez, F., Pérez, J., & Morales, N. (2020). A simulation-optimization framework for short-term underground mine production scheduling. *Optimization and Engineering, 21*, 939–971. https://doi.org/10.1007/s11081-020-09496-w

Mayer, S., Classen, T., & Endisch, C. (2021). Modular production control using deep reinforcement learning: proximal policy optimization. *Journal of Intelligent Manufacturing, 32*, 2335–2351. https://doi.org/10.1007/s10845-021-01778-z

May, M. C., Kiefer, L., Kuhnle, A., Stricker, N., & Lanza, G. (2021). Decentralized Multi-Agent Production Control through Economic Model Bidding for Matrix Production Systems. *Procedia CIRP, 96*, 3–8. https://doi.org/10.1016/j.procir.2021.01.043. https://linkinghub.elsevier.com/retrieve/pii/S2212827121000664

Mehlig, B. (2021). *Machine Learning with Neural Networks: An Introduction for Scientists and Engineers* (1st ed.). Cambridge University Press. https://doi.org/10.1017/9781108860604

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning. https://doi.org/10.48550/arXiv.1312.5602. arXiv:1312.5602

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*, 529–533. https://doi.org/10.1038/nature14236. www.nature.com/articles/nature14236

Mouelhi-Chibani, W., & Pierreval, H. (2010). Training a neural network to select dispatching rules in real time. *Computers & Industrial Engineering, 58*, 249–256. https://doi.org/10.1016/j.cie.2009.03.008. https://linkinghub.elsevier.com/retrieve/pii/S0360835209000953

Mourtzis, D. (2020). Simulation in the design and operation of manufacturing systems: state of the art and new trends. *International Journal of Production Research, 58*, 1927–1949. https://doi.org/10.1080/00207543.2019.1636321

Nasiri, M. M., Yazdanparast, R., & Jolai, F. (2017). A simulation optimisation approach for real-time scheduling in an open shop environment using a composite dispatching rule. *International Journal of Computer Integrated Manufacturing, 30*, 1239–1252. https://doi.org/10.1080/0951192X.2017.1307452

Nazari, M., Oroojlooy, A., Snyder, L. V., & Takáč, M. (2018). Reinforcement Learning for Solving the Vehicle Routing Problem. https://doi.org/10.48550/ARXIV.1802.04240. arXiv:1802.04240

Neto, A. A., Deschamps, F., Da Silva, E. R., & De Lima, E. P. (2020). Digital twins in manufacturing: an assessment of drivers, enablers and barriers to implementation. *Procedia CIRP, 93*, 210–215. https://doi.org/10.1016/j.procir.2020.04.131. https://linkinghub.elsevier.com/retrieve/pii/S2212827120307733

Oluyisola, O. E., Bhalla, S., Sgarbossa, F., & Strandhagen, J. O. (2022). Designing and developing smart production planning and control systems in the industry 4.0 era: a methodology and case study. *Journal of Intelligent Manufacturing, 33*, 311–332. https://doi.org/10.1007/s10845-021-01808-w

Overbeck, L., Hugues, A., May, M. C., Kuhnle, A., & Lanza, G. (2021). Reinforcement Learning Based Production Control of Semi-automated Manufacturing Systems. *Procedia CIRP, 103*, 170–175. https://doi.org/10.1016/j.procir.2021.10.027. https://linkinghub.elsevier.com/retrieve/pii/S2212827121008684

Panzer, M., & Bender, B. (2022). Deep reinforcement learning in production systems: a systematic literature review. *International Journal of Production Research, 60*, 4316–4341. https://doi.org/10.1080/00207543.2021.1973138

Panzer, M., Bender, B., & Gronau, N. (2022). Neural agent-based production planning and control: An architectural review. *Journal of Manufacturing Systems, 65*, 743–766. https://doi.org/10.1016/j.jmsy.2022.10.019. https://linkinghub.elsevier.com/retrieve/pii/S027861252200190X

