Applications

Article series Industry 4.0

Felix Gehlhoff* and Alexander Fay

On agent-based decentralized and integrated scheduling for small-scale manufacturing

Ein Beitrag zum dezentralen und integrierten Scheduling für Einzel- und Kleinserienproduktion mittels Agenten

<https://doi.org/10.1515/auto-2019-0105>

Received August 29, 2019; accepted October 2, 2019

Abstract: Small-scale manufacturing often relies on flexible production systems that can cope with frequent changes of products and equipment. Transports are a significant part of the production flow, especially in the domain of large and heavy workpieces that requires explicit planning to avoid unnecessary delays. This contribution takes a detailed look at how to create feasible integrated schedules within a decentralised or even heterarchical architecture and which information the agents have to exchange. These schedules incorporate constraints such as the blocking-constraint. They also consider dynamic setup and operation durations while finding a good-enough solution. The proposed agent-based solution applies to a wide variety of scheduling problems and reveals positive properties in terms of scalability and reconfigurability.

Keywords: integrated scheduling, heterarchical architecture, multi-agent system, detailed transport planning

Zusammenfassung: Die Produktion von Kleinserien ist häufig abhängig von Fertigungssystemen, die in der Lage sind, mit häufigen Änderungen der Produkte und der eingesetzten Ressourcen umzugehen. Speziell bei sehr großen und schweren Werkstücken spielen die Transporte eine entscheidende Rolle im Produktionsprozess. Sie sollten daher explizit geplant werden um unnötige Verzögerungen zu vermeiden. Dieser Beitrag zeigt eine Möglichkeit auf, integrierte Produktions- und Transportpläne mit Hilfe eines dezentralen Agentensystems zu erstellen. Es wird zudem darauf eingegangen, welche Informationen die Agenten dafür austauschen müssen. Die integrierten Pläne beinhalten dynamische Rüstzeiten und genügen ebenfalls

den speziellen Anforderungen, die sich durch die Größe der Werkstücke ergeben. Das entwickelte Agentensystem lässt sich auf verschiedene Anwendungsfälle übertragen und zeigt positive Eigenschaften in Bezug auf Skalierbarkeit und Rekonfigurierbarkeit.

Schlagwörter: Integriertes Scheduling, heterarchische Architektur, Agentensystem, detaillierte Transportplanung

1 Introduction

The centralized and hierarchical control structures of traditional manufacturing systems are designed for production optimization while assuming a rather static production and product environment. However, frequent changes in customer demand, the trend for individualized production and global competition drive the need for flexibility in manufacturing systems [1]. Making use of decentralized intelligence within products and resources while equipping them with ubiquitous communication technology is one way to deal with the rising complexity within manufacturing systems [2]. This pairing of communication abilities and digital representation of physical objects and functionalities is also known as Industry 4.0 (I4.0) [3]. The new technologies and concepts provide significant potential for change and improvement to manufacturers and their supply chains. Almost every aspect, ranging from product design and supply chain coordination to order management and adaptations within the factory is affected by I4.0 [4]. Especially relevant for the approach of this contribution are the scenarios of the Order-Controlled Production (OCP) and Self-organising Adaptive Logistics (SAL). Ever-increasing pressure to minimize costs and therefore stocks while at the same time speeding up throughput times has forced the logistic industry to undergo significant changes. It evolved from an inventory heavy supplementary function to a provider of intri-

*Corresponding author: Felix Gehlhoff, Helmut Schmidt University, Hamburg, Germany, e-mail: felix.gehlhoff@hsu-hh.de

Alexander Fay, Helmut Schmidt University, Hamburg, Germany, e-mail: alexander.fay@hsu-hh.de

cate logistics processes that are essential for making today's mass customized production possible. However, this complexity is also prone to disruptions that directly influence production due to the reduction of (safety) stocks. A reliable distribution logistic in this scenario needs agile and autonomous capabilities to provide the necessary robustness [4]. These can, for example, be provided by autonomous vehicles that fulfil the transportation needs within a factory. These vehicles need to negotiate about orders and path rights, autonomously determine their path routes as well as pickup and delivery sites [4]. With ongoing research into autonomous vehicles, this will at some point be also a reality for most if not all logistics processes outside the factory.

But not only can the logistics part be handled autonomously. Looking at the order processing and the coordination of production resources within a factory, there is the need and the potential to apply the outlined concepts of I4.0 as well. Companies can engage in market places where they automatically can buy or sell either missing capabilities or excess capacities and, based on the same (self-)descriptions of orders and capabilities, automatically configure the entire value chain [4]. In such a system there is no constraint on how (be it via email, website or entered manually) or when the orders come in. These orders can be incorporated according to the company's strategy at any time, if necessary at the moment the order is received. This feature of additional automation is also applicable for factories that do not operate within a fully automated environment, i. e., production and transportation system, as it is often the case for small and mid-sized factories. Many mid-sized factories compete on their flexibility by reconfiguring the layout of the production floor, possible connections between workstations and shapes of production parts according to the current needs. Due to these frequent changes that can also include new product sizes and shapes, a fully automated physical system, e. g., a Flexible Manufacturing System (FMS) with an integrated transportation system, is often dismissed. This is because the FMS's transportation system is usually quite limited in its range of possible product properties. However, automated production planning and scheduling is still an area of great importance for such factories. This paper tries to provide directions on how a decentralized coordination algorithm can achieve the integrated scheduling of production and transportation resources within dynamic manufacturing environments.

The paper is organized as follows: Section 2 takes a more detailed look into the outlined domain and problem formulation and deduces the requirements for the solution approach. Section 3 analyses the state of the art followed

by the description of the coordination process as well as the information exchange in Section 4. The practical application is demonstrated in Section 5 combined with a brief evaluation. Section 6 closes with a summary and outlook.

