information

Article

Production Scheduling Based on a Multi-Agent System and Digital Twin: A Bicycle Industry Case

Vasilis Siatras, Emmanouil Bakopoulos, Panagiotis Mavrothalassitis, Nikolaos Nikolakis and Kosmas Alexopoulos *

Laboratory for Manufacturing Systems and Automation, Department of Mechanical Engineering and Aeronautics, University of Patras, 26504 Patras, Greece; siatras@lms.mech.upatras.gr (V.S.); bakopoulos@lms.mech.upatras.gr (E.B.); mavrothalassitis@lms.mech.upatras.gr (P.M.); nikolakis@lms.mech.upatras.gr (N.N.)
* Correspondence: alexokos@lms.mech.upatras.gr; Tel.: +30-2610-910160

Abstract: The emerging digitalization in today's industrial environments allows manufacturers to store online knowledge about production and use it to make better informed management decisions. This paper proposes a multi-agent framework enhanced with digital twin (DT) for production scheduling and optimization. Decentralized scheduling agents interact to efficiently manage the work allocation in different segments of production. A DT is used to evaluate the performance of different scheduling decisions and to avoid potential risks and bottlenecks. Production managers can supervise the system's decision-making processes and manually regulate them online. The multi-agent system (MAS) uses asset administration shells (AASs) for data modelling and communication, enabling interoperability and scalability. The framework was deployed and tested in an industrial pilot coming from the bicycle production industry, optimizing and controlling the short-term production schedule of the different departments. The evaluation resulted in a higher production rate, thus achieving higher production volume in a shorter time span. Managers were also able to coordinate schedules from different departments in a dynamic way and achieve early bottleneck detection.

Keywords: production scheduling; multi-agent system; digital twin; asset administration shell; deep reinforcement learning; mathematical programming; heuristic optimization

check for updates

Citation: Siatras, V.; Bakopoulos, E.; Mavrothalassitis, P.; Nikolakis, N.; Alexopoulos, K. Production Scheduling Based on a Multi-Agent System and Digital Twin: A Bicycle Industry Case. *Information* **2024**, *15*, 337. https://doi.org/10.3390/info15060337

Academic Editor: Katsuhide Fujita

Received: 30 April 2024
Revised: 31 May 2024
Accepted: 4 June 2024
Published: 6 June 2024

1. Introduction

Bicycle manufacturing can be characterized by three distinctive production stages to deliver the final product, starting from certain customer demand and raw materials. The preparation and painting of bike components is the first stage in which semi-final products from the bill of material (BOM) are produced. In parallel, the assembly of wheels is performed to receive the complete BOM and proceed with the final stage, the bike assembly. While manufacturers plan for the mid- to long-term production horizon, line managers aim to deliver the planning objectives by making effective short-term scheduling decisions for the different stages [1]. These decisions affect the performance of the production in the short run and can achieve optimal material flow between the different stages to eliminate disruptions and bottlenecks. Unfortunately for the managers, the complexity of the individual schedules driven by the high level of detail required in sequencing, and combining the items entering the production lines, define a difficult optimization problem, for which existing Enterprise Resources Planning (ERP) software cannot provide an effective solution. As such, manufacturers rely on making empirical decisions which can result in slower reaction time and the insufficient utilization of resources.

Particularly for the painting stage, engineers need to prepare the components through a series of operations after which color, decals, and coating are applied, before moving them in stock. This stage requires sequencing and combining the items in different groups while entering the painting line, to maximize production volume and reduce bottlenecks between

the processes. The problem is known in research as the paint-shop scheduling problem (PSSP). Similarly, for the wheel assembly lines, managers need to arrange the sequence of executing the different orders in a dynamic way, given the lines' availability and the material delivery date for the bike assembly stage. This can be expressed as a dynamic job-shop scheduling problem. Finally, managers from the bike assembly lines collect the material from the previous stages and define the best sequence of the orders entering the lines to maximize the production rate affected by the different processing rates for each product type. This stage is usually a flexible flow-shop type of environment with sequential assembly steps which may or may not have available storage capacity in between.

As a method for supporting manufacturers dealing with composite optimization problems, multi-agent system technology has been introduced in research or market solutions to break the problem into individual smaller decisions coordinated in a decentralized manner [2]. Agents are frequently empowered with artificial intelligence methods, to drive their decision-making process in an optimal way. Deep reinforcement learning (DRL) is a popular method in modern production management which trains a model upon a specific environment to make calculated actions on any given state [3]. This has been particularly useful in dynamic scheduling cases, due to its ability to make quick and accurate short-term decisions [4]. Mathematical optimization and heuristic/metaheuristic algorithms are also branches of artificial intelligence suitable for making near-optimal decisions in complex scheduling problems and frequently used as the core logic of a scheduling agent [5].

Through innovations in the direction of Industry 4.0, modern multi-agent systems have evolved into a distributed network of agents. Standards from the Industry 4.0 paradigm start to be integrated in the communication of the different digital entities, including agents and physical assets [6]. Autonomous optimization and control, based on multi-agent technology, has had a great impact on industrial applications, leading research into the development of several frameworks to establish the methods for these entities to effectively operate and negotiate [7]. The administration shell concept is utilized to manage their behavior and achieve negotiations in a standardized manner. This level of industrial digitalization allows digital twin technology to analyze production performance in real time and project potential implications in the near future [8]. Engineers can benefit from such an integrated framework as it allows them to receive online decision support and assess potential risks on the run.

In the case of bicycle manufacturing industry, the scheduling of the day-to-day operations has a major effect on the daily production volumes and cross-department coordination for the avoidance of material overflows. Manufacturers have developed empirical policies and methods for optimizing the daily task sequence and bridge the gap between available planning software and detailed production scheduling. Nevertheless, as valuable as human experience is for the organization, manual scheduling lacks reactiveness and usually provides medium-quality solutions. The accumulation of such decisions in the long run impacts the overall profit of the organization and gradually increases production costs. The main objective of this study is to develop and evaluate a bicycle production scheduling framework that will support line managers in coordinating daily production and resolving unnecessary delays or bottlenecks.

The proposed framework encapsulates three different AI scheduling agents, capable of solving the three different segments of the bicycle production phase. Agents are deployed within a multi-agent system environment and guided by the asset administration shell technology, thus structuring and communicating information using standardized APIs and models. This behavior enables the system's scalability and empowers the dynamic deployment of additional scheduling methods. Moreover, capturing dynamic production information gives the opportunity of deploying digital twin models of the production environment as a way of evaluating and projecting the performance of the different scheduling decisions. Such an integrated scheduling solution allows line managers to examine the performance of different alternatives, react to production disturbances, and effectively coordinate production stages.

2. State of the Art

The manufacturing sector has undergone significant transformation with the integration of artificial intelligence (AI), which it presents innovative approaches to enhance productivity, quality, and efficiency [1]. Its ability to address complex challenges, analyze vast datasets, and provide accurate predictions has led to its increasing adoption across manufacturing operations [9]. Various processes such as quality assurance, preventive maintenance, supply chain management, and production scheduling have benefited from AI applications [10–14]. Among the critical tasks in the industrial domain, production planning and scheduling stand out [15–17]. These activities involve the development and implementation of strategies, methods, and technologies to create effective production plans and schedules. Production planning entails determining, in advance, what needs to be done and how, while scheduling involves allocating resources or manufacturing facilities to fulfill work orders. Efficient production scheduling reduces costs, enhances productivity, and ultimately enhances customer satisfaction. AI systems, renowned for their ability to tackle complex scheduling challenges and provide accurate solutions, have gained prominence in production scheduling. Machine learning (ML) stands out as one of the most widely used AI techniques [18]. Advanced production planning and scheduling algorithms have been developed using genetic algorithms, mathematical optimization, artificial neural networks, and reinforcement learning.

RL, in particular, has gained attention for its ability to learn optimal policies without prior knowledge of schedules [19]. One well known algorithm of RL is the Q-learning algorithm that learns to estimate the value of taking a particular action a_t at time t in a specified state s_t [20]. Q-learning iteratively updates the Q-table based on experienced rewards, gained from the interaction with the environment. The agent learns the optimal policy to maximize long-term rewards without requiring a model of the environment. Furthermore, the Deep Q-Network (DQN) extends Q-learning to handle high-dimensional state spaces by employing deep neural networks to approximate Q-Values. DQN is typically applied to problems modelled as Markov Decision Processes (MDPs), where the agent interacts with an environment composed of states, actions, transition probabilities, and rewards [21]. Recent research has demonstrated successful applications of RL agents in various manufacturing scenarios, improving decision-making and optimizing resource allocation [22]. However, challenges persist in scaling solutions for larger production systems and achieving multi-objective optimization. Future efforts are focused on integrating RL with dispatch rules, deep learning, and multi-agent training to address these challenges and enhance production planning and scheduling efficiency [23,24].

Mathematical optimization is pivotal for planning and scheduling in manufacturing, enabling companies to efficiently allocate resources, sequence jobs, and create production schedules that maximize productivity and minimize costs [25,26]. For instance, optimization models have been applied to determine optimal job sequences in production lines, minimizing setup times and maximizing throughput [27]. Similarly, in supply chain management [28,29], optimization techniques are used to optimize inventory levels [30,31], transportation routes [32], and production schedules [33], ensuring the timely delivery of goods while minimizing inventory holding costs and transportation expenses. Various production scheduling optimization problems have been addressed using heuristics. For instance, a combination of constructive heuristics and iterated greedy algorithms effectively minimized makespan in the distributed blocking flow-shop scheduling problem (DBFSP) [34]. In [35], the authors propose a heuristic-based approach for the stochastic optimization of mine production schedules, employing iterative improvement through block swapping to achieve the final solution. Moreover, heuristics have proven successful in optimizing scheduling tasks with the goal of reducing total energy consumption [36]. Additionally, in [37], a hybrid heuristic algorithm is developed to tackle the precedence constrained production scheduling problem (PCPSP) in the open-pit mining industry.

