Marcel Panzer | Benedict Bender | Norbert Gronau

A deep reinforcement learning based hyper-heuristic for modular production control

Suggested citation referring to the original publication:
International journal of production research (2023) , pp. 1 - 22
DOI: https://doi.org/10.1080/00207543.2023.2233641
ISSN: 0020-7543, 1366-588X, 0278-6125

Journal article | Version of record

Secondary publication archived on the Publication Server of the University of Potsdam:
Zweitveröffentlichungen der Universität Potsdam : Wirtschafts- und Sozialwissenschaftliche Reihe 173
ISSN: 1867-5808
URN: https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-605642
DOI: https://doi.org/10.25932/publishup-60564

INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH
https://doi.org/10.1080/00207543.2023.2233641

Taylor & Francis
Taylor & Francis Group

A deep reinforcement learning based hyper-heuristic for modular production control

Marcel Panzer, Benedict Bender and Norbert Gronau

Chair of Business Informatics, Processes and Systems, University of Potsdam, Potsdam, Germany

ABSTRACT

In nowadays production, fluctuations in demand, shortening product life-cycles, and highly configurable products require an adaptive and robust control approach to maintain competitiveness. This approach must not only optimise desired production objectives but also cope with unforeseen machine failures, rush orders, and changes in short-term demand. Previous control approaches were often implemented using a single operations layer and a standalone deep learning approach, which may not adequately address the complex organisational demands of modern manufacturing systems. To address this challenge, we propose a hyper-heuristics control model within a semi-heterarchical production system, in which multiple manufacturing and distribution agents are spread across pre-defined modules. The agents employ a deep reinforcement learning algorithm to learn a policy for selecting low-level heuristics in a situation-specific manner, thereby leveraging system performance and adaptability. We tested our approach in simulation and transferred it to a hybrid production environment. By that, we were able to demonstrate its multi-objective optimisation capabilities compared to conventional approaches in terms of mean throughput time, tardiness, and processing of prioritised orders in a multi-layered production system. The modular design is promising in reducing the overall system complexity and facilitates a quick and seamless integration into other scenarios.

ARTICLE HISTORY
Received 15 March 2023
Accepted 22 June 2023

KEYWORDS
Production control; modular production; multi-agent system; deep reinforcement learning; deep learning; multi-objective optimisation

1. Introduction

With growing challenges of fluctuating demand, market volatility, and increasingly complex manufacturing processes, there is a strong need for a resilient and adaptable production control (Kapoor et al. 2021). The production control must not only cope with unforeseen machine failures while managing high product individualisation levels and dynamic processes, but also handle large amounts of data under sustainable matters which requires sensible data collection and processing (Bueno, Godinho Filho, and Frank 2020; Tao et al. 2018). To cope with these challenges, companies must seize the opportunity to implement control approaches that can cope with varying production conditions and facilitate ongoing optimisation of performance indicators to increase competitiveness (Grassi et al. 2020; Lee et al. 2018; Parente et al. 2020). Recent advancements in cyber-physical systems, the industrial internet of things, and other related technologies already facilitated widespread data collection and processing in production systems (Lass and Gronau 2020; Lee, Bagheri, and Jin 2016; Lee, Bagheri, and Kao 2015; Ritterbusch and Rolf Teichmann 2023). By leveraging the *Industry 4.0* principles, these technologies can unlock significant process potentials and competitive advantages (Parente et al. 2020). In production control practice, however, conventional algorithms are often applied, such as the *First-in-First-out* rule (*FiFo*), which do not guarantee global optimality while others are hard-coded and layout specific (Kuhnle et al. 2021; Mönch, Fowler, and Mason 2013), which do not meet recent demands regarding flexibility.

A recent approach to process large amounts of input data, deep reinforcement learning (RL), was increasingly applied in production control in recent years (Panzer and Bender 2022; Samsonov et al. 2021; Sutton and Barto 2017). Deep RL is characterised by its interactive, trial-and-error learning principle and often demonstrated superior performance compared to conventional production control approaches. Its online optimisation and direct data processing capabilities make

CONTACT Marcel Panzer marcel.panzer@wi.uni-potsdam.de Chair of Business Informatics, Processes and Systems, University of Potsdam, Karl-Marx-Street 67, Potsdam 14482, Germany

This article was originally published with errors, which have now been corrected in the online version. Please see Correction (http://dx.doi.org/10.1080/00207543.2023.2245272)

it particularly well-suited for real-time decision making in fast-paced applications, setting it apart from other AI-based methods that may require longer computation times (Chang et al. 2022). Despite the considerable attention paid to deep RL-based single-agent systems, multi-agent-based systems have received comparatively less attention due to the significant challenges associated with agent orchestration and communication design (Panzer and Bender 2022). Yet, they can assist in achieving both, local and global, performance objectives and develop robust control policies (Tampuu et al. 2017).

To combine the advantages of deep RL and multi-agent-based systems to cope with recent demands, this paper proposes a novel hyper-heuristics based control approach for modular multi-agent production systems that utilises both, distributed resources and deep RL. The hyper heuristic is applied for control optimisation that utilises deep neural networks for the selection of low-level heuristics. Each agent deploys its own neural networks, tailored to its specific modular production environment which is transferable to similar systems. To leverage adaptability and scalabilty, our further motivation is to implement the approach within a semi-heterarchical production to cope with the prevailing organisational challenges of multi-layered production systems. The key contribution is an adaptive control approach that combines deep learning-based control with an adaptive and scalable production organisation that optimises pre-defined production performance indicators. The approach is designed to handle spontaneous events, such as machine failures and rush orders, while ensuring stability and facilitating a seamless transition to real-world production scenarios.

The remainder of the paper is organised as follows: In Section 2, basics of deep RL and multi-agent-based production control are outlined and the research objective is specified. The conceptual design and artifact requirements are defined in Section 3. Results are outlined and evaluated in Section 4, and transferred to a real test-bed in Section 5. A discussion is outlined in Section 6 and a conclusion is given in Section 7.

2. Problem statement

This section first discusses the basics and organisations of modular and matrix production systems and elaborates on the principles of (deep) RL. Finally, results of a systematic literature review of the combination of these in decentralised and multi-agent based production control is conducted for defining the specific research objectives.

2.1. Modular and semi-heterarchical production systems

Nowadays, adaptability is vital in production systems to handle machine failures, rush orders, and other disruptions. To cope with such internal and external disruptions, modular production systems were designed and often validated in simulated approaches (May et al. 2021). Such modular systems allow individualised production processes through line-less control and the ability to define arbitrary production flows by using automated guided vehicles, which enable a detached process execution (May et al. 2021; Mayer, Classen, and Endisch 2021; Tamás 2021). This flexibility leads to higher use of resources through sharing strategies, shorter transportation routes, and reduced buffer stocks (as in (Greschke et al. 2014; Schenk, Wirth, and Muller 2010)). However, despite its advantages, modular production approaches suffer from an increased control complexity due to the highly flexible operation of manufacturing modules and the large control solution space (Schenk, Wirth, and Muller 2010; Schmidtke, Rettmann, and Behrendt 2021). The increased number of potential process paths leads to large actions spaces that raises optimisation complexity and a proper extraction of relevant information through a neural network gets more complex.

With this regard, previous research emphasised the importance of decentralising decision-making among production agents, allowing them to make decisions based on their specific task and available resources to leverage their reasoning, perception, and action capabilities (Balaji and Srinivasan 2010; Parunak et al. 1986). Weiss (2001) particularly emphasises the flexible and reconfigurable properties of multi-agent structures as conventional decentralised control approaches. In a more recent review, Herrera et al. (2020) further emphasises the relevance of multi-agent systems for existing and planned real-world applications. A specific differentiation of such multi-agent systems is established by subdividing them into organisational forms, depending on the allocation, grouping, and interaction of the agents. While a hierarchy is characterised by a multitude of fixed master-slave relationships, a heterarchy consists primarily of peer-level relationships with distributed privileges to fulfil global and local objectives (Baker 1998; Bongaerts et al. 2000). Hierarchical systems are rather static, whereas heterarchical organisations suffer from local optimisation tendencies and myopic behaviour due to the lack of master-slave relationships (Sallez et al. 2010).

A semi-heterarchical production system seeks to combine the advantages of both hierarchical and heterarchical concepts. It achieves high integrity of the sub-components

through its hierarchical structure (Valckenaers et al. 1994) while maintaining a high reactivity and robustness through the distribution principle within the heterarchical systems (Groover 2019). The semi-heterarchical concept simultaneously enables both, long-term and short-term objectives to be reached, and allows the corresponding parameters to be optimised (Sallez et al. 2010). Implementations of this concept were made by Grassi et al. (2020) and Grassi et al. (2021) in different production levels. This facilitates a multi-agent system that enables the allocation of agents based on their functional scope. The semi-heterarchical concept was further implemented within a single control structure and by establishing domain-wise clustering by Borangiu et al. (2009) and Borangiu et al. (2010) in the field of product-driven scheduling and by Zambrano Rey et al. (2013) in the field of flexible manufacturing control. By using a 2-layer approach, Borangiu et al. (2010) fulfilled different objective horizons and obtained comparably higher robustness and agility of the system. Through the semi-heterarchical approach, Zambrano Rey et al. (2013) was further able to achieve control over the otherwise myopic agent behaviour.

2.2. Deep reinforcement learning based hyper-heuristic

To cope with the dynamic control requirements and allow for an adaptive control deep RL was implemented in production control approaches (Bahrpeyma and Reichelt 2022; Esteso et al. 2022). It made the leap to competitiveness especially with its successful implementation of the Atari environment, and has since become increasingly appealing for complex optimisation problems (Mnih et al. 2013). Due to its particularly interactive learning strategy and the neural network's ability to process large state inputs, deep RL can be tailored to a variety of data-centric online applications (Baer et al. 2020). As indicated in recent reviews, particularly value-based RL approaches were widely deployed in production control and demonstrated superior performances (Bahrpeyma and Reichelt 2022; Panzer, Bender, and Gronau 2022).

To facilitate the fundamental integration capability of deep RL to production control, the problem under consideration must satisfy the Markov property and correspond to a Markov Decision Process (MDP). Besides the rigid definition of the considered scope, the Markov assumption must be met, which implies that all future production states only depend on the current state. This constitutes the underlying assumption of our approach and of the later designed discrete-event based simulation (Sutton and Barto 2017). Q-learning is a variant of RL, which is a model-free, off-policy RL algorithm that exploits an action- or Q-value function. The Q-value function, see Equation (1), is typically defined based on an agent's expected cumulative reward in Equation (2), which follows its current policy, derived from the Bellman equation Bellman.

$$Q(s_t, a_t) = r + \gamma \ max(Q(s_{t+1}, a_{t+1})) \tag{1}$$

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

In following functions $Q(s_t, a_t)$ resembles the Q-value for a state s_t and action a_t at a certain time t. r is the immediate reward received after taking action a in state s_t, γ is the discount factor, and $max(Q(s_{t+1}, a_{t+1}))$ is the maximum Q-value over the next states s_{t+1} and actions a_{t+1} that can be executed from state s_{t+1} (Sutton and Barto 2017).