2 Problem description

As already outlined in Section 1, the layout of the production floor, the shape of products and generally the participants of a system and their properties are prone to change. This is especially relevant for the area of small-scale production or mid-sized factories. Therefore, the scheduling approach must be able to cope with the dynamic composition of the system during runtime as well as with the dynamically changing state of the system due to disruptions on the shop floor. One particularly challenging area of small-scale production with regards to the integration of transportation and production concerns large workpieces such as rotor blades for wind turbines or heavy machinery. For the production of goods with large dimensions, cranes, which are installed above the factory floor, are usually the industry's first choice for realizing the transportation processes. Operator teams that also provide the necessary handling procedures, i. e., fastening and loosening the workpieces to the crane, usually operate these cranes. These handling procedures often require a significant amount of time. Also, the total operation time can vary significantly, depending on the distances to cover. For these kind of transportation processes, it is especially relevant to mention that they can also be a bottleneck, i. e., influencing possible delivery and pickup dates at machines or workstations and therefore the production process itself [5]. This can be the case due to the general scarcity of resources but also due to failures in coordinating and planning the behaviour of such resources. For example, transport resources can block each other, which can influence the fulfilment of transportation operations considerably. This, as well as the consideration of pickup times, is not exclusively relevant for cranes but also for many other transportation systems like forklifts or Automated Guided Vehicles (AGV), assuming they are not available in an exceedingly high number (i. e., they can be a bottleneck) or can navigate in very large spaces (i. e., not having to consider problems of congestion). Another important aspect of such factories is the so-called "human factor". This human factor in many cases leads to a divergence from plans that can deteriorate the production flow [6]. One aspect that improves the acceptance of plans for workers is the plan stability. On the one hand, frequent changes to plans might lead to additional processes concerning the deliv-

ery of raw and auxiliary material. On the other hand, it decreases the workers' trust in and acceptance of the plan itself, which results in more deviations from the plan [7]. Many studies have analysed the drivers of scheduling instability and the impact of possible reactions like schedule freezing [8]. Even though the optimal length of the freezing period is an open research question and object of many studies themselves, it is a common means to freeze at least part of the production schedule to avoid too many disruptions on the factory floor. Furthermore, the scheduling of orders is often a process that does not only take place once for a defined interval with all information included a priori (static scheduling) but rather on-line, meaning that at every moment new orders and resources can enter or leave the system (dynamic scheduling) [9]. It can even be the case that there is more than one active order within the system at the same time, i. e., competing for free slots at machines in parallel. This makes it difficult for traditional, static scheduling algorithms to cope with the dynamic situation. Besides, they often rely on the possibility to arrange all orders according to their criteria or algorithm. If, however, the already scheduled orders are fixed due to considerations of plan stability, many of these techniques are not applicable.

The scheduling problem within flexible factories is commonly known as the standard job shop scheduling problem (JSSP) (see for example [10]). However, the outlined scenario necessitates some differences. Just like in the standard problem there is a set of k jobs (or tasks) $J = \{J_1, \dots, J_k\}$ that has to be performed on a set of m machines (or resources in general, for example buffers) $M = \{M_1, \dots, M_m\}$. Each job J_i is composed of a set of sequential production operations O_{ij} $i = 1, \dots, k$, $j = 1, \dots, m(i)$, $m(i) \leq m$, where i is the index of the job and j is the index of the step in the overall job. However, in contrast to the standard JSSP, $m(i) \leq m$, i. e., the number of steps is smaller than the number of machines, does not apply here because a job can be processed more than once on a machine. The standard definition of the deterministic processing time p_{ij} of an operation O_{ij} does also not apply because the processing times vary depending on the order size and the machine that the operation is processed on (not identical machines and the operations O_{ij} are not pre-assigned to machines). Sequence-dependent setup times $c_{i-1,j}(i)$ depending on the predecessor operation need to be considered as well.

Furthermore, it is not only the goal to create a feasible schedule for resources that is feasible in a sense that all precedence constraints of O_{ij} are met but also that after each manufacturing operation O_{ij} a transport operation T_{kl} $k = 1, \dots, m$, $l = 1, \dots, m$, $k \neq l$ where k indicates the

resource that performs operation O_{ij} and l resource O_{ij+1} resp. or a buffer place. The finish date of this transport operation follows the so-called no-wait constraint, which means that O_{ij+1} starts immediately after the finish of T_{ij} .

Due to the not negligible amount of time needed for load and unload processes, the transport operation needs to be further detailed so that $T = \{c_{i-1,j}(i), dL, dT, dU\}$ where dL = duration of load process (pickup), dT = duration of transport (depending for example on the distance to cover and the resource's speed) and dU = duration of unload process. All finish (marked with f) and start times (marked with s) of these sub-processes connect and must not be interrupted. Note that $sO_{i+1,j} \geq fdL_{kl}$ meaning that the start of the next production operation is only possible when the pickup procedure of the preceding job is completed and also $fdU_{kl} = sO_{i+1,j}$ meaning that the production operation starts immediately after the unload process is finished. Note that while $fdU_{kl} = sO_{i+1,j}$ it is not $fO_{ij} = sdL'_{kl}$ but $fO_{ij} \leq sT'_{i,j+1}$ where fO_{ij} marks the finishing time of operation O_{ij} and $sT'_{i,j+1}$ is the starting time of operation T_{ij} . Thus, a job remains on a production resource (and blocks it) after its operation is finished until it is picked up by a transport resource. The traditional job-shop model assumes infinite intermediate buffers where jobs can be stored. This means that the preparation for operation $O_{i+1,j}$ (or even O_{ij} itself provided no setup is necessary and that a part is available) can start right after O_{ij} is finished, or $fO_{ij} = sO_{i+1,j}$ where fO_{ij} marks the finishing time of operation O_{ij} and $sO_{i+1,j}$ is the starting time of operation $O_{i,j+1}$. This does also not apply to our case. The large dimensions of workpieces often hamper the buffering close to machines, which leads to the necessary consideration of the so-called blocking constraint. Mascis & Pacciarelli define the blocking constraint as the forced remainder of a job on a machine until the next (production) machine becomes available [11]. This will be adopted here so that the job remains on a production machine until picked up by a transport resource that delivers it either to a separate buffer place or to the next production machine.