Digital twin technology has garnered interest in both academia and industry, providing virtual representations of physical systems to inform decision-making [38]. Digital

twins integrate physical and digital environments, leveraging simulation to forecast system responses and optimize operations [39]. Simulation technology, a key component of digital twins, enables engineers to test and refine systems before implementation, reducing costs and improving efficiency [40]. Moreover, implementation methods support decision-making in production scheduling amid uncertainties [41]. Manufacturers have harnessed the power of machine learning (ML) techniques, including deep learning, to uncover intricate manufacturing patterns. These advancements have led to the creation of intelligent decision support systems tailored to various manufacturing activities. These tasks encompass continuous and intelligent inspections, predictive maintenance, quality enhancement, process optimization, supply chain management, and production scheduling. However, it is crucial to ensure the efficient utilization of datasets throughout the development phase of decision-making software. The digital twin effectively supports the training phase of data-driven approaches by facilitating the extraction of large datasets for training purposes [42]. Additionally, it enables more realistic validation to be conducted prior to implementation in the actual production environment. Standards such as OPC UA and the administration shell facilitate seamless information exchange between digital twin technologies and production hardware, ensuring real-time monitoring, control, and data exchange [43,44]. The administration shell concept offers a standardized framework for describing and managing assets, facilitating interoperability within the digital environment [45]. The asset administration shell supports the implementation of digital twins for Industry 4.0 and creates interoperability among assets [46,47].

The need to explore and address well-defined standards for production optimization agents is clearly revealed when there is a need for cooperation between different production agents, to formulate a multi-agent system [48]. Researchers from a variety of fields have given multi-agent systems a great deal of attention to break down complicated problems into smaller jobs [49]. In manufacturing, multi-agent systems can limit the complexity of order scheduling in production systems through a cooperative multi-agent system for production control optimization [50]. A similar approach was followed for the implementation of decentralized scheduling algorithms in a test-bed environment [51]. While multi-agent systems implementation methods have been explored in recent years, further investigation to address challenges is required. For example, the use of standards in a scheduling multi-agent system is crucial to develop systems that could be easily transformed into plug and play applications [52]. Additionally, agents that control or implement different applications and software should follow a hierarchical implementation to achieve better multi-agent system utilization and agent distribution. Lastly, if external applications are controlled through a multi-agent system functionality, standardization and interoperability are almost inevitable for smooth system operation.

The integration of multi-agent systems and digital twin technologies, along with the latest interoperability standards, is still an ongoing research topic of great interest for industrial end-users. For scheduling applications in particular, manufacturers can receive online decision support driven by digital twin calculations and be proactive about potential risks. In particular for the bicycle industry, there is still a shortage of effective adaptive scheduling solutions, which forces line managers and engineers to adapt empirical methods in order to coordinate production effectively and in a timely manner. As such, the opportunity for a multi-agent-based decision support framework, which integrates digital twin technology during scheduling and performance evaluation, shows potential for great industrial impact in production performance. This study aims to cover this gap and implement a multi-agent scheduling system in a framework which provides seamless communication with a production digital twin environment. The main objective of this work is to explore the impact of such an approach in terms of business performance and technological benefits for the user.

The implementation of the scheduling multi-agent system proposed in this work addresses these issues and provides an opportunity for a more flexible implementation of scheduling algorithms with different functionalities and heterogeneous optimization

techniques [21]. Multi-agent technologies are beneficial for planning and scheduling due to their flexibility, scalability, and parallelism [53,54]. They allow for decentralized decision-making, facilitating better adaptation to dynamic environments and enabling efficient resource allocation. In addition, they promote coordination and collaboration among agents, leading to resilient and flexible scheduling solutions that can effectively handle complex tasks [55–57].

As such, building upon previous algorithms and findings applied within the bicycle industry, this research focuses on integration and validation in a real industrial case. This involves the comprehensive revision and refinement of algorithms, models, submodels, and practical frameworks. These include the development of autonomous agents for scheduling optimization in paint shop environments, leveraging model-based and data-driven methods to improve production flowtime [27]. Additionally, an AAS-based information model is used to describe scheduling agents, facilitating interoperability and adaptability in Industry 4.0 environments [47]. As such, seamless integration with digital twin components can be established, allowing the development of deep learning and deep reinforcement learning agents in order to make adaptable online decisions for the system [42,58]. This integrated framework offers a holistic approach to addressing scheduling challenges in the bicycle industry, paving the way for enhanced operational efficiency and performance.

3. Method Description

The bicycle manufacturing phase is being divided into three primary stages, wheel assembly, the painting of semi-products, and final assembly operations. In principle, these three interdependent steps can characterize most of the bicycle production industry. The scheduling of these stages relies on coordinating the production environment in a way that the planning objectives are met, bottlenecks are avoided, and production works smoothly across the different departments. In practice, managers from the different departments are given certain production orders to execute in a given timeframe. The dependency between the different production stages poses the challenges not only of intra-department coordination, but also the coordination of the overall production for delivering orders on time. The proposed framework needs to provide individual decision support for scheduling internal department operations, but also provides awareness and adaptability to the needs of the proceeding production stages. Figure 1 illustrates the high-level architecture of the proposed framework for delivering such decision support to the user, while also achieving the granularity of the system and its interoperability with existing Industry 4.0 ecosystems.

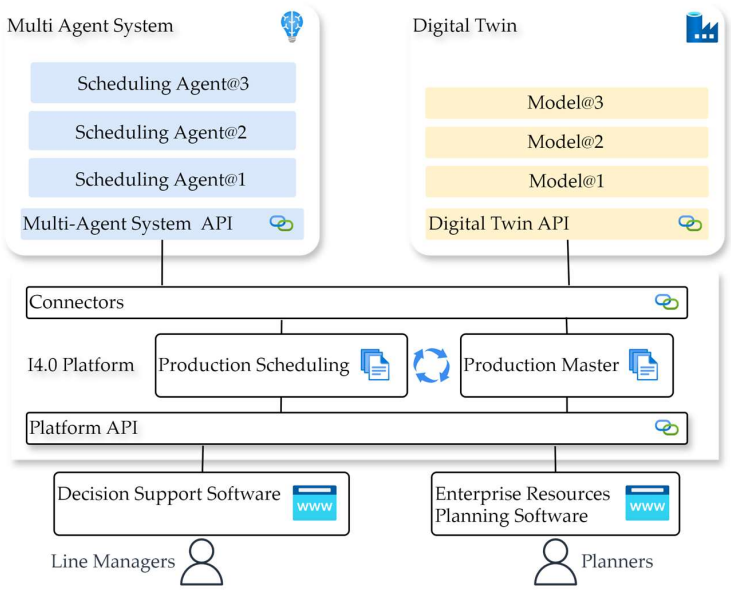

Figure 1. Integrated framework of multi-agent system, digital twin, and asset administration shell technologies.

At the top level of Figure 1, the business logic and functions of the proposed architecture are displayed. On the one hand, a multi-agent system is provided to accommodate optimization decisions and react to user demands and events. On the other hand, DT technology allows decisions to be evaluated and presented in a more holistic way to the user. In the middle level of Figure 1, the communication and information layer plays the role of middleware between the different assets/entities as well as the different levels, going from the shop floor level to the application logic and back. The use of the administration shell technology in this layer improves the interoperability of the framework and builds upon the latest I4.0 standards and interfaces. The bottom level of Figure 1 displays the integration with the shop floor environment, which is facilitated via the user interaction with the front-end application layers. In the proposed framework, the decision support software is configured by the administration shell ecosystem, thus making it reconfigurable to changes in the associated environment. In addition, a connection between certain AAS components and the associated data sources is established, configuring the system with input information to define a given production scenario.

3.1. Asset Administration Shell of Agents and Production

Interactions between the assets and external entities are facilitated via the asset administration shell concept. In this case, the assets are the multi-agent system and the production system. At first, the AAS for the multi-agent system was adapted from a previous research paper of the authors [47] upon integrating the AAS and multi-agent system technologies in a combined framework. The implementation of agent skills can thus be executed via an asset administration shell *Operation*, while also receiving events from the multi-agent system in the form of *BasicEvent* within the language of an administration shell. The production environment, on the other hand, plays the role of aggregating online production information, results from the multi-agent system, and quantitative results from the DT.

Figure 2 displays the information exchange between the different entities of the administration shell. The scheduling agent AAS enables the dynamic configuration of intelligent agents within the multi-agent system, as well as the real-time control of the agents through basic skills provided by the agent-based programming language. The reconfiguration of this entity is achieved by adding the corresponding *Operation* for a new agent/skill and implementing the corresponding services within the multi-agent system. In practice, the deployment of a new scheduling agent within the environment is configured through the following basic operations in the administration shell: "iMapping", "Optimization", and "oMapping". The input mapping component is responsible for generating agent-compatible information from any type of production-data structure that the system is implementing. The optimization module requests help from the agent upon optimizing the specific industrial scenario, which in this framework is requested by the user. In practice, the optimization sends a well-defined optimization problem to the multi-agent system for the purpose of allowing the agents to respond to the request based on their skillset. Output mapping is responsible for converting the data from agents into the initial production format. These operations allow the translation of any information model into the language of the corresponding agent implementation, and thus detach the production information model from the algorithmic/optimization part. Changing either the production information model or the agents would be easily achievable as long as new implementations for the aforementioned operations are provided to fit the mapping of the new models.

The production master AAS plays the role of the digital identity of the production, providing a real-time connection with ERP software, the description of the production system resources, and a connection with DT models for simulating the different results from the multi-agent system. What the doted lines connecting the two administration shells represent, however, is that interaction was modulated by the user, in this case the line managers, preventing the system from making unguided decisions.

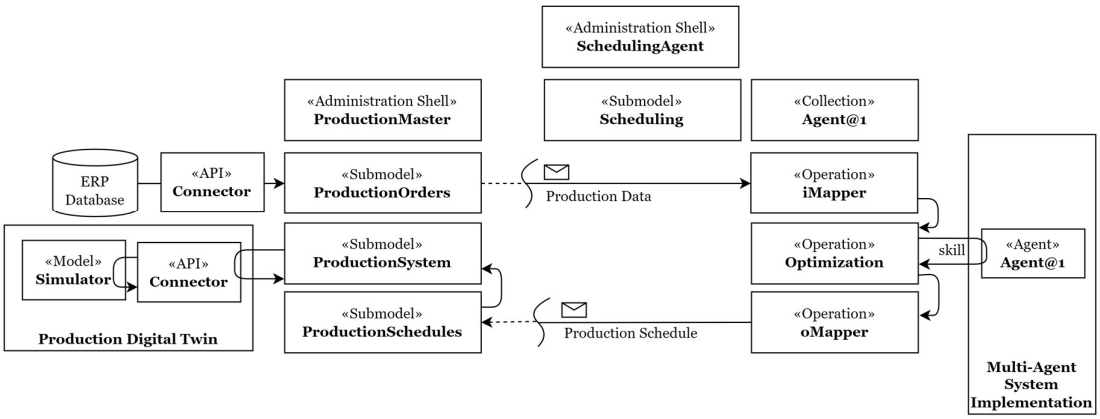

Figure 2. Administration shell components of multi-agent system and digital twin.