Whereas in conventional Q-learning a table is used to map Q-values, deep RL exploits a deep neural network to function approximator to map the policy of an agent, which is frequently updated based on past made experiences. The neural network enables the agent to learn from high-dimensional and complex input data such as raw sensory information. It can handle non-linear and non-convex environments better than traditional RL, allowing it to be more accurate and efficient in learning. During the learning process, often a batch replay is used, which iteratively trains and fits the network based on the stored batch data. The neural network, also known as the Q-network, takes the current state s_t of the agent as input and outputs a Q-value for each possible action a_t. The Q-network is trained to minimise the difference between the predicted Q-values and the target Q-values, which are calculated using the above mentioned Bellman equation. Thereby, Equation (3) is utilised for updating the weights of the Q-network in deep Q-learning (DQN), wherein w represents the weights of the Q-network, α denotes the learning rate, and E signifies the loss function, defined as the mean squared error between the predicted Q-values and the target Q-values, as shown in Equation (4).

$$w = w - \alpha \ \nabla w(E) \tag{3}$$

$$E = (Q(s, a) - (r + \gamma \ max(Q(s_{t+1}, a_{t+1}))))^2 \tag{4}$$

The DQN equation is derived by combining Equations (1)–(4), where the Q-network is trained to approximate the Q-values by minimising the loss function using the Bellman equation, as summarised in Equation (5). To further stabilise learning and performance, a target network with weights θ^- is introduced and used to calculate $Q(s_{t+1}, a_{t+1})$ for the next states (Mnih et al. 2013, 2015).

$$\begin{aligned} Q(s_t, a_t, \theta) \ &\leftarrow \ Q(s_t, a_t, \theta) \\ &+ \ \alpha \ [r \ + \ \gamma \ max \ Q(s_{t+1}, a_{t+1}, \theta^-) \ - \ Q(s_t, a_t, \theta)] \end{aligned} \tag{5}$$

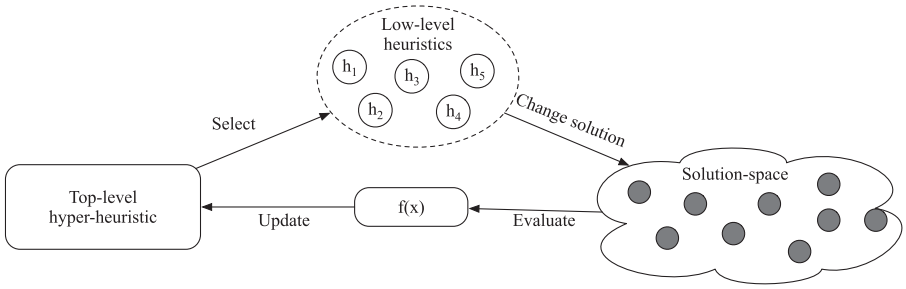

Figure 1. Hyper-heuristics based optimisation approach (Cowling, Kendall, and Soubeiga 2001; Swiercz 2017).

Building up on deep RL, a hyper-heuristic is an optimisation model that utilises a machine or deep learning algorithm such as the DQN to learn a high-level policy for selecting and adapting low-level policies. Due to the deep learning algorithm, the hyper-heuristic possess the capability to adapt to specific optimisation tasks, and thus effectively utilise the inherent capabilities and process logic's of low-level heuristics. This enables an automation of the design process, and allows for the utilisation of knowledge from online machine learning algorithms as an optimiser, resulting in the derivation of near-optimal scheduling and dispatching policies, which are based on established and more comprehensible low-level heuristics (Burke et al. 2010, 2019; Drake et al. 2020). During operation, the smart agent is trained to select one suitable heuristic from a pre-defined set depending on the received production state. The objective of a deep RL based hyper-heuristic is to improve the performance of the underlying optimisation problem by utilising the strengths and process implications of multiple low-level heuristics as illustrated in Figure 1 (Van Ekeris, Meyes, and Meisen 2021; Zhang et al. 2022).

2.3. Deep RL and multi-agent based production control

Prevailing approaches in multi-agent-based production control already try to leverage deep RL to benefit from a decentralised and online decision making and optimisation process. To get a comprehensive overview of the research field and trends, we searched the databases *Scopus* and *WebofScience* to identify relevant scientific papers.

Several studies have explored different approaches to improve production performance indicators such as utilisation rates or order tardiness. Malus, Kozjek, and Vrabič (2020) proposed an order bidding mechanism for autonomous mobile robots that utilised a joint global reward to minimise delays, optimising global utilisation efficiency upon part completion and locally accepted bids. The agents could bid between 0 and 1 based on their respective states and proximal policy optimisation (PPO) output, and the order with the highest bid was assigned for the dispatch task. The PPO ensured updates of the policy of not being too large, thus providing a balance between policy exploration and exploitation, making it more sample-efficient and stable compared to other RL algorithms Schulman et al. (2017). To cope with the dynamics of inherent order scheduling, Hammami, Mouelhi, and Ben Said (2017) proposed an multi-agent system based on simultaneous learning and information sharing between agents to reduce average delay. Decisional agents, responsible for overall decision-making process and order dispatching, were associated with choice agents that selected the best neural network for the decisional agent based on the desired performance optimisation criteria. In the dispatching approaches conducted by Dittrich and Fohlmeister (2020) and Hofmann et al. (2020), a centralised DQN decision module was employed for training purposes. This central module functioned as a repository for storing and updating the dispatching policy among all agents, and provided the current control policy upon request. Dittrich and Fohlmeister (2020) created a job shop with three process steps (turning, milling, and assembly), where agents could access local and global information to allocate orders to machine within a machine group to optimise mean cycle time. These agents could request necessary local and global system information to facilitate informed decision-making which was facilitated by globally defined rewards, which were then propagated to individual agents. Other studies focussed on different training strategies to improve production control. Waschneck et al. (2018) implemented a strategy where one network was trained at a time for stability and learning speed reasons, and subsequently each wafer manufacturing workstation was controlled by one neural network at a time to optimise for maximum uptime utilisation as a global goal. The network input comprised all possible lot positions and an idle option, while the network output included machine states (capacity, availability, etc.) and job states (type, progress). To optimise the product sequence in car manufacturing, Gros, Gros, and Wolf (2020) used an iterative learning strategy to prevent instabilities caused

by parallel training of several agents, determining the output sequence of cars to a buffer after finishing paint jobs. This minimised costs caused by inefficient car sequences and non-balanced flow of goods in subsequent manufacturing processes. Overbeck et al. (2021) utilised PPO agents and hyper-parameter tuning, to determine the optimal action in an automated assembly system adhering to Chaku-Chaku principles, where workers were tasked with loading machines and transporting orders. An evaluation in a real assembly cell for automotive parts demonstrated an improvement in decision quality over time and a more produced parts.

The aforementioned approaches primarily dealt with control problems in conventional job shops. Initial approaches in matrix systems were proposed by Gankin et al. (2021), May et al. (2021), and Hofmann et al. (2020). In Hofmann et al., agents received immediate rewards after each operational step for a chosen action and a delayed reward based on the total global cycle time after an order was completed, which accelerated the learning process and reduced order throughput times. The simulated system featured 10 workstations and several autonomous guided vehicles (AGVs) that could perform multiple process steps and were fully flexibly interconnected. Meanwhile, May et al. (2021) implemented an economic bidding approach to increase utilisation efficiency in a matrix-structured production system. The approach to maximised the operational profit for each agent independently and optimised the execution time and resource utilisation efficiency against conventional heuristics. Gankin et al. (2021) introduced a first large-scale matrix layout comprising 25 machines, which was based on the modular approach developed by Mayer, Classen, and Endisch (2021). Two distinct product types were manufactured, each involving 13 process steps. To reduce decision complexity, an action masking mechanism was implemented for preventing the selection of incorrect actions as each process step could be performed only at specific machines. All 20 transport units were trained in parallel as DQN agents, and the same neural network and buffer were used as the central decision instance and to facilitate experience sharing between the agents.

2.4. Problem formulation and contribution

From the previous literature set, several performant applications can be observed, however, most approaches are rather specific, such as the wafer fabrication or the car paint buffer re-ordering. A more scalable and adaptive approach is given with the matrix approaches of Mayer, Classen, and Endisch (2021) or Gankin et al. (2021). However, these assume a matrix structure and are less focussed on the clustering of production units, and, in case of Gankin et al. (2021), exploit a central decision entity. Also, all mentioned approaches deploy a single-staged control organisation at the operations level. In addition to the application and organisation scope, which is summarised in Table 1, the algorithmic approaches are often self-contained AI algorithms.

Table 1 highlights three fields of potential research in production control, algorithmic (1), organisational (2), and optimisation opportunities (3). The algorithmic field (1) currently lacks a deep RL-based hyper-heuristics approach, which operates at a higher level and can quickly select lower-level heuristics, as opposed to meta-heuristics that serve as search process optimisers or general guidelines (Tamás 2021). Previous research indicated that a deep RL-based hyper-heuristic can outperform population-based meta-heuristics, such as genetic algorithms, in terms of performance and interpretability (Kallestad et al. 2023; Zhang et al. 2022). Additionally, hyper-heuristics have benefit from fast computation of operations (as in Chang et al. 2022; Liu, Chang, and Tseng 2020), making them particularly suitable for real-time environments.

Regarding the organisational design (2), our approach deploys shopfloor and distribution layers to facilitate modular and semi-heterarchical production processes. The use of a deep RL-based multi-agent system is emphasised due to its collaborative possibilities and the ability to cope with larger systems requirements (Tampuu et al. 2017). Despite the current focus on single-agent environments (Esteso et al. 2022), our approach takes advantage of a distributed and semi-heterarchical agent organisation and manages system complexity by decomposing the overall complexity into respective fragments of only processing relevant information. The approach will be applied in a modular production environment that allows pre-defined tool bundling and configuration of machine groups, similar to real production. The modularity aims to exploit product-specific machine synergies while reducing coordination complexity and increasing scalability. This aligns with the requirement for a common modelling approach, as articulated by Mourtzis (2020), which is supported by the proposed standardised production modules within the underlying simulation framework. Consequently, the simulation can serve as an evaluation tool for our deep learning control framework prior to its deployment in the intended (hybrid) real-world environment, functioning as an progressive Industry 4.0 test-bed, as emphasised by de Paula Ferreira, Armellini, and Antonio De Santa-Eulalia (2020) and de Paula Ferreira et al. (2022).

Regarding the optimisation task (3), various approaches were proposed to optimise one or two objectives such

Table 1. Deep RL-based multi-agent approaches in production control.

Application	Algorithm	Training strategy	Control strategy	Agent interaction	Objective parameter	Orga. levels	Transfer scope	Year	Source
Car buffer	DQN	Iterative learning	Distributed	–	Cost/ decision time	1	Simulation	2020	Gros et al.
Chaku-chaku line	PPO	Shared PPO module	Central	–	Utilization/throughput	1	Simulation	2021	Overbeck et al.
	SA	Concurrent learning	Distributed	Agent information exchange	Mean tardiness	1	Simulation	2017	Hammami et al.
Job shop	DQN	Iterative DQN/ heuristics learning	Distributed	Global rewards	WIP/ uptime utilisation	1	Simulation	2018	Waschneck et al.
	DQN	Shared DQN module	Central	Agent information exchange	Mean cycle time	1	Simulation	2020	Dittrich et al.
	DRL (TD3)	Concurrent learning	Distributed	Order bidding mechanism	Tardiness	1	Simulation	2020	Malus et al.
	DQN	Shared DQN module	–	Agent state information	Throughput time	1	Simulation	2020	Hofmann et al.
Matrix production	DQN	Shared DQN module	Central	–	Throughput	1	Simulation	2021	Gankin et al.
	PPO	–	Distributed	Economic bidding	Execution time/utilisation eff.	1	Simulation	2021	May et al.
Matrix/modular production	DQN-based hyper-heuristic	Concurrent learning	Distributed	Agent and cell states	Throughput time/ priori-ties/tardiness	> 1	Simulation and reality	Our approach	

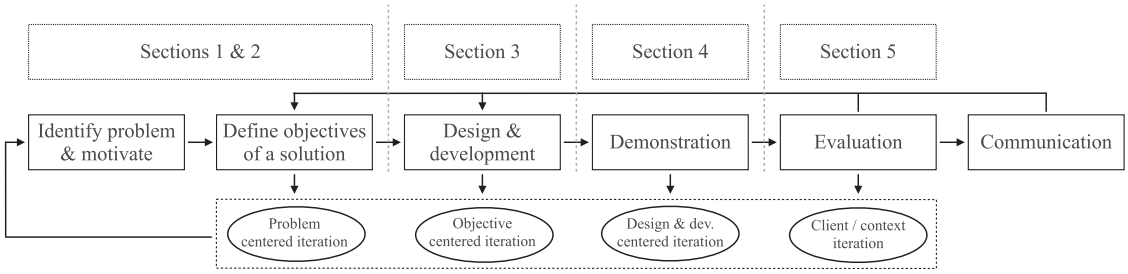

Figure 2. Pursued *DSRM* methodology (Peffers et al. 2007).

as tardiness, throughput, or utilisation. However, two variables that were not yet explored are order priority and urgency. Especially in modern production with customer-oriented services, order priority and urgency directly affect resource allocation and operational efficiency, to meet customers needs. Prioritized or premium customer groups as well as rush orders represent a significant source of revenue, and require a flexible production control that can adapt to fluctuating demands. To address this order features effectively, a combined measure or reward function is designed, that balances the prioritisation of orders according to the pre-defined objectives of mean throughput-time, tardiness as well as order priority and urgency.