It is also necessary to separate the delivery process of workpiece A to machine M and the pickup process for workpiece B at machine M . It is assumed here that the same transportation resource cannot carry out both processes because there is no available space to deliver workpiece A as long as the removal of workpiece B has not been carried out, i. e., no swaps are allowed [11].

The description of the domain and problem indicate that similar variations of this problem can be found in many different factories across different industries. Within these kind of factories, there can be different control solutions at work, ranging from traditional central Manufac-

turing Execution System (MES) controlled solutions to decentral Cyber-Physical Systems (CPS) into which the solution might need to be integrated. Additional emerging domains where these kind of problems arise are the development of flexible production networks [12] and even the so-called open-source production like for example makerspaces where in both cases there might be no central control entity that supervises the scheduling process. However, most existing approaches are not capable of dealing with scenarios that were not envisaged at design time [1]. When considering such a variety of domains and use cases, a generalizable solution should be decentral or even heterarchical because such an approach can be combined with traditional, centrally control solutions such as an MES, but a central approach is difficult to implement within a decentral architecture. A heterarchical architecture, i. e., an architecture that capsules all necessary functionalities within the participating entities, is also open for later integration within for example a holonic or hierarchal architecture if necessary. This can be easily envisaged in cases where the scheduling problem is part of a larger system that also includes for example (high level) supply chain or (low level) machine control functionalities. In addition, an approach that is based on local and thus incomplete information is usually attributed with a shorter timespan to reach an acceptable solution than an exact method [13]. This is especially relevant because the JSSP is known to be an NP-hard problem [14]. A problem that is NP-hard cannot be optimally solved in acceptable computation time for realistic instances [15]. To be applicable for creating the initial schedule as well as for rescheduling purposes that create ad-hoc solutions after a disruption occurs, Requirement 5 reflects the threshold for reaching a solution during ongoing production. It is assumed that longer waiting times are not tolerated by the operators on the production floor.

Summing up, the following requirements (R) for an integrated coordination algorithm that can solve the stated problems can be derived from the introduction and scenario description:

- R1. Consideration of buffer scarcity, especially in the proximity of machines, operation-specific durations as well as sequence or location-dependent setup times
- R2. Consideration of interrelations between transport processes and machine availability
- R3. Changes for already scheduled orders are not allowed for a general scheduling procedure without considering disruptions
- R4. The algorithm needs to solve dynamic and integrated scheduling heterarchically
- R5. The algorithm must reach a solution within less than one minute
- R6. The algorithm needs to be able to handle dynamically changing environmental conditions

The plans that such an algorithm will create are not optimal. This is a common concession for all heuristics when dealing with the NP-hard problem of manufacturing scheduling [15].

3 State of the art

This section takes a brief look into the definition and properties of on-line and dynamic scheduling problems as well as applicable models to solve the latter. Afterwards, the use of agent-based technologies is explained and the state of the art of agent-based systems within the manufacturing control and scheduling domain is analysed.

3.1 Dispatching and dynamic scheduling

It is important to distinguish between the concept of dispatching (scheduling) and ‘explicit’ scheduling. Dispatching scheduling is a technique that usually relies on a (priority) rule-based selection mechanism that selects jobs from a queue or buffer [10]. The classic term ‘scheduling’, however, mostly refers to the explicit assignment of start and end dates to operations on machines [15] even though scheduling and dispatching can be subsumed under the term of ‘detailed scheduling’ [16]. Scheduling problems differ in the availability of information. Static or deterministic scheduling refers to problems where all information is known a priori whereas on-line or dynamic scheduling refers to situations where part of the information only becomes available at later points in time [15]. This information includes the arrival of jobs as well as the configuration of the system’s resources [17]. The scenario described in Section 1 and 2 fits into the category of an on-line scheduling that follows the ‘scheduling one by one’ paradigm [18]. Here, a job can be scheduled to all free slots remaining without changing already assigned slots to other jobs. *Sgall* states that for ‘scheduling one by one’ problems it is sufficient to assign jobs to machines for a length of time and not for specific intervals [18]. This, however, is not the case here because the interrelations between transport and production resources require the exact definition of timeslots, especially due to the possibility of bottlenecks that stem from the scarcity of transportation resources. Note that it is a common approach to define

a detailed schedule and pass this schedule to dispatching that acts on this schedule while it is not restricted by it, e. g., to maintain necessary flexibility in case of machine failures [1]. A heuristic approach to dynamic scheduling that also incorporates rescheduling in case of disturbances is in principle capable of generating a new detailed schedule on short notice. Therefore, the use of dispatching rules becomes obsolete as the detailed schedule does reflect the current state of production and can be adhered to. However, the detailed algorithm for rescheduling in case of disturbances and (short term) manufacturing control are not the focus of this paper.

3.2 Applications of multi-agent systems within manufacturing control

The outlined problem of decentralised scheduling for the JSSP that also exactly reflects the interrelation between transport and production processes as well as dynamic durations of these operations is NP-hard. At the same time, the requirement of a decentralised approach prohibits many commonly used (meta-) heuristic solutions to the JSSP, such as Genetic, Particle Swarm or Tabu-Search algorithms. An exact solution that solves the outlined expanded JSSP by for example Mixed integer linear programming does not fulfil the requirements regarding the time to reach a solution. Several authors have concluded that Multi-agent systems (MAS) are an appropriate solution to such problems [14], [1], [19]: Agent-based systems are especially suitable for modelling complex problems due to the possibility to decompose the problem and encapsulate functionalities. The agents distribute the decision-making among different entities that work together to form a solution [20]. If the agent system is working in a decentralised and distributed architecture, i. e., distribute the computational resources across multiple processors, it can also apply parallel processing (and thus increase the speed to reach a solution) and avoid a single point of failure. Due to their flexibility regarding communication, an MAS is also an appropriate choice for dynamic problems where for example the configuration of the system can change during runtime, i. e., resources and orders can enter or leave the system at any time. For these reasons, agents have been applied to problems within the manufacturing control domain for many years. The Foundation for Physical Agents (FIPA) has developed a considerable amount of standardization work, ranging from the architectural components of an MAS to standardized communication technologies like the FIPA Agent Communication Language (ACL) and the FIPA Semantic Language (SL) that the agents can use

to relay specific content (like concepts of an ontology) to one another [21].