3.2. Digital Twin

Many scholars have used simulation to tackle a variety of production optimization and facility layout design issues [59]. To validate, test, and evaluate the proposed multi-agent system, a production digital twin module was developed. The goal is to apply the scheduling solutions for each department to a representative simulation model and calculate production performance in the given timeframe [60]. A discrete event simulation (DES) model for each department was developed. In this study, static models are utilized. These models operate on deterministic parameters and do not incorporate randomness or probability distributions for event occurrence. In contrast to stochastic models, which introduce uncertainty, the static model facilitates the precise simulation of system behavior under consistent conditions. As such, businesses can assess several alternatives to scheduling configurations and operating methods to aid in decision-making within the industrial context. Several crucial factors, such as schedule optimization and layout design, have an impact on the overall performance of manufacturing systems. To realize data interoperability and seamless architecture integration, supporting functionalities and API were developed and utilized. The architecture of the multi-agent system with the DT module can be seen in Figure 3.

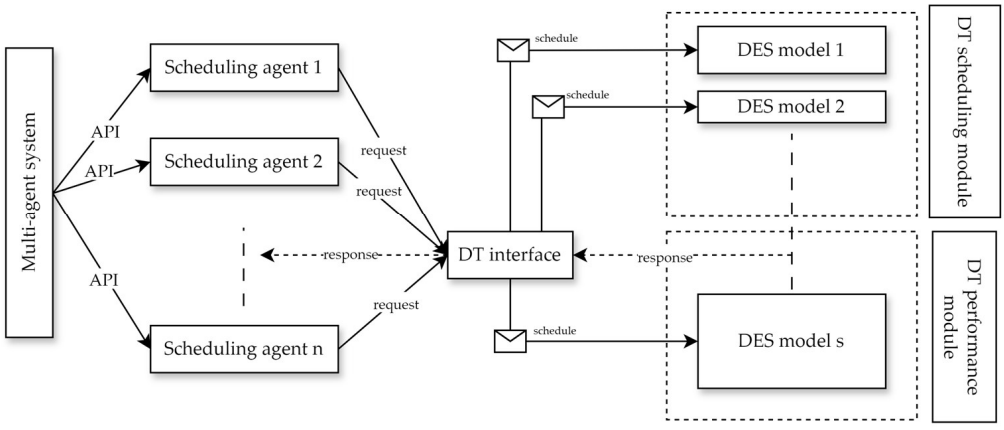

Figure 3. Multi-agent system and DT module implementation architecture.

DES and DT are distinct approaches used in modeling and simulation. DES primarily focuses on simulating complex systems by modeling discrete events and the flow of entities through the system. It is commonly employed for analyzing and optimizing processes in various domains. On the other hand, DT technology involves creating a virtual replica of a physical system, integrating real-time data to mimic its behavior. DTs are utilized for the real-time monitoring, prediction, and optimization of physical assets or processes. In this work, DES serves as a suitable DT for training and testing the DRL agent due to its ability

to accurately model the behavior of complex systems and simulate the effects of various events on system dynamics.

The multi-agent system handles the scheduling requests and reaches the scheduling agents via APIs. Then, each agent reaches the required DES model via a DT interface. There are two types of DT modules: (1) DT scheduling modules, where the simulation model is required for the scheduler operation and (2) DT performance modules, where the simulation model is used for performance observation. This distinction between these two types of DT models gives the multi-agent system the ability not only to apply full schedules, but also scheduling actions that will result to a schedule when they will be executed via the DES model. Thus, the multi-agent system and DT implementation can support static and dynamic schedulers at once, enhancing the flexibility of the proposed solution. The scheduling agents can request for a DES model to apply their whole schedule or specific scheduling actions and the DES models can respond with KPIs and production status, respectively (this process is also schematically presented in Figure 4).

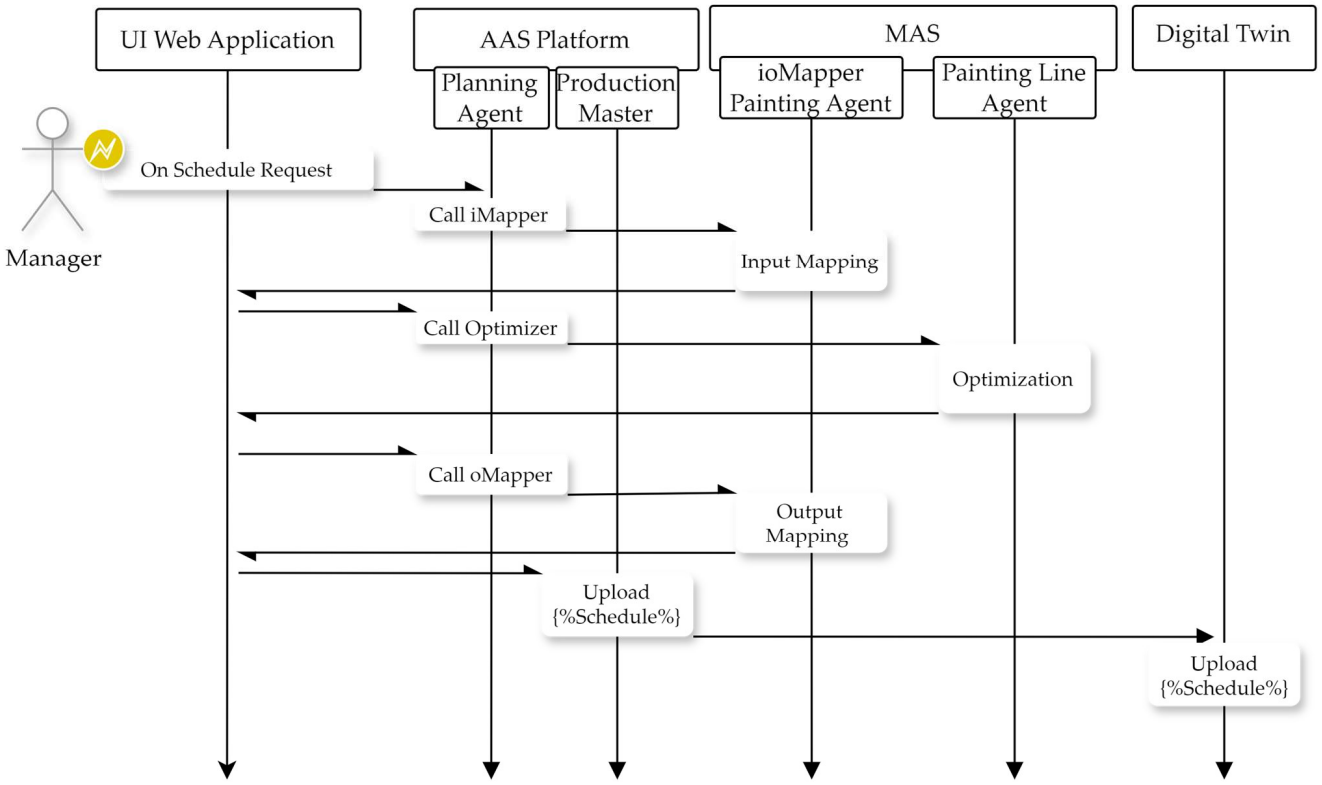

Figure 4. Flow diagram of interaction between the different modules of the framework.

4. Bicycle Industry Pilot Case

4.1. Industrial Scenario

The use case entails an Original Equipment Manufacturer (OEM) specializing in bicycles, producing various models for the market. Raw materials and components sourced from external suppliers initially pass through the painting and wheel assembly departments. These phases can operate independently and concurrently to prepare sub-components for the final product. Typically, due to the shorter duration required for wheel assembly, painting procedures follow suit for the same production orders. Once all components are in stock, the final stage involves bike assembly, where components listed in the product's BOMs are dispatched to the assembly department. The production undergoes a high-level of customization in the different orders, encompassing medium/small quantities of regular bikes, cargo bikes, and e-bikes from different Stock Keeping Units (SKUs). Sales enter the company's order system from different customers. Each sale includes the SKU type,

quantities, and delivery deadlines. Then, the company plans the production of the current sales order for a multi-week horizon.

ERP and Warehouse Management System (WMS) are used to manage the mid-to-long term plan of production and ensure that all the production's supplies are available when required. In the lower levels, however, this plan is not descriptive enough to accommodate the exact sequence of processes so that the production maintains a high production rate and respects the customer deadlines. In the "as-is" scenario, production managers receive production plans—indicating the list of customer orders that need to be produced in the following period—and create a manual schedule based on experience and empirical rules. ERP managers are responsible for creating a sequence of all the items that would enter the lines within this period. With no existing monitoring systems, line managers communicate to ensure that all processes go as scheduled. A change in the schedule may produce a large disturbance to the plan and require fast and reactive corrections to avoid order delay and production bottlenecks in the intermediate buffers. The bike assembly department poses another major challenge to the production performance as it requires smaller batches of orders to be scheduled on each assembly line. This department is scheduled each day, based on the stock material produced by the previous departments. The challenges of the bike assembly department drive the material requirements for the wheel assembly processes, thus creating the need for reactive planning solutions.

4.2. Multi-Agent System Configuration for Bicycle Industry

4.2.1. Wheel Assembly Lines Scheduling Agent

The assembly of wheels is characterized by manual operations, tire fitting, and the actual wheel assembly performed by dedicated lines, according to the wheel SKU. Wheel assembly lines can produce certain types of wheels, which may also require a different set of supplementary processes. Thus, although all wheels need to go through the assembly lines, the accompanying processes may vary based on the wheel SKU. As such, the scenario can be described as a job-shop scheduling problem, in which the line manager needs to determine the sequence of operations for the different lines and manual workstations. The problem was formulated based on the specifications and descriptions provided for the given use case. Product types, process plans, resources, orders, tasks, jobs, the suitability matrix, and the scheduling workload formulate the job-shop problem. The DRL agent for the wheels department has some requirements for the parameters:

- $maxP$, where $maxP$ is the maximum number of product types.
- $maxM$, where $maxM$ is the maximum number of resources.
- $maxT$, where $maxT$ is the maximum number of tasks of a product type.
- $maxJ$, where $maxJ$ is the maximum number of units to be processed of a job J.
- where maxO is the maximum number of orders that a job can have.
- Suitability matrix: if the value is 0, it means that the task is not suitable for the resource.
- The scheduling solution involves allocating jobs (J_c) to resources (M_r).