To the best of our knowledge, this is the first approach of a hyper-heuristics-based production control in a modular production system. We seek to leverage production performance and control production complexity through a layered approach to enable a robust and adaptive control. Furthermore, this will be the first approach that transfers a control approach of deep RL-based hyper-heuristics control to a real application.

3. Conceptual design

To ensure a systematic approach for reaching the research objectives, we followed the design science research methodology according to (Peffers et al. (2007) , see Figure 2). The first two steps of problem identification and objectives definition were addressed in the previous sections, which are now followed by constructing the research artefact as the third step. To satisfy the emerging dynamic requirements and deliver an adaptable and scalable simulation approach, an appropriate simulation framework must first be chosen. The framework should seamlessly integrate the hyper-heuristic control approach, to enable decentralised decision-making and leverage production performance.

3.1. Simulation approach

To implement the simulation, the production simulation framework *CoBra*, developed at our research department, was utilised. *CoBra* is based on the *SimPy* simulation library, commonly used in the field of discrete-event production simulation, as done in previous works (e.g. Kuhnle et al. 2021, 2020; Liu, Piplani, and Toro 2022). This tool enables the rapid creation of modular production environments, and supports the arbitrary design of production processes and control rules. A key requirement is the ability to seamlessly transfer the approach to a real environment. To achieve this, real-world requirements such as machine failures, maintenance efforts, randomly determined order sequences, and other process-dependent parameters were incorporated into the simulation. However, considering all the information for decision-making is impractical, necessitating a systematic construction of the state vector that is used for feeding the neural network, as outlined in Section 3.2.1.

In our approach, the modular production system consists of entities or groups of distributed autonomous agents, referred to as manufacturing (1) and distribution (2) modules (Giret and Botti 2004). The agents operate in parallel, making decisions based on the information they individually receive, thereby reducing the need for a centralised control model. The base/bottom layer is composed of several manufacturing agents, that are responsible for the processing of goods (refer to the bottom of Figure 3). These agents serve in modules that are typically specialised in conducting specific manufacturing processes, such as welding or assembly. The upper layers consist of distribution modules, that are responsible for the production coordination and distribution of goods (see top three layers in Figure 3). All agents can work in collaboration with other agents to process intermediate products. They have the capability to make decisions based on a shared policy and the individually received information. This facilitates the system's adaptability to adapt to changing production conditions in a flexible and efficient manner. The *CoBra* framework effectively enables the conceptual design of such semi-heterarchical systems up to a fully matrix-like production, that can be controlled by either deep learning based or conventional approaches. During the simulation, all agents are trained concurrently based on the obtained rewards, following the epsilon-greedy strategy. This facilitates an

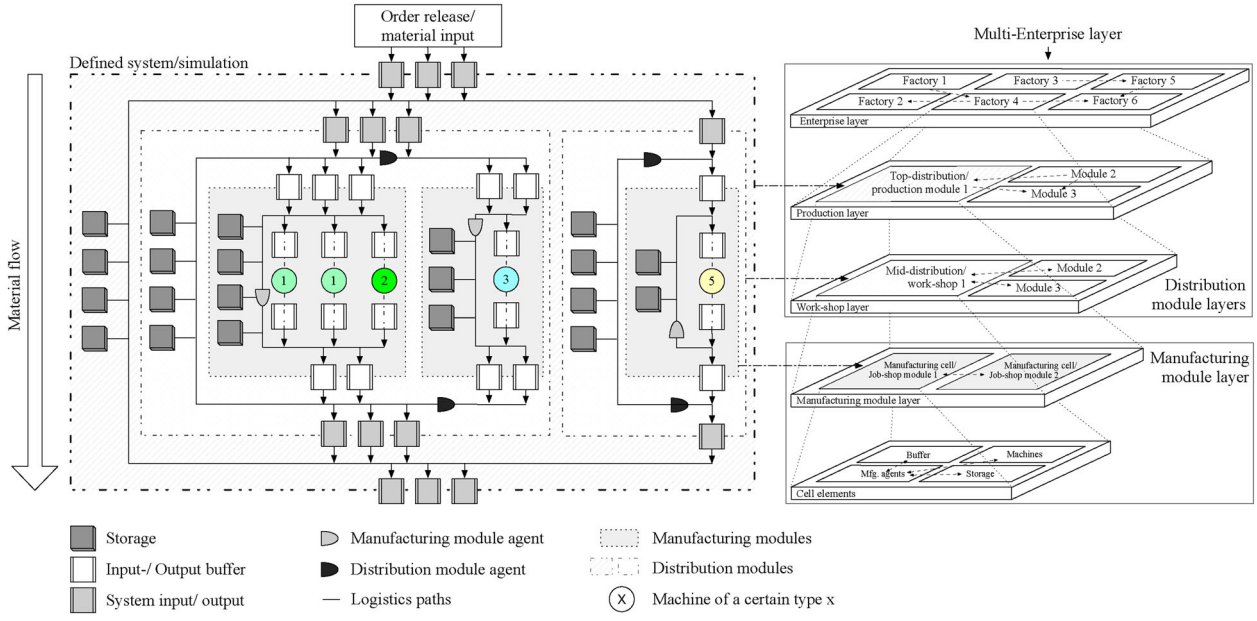

Figure 3. Projected multi-agent and semi-heterarchical system; right: adapted from Sallez et al. (2010).

exploration of the action space and potential policies and provides a broad database for the batch replay.

Illustration of our modular production system organised into hierarchical layers, with manufacturing layer/shop-floor at the bottom and distribution layers for logistics activities at higher levels. Each layer comprises multiple interconnected modules, following a semi-heterarchical organisation.

3.2. Variable hyper-heuristic design

The development of a hyper-heuristic involves multiple steps that address the third step of the *DSRM* methodology (Peffers et al. 2007). Following problem identification and objective definition, the hyper-heuristic must be designed in compliance with system constraints, available information, and the performance indicators that require optimisation. To enable an adaptive and online decision-making, and maintain the accessibility of the *CoBra* and *SimPy* simulation framework, *TensorFlow* was used for implementing deep learning functionalities. Based on that, the state vector design is first dealt with (see Section 3.2.1), that represents the current production state and also the system's interface that allows the hyper-heuristic to access the essential information for the decision making process. Second, in Section 3.2.2, lower-level heuristics are identified and the action space vector is constructed that addresses specific key performance criteria. Finally, in Section 3.2.3, the training and reward mechanisms are considered, which have an crucial impact on the learning process and system performance.

3.2.1. State space design

The state space design is an crucial step towards achieving an efficient and performant production control and should correlate to the targeted rewards and overall objectives (Kuhnle et al. 2020). The challenge is to select a state-set that contains all the essential information while avoiding the inclusion of unnecessary inputs. This may include general order information such as priorities, as well as process-related information pertaining to order throughput times or current tardiness. Machine information, including their operational status, maintenance needs, and current setup, could also be taken into account for specific scenarios if necessary. The state vector in our approach integrates buffers, storage, machine and agent order information, such as occupation type, along with associated order details, granting the specific agent's access to a comprehensive module state and global processing information, thereby facilitating situation-dependent decision-making.

Following other approaches (Kuhnle et al. 2021; Overbeck et al. 2021), we apply a *min-max-normalisation* for the various state inputs to scale the gathered values within a predefined and constrained range. This should leverage the performance of the neural network and not only enables a smoother mapping between states and actions while mitigating outliers and disproportionate input variables.

For time-related state inputs, the normalisation results in a state value range of $[-1, 1]$ for each order n out of all processable orders o within the respective module, as denoted in Equation (6). A processable order is defined by its feasibility of having one or more

possible subsequent steps, i.e. when the input buffer of the machine with the next needed processing step is unblocked, or when the order can be transported from module input to an available storage slot. Concerning the part of Equation (6), $s_{n,tpt}$ is calculated individually for each order concerning their global systemic and local module start time. For the discrete state space of order priorities, the state inputs are discretised to $[0, 1]$, signifying an input of $s_{n,prio} = 0$ for normal orders, 0.5 for prioritised orders, and 1 for high-priority orders. This is intended to ensure more stable gradients, faster training, and correct weight initialisation.

$$
\begin{aligned}
[1] \quad & s_{n,tpt} = (1 - 2\,\frac{t_{tpt,\,max} - t_{tpt,\,n}}{t_{tpt,\,max} - t_{tpt,\,min}}) \\
[2] \quad & s_{n,due\_to} = (2\,\frac{t_{dt,\,max} - t_{dt,\,n}}{t_{dt,\,max} - t_{dt,\,min}} - 1) \\
[3] \quad & s_{n,prio} = \begin{cases} 0 & if\ prio_n = 0 \\ 0.5 & if\ prio_n = 1,\ (prioritized\ order) \\ 1 & if\ prio_n = 2,\ (high\ priority\ order) \end{cases}
\end{aligned} \tag{6}
$$

The resulting state vector S_t, which is used as the input for the neural networks in subsequent processes, was derived from iterative testing and consistent mapping to targeted production performance indicators, which encompass the order throughput time, tardiness, and order priorities. To achieve this, the due date, local and global start time, and priority of all processable orders o at each available slot are concatenated, as outlined in Equation (7). For orders that are blocked or reserved by other agents, undergoing processing by a machine, or situated in input buffers, the state input for each metric ($s_{tpt,\_due\_to,\_prio}$) is assigned a value of 0 to maintain a constant state size. Additional module positions in Equation (7) correspond to those depicted in Figure 3.

$$
\begin{aligned}
S_t = (\overbrace{\underbrace{1,\ 0.8,\ -0.2,}_{\substack{Cell\ input \\ buffer\ slots}}\ \ldots,\ \underbrace{0,\ 0,\ -0.5,}_{Storage\ slots}}^{S_{tpt\ local}} \\
\underbrace{\qquad\qquad}_{Further\ module\ positions} \\
+ S_{tpt\ global} + S_{due\ to} + S_{prio})
\end{aligned} \tag{7}
$$

3.2.2. Action space design

The action space design refers to the process of defining the set of possible actions that the deep RL agent can take at any given state and defines the manufacturing sequence. With the generic optimisation approach according to Kanervisto, Scheller, and Hautamaki (2020), the objective is not a maximum number of actions, but a discretization of the action space and the selection of genuinely necessary actions. The former is given by the set of dispatching rules as control heuristics and the linking of each with a corresponding deep RL action. Dispatching rules can be deployed as low-level heuristics as they provide a quick and efficient selection of the next job to be processed, thereby reducing overall processing time and increasing the overall efficiency of the production process. Even though there are various dispatching rules available, some might be less effective, since they do not affect the desired performance parameter. Nevertheless, the idea of providing a wide range of production strategies can make it easier to tailor subsequent production scenarios and its specific optimisation problem and constraints.