Out of the abundant amount of literature regarding MAS in manufacturing control, most relevant papers have been selected based on surveys, as described below. These papers will be discussed in more detail in the following and are compared in Table 1.

Several studies have been conducted that summarize the developments of agent applications in manufacturing systems [20, 17, 1, 14]: *Caridi & Cavaliere* start by describing a taxonomy for the classification of MAS's. Notably, they distinguish three forms of organizations in MAS's: Hierarchical, centralized and heterarchical [20]. After analysing more than 100 contributions of MAS applications in production planning and control they conclude that the multi-agent paradigm is especially suitable for low volume production with a wide product range. [17] analysed more than 100 approaches as well and point out that a fully federated or decentral system that relies on message passing as a means of communication is well suited for developing scalable MAS architectures. [17] and [20] especially emphasize the approach of [10]. *Caridi & Cavaliere* find that heterarchical architectures are the most common, however, they mostly do not make the transition to prototypical or production phase and are very communication intensive while not guaranteeing the optimum [20]. The authors also emphasize the approach of [22] within this area of heterarchical approaches. Also highlighted by [17] within the domain of dynamic scheduling is the approach of [23] and [24].

In his survey [1], *Leitao* points out that purely heterarchical systems are rare to be found, even if it is often stated that the approaches are totally distributed, mainly to achieve global optimization [1]. *Leitao* describes a suitable approach in [25]. He emphasizes ontologies as an important technology to tackle problems of communication and interoperability, which are crucial factors for the development of distributed production control applications [1]. There are several ontologies within the manufacturing domain, [26] being the most suitable for this contribution. This ontology defines the concepts of *Resource*, *WorkOrder* (which has specific start and end times and executes an operation), *Product* and *Operation* as well as further additional concepts like *Disturbances* as well as relations between the concepts, such as the *has allocated* relation between *Resource* and *WorkOrder* [26]. Similarly relevant within the field of manufacturing control are models and meta-models. An interesting approach that, however, is designed for stationary material flow modules (such as conveyers) is presented in [27]. *Fischer et al.* develop an agent-based control system that automatically detects

Table 1: Summary of the conducted literature review.

Contribution	Scheduling granularity	Architecture	Notable features	Notable differences	R1	R2	R4
[22]	Dispatching	Heterarchical	Market/price mechanism, overlaps (fixture and machine)	No explicit scheduling	no	no	no
[10]	Dispatching	Heterarchical	Forecast (estimated tardiness) and relaxed due dates	Assume buffers at machines (before and after)	no	no	yes
[23]	Dispatching	Hierarchical	Supervisor selects dispatching rules	No explicit scheduling	no	no	no
[24]	Scheduling	Holonic	Backwards and forwards messaging	No transport integration, higher communication intensity	no	no	no
[25]	Scheduling	Holonic & Hybrid	Pheromone-based communication, coordination scheme changes	No transport scheduling	no	no	partially
[28]	Scheduling & Dispatching	Hybrid	Integrated with RFID and simulations	Assumes unlimited buffers and does not consider blocking constraints	no	no	yes
[29]	Scheduling	Hybrid (hierarchical elements)	Incorporates preferences from higher levels via voting	During the integrated scheduling only the transport costs are considered	partially	no	partially
[13]	Scheduling	Hierarchical	Considers maintenance and production tasks	Stronger degree of centralization (blackboard & supervisor)	no	no	partially
[30]	Dispatching	Heterarchical	Forecasts of estimated are tardiness	No explicit scheduling	no	no	partially
[31]	Scheduling	Hybrid	2-step negotiations with multiple criteria	Only schedules production step I after step i-1 is finished	no	no	partially

changes in the system configuration, updates the coordinating agent's knowledge base and can calculate possible routes through the system [27]. The meta-model includes concepts like *Ability* (Transport and Manipulation) of a resource or *Orders* that can be processed by the system on specific *Routes*, i.e., combinations of transport and manipulation activities.

The results of the literature review are depicted in Table 1.

All examined approaches are capable of fulfilling Requirement 3 and Requirement 5, therefore they are not displayed in Table 1. These requirements are relevant when comparing the approach to central algorithms, such as a genetic algorithm, or techniques like MILP. However, this is not the focus of this review. The approach that is closest to fulfilling the requirements is the approach of *Badr et al.* The approach develops a hierarchical archi-

tecture where agents are grouped and represented by a group agent to fulfil certain tasks [29]. They coordinate the scheduling of tasks by using auctions that follow the Contract Net (CNET) [32] as well as a voting mechanism that incorporates preferences of agents on higher levels to achieve better global optimization. However, during integrated scheduling, only the transportation costs are included in the approach and the machine schedule is adjusted in case a specific date cannot be matched by the transport. In [25] the well-known holonic ADACOR architecture is described. Within ADACOR, there are holons for products, tasks, operations and supervisory functions, which incorporate coordination and global optimization. An adaptive autonomy factor of the holons leads to hierarchical coordination during normal allocation and heterarchical interactions during rescheduling. In contrast to many other agent-based systems, the communication