Initially, the scheduling workload consists of all the orders O_g, jobs J_c, and tasks T_b that need to be executed. Nevertheless, the scheduling solution refers to the jobs' allocation to resources (J_c allocation to M_r).

The building blocks and procedures required to link the DRL scheduling agent to the scheduling input and environment supporting features are defined as part of the training and testing execution framework. Algorithm 1 presents the pseudocode for the training phase of the DRL. The core of the agent is a deep Q-network, which is a deep neural network implementing the Q-function.

Algorithm 1. DRL training pseudocode.

Input Data: Episodes
Results: DQN weights
Parameters: Episode
Procedure:

1. Initialization of the DES and DRL agent
2. Create initial status of a production scenario (resources availability and workload)
3. Pass state in the DRL agent (DQN)
4. Agent proposes an action
5. Action masking
6. Pass proposed action to simulation
7. Perform simulation run until next decision point
8. Create current state and calculate reward
9. If (Production is finished)
    a. Reset simulation to initial state for a new production scenario
    b. If (episode == Episodes)
        i. Store DQN weights
        ii. End
    c. Else
        i. Episode += 1
        ii. Go to Step 2

The agent is trained in a simulated production system representing the environment. The operations in a manufacturing system can be modelled as discrete processes, and thus a discrete event simulator fits very well with this type of environment. The environment where the agent should be trained in should be adequately explored. Thus, the modelling of the environment from the agent perspective is one of the most important aspects in such a machine learning approach. The overall status of the production at time t should be mapped in some predefined parameters that formulate the state s at time t. This state will be given as input to the DRL agent to calculate the output, which is the next action a in the environment. State s_t is a vector of features representing the production status at time t. The state consists of information regarding the status of jobs and resources. The DRL agent explores the environment by visiting the various states. The transition from a state s to a state $s\prime$ is defined as action a. The agent shall perform actions to visit the states of the environment under exploration, and the set of all possible actions formulates the action space. The DRL agent decides which dispatch rule fits best in each decision point. The agent is then rewarded with a reward, which includes the achieved makespan value, indicating the score of the proposed action in a particular state.

To enable this DRL agent to link with the simulation environment, a custom-built part known as the execution controller has been created, carrying out a number of essential tasks. First, the process requires pre-processing by converting the DES model's output, which contains data on resources and task status, into a DQN input vector. It also uses the makespan which resulted from the simulation to determine the episode reward for the DRL agent. On the other hand, post-processing is handled by the execution controller. It does this by creating the DES model input from the DQN output, which is a vector containing encoded dispatch rules. Additionally, it manages simulation processes, guaranteeing the framework's stability by coordinating the DQN and DES models' operations. Furthermore, scheduling input is managed by the execution controller and is generated internally or through outsourcing based on the framework's mode (training or testing). Figure 5 summarizes the features and parts of the DRL scheduling framework and shows how components work together to provide effective scheduling agent training and performance. In detail, Figure 5 illustrates the overall architecture of the DRL framework. In this architecture, the DRL agent interacts with the environment via the operation controller for both training and testing purposes. The environment is modeled as a shop floor within a DES. The subsequent

sections provide detailed descriptions of the individual components shown in Figure 5, including the scheduling input, environment (DES), DRL agent, and operation controller.

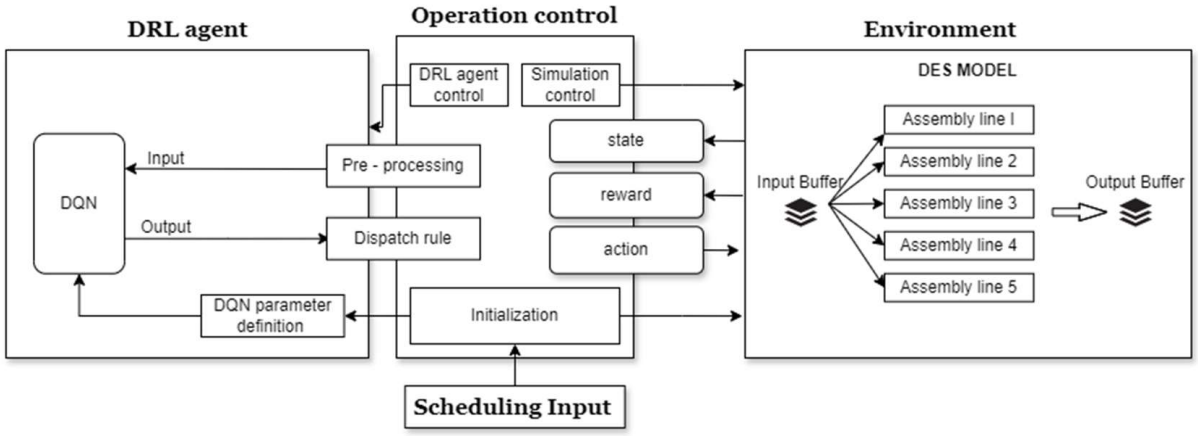

Figure 5. DRL scheduling framework architecture.

The DRL agent was implemented with Python and the Deep-Q-network was realized with the use of Python libraries Tensorflow and Keras. The DES model was realized with the use of WITNESS HORIZON. The connection that the execution controller serves between the agent and the simulation was possible with the use of a custom-made API, able to control the simulation runs.

4.2.2. Painting Line Scheduling Agent

The painting line scheduling problem deals with a rapid material flow in which the manager aims to define the right sequence and combination of material entering the discrete conveyor line of the paint shop. Differently to other industrial sectors, bike BOMs contain a combination of smaller parts that need to be painted, with varying dimensions and colors, posing one of the initial spatial/capacity constraints for the different carriers of the line. Moreover, sequence-dependent setup delays are usually required when switching between different colors which also makes this a sequencing problem. In some cases, there are also additional bottleneck constraints, associated with the production rate of the operation before and after the painting line. Buffer capacity constraints before or after the painting line pose the main risk for buffer overflow and potential line stoppages.

Figure 6 provides a schema of the scenario and the challenges for the line manager. The schema represents the conveyor line, consisting of multiple carriers (usually up to a few hundred) which transfer the material throughout the paining cabins. Carriers have capacity constraints based on the sizes of the material and also, in some cases, limitations in combining certain material types due to the configuration of the carrier. The way that the material is grouped plays a significant role in the percentage of the carrier that will be fulfilled. The higher the percentage, the higher the production rate of the line (in other words, the parts painted per time unit). Moreover, the sequence of the groups of items entering the carriers also plays a significant role in the setup delay of the line. This aspect is associated with the color and the time delay when switching colors in the painting cabins. Usually, the setup delay is expressed in empty carriers, leaving time for the operators to make all required adjustments on the color setup. Considering that buffers in the output have a limited spatial capacity, when defining the sequence of items entering the line, line managers need to consider the output rate from the buffers to avoid material concentration and potential stoppages of the line.

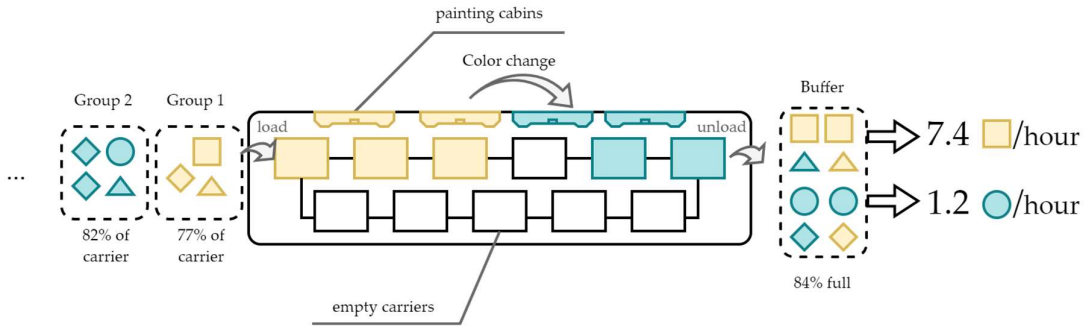

Figure 6. Schematic representation of the paint-shop scheduling problem with explanatory data.

The specific scheduling agent utilizes a mathematical optimization model for retrieving the scheduling decisions to the problem. The decision for the system is to define the sequence and groups of items entering the painting line by selecting which item types, from which orders, with what color, and in what quantity will be placed on each consecutive carrier of the line, up to the point where all customer orders are satisfied. The main objective of the underlying model is to optimize the output production rate of the lines.

The problem formulation was inherited from a previous work of the authors [27], which provides the decision variable $x_{p,i,t} \in \mathbb{Z}_{\geq 0}$ for the number of items from item type $i \in I$ for product $p \in P$ that will enter the line at time $t \in \mathbb{Z}_{\geq 0}$. This creates the sequence and grouping of items entering the line, considering each timestep (t) as the sequence of carriers moving at a constant speed. To constraint the environment into a space of feasible solutions, the model provided $n_p n_i + 6\, n_p n_i n_t + n_c n_t + n_t d(n_c + 1)$ number of different expressions, where n_p, n_i, n_t, n_c, d are the number of products, item types, timesteps available, and colors, and the setup delay, respectively. The model was reduced into a linear mixed-integer model, containing a ${n_p}^2 + n_i(n_p + 1)$ number of decision variables. The proposed model focused on the minimization of the total flowtime for producing the product demand L_{total}; nevertheless, it did not consider the possibility of buffer overflow disturbances within the schedule.

The output rate per item type from the painting line can be an extension to the model, guiding the optimization process in avoiding buffer overflows and potential bottlenecks. The output rate in each window of carriers can be expressed as the average number of items, mixed on that part of the line accordingly (as shown in Equation (1)).

$$\lambda_{i,\,k} = \frac{\sum_{t=k\,L}^{(k+1)\,L} \sum_{\forall p \in P} x_{p,\,i,\,t}}{L} \text{ for } k = 0,\ 1,\ 2, \ldots, n_t \tag{1}$$

where L is the window of carriers to look at. This can then be accumulated within the objective function as the squared error from the desired output rate, and be presented as a quadradic minimization criterion as shown in Equation (2).

$$\min\left\{ \sum_{\forall i \in I} \sum_{k=0}^{\infty} \left(\lambda_{i,\,k} - \lambda_i^{\text{desired}} \right)^2 \right\} \tag{2}$$

where $\lambda_{i,\,k} \leq \lambda_i^{\text{desired}}$, $k = 0,\ 1,\ 2,\ \ldots, n_t, \forall i \in I$. This new criterion provides the capability to adapt the desired output rate ($\lambda_i^{\text{desired}}$), thus managing to create an equilibrium and avoid overflows.