Parente, M., Figueira, G., Amorim, P., & Marques, A. (2020). Production scheduling in the context of Industry 4.0: review and trends. *International Journal of Production Research, 58*, 5401–5431. https://doi.org/10.1080/00207543.2020.1718794

Pawar, S., & Maulik, R. (2021). Distributed deep reinforcement learning for simulation control. *Machine Learning: Science and Technology, 2*, 025029. https://doi.org/10.1088/2632-2153/abdaf8

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems, 24*, 45–77. https://doi.org/10.2753/MIS0742-1222240302

Phanden, R. K., Palková, Z., & Sindhwani, R. (2019). A Framework for Flexible Job Shop Scheduling Problem Using Simulation-Based Cuckoo Search Optimization. In K. Shanker, R. Shankar, & R. Sindhwani (Eds.), *Advances in Industrial and Production Engineering* (pp. 247–262). Singapore: Springer Singapore. https://doi.org/10.1007/978-981-13-6412-9_23.

Rauf, M., Guan, Z., Sarfraz, S., Mumtaz, J., Shehab, E., Jahanzaib, M., & Hanif, M. (2020). A smart algorithm for multi-criteria optimization of model sequencing problem in assembly lines. *Robotics and Computer-Integrated Manufacturing, 61*, 101844. https://doi.org/10.1016/j.rcim.2019.101844. https://linkinghub.elsevier.com/retrieve/pii/S0736584518301959

Rocchetta, R., Bellani, L., Compare, M., Zio, E., & Patelli, E. (2019). A reinforcement learning framework for optimal operation and maintenance of power grids. *Applied Energy, 241*, 291–301. https://doi.org/10.1016/j.apenergy.2019.03.027. www.sciencedirect.com/science/article/pii/S0306261919304222

Rodríguez, M. L. R., Kubler, S., De Giorgio, A., Cordy, M., Robert, J., & Le Traon, Y. (2022). Multi-agent deep reinforcement learning based Predictive Maintenance on parallel machines. *Robotics and Computer-Integrated Manufacturing, 78*, 102406. https://doi.org/10.1016/j.rcim.2022.102406. https://linkinghub.elsevier.com/retrieve/pii/S0736584522000928

Rojas, R. A., & Rauch, E. (2019). From a literature review to a conceptual framework of enablers for smart manufacturing control. *The International Journal of Advanced Manufacturing Technology, 104*, 517–533. https://doi.org/10.1007/s00170-019-03854-4

Sakr, A. H., Aboelhassan, A., Yacout, S., & Bassetto, S. (2021). Simulation and deep reinforcement learning for adaptive dispatching in semiconductor manufacturing systems. *Journal of Intelligent Manufacturing, 34*, 1311–1324. https://doi.org/10.1007/s10845-021-01851-7

Sallez, Y., Berger, T., Raileanu, S., Chaabane, S., & Trentesaux, D. (2010). Semi-heterarchical control of FMS: From theory to application. *Engineering Applications of Artificial Intelligence, 23*, 1314–1326. https://doi.org/10.1016/j.engappai.2010.06.013. https://linkinghub.elsevier.com/retrieve/pii/S0952197610001363

Samsonov, V., Kemmerling, M., Paegert, M., Lütticke, D., Sauermann, F., Gützlaff, A., Schuh, G., & Meisen, T. (2021). Manufacturing Control in Job Shop Environments with Reinforcement Learning:. (pp. 589–597). Online. https://doi.org/10.5220/0010202405890597. https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0010202405890597

Samsonov, V., Ben Hicham, K., & Meisen, T. (2022). Reinforcement Learning in Manufacturing Control: Baselines, challenges and ways forward. *Engineering Applications of Artificial Intelligence, 112*, 104868. https://doi.org/10.1016/j.engappai.2022.104868. https://linkinghub.elsevier.com/retrieve/pii/S0952197622001130

Schmidt, M., & Nyhuis, P. (2021). *Produktionsplanung und -steuerung im Hannoveraner Lieferkettenmodell: innerbetrieblicher Abgleich logistischer Zielgrößen*. Berlin [Heidelberg]: Springer Vieweg. iSBN: 978-3-662-63896-5.