is not entirely message based but also uses pheromones to propagate disturbances within the system. However, transportation scheduling within the ADACOR architecture is part of the plan execution, i. e., is requested whenever an operation is finished and not during the planning activity [33]. Wang & Lin propose an approach that solves dynamic scheduling and manufacturing control by integrating the MAS with a simulation (to determine the performances of schedules) and RFID tags, which track the progress of orders within the system [28]. Within this approach, the production schedule is created by applying a centralized high-level scheduling technique, and the agents afterwards try to implement it using a negotiation based coordination scheme. However, if a resource is busy at a specified time, the start of the next operation is simply scheduled at the end of the preceding process without considering if an arrival at that time is possible from a transportation point of view. The approach also does not consider blocking constraints or limited buffers in front of or behind machines. This is also true for many other approaches, like for example [9], where the emphasis lies on incorporating information of queue lengths on production stages and the use of predicted completion times to adjust selection heuristics. Bencheikh *et al.* address the problem of scheduling production and maintenance tasks [13]. A central message storage system or so-called blackboard is used by the agents to communicate. A supervisory agent restricts the access to this blackboard. Objects in this environment serve as containers for potential slots of production and maintenance operations. The schedules are optimized incrementally until the best (applicable) solution is found. However, this approach heavily relies on centralized communication and coordination components. A decentral (dispatching) scheduling is implemented in [30] where the agents exchange information about utilization, due dates, order priorities etc. but there is no exact schedule where start and end dates are defined.

Summarizing the results of the literature it can be stated that no approach models the integrated scheduling problem in sufficient detail regarding the stated requirements.

4 The approach

The dynamic and integrated scheduling problem including resource and sequence-dependent setup times is, as outlined in Section 1 and 2, generally not solvable within polynomial time and therefore has to be solved heuristically [9]. The approach of this paper solves this problem

Figure 1: Exemplary production and logistics system.

with a multi-level heuristic, i. e., heuristics will be used on different levels or steps of the coordination process. The agent paradigm is applied to distribute calculations between different agents. This section explains in detail the agents and the heuristics they apply. The following scenario, which is derived from an industrial project, will serve to exemplify the approach:

In this scenario, there are machines (M) and buffers (B) that are connected by an overhead crane (transport resource (T1)). The dashed line behind buffer 1 indicates that in principle there can be more than one buffer close to that space. M2.1 and M2.2 provide the same capabilities, i. e., they can perform the same operations but can implement different setup and process durations. It is important to mention again that it is not possible to derive general durations of transportation times between production resources as the behaviour of the transportation resources can differ and the total duration depends on the preceding operation and end state (see Requirement 1, Section I).

4.1 Derived agents and interaction scheme

Similar to many other approaches [20, 17, 1], an order agent is created for every order that enters the system. These agents need to interact with resource agents that provide the necessary capabilities (transport, production and buffer) to fulfil the desired operations. Every resource and order agent manages its schedule, i. e., the specific Work Orders it received as well as the planned start and end times of each Work Order. Another agent is responsible for the database connection. This agent receives requests to get information from or store new information into a MySQL database, which serves as the basis for connecting to an MES. In addition to these agents, the approach uses the FIPA specified Directory Facilitator (DF) agent [21] to store information about and find agents within the system.

Figure 2: Possible solution mechanisms and parameters.

The interactions necessary to coordinate the completion of a production order follow the CNET-Protocol [32]. However, the standardized CNET does only display the communication necessary for a single operation. There are two open questions to consider: First, it is important to define how many operations the order agent should incorporate into a single auction run or round of the bidding procedure. The freedom of choice here ranges from organizing one operation after another to organizing all operations at once, i. e., calling for proposals for all operations before making any commitment. The second question to consider is if the time between production operations is initially supposed to be an estimation, which is tried to be filled with a transport operation at a later stage, i. e., the booking procedure of production and transport operations are separated. Figure 2 displays these different possibilities from an already scheduled (base) operation.

The approach here considers only one production stage at a time, i. e., the number of production stages included within one round of auctions is one. The limited amount of stages within one round of negotiations between the agents considerably reduces the algorithm's complexity. The number of possible combinations increases very rapidly with growing system size, especially if there are also different buffers available. For a moderate system with one system entry and exit point, six production stages, four machines on each stage and four transport resources, the number of possibilities already exceeds $4^{13} = 67$ million (without considering buffers). This makes it computationally undesirable to take all possible paths through the system into account.

In addition, while building up possible schedules, the algorithm needs to take already given proposals into account (this 'indecision problem' is also mentioned in [24] where this problem is avoided by preventing simultaneous negotiation of tasks with resources through a task manager that restricts the launch of new tasks for the entire system, but this approach slows down the decision-making process). The approach here uses a locking mechanism

that prevents the resource agent from offering more slots as long as the confirmation or rejection of already given proposals is still missing. The planning process for the outlined scenario does not depend on hard real-time restrictions. The only constraint here, as mentioned in Requirement 5, is that the algorithm reaches a solution within one minute. This favours the approach of the locking mechanism, even though it might lead to marginally longer planning times in case of simultaneously active order agents because it significantly reduces the coordination mechanism's complexity, i. e., number of necessary interactions. It would also be possible to ignore already promised timeslots in the first place and check at a later stage in the process if a given proposition can be fulfilled (or if it is no longer valid as the timeslot has been promised to more than one operation). This, however, increases the necessary interactions considerably as each of those initially given proposals that cannot be fulfilled requires to restart the scheduling process. This is because all steps following the cancelled step are no longer valid either.

Another aspect to consider when including more than one production stage at a time is that the number of possible paths to include within a single offer also rises very quickly. Assume that the agent T1 has proposed to transport a workpiece from M1 to M2.1 and M2.2 at specified times (proposal 1 and 2). Now it is asked to submit proposals for transporting the workpiece from M2.1 or M2.2 to M3 without having received an order confirmation for proposal 1 or 2. For that, it has to consider different starting points for each proposal, as it does not know which proposal (if any) will be accepted. For example, for the transport from M2.1 to M3 it can either be the case that T1 is at M2.1 as a starting point (and has no setup time) or still at the point where it currently resides in Figure 1, which results into the necessary consideration of a setup time. This is because it has not received the order for transporting the workpiece to M2.1 yet.