4.2.3. Bike Assembly Line Scheduling Agent

In the bike assembly stage, all the product's BOM has been manufactured (or supplied) and the final product is being assembled. The assembly process is performed in different assembly lines, each one capable of producing specific groups of products (e.g., regular bikes versus e-bikes). The bike assembly line itself usually consists of sequential manual

operations performed in a flow shop type of environment, which, depending on the type of product assembled, may vary in processing time. Due to this consecutive manner, the line moves at a constant speed (takt time), since products cannot proceed to the following assembly step, unless the space on the next step is freed.

Figure 7 displays the assembly line environment for a specific product sequence example. The ultimate objective for the line manager is to increase the production rate of the lines and manage to deliver products on time. Since this is the final step of the product manufacturing process, material deliveries from the wheel and painting lines are necessary for that process to be completed. In this case, however, the bike assembly stage is the key driver of production, and the multi-agent system is designed to adapt the previous stages into the schedule of the assembly lines. Scheduling the assembly line is a sequencing task, in which the right sequence of the products entering the lines must be given in a way that minimizes the waiting time of the items due to *blocking* (blocking is a phenomenon in which the current stage must remain a work-in-process and cannot release any task to the downstream stage once the buffer is stuffed.). Given the description of the problem, it can be abstracted into a *flow shop scheduling* problem with *blocking*, considering that there are no intermediate buffers between the stages.

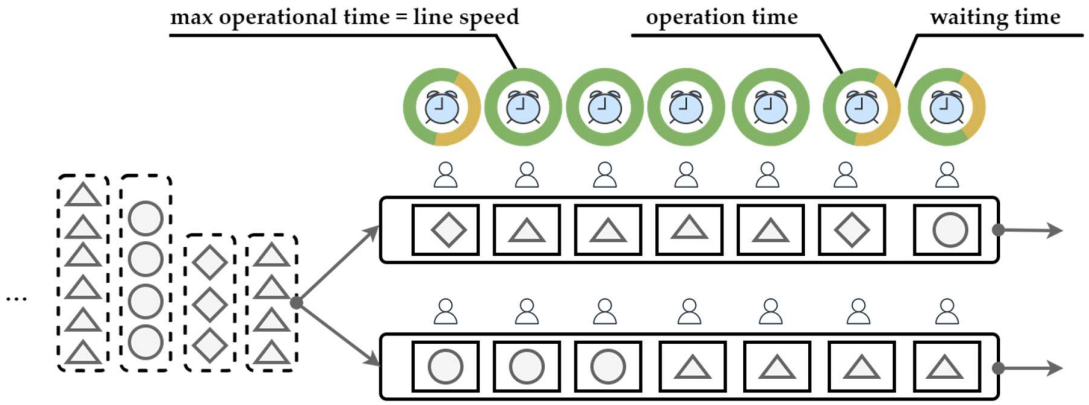

Figure 7. Bike assembly line scheduling problem schematic representation.

The encountered scheduling problem considers the assumption that production orders have already been allocated to the assembly lines in advance, and the decision that line managers must make is to sequence the products in the best possible way. In other words, considering a set of assembly jobs/products $J = \{1, 2, 3, 4, \ldots, n\}$, the sequence $S = \{j_k | k \text{ is even or odd depending on the sequence and } j_k = J(k)\}$ can be considered the optimum sequence if it minimizes the waiting time of each product in every assembly step in that sequence. As described before, the line must be moving at a constant speed which must be equal to the highest processing time (per assembly step) of the products present in the line at that time. Equation (1) can represent the processing time of the whole line (per step) at the time that job i has entered the line.

$$y_i = \max_{[i-N,\, i]} \left(p_j : j = S(i) \right) \tag{3}$$

The interval $[i - N, i]$ defines that after N assembly steps, the assembled product will exit the line and will no longer add any delay to the overall processing time. y_i expresses $\left[\frac{\min}{\text{step}}\right]$ and thus the time at which the ith item will enter the line can be expressed as shown in Equation (2), and equally, it will exit the line at the time defined by Equation (3). It

should be noted that the sum has to be up to i − 1 index since it represents the time at which the line has finished the first assembly step from the previous product (Figure 8).

$$e_i = \sum_{k=1}^{i-1} y_i \tag{4}$$

$$r_i = \sum_{k=1}^{(i+N-1)} y_i \tag{5}$$

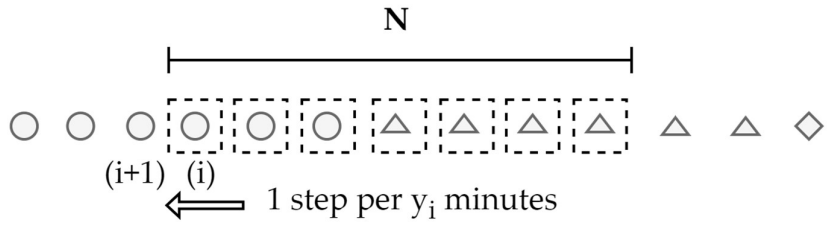

Figure 8. Schematic representation of the problem considering a slide window of length N.

The scheduling agent which was developed to address this problem was developed using a heuristic optimization algorithm [61], which generates near-optimum solutions based on a guided solution search. The algorithm starting from a root sequence (a subsequence, if it exists) generates a set of feasible alternative entries up to the point that the maximum number of alternatives (MNA) or the decision horizon (DH) has been reached. DH drives the depth of the search in different entries following the root sequence, and MNA defines the maximum number of branches to search. For each of the branches, a predefined sample of complete sequences will be generated, thus producing a full-scale solution upon which its performance can be evaluated. This is also referred to as sample rate (SR), since it generates random samples for evaluating the selections made in the search until the decision horizon. The best option is selected, and the root is updated, integrating the best entries up to the decision horizon. The process is repeated until the planning horizon is reached and a full sequence is produced. The proposed algorithm (also displayed in Figure 9) is incorporated within the optimization logic of the agent, driving the scheduling decisions for the assembly lines.

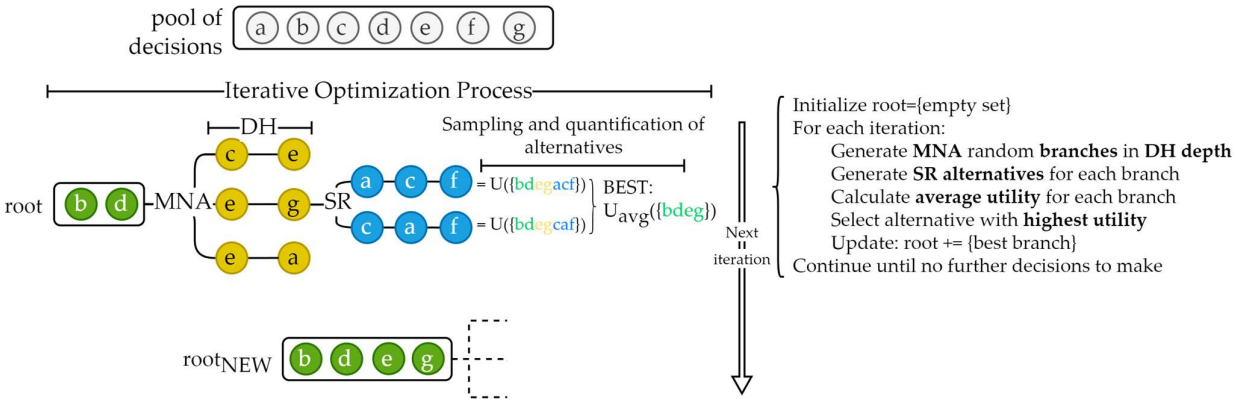

Figure 9. Optimization algorithm procedure displayed in a schematic way.

4.3. Implementation

The computer that was used for the implementation is equipped with Windows 10 Enterprise (64-bit) operating system. It features an Intel (R) Core (TM) i7-10700 CPU running at 2.90 GHz and has 32.0 GB of RAM, of which 31.9 GB is usable. In an overview, the proposed architecture consists of the multi-agent system, DT, web interfaces, AAS platform,

and certain connections for a coherent integration of those components. First, the AAS platform, built as a docker container, was installed on a virtual machine on the industrial premises. A virtual private network (VPN) connection was also a requirement to secure any information transactions with the organization's ERP software. This connection was developed in Java, mapping the AAS components of the framework, with the corresponding tables and columns on the Structured Query Language (SQL) server of the organization's ERP. This connection ensures that all new entries would also be translated into submodel elements in the AAS instances. The user web interfaces were also deployed as a spring-boot application in the virtual machine on the factory, working well in a closed network. The configuration of the user's web application allows engineers to modify these modules by exchanging or upgrading several components independently. The multi-agent system was developed in Janus/SARL, defining a certain number of capabilities, skills, and agents to be spawned dynamically during the initialization of the framework [47]. Each of the agents implements certain functions during runtime, inherited from the different scheduling solutions presented within this framework. As such, a one-to-one connection is established between the agent within the multi-agent system and the computational operation on a remote REST service. The wheel agent was deployed as a Flask application in Python using TensorFlow library for the training and implementation of the deep reinforcement learning methods [58]. The DQN model was configured with 1943 nodes in the input layer and two mid-layers, each with 2048 nodes. The output layer nodes varied, with DRLA1-2 having 2 nodes and DRLA3-4 having 16 nodes. The policy used was a Q-Policy with an epsilon value of 0.1. Memory was sequential, and the target model update rate was set at 1×10^{-2}. The gamma factor was 0.99, and the learning rate was 1×10^{-3}. The optimizer employed was Adam. Training involved 2000 episodes for DRLA1-2 and 8000 episodes for DRLA3-4. The painting agent was developed using Gurobi optimizer v10.0.0 (https://www.gurobi.com/ accessed on 17 April 2024) and was built as a Flask application in Python. Finally, the assembly agent was developed in Java, and launched as a spring-boot application. The I/O mapping components of the AAS were also launched as spring-boot autonomous applications, which allows the manufacturer to update them during the lifetime of the solution without creating conflicts with the rest of the framework. DT was developed as a standalone component, using the commercial tool WITNESS Horizon, on top of which a Python API was built to manipulate the environment and establish connection with the AAS components. The above services were connected to the AAS platform using standardized connectors, which ensures the code-free connection of the components.