Shavandi, A., & Khedmati, M. (2022). A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets. *Expert Systems with Applications, 208*, 118124. https://doi.org/10.1016/j.eswa.2022.118124. https://linkinghub.elsevier.com/retrieve/pii/S0957417422013082

Shiue, Y.-R., Lee, K.-C., & Su, C.-T. (2018). Real-time scheduling for a smart factory using a reinforcement learning approach. *Computers & Industrial Engineering, 125*, 604–614. https://doi.org/10.1016/j.cie.2018.03.039. www.sciencedirect.com/science/article/pii/S036083521830130X

Su, J., Huang, J., Adams, S., Chang, Q., & Beling, P. A. (2022). Deep multi-agent reinforcement learning for multi-level preventive maintenance in manufacturing systems. *Expert Systems with Applications, 192*, 116323. https://doi.org/10.1016/j.eswa.2021.116323. https://linkinghub.elsevier.com/retrieve/pii/S0957417421016249

Sutton, R. S., & Barto, A. G. (2017). *Reinforcement learning: an introduction*. Adaptive computation and machine learning series (2nd ed.). Cambridge, Massachusetts: The MIT Press. ISBN: 978-0-262-03924-6.

Swiercz, A. (2017). Hyper-Heuristics and Metaheuristics for Selected Bio-Inspired Combinatorial Optimization Problems. *Heuristics and Hyper-Heuristics - Principles and Applications, 1*, 3–20. https://doi.org/10.5772/intechopen.69225. www.intechopen.com/chapters/55554

Tao, H., Qiu, J., Chen, Y., Stojanovic, V., & Cheng, L. (2023). Unsupervised cross-domain rolling bearing fault diagnosis based on time-frequency information fusion. *Journal of the Franklin Institute, 360*, 1454–1477. https://doi.org/10.1016/j.jfranklin.2022.11.004. https://linkinghub.elsevier.com/retrieve/pii/S0016003222008055

Tassel, P., Gebser, M., & Schekotihin, K. (2021). A Reinforcement Learning Environment For Job-Shop Scheduling. https://doi.org/10.48550/ARXIV.2104.03760. arXiv:2104.03760.

Umlauft, M., Schranz, M., & Elmenreich, W. (2022). SwarmFabSim: A Simulation Framework for Bottom-up Optimization in Flexible Job-Shop Scheduling using NetLogo:. In *Proceedings of the 12th International Conference on Simulation and Modeling Methodologies, Technologies and Applications* (pp. 271–279). Lisbon, Portugal. https://doi.org/10.5220/0011274700003274. https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0011274700003274

Uzsoy, R., Church, L. K., Ovacik, I. M., & Hinchman, J. (1993). Performance evaluation of dispatching rules for semiconductor testing operations. *Journal of Electronics Manufacturing, 03*, 95–105. https://doi.org/10.1142/S0960313193000115

Valckenaers, P., Bonneville, F., Van Brussel, H., Bongaerts, L., & Wyns, J. (1994). Results of the holonic control system benchmark at KU Leuven. In *Proceedings of the Fourth International Conference on Computer Integrated Manufacturing and Automation Technology* (pp. 128–133). Troy, NY, USA: IEEE Comput. Soc. Press. https://doi.org/10.1109/CIMAT.1994.389083. http://ieeexplore.ieee.org/document/389083/

Venturelli, D., Marchand, D. J. J., & Rojo, G. (2015). Quantum Annealing Implementation of Job-Shop Scheduling. *arXiv, Quantum Physics*. . https://doi.org/10.48550/ARXIV.1506.08479. arXiv:1506.08479