The selection of low-level dispatching rules is a crucial step before the training and optimisation procedure and results in a representative rule-set that was derived from benchmarks and related approaches (Bergmann, Stelzer, and Strassburger 2014; Kaban, Othman, and Rohmah 2012; May et al. 2021; Tay and Binh Ho 2007). Also, due to the pre-defined set of dispatching rules, the action space does not increase with large layout sizes and there is no need to introduce masked actions for learning as the logic is mapped intrinsically. In the further course, we apply the local and global first-in-first-out (*FiFo*), shortest processing time (*SPT*), earliest due date (*EDD*) and highest priority (*HP*) dispatching rules as the low-level rule-set. The local and global *FiFo* rule determine the next order, out of the processable order set o, based on their local or global processing start time. The *SPT* rule selects an order based on the time required to complete the remaining process step, which particularly beneficial in resource-constrained scenarios. The *EDD* rule selects an order based on its due date, to meet tardiness objectives and the *HP* rule selects orders with a higher priority first.

3.2.3. Reward function design

The reward function is a fundamental component of the deep RL algorithm as it provides a scalar feedback signal to the agent, guiding its behaviour towards maximising the cumulative reward (Sutton and Barto 2017). In the context of hyper-heuristics based production control and multi-objective optimisation, the reward function is utilised to evaluate the performance of different low-level heuristics and guide the agent's selection towards a situation–specific and optimal control policy. To capture the desired performance criteria is crucial for the task completion and significantly affects system dynamics.

Based on the optimisation criteria of throughput time, tardiness, and respective order priorities and urgencies, we derived a combined reward function that can be transferred to other scenarios. For this purpose, the total

reward $R_{total} = \sum R_i$ for the chosen order n is composed of the mentioned optimisation criteria i according to Equation (8). Normalizing the total return proved to be negative in the later tests. Although an optimal total return can be calculated for each state, it fluctuates and is complex to interpolate in between.

$$R_{total} = R_{tpt\ local} + R_{tpt\ global} + R_{due\ to} + R_{prio} \tag{8}$$

The rewards for the throughput time $R_{TPT\ local}$ and $R_{TPT\ global}$ are a measure of how long it takes for an order to be completed from start to finish and can be used to prioritise policies that support faster completion times within a single module or the whole system. The tardiness-related reward, $R_{due\ to}$, reflects the delay in completing an order, incentivizing the algorithm to prioritise solutions that minimise delays and achieve earlier completion times. Priority related rewards R_{prio} are used to assign different levels of relevance to the orders to prioritise orders with a higher priority orders first. The deep RL algorithm takes the rewards and penalties that are outlined in Equation (9) to adjust its decision-making process and improve the performance over time. To emphasise positive and negative actions, we normalised and raised the evaluation parameters in the respective range to calculate the reward with respect to the specific maximum reward $R_{dt}, R_{tpt}, R_{prio,1/2}$.

$$[1]\quad R_{n,tpt} = (1 - 2\,\frac{t_{tpt,\,max} - t_{tpt,\,n}}{t_{tpt,\,max} - t_{tpt,\,min}})^5 * R_{tpt}$$

$$[2]\quad R_{n,due\ to} = (2\,\frac{t_{dt,\,max} - t_{dt,\,n}}{t_{dt,\,max} - t_{dt,\,min}} - 1)^5 * R_{dt}$$

$$[3]\quad R_{n,prio} = \begin{cases} 0 & if\ prio_n = 0 \\ R_{prio,1} & if\ prio_n = 1,\ (prioritized\ order) \\ R_{prio,2} & if\ prio_n = 2,\ (high\ priority\ order) \end{cases} \tag{9}$$

The variable R_{tpt} represents throughput time and can be defined separately for global and local measures. Additionally, R_{prio} has a constraint that ensures its values are greater than zero, specifically $0 < R_{prio,1} < R_{prio,2}$. The testings under consideration involve rewards for orders and utilise the values $R_{dt} = 100$, and for both local and global $R_{tpt} = 100$. The rewards were iteratively determined through a sensitivity analysis and can be modified depending on the objectives. In the case that a higher-priority order is selected, it will receive a reward of either $R_{prio,1} = 100$ or a significantly increased $R_{prio,2} = 500$. Conversely, selecting a regular order when a high-priority order is available, will result in a penalty.

4. Demonstration

In accordance with the *DSRM* demonstration step (Peffers et al. 2007), we will present the simulation pre-requisites in Section 4.1) and analyse its performance regarding the fulfilment of the pre-defined optimisation performance criteria and conduct benchmarks against conventional heuristics in Section 4.2.2. The results will facilitate making in-depth conclusions about the potential benefits and limitations of using deep RL as a top-level policy in multi-agent and multi-objective production control. The computations were carried out on an Intel Core i9-12900k CPU and 32GB of RAM. For each performance indicator, several simulations were run to obtain a representative set for training and benchmarking purposes.

4.1. Experimental settings

The simulation approach involves a *CoBra* model to emulate the behaviour of the agents within the manufacturing system. The simulated system contains 3 layers for distributing orders and respective materials within the manufacturing system to fulfil machining steps at the distributed resources. The top distribution module $D1$ resembles the high-level control module, overseeing the operation of the mid-layer modules and system in- and output. Below, there are 2 mid-layer distribution modules $D1.1/D1.2$ that control the flow of goods between the underlying manufacturing modules. Each manufacturing module possesses specific process capabilities, and there are 2 manufacturing mid-layer modules as illustrated in Figure 4 ($M1.1.1/2$; $M1.2.1/2$). The distances within and between the modules are determined based on 1 m base heights and widths, with a distance between modules of 1 m and a safe distance of 0.4 m for the agents, respectively autonomous vehicles. The outer dimensions of the high-level distribution module are 13 m in width and 6 m in height. The speed of the agents is set to 0.5 m per second, with a pick-up and unloading time of 0.1 min.

The conducted simulations indicate that in the case of less complex modules, which i.e. contain a single machine and few add-on positions (as $M1.2.1$ or $M1.2.2$), a heuristic control mechanism is similar in effectiveness compared to utilising a dedicated hyper-heuristic. Conversely, in cellular systems with larger state spaces and more complex dynamics and interactions, the deep RL approach proves to be superior. As a result, a hybrid approach, which employs both, heuristics and deep learning-based hyper-heuristic was implemented. Due to the system modularity, this approach allows to leverage the advantages of both control approaches. For the smart modules, varying input layers were incorporated according to the module state space. The distribution agents in $D1$, $D1.1$ and $D1.2$ deploy 88, 84 and 52 neurons for the input layer, and agents in $M1.1.1$ and $M1.1.2$ deploy 80 and 60 neurons. In total, 20 neural networks

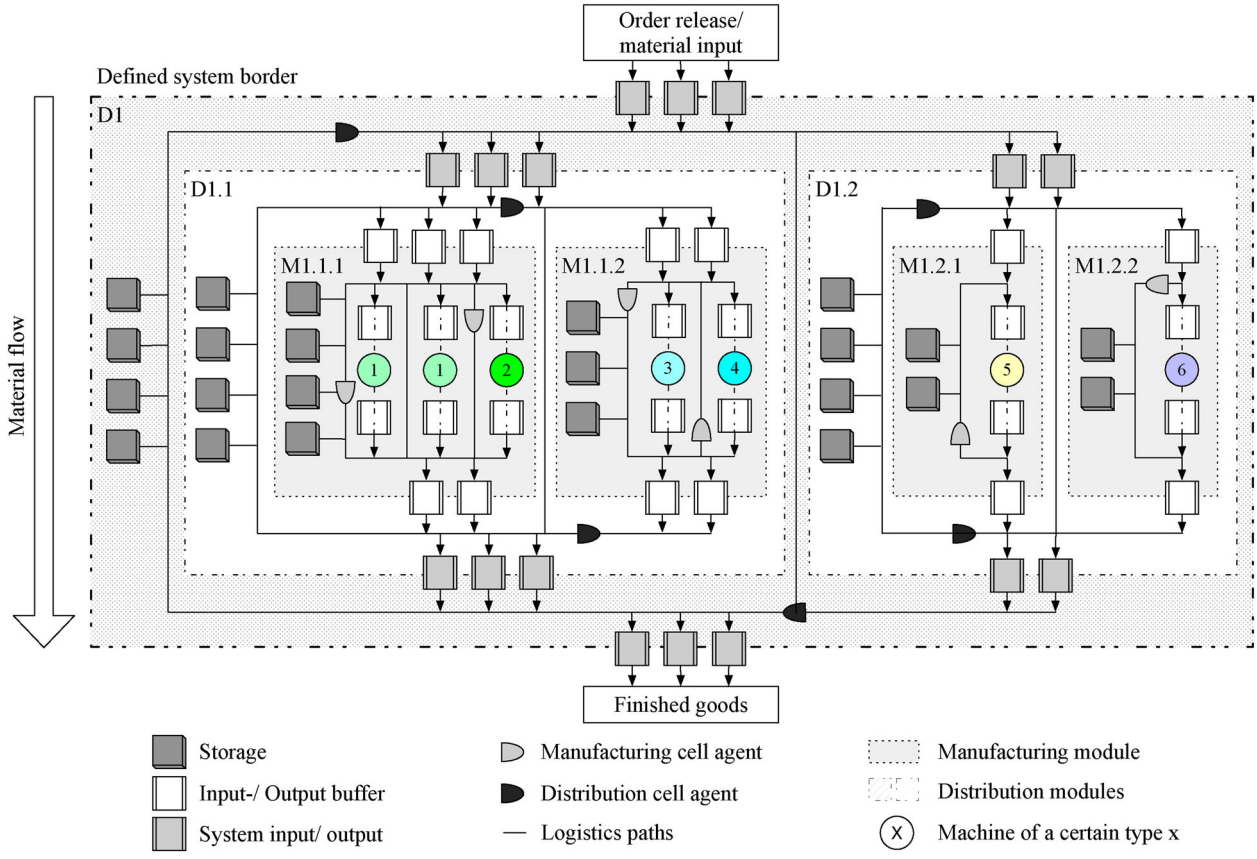

Figure 4. Simulated three-layer modular production system

Table 2. Parameter settings for the deep RL agents.

Parameter	Value
Batch size	128
Neurons in hidden layers	128/64
Learning rate α	0.005
Discount factor γ	0.98
Drop out ratio	0.01
Target update step	5
Minimum ϵ	0.01
ϵ-decay	0.997

are trained for the 10 deep learning-based agents, with each agent deploying one online and one target network. The agents in the manufacturing modules *M*1.2.1 and *M*1.2.2 are controlled by a *FiFo* rule, following its benchmarking results in Balaji and Srinivasan (2010) or Kaban, Othman, and Rohmah (2012). The iteratively optimised algorithmic parameters were initially related to similar approaches (as in Gankin et al. 2021) and are listed in Table 2. A batch size of 128 was deployed as a balance between training performance and computational efficiency. Batch sizes smaller than 128 resulted in decreased performance, especially for rush and high-priority orders, due to limited exploitation of solution spaces and prioritisation. Conversely, larger batch sizes resulted in significantly increased training times.

The simulation seeks to represent a real production scenario with stochastically varying urgency and priority of orders, where the number of orders processed depends on the pre-defined system load. The order sequence is randomly determined based on the order frequency (see Table 3, left). The order urgency is reflected in the due to time, with 20% of orders designated as rush orders, receiving a due to time $T_{dt,n}$, which is the sum of the order release time $T_{release,n}$, the predicted processing time $T_{proc,n}$, assuming a low system load (see Table 3, right), and a load-dependent factor T_{load}. The remaining orders (80%) are classified as standard orders, with an additional random distributed time buffer T_{buffer} between 30 and 60 min. Orders are further categorised as high-priority (10%, i.e. for a highly valuable customer), prioritised (15%), or standard orders (75%).