4.4. User Interaction

The user was integrated via a web interface (Figure 10), acting as a portal for managing the underlying computational environments. The proposed interface was designed and developed to use the AAS platform as the basis for creating all required connections with those modules. In other words, unlike typical web applications, this software did not provide any backend services, but rather uses the standardized API of the AAS, on top of the AAS platform which contains all associated information. The configuration of the application was structured by the following properties: "aas_repo_url", "aas_auth_url", "username", "password", "production_master_aas", and "scheduling_agent_aas". These components allow the user to define the location of the AAS platform as well as authorization aspects, and define the corresponding production master/system and multi-agent system AASs. Once the application is connected to the AAS platform, the configured AASs will be retrieved, and connections will be evaluated. Failing to evaluate the connection with certain components could result in the partial or overall inability of the application to work properly. In addition, failing to identify the AASs or certain parts of the AAS would also result in issues during the initialization of the application. As such, information must be modelled in the corresponding AAS templates and deployed on an Industry 4.0 compatible platform. As long as the AAS is structured properly, the user is provided with

functionalities for interacting with the scheduling agents via the execution of certain skills provided, interaction with the DT, and, finally, the execution of the schedule and dynamic re-scheduling actions, if needed.

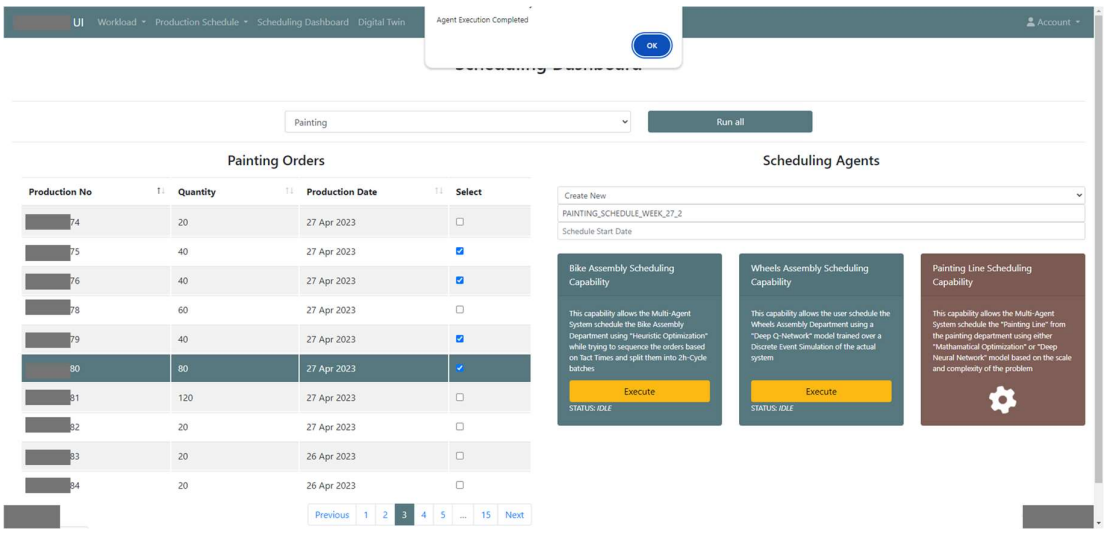

Figure 10. Screenshot of the multi-agent control panel from Web User Interface.

5. Evaluation Results and Discussion

Previous performance results from three separate days of operation in production were collected and analyzed as a base for evaluating the research results over the existing scheduling practices of the organization. The system was deployed in production and tested by the line managers as a decision-support tool for managing the daily schedules of the different stages. During that process, line managers needed to operate the painting, wheel, and assembly departments and coordinate the material flows in between. The goal was to support the management of these departments for the purpose of achieving higher production volume during the shift and avoid disruptions. Results from these experiments were tracked and are analyzed in the following section. In addition, the performance with respect to computational time and resources was tracked, identifying the user experience in terms of optimization delay and more. Deviations between the digital twin calculations and the actual production performance were analyzed, revealing the accuracy of the model. Offline, testing through a collection of different production datasets was also performed to identify the performance of the framework in different scenarios, such as line breakdowns, and production scale.

The focus was to evaluate the integration of the modules in a unified framework, evaluating both business and technology performance. From a business point of view (business KPIs), the MAS tried to optimize the production volume per day, or, in other words, minimize the makespan of the whole schedule and increase the utilization of the lines to the maximum percentage. From a system point of view, it was important to validate the framework response time and its ability to dynamically provide fast and accurate solutions in case of a production event, and, finally, to try to provide some quality indicators of the benefits of the system within the proposed production scenario. The comparison focused on pinpointing the improvement of the "as-is" scenario with the "to-be" scenario, meaning the proposed framework implementation. Specific information highlighting the productivity of the underlying environment (e.g., products per shift) are kept hidden as sensitive information of the organization. As such, the evaluation results were provided in a parametric manner, with respect to the "as-is" production performance. Three primary indicators were provided from the evaluation study, namely, worst (minimum performance), average, and best (maximum performance) solution. A set of experiments was performed with different production workloads. The differences were mainly due to the variation

in the types of products included in the workload. This resulted in variations in the performance of the scheduling agent. To generalize this aspect, the average performance was calculated from the evaluation of multiple daily schedules, giving an indication of the expected results if this system was implemented in production. On top of that, the extreme values (worst/minimum and best/maximum performance) from these datasets were provided, to show the variation in performance in different production instances.

The painting line scheduling was evaluated in terms of the number of hangers (painting line hanger or hook is a specific type of carrier on which items are hung at certain predefined positions.) used to produce a specific workload of items. The fewer hangers the manager can use, the more the items that can be mixed, and the less the time required for the total amount of items to pass through the whole painting line. Delays (or else empty/not fully occupied hangers) could be the result of not creating optimum mixtures or unnecessary setup delays caused by changing from one color to another. Two possible scenarios were tested for the schedule, one which is a realistic scenario which, given the current production environment, consisted of carriers with limited flexibility in mixing different item types in the same carriers (due to the carrier design itself), and one which is for potentially updating the carrier in the future and enabling all items to be mixed into the same carrier, naturally considering its capacity, as has already been done. The results from the evaluation process were measured as a percentage increase over the previous average production performance in production rate. The results presented in Table 1 were calculated using the following formula $\left(\frac{\mathrm{KPI}^{\mathrm{X}}_{\mathrm{TO-BE}}}{\mathrm{KPI}^{\mathrm{AVG}}_{\mathrm{AS-IS}}} - 1\right) \cdot 100\%$ where $\mathrm{X} : \{\mathrm{Worst},\ \mathrm{Average},\ \mathrm{Best}\}$ is the different performances obtained from the evaluation dataset.

Table 1. Evaluation results from the painting line scheduling agent versus the production performance of the "as-is" scenario without the multi-agent system implementation.

	KPI: Production Rate (Items per Hanger)		
	Worst	**Average**	**Best**
Painting (No-hanger Upgrade)	+0.36%	+8.79%	+21.06%
Painting (Hanger Upgrade)	+65.51%	+74.28%	+82.07%

As the results indicate, in the scenario where no hanger upgrade is performed and production works in its current state, manufacturers should expect to see an average increase of 8.79% in the production rate by using the proposed solution in scheduling the painting line. The performance results ranged between the minimum and maximum increase displayed in Table 1 due to the workload diversity in-between different production days. It should also be noted that this scenario also considers that orders of same color are not mixed within the hangers, which is mandatory for smaller items, yet not infeasible for larger items. In the case that the production considers investing in upgrading the existing hangers with more flexible ones that can handle more item types at once, a considerable average increase would be shown in the production rate, as shown in Table 1, as it would require fewer hangers to hold the same number of items. In particular, the agent was provided with the ability of making combinations over the painting line considering that the hangers could physically support these combinations. The responsiveness of the agent also played a significant role for the line manager. The proposed results indicated quality performance giving an average of 1154 s for a workload scale of 543% of the average daily production volume.

For the wheel assembly case, the agent was compared against the actual ("as-is") average performance KPIs (Table 2), which was documented on a full-scale shop floor, with no abnormalities included. Simulation was used, and several hypothetical scenarios within this evaluation phase were tested, to emulate the potential abnormalities and uncertainties that are found in this production department. Regarding the scenarios, machine breakdowns or completely unavailable production lines were considered in the simulation

runs, which is something that affects the production rate. Moreover, the workload was not always equally distributed across the lines due to process plan routing constraints (e.g., wheels that can only be produced in one line). Additionally, workload variations for each scenario affected agent performance, since variability is introduced in the wheel assembly processing times and process plans. In general, the production rate (measured in wheels produced per hour) achieved by the agent at a fully operational environment mimicking the actual full-scale shop floor operation showed a valuable increase of 14.78% in a steady environment where no abnormalities occurred. In this case, the resource availability was the same, but the workload scenarios were different since they were randomly generated for the simulation scenarios. In such scenarios, a makespan reduction of 2–10% was observed. On the other hand, when it comes to a case with breakdowns or unavailable production lines, and dynamic scheduling was needed, the agent performed well again, reaching an increase in the production rate of 1.9%. It must be mentioned that the comparison here is between the fully operational real environment and a simulation environment with uncertainties. As such, although the evaluation was between the physical system without breakdowns and a digital system with breakdowns, the results showed that the agent managed to handle the deviations and resulted in an average to better performance than the fully operational current setup. It was found that the current shop-floor rescheduling strategy would lead to greater tardiness, which is also affected by the workload variability. In terms of responsiveness, the user experienced after the scheduling input selection an average of 15 s response time, of which 0.067 s was pre-processing, 0.73 s was used for DQN model execution, 13.5 s for the simulation of the scheduling command, and the remaining time was allocated to modelling the I/O and transferring data.

Table 2. Wheel assembly department DRL agent performance results.