Waschneck, B., Reichstaller, A., Belzner, L., Altenmuller, T., Bauernhansl, T., Knapp, A., & Kyek, A. (2018). Deep reinforcement learning for semiconductor production scheduling. In *2018 29th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*. Saratoga Springs, NY, USA. https://doi.org/10.1109/ASMC.2018.8373191. https://ieeexplore.ieee.org/document/8373191/

Waubert De Puiseau, C., Peters, J., Dörpelkus, C., Tercan, H., & Meisen, T. (2023). schlably: A Python framework for deep reinforcement learning based scheduling experiments. *SoftwareX, 22*, 101383. https://doi.org/10.1016/j.softx.2023.101383. https://linkinghub.elsevier.com/retrieve/pii/S2352711023000791

Weichert, D., Link, P., Stoll, A., Rüping, S., Ihlenfeldt, S., & Wrobel, S. (2019). A review of machine learning for the optimization of production processes. *The International Journal of Advanced Manufacturing Technology, 104*, 1889–1902. https://doi.org/10.1007/s00170-019-03988-5

Zhang, C., Song, W., Cao, Z., Zhang, J., Tan, P. S., & Xu, C. (2020). Learning to Dispatch for Job Shop Scheduling via Deep Reinforcement Learning. https://doi.org/10.48550/ARXIV.2010.12367. arXiv:2010.12367.

Zhang, Y., Bai, R., Qu, R., Tu, C., & Jin, J. (2022). A deep reinforcement learning based hyper-heuristic for combinatorial optimisation with uncertainties. *European Journal of Operational Research, 300*, 418–427. https://doi.org/10.1016/j.ejor.2021.10.032

Zhang, J., Ding, G., Zou, Y., Qin, S., & Fu, J. (2019). Review of job shop scheduling research and its new perspectives under Industry 4.0. *Journal of Intelligent Manufacturing, 30*, 1809–1830. https://doi.org/10.1007/s10845-017-1350-2

Zhang, H.-C., & Huang, S. H. (1995). Applications of neural networks in manufacturing: a state-of-the-art survey. *International Journal of Production Research, 33*, 705–728. https://doi.org/10.1080/00207549508930175

Zhang, H., & Roy, U. (2019). A semantics-based dispatching rule selection approach for job shop scheduling. *Journal of Intelligent Manufacturing, 30*, 2759–2779. https://doi.org/10.1007/s10845-018-1421-z

Zhao, Y., & Zhang, H. (2021). Application of Machine Learning and Rule Scheduling in a Job-Shop Production Control System. *International Journal of Simulation Modelling, 20*, 410–421. https://doi.org/10.2507/IJSIMM20-2-CO10

Zheng, S., Gupta, C., & Serita, S. (2020). Manufacturing Dispatching Using Reinforcement and Transfer Learning. In U. Brefeld, E. Fromont, A. Hotho, A. Knobbe, M. Maathuis, & C. Robardet (Eds.), *Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 655–671). Würzburg, Germany. https://doi.org/10.1007/978-3-030-46133-1_39.

Zhou, L., Jiang, Z., Geng, N., Niu, Y., Cui, F., Liu, K., & Qi, N. (2022). Production and operations management for intelligent manufacturing: a systematic literature review. *International Journal of Production Research, 60*, 808–846. https://doi.org/10.1080/00207543.2021.2017055

Zhou, C., Tao, H., Chen, Y., Stojanovic, V., & Paszke, W. (2022). Robust point-to-point iterative learning control for constrained systems: A minimum energy approach. *International Journal of Robust and Nonlinear Control, 32*, 10139–10161. https://doi.org/10.1002/rnc.6354

Zhou, Y., Yang, J.-J., & Huang, Z. (2020). Automatic design of scheduling policies for dynamic flexible job shop scheduling via surrogate-assisted cooperative co-evolution genetic programming. *International Journal of Production Research, 58*, 2561–2580. https://doi.org/10.1080/00207543.2019.1620362