The simulation comprises two types of orders, steel shafts (1.) and aluminum shafts (2.). The processing steps, times, and order frequencies for each type and value-added service (e.g. labelling) are listed in Table 3, accounting for the unique properties of steel and aluminum, such as their strength, weight, and durability, that affect processing times. The value-added services, including labelling, coating, and packaging, can increase throughput time through additional steps. Process steps progress from left to right and are completed at the machine with the corresponding number in brackets, as pointed out in Figure 4. The time for a machine tool-set

Table 3. Order type specifications with associated processing times [min.] and sequences.

Order type		Order frequency	Processing time (at machine)			Avg. throughput time w. transportation (low-load)
			Milling	Grinding	Value-added-service	
Steel shaft	1.1	13%	8 (1)	4 (2)	–	18.9
	1.2	34%	8 (1)	4 (2)	Labelling/ packaging: 4 (6)	26.6
Aluminium shaft	2.1	13%	3 (3)	3 (4)	–	13.8
	2.2	40%	6 (3)	6 (4)	Coating/ packaging: 4 (5)	21.5

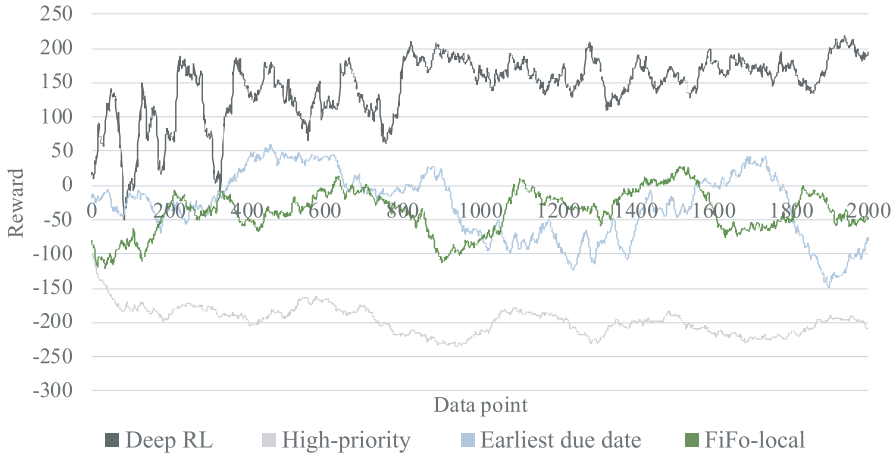

Figure 5. Moving average of obtained rewards for the top-layer *D*1 agent

exchange between product groups and the time required to load and release an item is 0.5 minutes each. Machine failures occur at an average rate of 4 per 1000 min and have a stochastic duration with a repair time ranging from 10 to 20 min.

4.2. Experimental results

In Section 4.2.1, a high-load scenario was adapted during the training process with a maximum system congestion, which increases task complexity of order selection and allows the mapping of a substantial number of state-action pairs for later operation. This implies an infinite number of planned orders and immediate order release to the input buffer throughout the whole training process. This scenario presents challenges to learn effective strategies to maintain control performance and efficiency in terms of order allocation and system management. In Section 4.2.2, a normal load scenario was adapted that reflects a real processing equilibrium for benchmarking purposes. In Section 4.2.3, the scalability and robustness of our approach are trial to ensure stability and efficiency during operation, thereby confirming its reliability for real-world applications. For conducting the performance analysis, the mean tardiness ($T_{td,mean} = \frac{1}{n}\,\Sigma_{i=1}^{n} max(0, C_i - d_i)$), and the mean global throughput time ($T_{tpt,mean} = \frac{1}{n}\,\Sigma_{i=1}^{n} T_{tpt,i}$), for all n orders, were considered. A particular emphasis is put on the interconnected order indicators, regarding their priority and urgency.

4.2.1. Training process

Incorporating the comparative analysis of the hyper-heuristic approach against conventional heuristics within the training process, as depicted in Figure 5, we observe that the hyper-heuristic outperforms these traditional rules in terms of the received rewards. Despite of the initial random rule selection at the beginning of the training phase (with $\epsilon = 1$, see left side in Figure 5), the hyper-heuristic demonstrates performance levels close to those of traditional dispatching rules at the start of the training. It maintains a functional policy that complies with process requirements which results in a working policy right from the start of training, without the need for action masking or other acceleration mechanisms. As the training process progresses, the optimisation criteria are increasingly satisfied, leading to continuous learning and performance enhancements. This trend highlights the progressive performance convergence of the approach.

Upon completion of the training, the hyper-heuristic achieved a moving average score of 200, a significant improvement over conventional heuristics such as FiFo local (-50), EDD (-78), and high priority rule (-207). This demonstrates the superiority of the deep learning-based rule selection approach within the hyper-heuristic framework, which has significantly exploited its optimisation potential compared to conventional heuristics.

Figure 6 illustrates a throughput time related performance analysis of the hyper-heuristic, which is crucial for evaluating the true effectiveness of our approach. The

Figure 6. Moving average of throughput times related to order priorities

results indicate a significant decrease in throughput time as the simulation progresses, which can be attributed to higher utilisation rates despite the capacity limitations that restrict the simultaneous processing of orders at a machine. Regarding standard priority orders, the throughput time decreases by 43% and 51%, 59% for prioritised and high-priority orders. In contrast to the conventional rules, a noticeable decrease in fluctuations is observed with an increasing number of episodes, indicating that the deep RL algorithm is trending towards its optimal policy. Additionally, the analysis indicates that higher-priority orders have significantly faster throughput times compared to standard orders. Specifically, high-priority orders have a throughput time of 74.1 min, representing a 32.6% lower throughput time compared to standard orders, while prioritised orders exhibit a throughput time of 89.3 min, corresponding to an 18.7% decrease with a throughput time of 109.9 min. The green dotted line indicates the average Work-In-Progress (WIP) level. Despite a slight increase in throughput times as WIP levels rise, the robustness of control and optimisation is evident with the occurring WIP peaks in Figure 6.

4.2.2. Benchmark results

In this Section, we conducted a benchmarking scenario that encompassed a simulation range of 7200 min, corresponding to three 8-hour shifts over a period of 5 days. The order amount of 2800 was iteratively determined to be near the system equilibrium. Unlike the training mode, in which batch replay is required, the trained agents are now applied in an operational mode. To facilitate comprehensive analysis, we included a random rule in addition to the average benchmarks, which are summarised in Table 4. The left column of the table presents the hyper-heuristics results, while the rightmost column lists comparison indicators of the hyper-heuristic approach with the average of the individual dispatching rules (improved values are highlighted in green, worsened in red). The inclusion of the random rule allows for a more thorough evaluation of the other rules performances.

The results demonstrate that the summarised order values in the upper part were all improved using our approach. The mean tardiness was reduced by nearly 39.5% and the throughput was increased by 1.4%. In the individual benchmark comparison, the throughput time is comparable to the local *FiFo* rule, but priority-related indicators are improved. The order priority-related performance indicators are listed in the lower part for standard (0), prioritised (1) and high-priority (2) orders. With our approach, high priority orders were delivered with 60.8% reduced tardiness, and with a shorter throughput time. In contrast, both *FiFo* rules, as the most commonly used ones, performed worse regarding total order indicators compared to the proposed hyper-heuristic.

However, it was expected that rules that are specifically designed to optimise a particular indicator would perform better in that specific case. For instance, the *SPT* rule indicates shorter throughput times and the lowest work in progress levels, but it produced 78 fewer orders due to the blocking of buffers and storages by orders with additional process steps that are scheduled with higher throughput times. This led to a considerably high tardiness for these orders of 21.5 min. Consequently, all orders that comprised value-added services had to wait for orders with fewer remaining processing time. The mean tardiness was lowest with the *EDD* rule, which optimises based on order due dates. However, none of the conventional dispatching rules were performing in combined measures and multi-objective view.

In the following, we compare the hyper-heuristic approach with individual dispatching rules in relation to the order priorities. In this regard, the hyper-heuristic

Table 4. Multi-objective optimization benchmark incorporating order priorities

		Hyper-heuristic			FiFo local			FiFo global			Earliest due date first			Shortest processing time first			Highest priority first			Random (excl. in avg. benchmark)			Hyper-heuristic against avg. benchmark		
Total order set	Total throughput [#]	2734			2722			2709			2718			2656			2678			2645			1.4%		
	Throughput time [min.]	86.2			91.7			88.8			88.4			83.0			79.9			87.8			−0.2%		
	Mean tardiness [min.]	8.9			12.0			10.7			8.5			21.5			20.8			22.3			−39.5%		
	Work-in- progress [#]	28.3			29.5			30.2			29.7			27.0			28.9			29.9			−2.6%		
Priority related order set	Order priority	**0**	**1**	**2**	**0**	**1**	**2**	**0**	**1**	**2**	**0**	**1**	**2**	**0**	**1**	**2**	**0**	**1**	**2**	**0**	**1**	**2**	0	1	2
	Total throughput [#]	2066	402	266	2059	401	263	2049	399	261	2055	399	263	2012	389	254	2021	398	259	2005	387	253	1.3%	1.2%	2.3%
	Throughput time [min.]	88.3	83.6	73.3	91.1	93.9	93.1	88.4	89.6	90.5	88.1	90.3	88.0	81.4	85.1	93.2	91.5	44.5	44.1	87.6	85.9	87.8	0.2%	3.6%	−10.4%
	Mean tardiness [min.]	10.0	6.2	5.0	11.8	12.7	12.5	10.6	10.6	10.7	8.4	8.8	8.9	20.0	23.1	31.4	28.0	0.2	0.3	22.4	19.7	25.6	−36.5%	−44.0%	−60.8%
	Work-in- progress [#]	22.2	3.8	2.3	22.7	4.1	2.8	23.2	4.1	2.9	22.8	4.1	2.7	20.6	3.6	2.8	25.2	2.2	1.5	23.0	3.9	3.0	−3.1%	5.0%	−9.4%

outperformed the *FiFo*, *EDD*, and *SPT* rules, particularly for prioritised and high-priority orders. Specifically, for prioritised and high-priority orders, the hyper-heuristic reduced throughput by almost 7% to 83.6 min, and 20% 73.3 min, respectively, compared to the average time of the previously mentioned rules. Individually, the *HP* rule performed best for prioritised and high-priority orders, but had a 13.5% higher throughput time for standard orders than our approach, which account for 75% of the total order-set. Although the higher-priority orders are completed with a minimum mean tardiness with the *HP* rule, this leads to longer processing times for the much larger set of standard orders. Despite the hyper-heuristic's slightly poorer performance in the prioritised order class, it still outperformed the other rules from a multi-objective perspective. Additionally, the hyper-heuristic produced 2.7% more high-priority orders than the *HP* rule.

For a further analysis, we included Table 5 which demonstrates the relationship between order urgencies and priorities for the optimisation of throughput times. The table includes a comparison of the corresponding throughput times with a high-priority and rush order as the base value in the lower section. As previous results indicate, the global *FiFo* and *random* rules were less performant compared to other the benchmarks, which is why they are excluded in further analysis.

One notable finding is the significant increase in throughput times of the hyper-heuristic (Table 5, left) for low prioritised and non-rush orders. The development is progressive, with a clear 22.8% increase from the highest to the lowest urgency and priority class. This suggests that non time-critical orders are processed at higher throughput times. However, due to the high priority relevance within the reward function, the increase is observable, but rather small with 2.1%. On the other hand, the *EDD* rule, as a single-criterion rule, only optimises one criterion, which is the due date, equally for all priorities. Similarly, the *HP* rules also indicates a similar pattern for priorities, in which rush orders were processed 3% or 1% slower. The hyper-heuristic approach appears to be the more effective in optimising combined urgency and priority measures. While the *EDD* and *HP* rules may be suitable for constrained optimisation, they may not perform well in stand-alone operation.