Regarding the second question, whether the approach should consider the exact transportation times or just esti-

Figure 3: Exemplary operation combinations.

mations in the first place, the decision has been made to incorporate the exact transportation times within an interrelated booking process. It is possible to collect offers for every production stage (assuming here that a product needs one operation on every stage) from every machine, derive a production schedule and then try to implement that schedule by trying to book transportation resources that enable the scheduled production operations (see for example [29]). In the case of transportation resources being the bottleneck of the process, this algorithm is likely to require a significant amount of additional iterations when a production slot is impossible from a transportation point of view. It either has to be repeated with the new transportation constraint imposed, or the machine schedule needs to be adjusted, which is rarely possible because later slots are already promised to other jobs. Also, the procedure of booking the best production process on a certain stage, checking for possible transportation resources for this step, and possibly repeating production scheduling in case of a bottleneck in transportation can lower the solution's quality as well and might lead to an increased load of communication. If one considers both production and transportation in combination, i. e., selecting the best combination of both instead of selecting the best production process first, the algorithm can achieve better results.

A possible set of offers for production step 2, i. e., an operation on machine M2.1 or M2.2, with the included transport and buffer processes is displayed in Figure 3.

Each production proposal that the order agent receives constitutes the base for a possible operation combination (OC). Therefore, an OC includes one production

proposal as well as optional buffer proposals (if necessary, as for OC2) as well as transport proposals for each necessary transport. The number of offers depends on the number of resources that can make an offer for a certain operation. For example, the buffers can be assigned to certain areas, thus offering up capacity only for buffering operations between machines in the specified area. A buffer is necessary if the time between the start of a production process P_i minus the estimated transport time (ETT) and the end of the process P_{i-1} plus ETT exceeds a certain threshold, e. g., 30 minutes (OC2). Otherwise, the workpiece is delivered directly from machine to machine (OC1). Depending on the machine properties, it can be possible to deliver a workpiece to a machine while the setup procedure is still running (Figure 3, B). Otherwise, there can be no "Pickup Overlap" as indicated in Figure 3 because the workpiece can only be delivered after the setup has been completed (Figure 3, A). Figure 3, C indicates that within the offers for one OC there must be no overlap between operations because the order agent might award both to that transport agent. This does not hold for operations in different OCs because these are mutually exclusive – the workpiece will be delivered to either machine M2.1 or M2.2. Note that in case of large workpieces there usually is one order agent per workpiece. However, it is also possible to subsume multiple workpieces within one order (agent), which does not affect on the coordination principles. It only possibly influences the number and thus the duration of operations that have to be performed on the order.

To determine the best OC, each possible path is calculated and selected according to defined criteria, e. g., earli-

Figure 4: Order agent's control flow.

est finish time or smallest setup duration. This selection is applied on multiple levels. First, when calculating the best single path within an OC, i. e., selecting the transport and buffer operations (for a specific production operation) that are most suitable according to the defined criteria. Second, when choosing between different OCs, i. e., different production proposals and their corresponding buffer and transport proposals. Third, when calculating appropriate intervals for a proposal, a resource agent also applies this selection heuristic, e. g., selecting the interval with the earliest arrival date (see Section 4.3). Figure 4 displays the order agent's control flow during the collection of proposals for one production stage.

After the best possible path has been determined, the order agent accepts the proposals that are included within the best path and rejects the rest. This process is repeated for every production stage. If, however, no transport proposal meets the desired criteria, i. e., starting and finishing within the possible timeframe, the process is restarted with the earliest possible delivery date as an input for new production proposals. The algorithm in its current form can terminate in a state where there is no feasible solution to proceed. This is due to the strict adherence to Requirement 3 (no changes allowed to already scheduled operations), which for example yields no feasible solution if there is no transport resource that can pick up the work-

piece in time for the next (already scheduled) operation to start. The occurrence of such instances is relayed to a human operator, who has to rearrange the tasks until a new feasible solution is possible, i. e., relaxes Requirement 3.

The following paragraph examines the calculation of possible intervals as well as the necessary exchange of information between the agents in more detail.

4.2 Adjustments of ontology concepts and information exchange

The coordination algorithm in principle follows the well-known CNET Protocol, which is a simple single-round auction protocol [32]. In this protocol, an initiating agent requests participants to make offers (Call for Proposal, CFP), which are answered positively or negatively in turn. The emphasis here, however, lies on the information content within the CFP, proposal and accept proposal messages as well as the algorithm that is used by the agents to calculate possible intervals. The basis for the information exchange is an ontology that builds on the already mentioned ADACOR ontology [26]. Following the use case described in Section 1, not all concepts (and relations) of the ADACOR ontology are necessary and in some cases, they need to be further refined.