	Scenarios	
	Environment with Breakdowns	**Fully Operational Environment**
Production Rate (%)	+1.9%	+14.78%
Machine Utilization (%)	67.69%	74%
Tardiness (minutes)	+191 min (worst)/+81.95 min (average)	−191 min (best)

For the bike assembly department, production rate is also the most important attribute for line managers. The agent achieved a considerable improvement in production rate and makespan, as shown in Table 3. That improvement is expressed with respect to the average production volume of the "as-is" scenario in a day shift, estimated from three separate days' workload. Regarding the responsiveness of the agent, the algorithm indicated an average response time of 1 min for a production scale of 6 to 14 h of operation (or one-to-two shifts), with the production rate achieved by the agent. As such, the line managers could rapidly adapt the production schedule in real time and adjust dynamically to the environment. Each change in the bike assembly department triggered a dynamic scheduling request for the wheel assembly stage, and line managers received a new schedule configuration for the lines, which could be reviewed and implemented, if found to be beneficial for the production.

Table 3. Evaluation results from the bike assembly scheduling agent versus the average performance of the system before applying the proposed solution.

	Bike Assembly Department Performance Results		
	Worst	**Average**	**Best**
Makespan	−1.97%	−19.39%	−29.67%
Production Rate (bikes/shift)	+1.43%	+4.63%	+9%

The user was able to evaluate the scheduling solutions through the digital twin simulation. Results from the agent were sent to the digital twin models and several performance indicators were calculated within the discrete event simulation model. After these scheduling results were applied in production, several deviations between the calculations within the digital twin and the actual execution time were found. On average, the digital twin deviated $\pm 0.019\%$ over the actual production time, which was reflected in a ± 132.14 min deviation when calculating the system performance in a multiple shift timespan. The causes of this deviation can be inherited from multiple aspects, such as model assumptions and generalizations, unforeseen events and stochasticity, manual operations, and human errors. In this industrial case, the level of human involvement in multiple production stages seemed to play a prevailing role in the deviation between the quantitative model and the actual production. Examples of such deviations were identified in the hanging/unhanging area of the painting line, as well as slight deviations in the bike and wheel assembly manual operations time, which accumulated in the overall deviation percentage identified above. It should be noted, however, that further investigation into the causes of deviation was not performed. Regarding the response time for the digital twin, the user experienced an average of 62.89 s. The response time fluctuated based on the problem scale, i.e., the number of items/products included within the schedule. On average, 0.01597 s were added for every new item added onto the schedule. This meant that for an extra 1000-item workload, the digital twin would need an average of 15.97 s of extra calculation time.

From a holistic standpoint, the multi-agent system was able to accumulate these scheduling decisions on top of existing schedules, thus building an overall solution for the production. This allowed line managers to dynamically adjust their production lines automatically, while taking into consideration scheduling decisions from other sequential stages. As such, a better coordination of production was achieved, and the system was empowered with an adaptable solution to react to production disturbances.

6. Conclusions

The proposed research study focused on the deployment and evaluation of different Industry 4.0 technologies integrated into a single framework for supporting production managers with the scheduling problems of the bicycle industry. The proposed framework was built upon a multi-agent system and DT technology, providing online decision-making and the evaluation of scheduling decisions on the run. Information was structured in the form of the AAS concept, to enable the reconfigurability and interoperability of the solution. Three different scheduling algorithms were utilized to address three different problems found in the bicycle industry, namely, mathematical optimization, deep reinforcement learning, and heuristic optimization. Each algorithmic scheduling solution was integrated into the administration shell model and represented as an agent entity within the multi-agent system. The results from this study focused on pinpointing the performance of the individual agents on the different problems, while also evaluating the performance of the multi-agent system. The DT technology was thus a valuable tool for quantifying the performance of the production system in the short term, based on decisions made by the agents. The evaluation results produced by the industrial case study indicated key improvements in the individual performance of certain lines in production, as well as a better coordination of the material flow across the different departments. This aspect was mainly supported by the DT components, which revealed inconsistencies in the scheduling results across two consecutive departments and triggered re-scheduling operations for the corresponding agents managing the operations of these departments.

In addition, certain limitations and opportunities for future research were identified based on the results from the proposed study. The information model structuring production data from ERP system into the AAS models is a crucial aspect which can either improve or limit the scalability of the solution and its potential applicability in different industrial setups. Although the proposed paper exploits the methodology of the AAS, it does not particularly focus on the structure of information regarding production data within

the AAS. Coupling the aforementioned information using existing production standards within the AAS concept is still an open issue for such research applications. Moreover, the development of an orchestration agent was found to be particularly useful for guiding the actions of the scheduling agents and boost coordination across the different departments. At this stage of the research study, coordination was addressed through the DT, indicating the need for rescheduling operations after certain conflicting decisions are made for the interdependent departments. Acquiring an orchestration entity can support foreseeing such production disturbances in advance and guide scheduling agents to make accurate scheduling decisions that will address potential disturbances of the material flows between these departments. Finally, great industrial interest may arise from extending the proposed solution to other industrial sectors. To do so, the proposed framework could be reused, while exploiting different scheduling agent implementations as well as information communication models, tailored to the specific needs of the case.

Author Contributions: Conceptualization, V.S. and K.A.; Methodology, V.S., E.B., P.M., N.N. and K.A.; Software, V.S., E.B. and P.M.; Validation, V.S., E.B. and P.M.; Resources, N.N. and K.A.; Writing—original draft, V.S., E.B. and P.M.; Writing—review & editing, V.S., E.B., P.M., N.N. and K.A.; Project administration, N.N. and K.A.; Funding acquisition, K.A. All authors have read and agreed to the published version of the manuscript.

Funding: This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 957204 MAS4AI. The dissemination of results herein reflects only the authors' view, and the Commission is not responsible for any use that may be made of the information it contains.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy restrictions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chryssolouris, G.; Alexopoulos, K.; Arkouli, Z. Artificial Intelligence in Manufacturing Systems. *Stud. Syst. Decis. Control* **2023**, *436*, 79–135.
2. Liu, Y.; Fan, J.; Zhao, L.; Shen, W.; Zhang, C. Integration of deep reinforcement learning and multi-agent system for dynamic scheduling of re-entrant hybrid flow shop considering worker fatigue and skill levels. *Robot. Comput. Manuf.* **2023**, *84*, 102605. [CrossRef]
3. Kayhan, B.M.; Yildiz, G. Reinforcement learning applications to machine scheduling problems: A comprehensive literature review. *J. Intell. Manuf.* **2023**, *34*, 905–929. [CrossRef]
4. Shyalika, C.; Silva, T.; Karunananda, A. Reinforcement Learning in Dynamic Task Scheduling: A Review. *SN Comput. Sci.* **2020**, *1*, 1–17.
5. Guzman, E.; Andres, B.; Poler, R. Models and algorithms for production planning, scheduling and sequencing problems: A holistic framework and a systematic review. *J. Ind. Inf. Integr.* **2022**, *27*, 100287. [CrossRef]
6. Jacoby, M.; Baumann, M.; Bischoff, T.; Mees, H.; Müller, J.; Stojanovic, L.; Volz, F. Open-Source Implementations of the Reactive Asset Administration Shell: A Survey. *Sensors* **2023**, *23*, 5229. [CrossRef]
7. Talebi, S.P.; Werner, S. Distributed kalman filtering and control through embedded average consensus information fusion. *IEEE Trans. Autom. Control* **2019**, *64*, 4396–4403. [CrossRef]
8. Soori, M.; Arezoo, B.; Dastres, R. Digital twin for smart manufacturing, A review. *Sustain. Manuf. Serv. Econ.* **2023**, *2*, 100017. [CrossRef]
9. De Simone, V.; Di Pasquale, V.; Miranda, S. An overview on the use of AI/ML in Manufacturing MSMEs: Solved issues, limits, and challenges. *Procedia Comput. Sci.* **2023**, *217*, 1820–1829. [CrossRef]
10. Colledani, M.; Tolio, T.; Fischer, A.; Iung, B.; Lanza, G.; Schmitt, R.; Vancza, J. Design and management of manufacturing systems for production quality. *CIRP Ann.* **2014**, *63*, 773–796. [CrossRef]
11. Arora, A.; Gupta, R. A Comparative Study on Application of Artificial Intelligence for Quality Assurance in Manufacturing. In Proceedings of the 4th International Conference on Inventive Research in Computing Applications, ICIRCA 2022-Proceedings, Coimbatore, India, 21–23 September 2022; pp. 1200–1206. [CrossRef]