4.2.3. Analysis of optimisation robustness and scaleability

In addition to analysing the performance of our approach, we also evaluated its robustness and scalability. It is essential to ensure that the approach can operate efficiently and reliable, even with increased order volumes, to guarantee long-term effectiveness. Previously, we demonstrated the control efficiency of our approach with increasing WIP levels in Figure 6. Now, we evaluate the robustness of our approach by measuring the rewards received during operation, which serve as an indicator of how well the desired production objectives were fulfilled. As illustrated in Figure 7, the received rewards of each dispatching rule were analysed, with the random rule serving as the lower benchmark. The rewards of conventional rules exhibit significant fluctuations in both, short-term and long-term rewards. Despite occasional near-parity with the hyper-heuristic, on average, conventional rules were less robust than the hyper-heuristic in fulfilling the multiple defined optimisation objectives.

Table 6 summarises the statistical values obtained from our robustness analysis. It can be observed that the hyper-heuristic approach achieved good performance in terms of robustness, which is in line with our earlier findings. However, the performance of the *HP* rule was surprisingly poor, which can be explained by the predefined rewards. Although its policy of favouring higher prioritised orders was followed, the larger amount of neglected standard orders led to a corresponding deterioration in performance. This trend is reflected in the decreasing reward trend, as the tardiness of the orders in the modules continued to increase over time (as indicated in Figures 5/7) which results in poor tardiness and throughput values and respective negative rewards. This highlights the importance of considering combined objectives, as neglecting standard orders, although being less valuable per order, can have a detrimental effect on optimisation objectives. Therefore, it is crucial to find a balance between prioritising high-priority orders and ensuring that standard orders are also processed efficiently.

The scalability of our approach is supported by the assumption of decentralised decision-making and the small state spaces of the neural networks that are used for action calculation. Due to the modular layout, also the overall task complexity was broken down among the agents which could thereby deploy compact neural networks. As a result, all agent computations were carried out in real-time, taking less than 0.01 seconds. We distinguish between the cases of pure training and the change of production organisation during operation. Thereby, adding a new manufacturing module does not require the re-training of the entire system, as it would be the case with a central control instance. Instead, only the affected module and its overlying distribution module would need to be trained. Furthermore, if an identical manufacturing module is added, the logic can be learned via transfer learning, which reduces the training time to just training the overlying distribution module. Although training our scenario for 10,000 simulated

Table 5. Order urgency and priority dependent optimisation of throughput times [min.].

	Hyper-heuristic		FiFo local		EDD		HP	
Total orders	Rush order	Standard order	Rush order	Standard order	Rush order	Standard order	Rush order	Standard order
High priority	72.1	73.6	90.0	90.7	59.1	96.3	45.1	43.9
Prioritized	72.5	86.2	86.2	90.4	56.1	98.6	44.0	44.6
Standard	87.1	88.6	87.5	88.7	57.5	96	93.8	90.9
In relation to high-priority and rush orders								
High priority	1	2.1%	1	0.8%	1	62.8%	1	-3%
Prioritized	0.6%	19.5%	-4.2%	0.4%	-5.2%	66.8%	-2%	-1%
Standard	20.8%	22.8%	-2.7%	-1.4%	-2.8%	62.6%	108%	101%

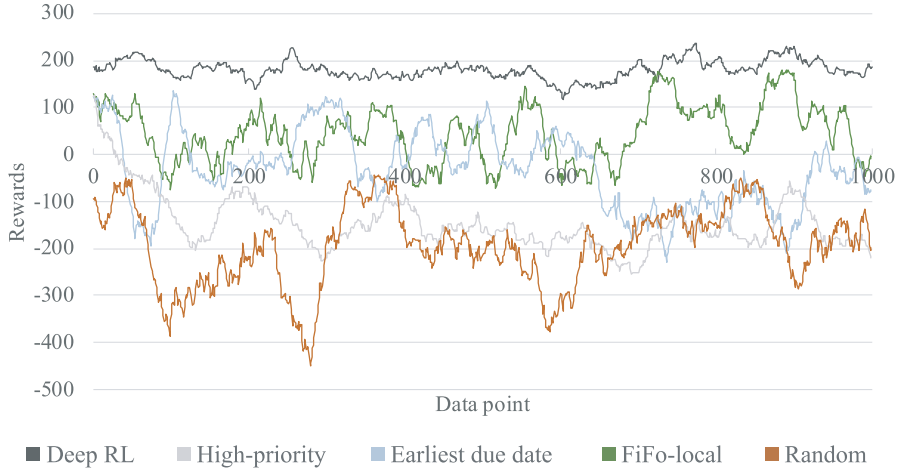

Figure 7. Moving average of agent rewards for the D1 distribution module.

minutes required approximately 36 h (due to the iterative calculation), re-training a new network for a new module takes only a fraction of this time. Re-training the *D*1.2 layer (see Figure 4) after an additional manufacturing module was added took only about 4 h. From an organisational perspective, the scalability of our is not dependent on the complexity of the layout, as it can be broken down into sub-modules. Thus, our approach can be extended to more complex systems without requiring a complete overhaul of the existing training model. This scalability feature, combined with the real-time processing capabilities, leads to an suitable option for transferring it into a real production system.

5. Simulation to reality transfer

The transfer of the evolved and simulated approach was further deployed within the *Center for Industry 4.0* and validated using its modular manufacturing system. Figure 8 contains the real production environment (1), the updated simulation model (2), and the corresponding layout (3). The manufacturing cells are projected into the available machine cubes (numbers 1–5), and the input and output buffers are located within these. The real layout is similar to the simulated layout, with an additional robot manufacturing cell (see cell 4) and an additional storage slot in the distribution cell to use the full storage capacity of the real environment (see number 8). For the transport of an order on the distribution level, the load carrier cubes are used as autonomous agents (9). If such a cube arrives at one of the machine cubes (1–5), an order is placed or picked up there and can then be transported to the next machine cube or to the system output. In the system input (6) and output (7), orders are generated (including the buffer capacity) and collected or deposited by the autonomous load carriers. The transfer of deep learning-based instructions from the simulated agents to the real-world logistics elements was conducted by implementing the simulated system's logic into the workload carriers. The instructions of the neural network were processed by the *FabOS* (factory operating system, Lass and Gronau (2020)). The action-set and state-vector remained consistent, with only the destinations being modified.

Since the agents are the bottleneck of this simulation due to their low speed, we based the performance evaluation on a fixed period of 5 h. The path of the load carriers from one location to another is determined by the *FabOS* to avoid collisions and blockages. Due to time restrictions, the operation was carried out on the basis of previously trained agents. Control decisions were made after examining the status of new orders, after arriving at

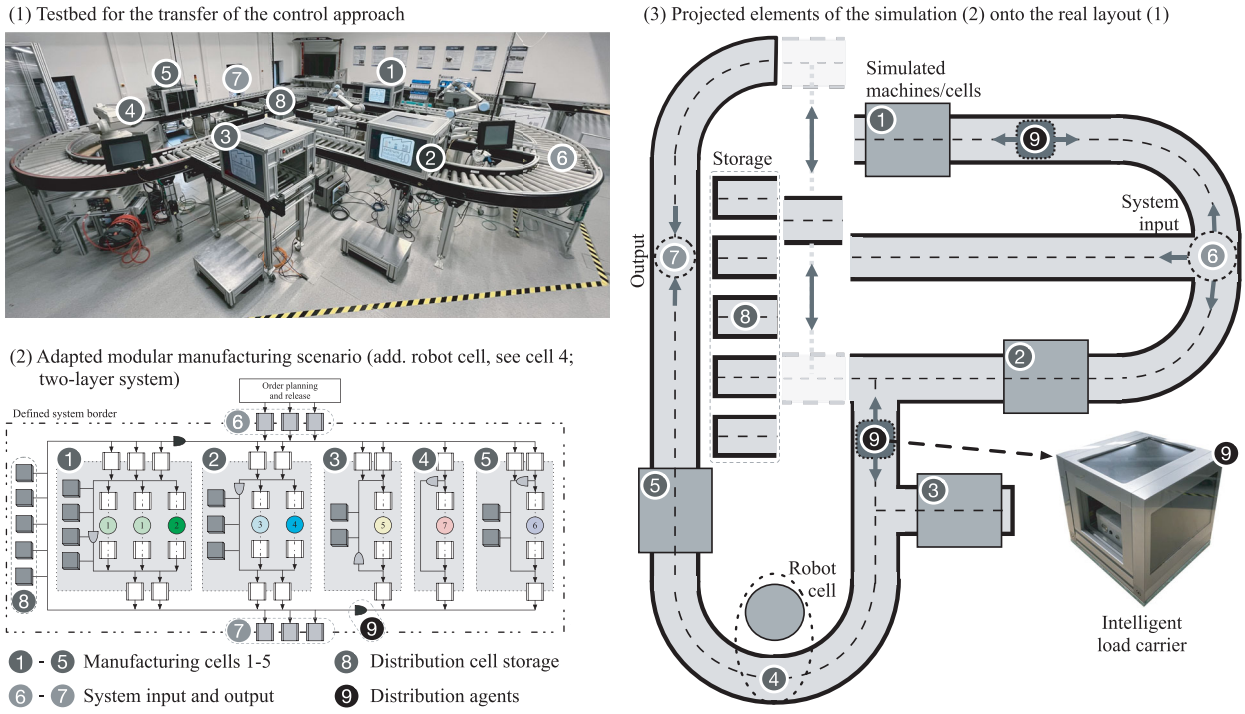

Figure 8. Testing setup of the hyper-heuristic within the hybrid production environment.

Table 6. Summary of reward mean and standard deviation.

	Hyper-heuristic	FiFo local	Earliest due date	High-priority	Random
Arithmetic mean [-]	178.8	49.2	−24.1	−156.9	−193.2
Standard deviation [-]	21,2	60,9	85,5	59,8	84,6

a destination, and after any changes in the system, when the agents had no assigned task. The results of the real test are listed in Table 7 and are compared against the *FiFo local* dispatching rule, due to its performance, but also its wide-spread use in real production systems. The best values are indicated in bold letters.

Although there was a slight decrease of one unit in throughput when compared to simulation performance, it is important to note that this may be attributed to the rather short duration of the conducted testing. Conversely, there was a 1.7% reduction in throughput time while maintaining a comparable level of tardiness for the total order-set. Additionally, higher-priority orders were processed more frequently and with an average tardiness that was 90% lower than that of lower-priority orders.

6. Discussion

Today's production systems must cope with increasingly demanding customer requirements, shorter product and development cycles and short-term fluctuations in demand. One approach to address these challenges in production control is deep RL as a data-driven optimisation tool, which differs from other machine learning methods mainly in its online adaptability and real-time processing of sensor data. In our approach, we have demonstrated the superior performance of deep RL compared to conventional rules that are still in widespread use. We also transferred our approach into a hybrid production environment, highlighting is real-world capabilities. It becomes clear that deep RL can master the link between input states, optimisation goal and the derivation of necessary actions and can help to achieve individual production goals. A multi-level system with distributed production resources was considered, which is composed of agents with different tasks. This is intended to cover different industry backgrounds and maximise transferability to specific applications.

Managerial insights

The increasing connectivity of future factories and the growing complexity of products and processes require accelerated corporate adaptation cycles, especially in manufacturing. To meet these challenges, companies are well-advised to consider more sophisticated approaches to increase flexibility, mitigate process risks, and maximise production robustness. However, in addition to

Table 7. Performance benchmark within the hybrid production environment.