The concept of the Work Order has been adjusted: The scheduled start and end dates within a Work Order are subsumed within an additional concept "timeslot" that also has the attribute "length" which simplifies further calculations. The Work Order is further enhanced with the attributes "BufferBeforeTimeslot" and "BufferAfterTimeslot". These attributes indicate the flexibility or slack of the proposed timeslot, i. e., the time a scheduled operation can start earlier or finish later. As these concepts are also used by the agents internally, there is also the addition of the object properties "StartState" and "EndState" which are used by the resource agents to capture the start and end states for each operation they are requested to make a proposal for. These states can either be locations (see next paragraph) in case of transport resources or "setup states" that indicate a certain configuration of the equipment of production resources. For this use case, there is no need to depict the setup as a separate concept itself (as it is modelled in the ADACOR ontology). Rather it is handled as a simple data property (e. g., a float number) that reflects its duration and is dynamically filled according to the actual and desired start states. Note that a setup can be necessary not just for production operations but also for transport operations. In the latter case, it simply reflects the time needed to reach the workpiece from the last destination

before that order. Within the CFP messages to production agents on stage n , the order agent sends the requested operation, the quantity, as well as a timeslot that contains the earliest start and (desired) latest end. The earliest start is calculated by either the release date of the order or the scheduled end of production step $n-1$ plus the ETT. The latest end can be used if for example a backwards scheduling algorithm is implemented that assigns a planned production time to each stage (see for example [28]). The proposal by the production agents include duration and setup times as well as the price, buffer time before and after the operation, the exact timeslot and information about the resource, i. e., name and location. The price can be calculated in multiple ways, for example reflecting the utilization of a machine [22] or reflecting costs or profit margins [12]. In the approach here, it is simply calculated by summing up the setup and operation duration as well as the time increment (see next paragraph). The order agent uses this information to either first get proposals for buffers or directly create the CFP messages for the transport agents (that contain end of operation $n-1$, start of operation n , locations and quantity for each separate operation).

4.3 Applied heuristics by the agents

In the process that is depicted in Figure 5, the transport and production resource agents calculate intervals for each requested operation.

The parameters that are calculated in step 1, Figure 5 constitute all aspects of the proposal that do not depend on the exact timeslot (and its predecessor and successor) to be calculated. These include the number of tours or operations needed (calculated from the resource's capacity and the quantity) and the duration of process without setup. Afterwards, the agent goes through the list of its free intervals, starting from the earliest possible, until a suitable slot is found. First, the dependent parameters are calculated (step 2, Figure 5). These are the setup time and the time increment (TI). The latter marks the duration by which the *following* Work Order's timeslot must be adjusted, due to the addition and end state of the current Work Order within that free interval. This TI can be both positive and negative depending on the necessary start state of the following process. Note that this TI has to be accounted for also in case of calculating if a certain operation can be fulfilled within a specified timeslot.

After the dependent parameters have been calculated (step 2 in Figure 5), the possible slots for the given free interval are determined (step 3, Figure 5). The slots mark different possible start- and endpoints that stem from the

Figure 5: Resource control flow for proposal calculation.

given timeslot of the CFP, the buffer before the operation as well as the resource agent's specific free intervals. These slots are analysed in two steps. First, it is checked if they are feasible with regards to the given constraints of the CFP, i. e., if the estimated start is after the earliest start and the estimated end is later then the earliest end (step 4, Figure 5). Second, the remaining slots are checked against the agent's own schedule to determine if the start of the interval minus the setup duration is greater than the lower bound of the free interval and if the end of the proposed in-

terval is smaller than the upper bound of the free interval (step 5, Figure 5). If one or more slots remain, these slots will be sorted, e. g., by earliest arrival, and the best one is chosen to be included in the proposal. If there is no slot remaining, the next free interval will be analysed until a suitable slot is found. However, this slot can end after the latest end (from production resource side) and start after the latest start (from production resource side) as these criteria might not be achievable by the transport agent. The latest start and latest end are not included into the feasibility analysis at this point because otherwise, the agent would possibly disregard all intervals thus sending no offer at all. It is advisable, however, to send an offer with the earliest timeslot possible from transport resource side as an indication for the next round of the coordination process. After the most suitable slot for a CFP has been determined, the agent calculates the price (step 6, Figure 5) and creates the respective proposal. This proposal is a concept of the ontology as well that contains the Work Order, the price as well as the already mentioned parameters. All of the proposals that are created for the response to a CFP are included within one proposal message to the order agent.

The order agent starts the calculation of OCs either after having received all proposals or after the specified deadline has expired. The calculation of the best possible OC is an iterative process. The order agent creates one OC for each production proposal. Within one OC, a separate path is created for each buffer proposal, which means that the agent analyses $m * b$ possible path for a single production stage, where m is the number of production resources and b is the number of possible buffers. The transport proposals are assigned to these paths by start and end locations. Each transport operation is analysed and selected by applying the defined criteria, e. g., choosing the transport operation for M1_M2.2 with the earliest finish date and the least setup time. When all OCs are calculated the total setup time and the finish date for each OC is calculated and the best one is selected. After the order agent has calculated the best path it confirms the respective proposals with an accept proposal message. This message contains the exact start and end dates that the OC requires, e. g., if an accepted slot uses the buffer time before the operation the start and end dates are transmitted accordingly. The accept proposal message also contains the actual pickup times that are associated with the transport resource that is considered. Before, the production resource agent could only use an estimation as these times can differ between different resources. After the operation combination is confirmed, the order agent also sends an “Inform Departure” message to the production resource $n - 1$ as this departure date was not known before. If the LS date

is violated at this point, i. e., the earliest transport possible to a buffer or production resource n is too late for the production resource $n - 1$, the already scheduled plan cannot be adhered to and changes have to be made. For example, later operations have to be postponed.

Note that the order agent does not manage the same times in its schedule as the resource agents. This is because the resource agents do not relay the starting times of their setup operations but only of the production process. Due to this divergence in schedules, there are two separate sources for identifying the current state of the production system. First, the order agent’s schedule, which at all times represents the physical location of the workpiece. Second, the schedule from a resource perspective that also accounts for setup times. These two sources need to be separate because, as already explained, the resources’ setup times are dynamically adjusted with the ongoing scheduling procedures. Therefore, the order agent’s schedule does not contain the updated information at all times.