12. Lee, W.J.; Wu, H.; Yun, H.; Kim, H.; Jun, M.B.; Sutherland, J.W. Predictive Maintenance of Machine Tool Systems Using Artificial Intelligence Techniques Applied to Machine Condition Data. *Procedia CIRP* **2019**, *80*, 506–511. [CrossRef]
13. Pournader, M.; Ghaderi, H.; Hassanzadegan, A.; Fahimnia, B. Artificial intelligence applications in supply chain management. *Int. J. Prod. Econ.* **2021**, *241*, 108250. [CrossRef]
14. Helo, P.; Hao, Y. Artificial intelligence in operations management and supply chain management: An exploratory case study. *Prod. Plan. Control.* **2022**, *33*, 1573–1590. [CrossRef]
15. Cadavid, J.P.U.; Lamouri, S.; Grabot, B.; Pellerin, R.; Fortin, A. Machine learning applied in production planning and control: A state-of-the-art in the era of industry 4.0. *J. Intell. Manuf.* **2020**, *31*, 1531–1558. [CrossRef]
16. Bueno, A.F.; Filho, M.G.; Frank, A.G. Smart production planning and control in the Industry 4.0 context: A systematic literature review. *Comput. Ind. Eng.* **2020**, *149*, 106774. [CrossRef]
17. Serrano-Ruiz, J.C.; Mula, J.; Poler, R. Smart manufacturing scheduling: A literature review. *J. Manuf. Syst.* **2021**, *61*, 265–287. [CrossRef]
18. Alexopoulos, K.; Nikolakis, N.; Chryssolouris, G. Digital twin-driven supervised machine learning for the development of artificial intelligence applications in manufacturing. *Int. J. Comput. Integr. Manuf.* **2020**, *33*, 429–439. [CrossRef]
19. Sutton, R.S.; Barto, A.G. *Book Reviews Reinforcement Learning*; MIT Press: Cambridge, MA, USA, 1999.
20. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef]
21. Chien, C.-F.; Lan, Y.-B. Agent-based approach integrating deep reinforcement learning and hybrid genetic algorithm for dynamic scheduling for Industry 3.5 smart production. *Comput. Ind. Eng.* **2021**, *162*, 107782. [CrossRef]
22. Mohan, J.; Lanka, K.; Rao, A.N. A Review of Dynamic Job Shop Scheduling Techniques. *Procedia Manuf.* **2019**, *30*, 34–39. [CrossRef]
23. Tang, J.; Salonitis, K. A Deep Reinforcement Learning Based Scheduling Policy for Reconfigurable Manufacturing Systems. *Procedia CIRP* **2021**, *103*, 1–7. [CrossRef]
24. Waschneck, B.; Reichstaller, A.; Belzner, L.; Altenmüller, T.; Bauernhansl, T.; Knapp, A.; Kyek, A. Optimization of global production scheduling with deep reinforcement learning. *Procedia CIRP* **2018**, *72*, 1264–1269. [CrossRef]
25. Jasinski, M.; Najafi, A.; Homaee, O.; Kermani, M.; Tsaousoglou, G.; Leonowicz, Z.; Novak, T. Operation and Planning of Energy Hubs Under Uncertainty—A Review of Mathematical Optimization Approaches. *IEEE Access* **2023**, *11*, 7208–7228. [CrossRef]
26. Thörnblad, K. Mathematical Optimization in Flexible Job Shop Scheduling Modelling, Analysis, and Case Studies. Ph.D. Thesis, Department of Mathematical Sciences, Division of Mathematics, Chalmers University of Technology and University of Gothenburg, Göteborg, Sweden, 2013.
27. Vasilis, S.; Nikos, N.; Kosmas, A.; Dimitris, M. A toolbox of agents for scheduling the paint shop in bicycle industry. *Procedia CIRP* **2022**, *107*, 1156–1161. [CrossRef]
28. Nurjanni, K.P.; Carvalho, M.S.; Costa, L. Green supply chain design: A mathematical modeling approach based on a multi-objective optimization model. *Int. J. Prod. Econ.* **2017**, *183*, 421–432. [CrossRef]
29. Habib, M.S.; Lee, Y.H.; Memon, M.S. Mathematical Models in Humanitarian Supply Chain Management: A Systematic Literature Review. *Math. Probl. Eng.* **2016**, *2016*, 3212095. [CrossRef]
30. Alkahtani, M. Mathematical Modelling of Inventory and Process Outsourcing for Optimization of Supply Chain Management. *Mathematics* **2022**, *10*, 1142. [CrossRef]
31. Bork, W.; Giedraitis, M. Optimizing within the Supply Chain: A Mathematical Model for Inventory Optimization with Respect to Demand Planning. 2023. Available online: https://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-341853 (accessed on 17 April 2024).
32. Zhao, F.; Ubaka, I. Transit Network Optimization-Minimizing Transfers and Optimizing Route Directness. *J. Public Transp.* **2004**, *7*, 63–82. [CrossRef]
33. Kolus, A.; El-Khalifa, A.; Al-Turki, U.M.; Duffuaa, S.O. An integrated mathematical model for production scheduling and preventive maintenance planning. *Int. J. Qual. Reliab. Manag.* **2020**, *37*, 925–937. [CrossRef]
34. Chen, S.; Pan, Q.-K.; Gao, L. Production scheduling for blocking flowshop in distributed environment using effective heuristics and iterated greedy algorithm. *Robot. Comput. Manuf.* **2021**, *71*, 102155. [CrossRef]
35. Montiel, L.; Dimitrakopoulos, R. A heuristic approach for the stochastic optimization of mine production schedules. *J. Heuristics* **2017**, *23*, 397–415. [CrossRef]
36. Aghelinejad, M.; Ouazene, Y.; Yalaoui, A. Production scheduling optimisation with machine state and time-dependent energy costs. *Int. J. Prod. Res.* **2018**, *56*, 5558–5575. [CrossRef]
37. Jélvez, E.; Morales, N.; Nancel-Penard, P.; Cornillier, F. A new hybrid heuristic algorithm for the Precedence Constrained Production Scheduling Problem: A mining application. *Omega* **2020**, *94*, 102046. [CrossRef]
38. Negri, E.; Fumagalli, L.; Macchi, M. A Review of the Roles of Digital Twin in CPS-based Production Systems. *Procedia Manuf.* **2017**, *11*, 939–948. [CrossRef]
39. Botín-Sanabria, D.M.; Mihaita, A.-S.; Peimbert-García, R.E.; Ramírez-Moreno, M.A.; Ramírez-Mendoza, R.A.; Lozoya-Santos, J.d.J. Digital Twin Technology Challenges and Applications: A Comprehensive Review. *Remote Sens.* **2022**, *14*, 1335. [CrossRef]
40. Boschert, S.; Rosen, R. Digital twin-the simulation aspect. In *Mechatronic Futures: Challenges and Solutions for Mechatronic Systems and Their Designers*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 59–74.

41. Negri, E.; Pandhare, V.; Cattaneo, L.; Singh, J.; Macchi, M.; Lee, J. Field-synchronized Digital Twin framework for production scheduling with uncertainty. *J. Intell. Manuf.* **2021**, *32*, 1207–1228. [CrossRef]
42. Bakopoulos, E.; Siatras, V.; Mavrothalassitis, P.; Nikolakis, N.; Alexopoulos, K. Digital-Twin-Enabled Framework for Training and Deploying AI Agents for Production Scheduling. In *Artificial Intelligence in Manufacturing*; Springer: Cham, Switzerland, 2024. [CrossRef]
43. Jhunjhunwala, P.; Atmojo, U.D.; Vyatkin, V. Applying Skill-based Engineering using OPC-UA in Production System with a Digital Twin. In Proceedings of the 2021 IEEE 30th International Symposium on Industrial Electronics (ISIE), Kyoto, Japan, 20–23 June 2021. [CrossRef]
44. Fuchs, J.; Schmidt, J.; Franke, J.; Rehman, K.; Sauer, M.; Karnouskos, S. I4.0-compliant integration of assets utilizing the Asset Administration Shell. In Proceedings of the 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 10–13 September 2019. [CrossRef]
45. Pribiš, R.; Beňo, L.; Drahoš, P. Asset Administration Shell Design Methodology Using Embedded OPC Unified Architecture Server. *Electronics* **2021**, *10*, 2520. [CrossRef]
46. Lu, Q.; Li, X.; Li, G.; Li, M.; Zhou, J.; Liu, J.; Shen, X.; Zhu, F. A general asset administration shell platform for production line design. In Proceedings of the 2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence, DTPI 2021, Beijing, China, 15 July–15 August 2021; pp. 192–195. [CrossRef]
47. Siatras, V.; Bakopoulos, E.; Mavrothalassitis, P.; Nikolakis, N.; Alexopoulos, K. On the Use of Asset Administration Shell for Modeling and Deploying Production Scheduling Agents within a Multi-Agent System. *Appl. Sci.* **2023**, *13*, 9540. [CrossRef]
48. Dorri, A.; Kanhere, S.S.; Jurdak, R. Multi-Agent Systems: A Survey. *IEEE Access* **2018**, *6*, 28573–28593. [CrossRef]
49. Cardoso, R.C.; Ferrando, A. A Review of Agent-Based Programming for Multi-Agent Systems. *Computers* **2021**, *10*, 16. [CrossRef]
50. Dittrich, M.-A.; Fohlmeister, S. Cooperative multi-agent system for production control using reinforcement learning. *CIRP Ann.* **2020**, *69*, 389–392. [CrossRef]
51. Egger, G.; Chaltsev, D.; Giusti, A.; Matt, D.T. A deployment-friendly decentralized scheduling approach for cooperative multi-agent systems in production systems. *Procedia Manuf.* **2020**, *52*, 127–132. [CrossRef]
52. Renna, P. Flexible job-shop scheduling with learning and forgetting effect by multi-agent system. *Int. J. Ind. Eng. Comput.* **2019**, *10*, 521–534. [CrossRef]
53. Mezgebe, T.T.; El Haouzi, H.B.; Demesure, G.; Pannequin, R.; Thomas, A. Multi-agent systems negotiation to deal with dynamic scheduling in disturbed industrial context. *J. Intell. Manuf.* **2020**, *31*, 1367–1382. [CrossRef]
54. Nair, A.S.; Hossen, T.; Campion, M.; Selvaraj, D.F.; Goveas, N.; Kaabouch, N.; Ranganathan, P. Multi-Agent Systems for Resource Allocation and Scheduling in a Smart Grid. *Technol. Econ. Smart Grids Sustain. Energy* **2018**, *3*, 15. [CrossRef]
55. Wang, J.; Deng, X.; Guo, J.; Zeng, Z. Resilient Consensus Control for Multi-Agent Systems: A Comparative Survey. *Sensors* **2023**, *23*, 2904. [CrossRef]
56. Komesker, S.; Motsch, W.; Popper, J.; Sidorenko, A.; Wagner, A.; Ruskowski, M. Enabling a Multi-Agent System for Resilient Production Flow in Modular Production Systems. *Procedia CIRP* **2022**, *107*, 991–998. [CrossRef]
57. Moghadam, R.; Modares, H. Resilient adaptive optimal control of distributed multi-agent systems using reinforcement learning. *IET Control Theory Appl.* **2018**, *12*, 2165–2174. [CrossRef]
58. Alexopoulos, K.; Nikolakis, N.; Bakopoulos, E.; Siatras, V.; Mavrothalassitis, P. Machine Learning Agents Augmented by Digital Twinning for Smart Production Scheduling. *IFAC-PapersOnLine* **2023**, *56*, 2963–2968. [CrossRef]
59. Mourtzis, D.; Papakostas, N.; Mavrikios, D.; Makris, S.; Alexopoulos, K. The role of simulation in digital manufacturing: Applications and outlook. *Int. J. Comput. Integr. Manuf.* **2015**, *28*, 3–24. [CrossRef]
60. Mavrothalassitis, P.; Nikolakis, N.; Alexopoulos, K. Discrete Event Simulation for Improving the Performance of Manufacturing Systems: A Case Study for Renewable Energy Sources Production. *IFIP Adv. Inf. Commun. Technol.* **2023**, *692*, 650–665.
61. Alexopoulos, K.; Koukas, S.; Boli, N.; Mourtzis, D. Resource planning for the installation of industrial product service systems. *IFIP Adv. Inf. Commun. Technol.* **2017**, *514*, 205–213.