	Hyper-heuristic				FiFo local				Comparison (FiFo as base)			
Scope/ order priority	Total	0	1	2	Total	0	1	2	Total	0	1	2
Total throughput [#]	121	92	17	12	122	95	16	11	−0.8%	−3.2%	6.3%	9.1%
Throughput time [min.]	56,9	59,4	55,4	51,5	57,9	56,8	67,6	53,6	−1.7%	4.6%	−18.0%	−3.9%
Mean tardiness [min.]	9,8	12,8	1,6	1,1	9,8	9,4	13,9	14,1	0.0%	36.2%	−88.5%	−92.2%

sustaining processes, it is also important to fully exploit competitiveness through the deliberate use of new algorithmic approaches and organisational capabilities. By using hyper heuristics, companies can limit their dependence on scarce human capital and proactively use data-driven operations to reduce costly manual processes. In contrast to conventional approaches, which can only react to changing conditions to a limited extent, the modular framework presented in our approach has a significantly higher transferability and can be adapted according to market requirements. In addition, the defined objectives were achieved in a combined manner more effectively, which increases cash flow and can reduce conversion costs. Furthermore, additional services such as rush orders and prioritised orders were integrated, which not only enable additional cash flow, but also integrate customer-oriented services on the shop floor.

7. Conclusion

This paper presents a novel hyper-heuristic based control approach for modular designed holonic production systems. Holons were modelled as autonomous manufacturing entities, providing high adaptability in a semi-heterarchical structure. Unlike previous multi-agent approaches that were limited to a single operational level or that implemented confined deep learning techniques, we deployed dispatching rules for facilitating a deep learning-based performance optimisation. Each agent within the production system had its own control policy and shared experiences with agents within the same cell. The differentiated contemplation of manufacturing agents at the shop-floor layer and the use of distribution agents within the upper transport layers was emphasised.

The control approach targeted several parameters for optimisation, including global throughput time, adherence to due dates, and the processing of rush and prioritised orders. Simulations and real-world scenarios demonstrate the superiority of the hyper-heuristic approach in multi-objective optimisation of average throughput time, total throughput, and tardiness. Prioritized and rush orders, which often emerge in practice, were processed with better performance than with dedicated dispatching rules. Likewise, not only were more orders processed within the real environment, but the existing rule-set was outperformed with regard to the defined performance indicators.

The hybrid and decentralised hyper-heuristic control approach integrates both, deep learning based and conventional elements, to facilitate a scenario-specific process optimisation. Whereas the holonic approach allows for an easy adoption of new resources and processes, the hyper-heuristic prevents the selection of misleading actions by leveraging the rule-set integrated process logic. It resembles a self-configuring system that automatically adapts to changing production conditions without human intervention and leverages system scaleability. The addition of a cell does not necessitate re-training of the entire system which leverages utilisation efficiency. The distributed resources further avoid the need for large state and action spaces, as the modules and associated state or action parameters have a pre-defined scope.

Future research should focus on reducing the training effort required for comparable modules through the use of parameter learning strategies and the freeze and transfer of network layers to differing cells for faster scenario adoption. Moreover, to better adapt to varying environmental conditions, agents should be assigned specific action spaces based on cell objectives and levels, which can be kept variable, allowing different strategies to be executed. This will allow for an hybrid operational model, avoiding the prevalent pre-training in a digital twin, which further leverages production performances and adaptability. Future research should also target bridging the gap between the real-world and simulated environments, which will ultimately reduce operational barriers.

Data availability statement

The authors confirm that the data supporting the findings of this study are available within the article and its supplementary materials.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Notes on the contributors

Marcel Panzer, M.Sc., is a mechanical engineering graduate from the Karlsruhe Institute of Technology with extensive industrial experience and a track record of working on various transfer projects. Currently, he holds the position of research assistant at the University of Potsdam, Chair of Business Informatics, esp. Process and Systems since 2020. Additionally, he serves as the head of the Center for Industry 4.0, which aims to bridge the gap between research and industry by transferring novel technologies into real production environments. Marcel's research primarily focuses on production planning and control, leveraging deep learning-based technologies to improve manufacturing processes.

Benedict Bender is a Post-Doc Lecturer who studied business informatics at the University of Potsdam, the Humboldt University of Berlin as well as the University of St. Gallen. His research interests include aspects of IT security and privacy as well as digital platforms and business ecosystems. He is actively involved in collaborating with industry partners to bridge the gap between academia and real-world applications.

Norbert Gronau is a Professor for Business Information Systems at the University of Potsdam. He studied mechanical engineering and business administration at the Technical University of Berlin (TU). In 1994, he received his doctorate in the Department of Computer Science (TU). Up to March 2000, he was head of the teaching and research group Production-Oriented Business Information Systems at the TU Berlin. He was head of the Department of Business Information Systems at the University of Oldenburg from 2000 to 2004. Since 2004 he holds the chair of Business Informatics, Processes and Systems.

ORCID

Marcel Panzer http://orcid.org/0000-0003-4099-0179

References

Baer, Schirin, Danielle Turner, Punit Mohanty, Vladimir Samsonov, Romuald Bakakeu, and Tobias Meisen. 2020. "Multi Agent Deep Q-Network Approach for Online Job Shop Scheduling in Flexible Manufacturing." In *International Conference on Manufacturing System and Multiple Machines*, Tokyo: Japan.

Bahrpeyma, Fouad, and Dirk Reichelt. 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.

Baker, Albert D. 1998. "A Survey of Factory Control Algorithms that Can Be Implemented in a Multi-Agent Heterarchy: Dispatching, Scheduling, and Pull." *Journal of Manufacturing Systems* 17 (4): 297–320. https://doi.org/10.1016/S0278-6125(98)80077-0.

Balaji, P. G., and D. Srinivasan. 2010. "An Introduction to Multi-Agent Systems." In *Innovations in Multi-Agent Systems and Applications – 1*, edited by J. Kacprzyk, D. Srinivasan, and L. C. Jain, Vol. 310, 1–27. Berlin, Heidelberg: Springer Berlin Heidelberg.

Bellman, Richard. 1957. "A Markovian Decision Process." *Indiana University Mathematics Journal* 6 (5): 679–684. https://doi.org/10.1512/iumj.1957.6.56038.

Bergmann, S., S. Stelzer, and S. Strassburger. 2014. "On the Use of Artificial Neural Networks in Simulation-Based Manufacturing Control." *Journal of Simulation* 8 (1): 76–90. https://doi.org/10.1057/jos.2013.6.

Bongaerts, Luc, László Monostori, Duncan McFarlane, and Botond Kádár. 2000. "Hierarchy in Distributed Shop Floor Control." *Computers in Industry* 43 (2): 123–137. https://doi.org/10.1016/S0166-3615(00)00062-2.

Borangiu, Theodor, Pascal Gilbert, Nick-Andrei Ivanescu, and Andrei Rosu. 2009. "An Implementing Framework for Holonic Manufacturing Control with Multiple Robot-Vision Stations." *Engineering Applications of Artificial Intelligence* 22 (4-5): 505–521. https://doi.org/10.1016/j.engappai.2009.03.001.

Borangiu, Theodor, Silviu Râileanu, Damien Trentesaux, and Thierry Berger. 2010. "Semi-Heterarchical Agile Control Architecture with Intelligent Product-Driven Scheduling." *IFAC Proceedings Volumes* 43 (4): 108–113. https://doi.org/10.3182/20100701-2-PT-4011.00020.

Bueno, Adauto, Moacir Godinho Filho, and Alejandro G. Frank. 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.

Burke, Edmund K., Matthew R. Hyde, G. Kendall, G. Ochoa, E. Özcan, and John R. Woodward. 2010. "A Classification of Hyper-Heuristic Approaches." In *Handbook of Metaheuristics*, edited by M. Gendreau and J.-Y. Potvin, Vol. 272, 453–477. Cham: Springer International Publishing.

Burke, Edmund K., Matthew R. Hyde, G. Kendall, G. Ochoa, E. Özcan, and John R. Woodward. 2019. "A Classification of Hyper-Heuristic Approaches: Revisited." In *Handbook of Metaheuristics*, edited by Michel Gendreau and Jean-Yves Potvin, Vol. 272, 453–477. Cham: Springer International Publishing.

Chang, Jingru, D. Yu, Z. Zhou, W. He, and L. Zhang. 2022. "Hierarchical Reinforcement Learning for Multi-Objective Real-Time Flexible Scheduling in a Smart Shop Floor." *Machines* 10 (12): 1195. https://doi.org/10.3390/machines10121195.

Cowling, Peter, Graham Kendall, and Eric Soubeiga. 2001. "A Hyperheuristic Approach to Scheduling a Sales Summit." In *Practice and Theory of Automated Timetabling III*, edited by G. Goos, J. Hartmanis, J. van Leeuwen, E. Burke, and W. Erben, Vol. 2079, 176–190. Berlin, Heidelberg: Springer Berlin Heidelberg.

de Paula Ferreira, William,Fabiano Armellini, and Luis Antonio De Santa-Eulalia. 2020. "Simulation in Industry 4.0: A State-of-the-Art Review." *Computers & Industrial Engineering* 149:106868. https://doi.org/10.1016/j.cie.2020.106868.

de Paula Ferreira, William, Fabiano Armellini, Luis Antonio de Santa-Eulalia, and Vincent Thomasset-Laperrière. 2022.

"A Framework for Identifying and Analysing Industry 4.0 Scenarios." *Journal of Manufacturing Systems* 65:192–207. https://doi.org/10.1016/j.jmsy.2022.09.002.

Dittrich, M.-A., and S. Fohlmeister. 2020. "Cooperative Multi-Agent System for Production Control Using Reinforcement Learning." *CIRP Annals* 69 (1): 389–392. https://doi.org/10.1016/j.cirp.2020.04.005.

Drake, John H., Ahmed Kheiri, Ender Özcan, and Edmund K. Burke. 2020. "Recent Advances in Selection Hyper-Heuristics." *European Journal of Operational Research* 285 (2): 405–428. https://doi.org/10.1016/j.ejor.2019.07.073.

Esteso, Ana, David Peidro, Josefa Mula, and Manuel Díaz-Madroñero. 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.

Gankin, Dennis, Sebastian Mayer, Jonas Zinn, Birgit Vogel-Heuser, and Christian Endisch. 2021. "Modular Production Control with Multi-Agent Deep Q-Learning." In *26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Vasteras, Sweden, Sep., 1–8. IEEE. https://doi.org/10.1080/00207543.2022.2104180.

Giret, Adriana, and Vicente Botti. 2004. "Holons and Agents." *Journal of Intelligent Manufacturing* 15 (5): 645–659. https://doi.org/10.1023/B:JIMS.0000037714.56201.a3.

Grassi, Andrea, Guido Guizzi, Liberatina Carmela Santillo, and Silvestro Vespoli. 2020. "A Semi-Heterarchical Production Control Architecture for Industry 4.0-Based Manufacturing Systems." *Manufacturing Letters* 24:43–46. https://doi.org/10.1016/j.mfglet.2020.03.007.

Grassi, Andrea, Guido Guizzi, Liberatina Carmela Santillo, and Silvestro Vespoli. 2021. "Assessing the Performances of a Novel Decentralised Scheduling Approach in Industry 4.0 and Cloud Manufacturing Contexts." *International Journal of Production Research* 59 (20): 6034–6053. https://doi.org/10.1080/00207543.2020.1799105.

Greschke, P., M. Schönemann, S. Thiede, and C. Herrmann. 2014. "Matrix Structures for High Volumes and Flexibility in Production Systems." *Procedia CIRP* 17:160–165. https://doi.org/10.1016/j.procir.2014.02.040.

Groover, Mikell P. 2019. *Automation, Production Systems, and Computer-Integrated Manufacturing*. 5th ed. Hudson Street, New York: Pearson Education.

Gros, Timo P., Joschka Gros, and Verena Wolf. 2020. "Real-Time Decision Making for a Car Manufacturing Process Using Deep Reinforcement Learning." In *Winter Simulation Conference (WSC)*, Orlando, FL, USA, Dec., 3032–3044. IEEE. https://doi.org/10.1109/WSC48552.2020.9383884.

Hammami, Zeineb, Wiem Mouelhi, and Lamjed Ben Said. 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.

Herrera, Manuel, Marco Pérez-Hernández, Ajith Kumar Parlikad, and Joaquín Izquierdo. 2020. "Multi-Agent Systems and Complex Networks: Review and Applications in Systems Engineering." *Processes* 8 (3): 312. https://doi.org/10.3390/pr8030312.

Hofmann, Constantin, Carmen Krahe, Nicole Stricker, and Gisela Lanza. 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.

Kaban, A. K., Z. Othman, and D. S. Rohmah. 2012. "Comparison of Dispatching Rules in Job-Shop Scheduling Problem Using Simulation: A Case Study." *International Journal of Simulation Modelling* 11 (3): 129–140. https://doi.org/10.2507/IJSIMM11(3)2.201.

Kallestad, Jakob, Ramin Hasibi, Ahmad Hemmati, and Kenneth Sörensen. 2023. "A General Deep Reinforcement Learning Hyperheuristic Framework for Solving Combinatorial Optimization Problems." *European Journal of Operational Research* 309 (1): 446–468. https://doi.org/10.1016/j.ejor.2023.01.017.

Kanervisto, Anssi, Christian Scheller, and Ville Hautamaki. 2020. "Action Space Shaping in Deep Reinforcement Learning." In *IEEE Conference on Games (CoG)*, Osaka, Japan, Aug., 479–486. IEEE. https://doi.org/10.1109/CoG47356.2020.9231687.

Kapoor, Kawaljeet, Ali Ziaee Bigdeli, Yogesh K. Dwivedi, and Ramakrishnan Raman. 2021. "How is COVID-19 Altering the Manufacturing Landscape? A Literature Review of Imminent Challenges and Management Interventions." *Annals of Operations Research* 1:1–33. https://doi.org/10.1007/s10479-021-04397-2.

Kuhnle, Andreas, Marvin Carl May, Louis Schäfer, and Gisela Lanza. 2021. "Explainable Reinforcement Learning in Production Control of Job Shop Manufacturing System." *International Journal of Production Research* 1–23. https://doi.org/10.1080/00207543.2021.1972179.

Kuhnle, Andreas, Jan-Philipp Kaiser, Felix Theiß, Nicole Stricker, and Gisela Lanza. 2020. "Designing an Adaptive Production Control System Using Reinforcement Learning." *Journal of Intelligent Manufacturing* 32 (3): 855–876. https://doi.org/10.1007/s10845-020-01612-y.

Lass, Sander, and Norbert Gronau. 2020. "A Factory Operating System for Extending Existing Factories to Industry 4.0." *Computers in Industry* 115:103128. https://doi.org/10.1016/j.compind.2019.103128.

Lee, Jay, Behrad Bagheri, and Chao Jin. 2016. "Introduction to Cyber Manufacturing." *Manufacturing Letters* 8:11–15. https://doi.org/10.1016/j.mfglet.2016.05.002.

Lee, Jay, Behrad Bagheri, and Hung-An Kao. 2015. "A Cyber-Physical Systems Architecture for Industry 4.0-Based Manufacturing Systems." *Manufacturing Letters* 3:18–23. https://doi.org/10.1016/j.mfglet.2014.12.001.

Lee, Jay, Hossein Davari, Jaskaran Singh, and Vibhor Pandhare. 2018. "Industrial Artificial Intelligence for Industry 4.0-Based Manufacturing Systems." *Manufacturing Letters* 18:20–23. https://doi.org/10.1016/j.mfglet.2018.09.002.

Liu, C., C. Chang, and C. Tseng. 2020. "Actor-Critic Deep Reinforcement Learning for Solving Job Shop Scheduling Problems." *IEEE Access* 8:71752–71762. https://doi.org/10.1109/ACCESS.2020.2987820.

Liu, Renke, Rajesh Piplani, and Carlos Toro. 2022. "Deep Reinforcement Learning for Dynamic Scheduling of a Flexible Job Shop." *International Journal of Production Research* 60 (13): 4049–4069. https://doi.org/10.1080/00207543.2022.2058432.

Mönch, Lars, John Fowler, and Scott J. Mason. 2013. *Production Planning and Control for Semiconductor Wafer Fabrication Facilities: Modeling, Analysis, and Systems*. Operations Research/Computer Science Interfaces Series, Vol. 52. New York: Springer. OCLC: ocn794710214.

Malus, Andreja, D. Kozjek, and R. Vrabič. 2020. "Real-Time Order Dispatching for a Fleet of Autonomous Mobile Robots Using Multi-Agent Reinforcement Learning." *CIRP Annals* 69 (1): 397–400. https://doi.org/10.1016/j.cirp.2020.04.001.

May, Marvin Carl, Lars Kiefer, Andreas Kuhnle, Nicole Stricker, and Gisela Lanza. 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.

Mayer, Sebastian, Tobias Classen, and Christian Endisch. 2021. "Modular Production Control Using Deep Reinforcement Learning: Proximal Policy Optimization." *Journal of Intelligent Manufacturing* 32 (8): 2335–2351. https://doi.org/10.1007/s10845-021-01778-z.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. "Playing Atari with Deep Reinforcement Learning." arXiv:1312.5602.

Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, et al. 2015. "Human-Level Control Through Deep Reinforcement Learning." *Nature* 518 (7540): 529–533. https://doi.org/10.1038/nature14236.

Mourtzis, Dimitris. 2020. "Simulation in the Design and Operation of Manufacturing Systems: State of the Art and New Trends." *International Journal of Production Research* 58 (7): 1927–1949. https://doi.org/10.1080/00207543.2019.1636321.

Overbeck, Leonard, Adrien Hugues, Marvin Carl May, Andreas Kuhnle, and Gisela Lanza. 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.

Panzer, Marcel, and Benedict Bender. 2022. "Deep Reinforcement Learning in Production Systems: A Systematic Literature Review." *International Journal of Production Research* 60 (13): 4316–4341. https://doi.org/10.1080/00207543.2021.1973138.

Panzer, Marcel, B. Bender, and N. Gronau. 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.

Parente, Manuel, Gonçalo Figueira, Pedro Amorim, and Alexandra Marques. 2020. "Production Scheduling in the Context of Industry 4.0: Review and Trends." *International Journal of Production Research* 58 (17): 5401–5431. https://doi.org/10.1080/00207543.2020.1718794.

Parunak, H. Van Dyke, John F. White, P. W. Lozo, R. Judd, B. W. Irish, and J. Kindrick. 1986. "An Architecture for Heuristic Factory Control." In *American Control Conference*, Seattle, WA, USA, June, 548–558. IEEE. https://doi.org/10.23919/ACC.1986.4789001.

Peffers, Ken, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee. 2007. "A Design Science Research Methodology for Information Systems Research." *Journal of Management Information Systems* 24 (3): 45–77. https://doi.org/10.2753/MIS0742-1222240302.

Ritterbusch, Georg David, and Malte Rolf Teichmann. 2023. "Defining the Metaverse: A Systematic Literature Review." *IEEE Access* 11:12368–12377. https://doi.org/10.1109/ACCESS.2023.3241809.

Sallez, Y., T. Berger, S. Raileanu, S. Chaabane, and D. Trentesaux. 2010. "Semi-heterarchical Control of FMS: From Theory to Application." *Engineering Applications of Artificial Intelligence* 23 (8): 1314–1326. https://doi.org/10.1016/j.engappai.2010.06.013.

Samsonov, Vladimir, Marco Kemmerling, Maren Paegert, Daniel Lütticke, Frederick Sauermann, Andreas Gützlaff, Günther Schuh, and Tobias Meisen. 2021. "Manufacturing Control in Job Shop Environments with Reinforcement Learning." In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence*, Online Streaming, – Select a Country –, 589–597. SCITEPRESS – Science and Technology Publications. https://doi.org/10.5220/0010202405890597.

Schenk, M., S. Wirth, and E. Muller. 2010. *Factory Planning Manual: Situation-Driven Production Facility Planning*. Berlin, London, New York: Springer. OCLC: ocn428029418.

Schmidtke, Niels, A. Rettmann, and F. Behrendt. 2021. "Matrix Production Systems – Requirements and Influences on Logistics Planning for Decentralized Production Structures." https://doi.org/10.24251/HICSS.2021.201.

Schulman, John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. "Proximal Policy Optimization Algorithms." *arXiv*. https://doi.org/10.48550/arXiv.1707.06347.

Sutton, Richard S., and Andrew G. Barto. 2017. *Reinforcement Learning: An Introduction*. 2nd ed., Adaptive Computation and Machine Learning Series. Cambridge, Massachusetts: The MIT Press.

Swiercz, A. 2017. "Hyper–Heuristics and Metaheuristics for Selected Bio–Inspired Combinatorial Optimization Problems." In *Heuristics and Hyper-Heuristics – Principles and Applications*, edited by J. Del Ser Lorente. InTech. https://doi.org/10.5772/intechopen.69225.

Tamás, Bányai. 2021. "Optimization of Material Supply in Smart Manufacturing Environment: A Metaheuristic Approach for Matrix Production." *Machines* 9 (10): 220. https://doi.org/10.3390/machines9100220.

Tampuu, Ardi, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. 2017. "Multiagent Cooperation and Competition with Deep Reinforcement Learning." *PloS One* 12 (4): e0172395. https://doi.org/10.1371/journal.pone.0172395.

Tao, Fei, Q. Qi, A. Liu, and A. Kusiak. 2018. "Data-Driven Smart Manufacturing." *Journal of Manufacturing Systems* 48:157–169. https://doi.org/10.1016/j.jmsy.2018.01.006.

Tay, Joc Cing, and Nhu Binh Ho. 2007. "Designing Dispatching Rules to Minimize Total Tardiness." In *Evolutionary Scheduling*, edited by Janusz Kacprzyk, Keshav P. Dahal, Kay Chen Tan, and Peter I. Cowling, Vol. 49, 101–124. Berlin, Heidelberg: Springer Berlin Heidelberg.

Valckenaers, P., F. Bonneville, H. Van Brussel, L. Bongaerts, and J. Wyns. 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*, Troy, NY, USA, 128–133. IEEE. https://doi.org/10.1109/CIMAT.1994.389083.

Van Ekeris, Tilo, Richard Meyes, and Tobias Meisen. 2021. "Discovering Heuristics and Metaheuristics for Job Shop Scheduling from Scratch Via Deep Reinforcement Learning." https://doi.org/10.15488/11231.

Waschneck, Bernd, A. Reichstaller, L. Belzner, T. Altenmuller, T. Bauernhansl, A. Knapp, and A. Kyek. 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.

Weiss, G. ed. 2001. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. 3rd ed. Cambridge, MA: MIT Press.

Zambrano Rey, G., C. Pach, N. Aissani, A. Bekrar, T. Berger, and D. Trentesaux. 2013. "The Control of Myopic Behavior in Semi-Heterarchical Production Systems: A Holonic Framework." *Engineering Applications of Artificial Intelligence* 26 (2): 800–817. https://doi.org/10.1016/j.engappai.2012.08.011.

Zhang, Yuchang, R. Bai, R. Qu, C. Tu, and J. Jin. 2022. "A Deep Reinforcement Learning Based Hyper-Heuristic for Combinatorial Optimisation with Uncertainties." *European Journal of Operational Research* 300 (2): 418–427. https://doi.org/10.1016/j.ejor.2021.10.032.