5 Practical application and evaluation

To exemplify the approach, a scenario based on an industrial application has been chosen that highlights the behaviour in an on-line scheduling case. An order of 10 pieces of a product P1 was introduced to the system described in Section 1 with the production plan M1–M2–M4–M5 (subsumed in one order) with necessary transports between every machine (only the transport to M1 is disregarded). The sequence-dependent setup times (as well as the process times per piece) for the example are 15 minutes from P1 to P2 (resp. the operation time for products P1 and P2) and 30 minutes from P2 to P1 respectively for every machine. The product P2 is an alternative product that requires a different configuration. The speed of the transportation resource is set to 0.5 m/s and the pickup time to 10 minutes, which are realistic assumptions from the industry project for crane transports. The transport resource can transport all ten pieces within one operation (capacity ≥ 10). The pickup times as well as the resource’s speed are adjustable according to the needs of the applicable use case and are stored in a database that is used by the transport agent at its creation time. The following table shows the a priori scheduled timeslots that the agents derive from the database and integrate into their schedule. The states in column 4 also reflect the start state of each resource. The start state and end state are separated as the start state

Figure 6: Scheduling result for the application scenario.

Table 2: Predefined blocked slots.

Resource	Start [time]	End [time]	State (Start-End) (configuration / [x,y])
M2.1	16:15	18:15	P1-P1
M2.1	00:00	08:00	P1-P1
M2.2	16:15	22:30	P1-P1
M4.1	16:15	20:00	P1-P1
T1	19:30	21:00	[5,5]–[5,5]

is, as already mentioned, dynamically adjusted and influences the necessary setup time.

The following figure displays a total schedule for the order from the resource side as well as the schedule from the order side. In this scenario, parallel delivery to a machine while the machine is still in its setup routine is not possible. The order agent's schedule tracks the location of the order (and its workpieces) without considering the overlaps due to the pickup process at the machine. These are disregarded for reasons of easier throughput time calculation. However, these overlaps are included in the total schedule.

All dates from order and resource perspective are stored within the database. Upon receiving an order completion notice, this data is used by another agent to create the GANTT charts. The interactions as well as parts of the information that the agents exchange for proceeding one production step within the coordination process is displayed in Figure 7.

This scenario highlights different aspects of the agent system's behaviour. First, the slot at M2.1 between the busy intervals is big enough to process the order. However, the

blocked interval of T1 from 19:30 to 21:00 makes it impossible to pick up the workpiece from M2.1 in time, as M2.1 only has a buffer after the operation of 100 minutes (22:05 + 100 minutes = 23:45 + 15 minutes time increment for the additional setup for the next task). The earliest possible arrival from T1 at 21:30, however, would result in a later finish and thus conflict with the already scheduled interval. Therefore, the OC that uses M2.2 is chosen, even though it requires an additional buffer process. Second, the difference in the proposed prices from M2.1 and M2.2 also reflects the time increment. If the proposed slots were alike and the transport resource would be able to fulfil both, this price difference would lead to the selection of M2.2 over M2.1. Third, the order agent accepts the proposal from M4.1 over M4.2, even though the proposals by these agents are exactly alike. This is because the combination of transport and production is slightly better in case of M4.1 because the distance to cover from M2.2 to M4.1 is shorter and therefore M4.1 can start and finish earlier. Fourth, the approach is quite communication-intensive. To reduce the number of messages, multiple CFPs and proposals are included within one respective message (as indicated in Figure 7 “cfps” and “proposals”). Each resource receives a CFP message, answers with a proposal and receives feedback for that message (if necessary, an “inform scheduled” messages can be added here as well). The resource agent of stage $n-1$ also receives an update of the departure time. Afterwards, every agent requests a database entry. Thus, the number of necessary messages for one production stage can be estimated at $4 * (m + b + t) + 3$ where m = number of production resources for that stage, b = number of buffers and t = number of transport resources. For the depicted scenario, this results in 23 messages for a stage with

Figure 7: Booking interactions for one production stage.

two production and buffer resources plus additional messages for registration and agent search messages to the DF-Agent.

The agent system has been implemented in JADE [34]. JADE is a framework for agent-based systems in Java that provides basic classes and communication tools. An agent in the JADE framework carries out different behaviours that can, for example, be triggered by certain events or run continuously until the agent is taken offline. Each agent, and if necessary specific behaviours, run in their own thread, which enables parallel processing of tasks. This speeds up the coordination process as the agents send updates of their schedule to a dedicated database agent, which enters the data into the database and runs in a separate thread.

An evaluation has been carried out to analyse the application's performance during runtime. The following figure illustrates the timing behaviour of the system with an ongoing arrival of orders.

Figure 8 displays that the duration of the coordination is about 230 ms and increases only slowly with the number of orders in the system. The increase can be explained by the increasing number of intervals that have to be checked for each order. The products A and B represent different process plans that encompass four and respectively five pro-

Figure 8: Analysis of the coordination's duration over number of orders.

duction steps. The difference regarding the average duration is negligible.

Figure 9 shows the orders grouped by the number of processes (i.e., production, transport and buffering) which need to be scheduled (9, 11, or 13 processes, respectively). As can be seen in Figure 9, this does not influence the duration of the calculation.

This was also approved in a further test, which revealed that doubling the number of production resources

Figure 9: Analysis of the coordination's duration grouped by number of processes.

on a production stage does not influence the runtime of the algorithm significantly. The runtime increases, however, with more transport resource agents being added (approx. increase of 15 per cent for twice the number of transport resources).

The experimental system consisted of the resources described in Figure 1 plus an additional transport resource. The experiments have been conducted on a Windows 7 Notebook with Intel® Core™ i7-3610QM CPU @ 2.30 GHz and 16,0 GB RAM. The duration of the coordination scheme is in line with the requirements (no hard real-time restrictions).

To evaluate if the created plans are feasible and to perform future performance evaluations two scenarios have been implemented coupled to discrete-event simulations. The first simulation models a large factory with multiple cranes that are operated by a scarce number of operator teams. A detailed description of the simulation model and feasibility results can be found in [35]. The plans created did comply with all restrictions (operator scarcity and physical constraints of the shared crane rail). Another simulation has been created for the scenario outlined in Section I. There, plans that include transports, as well as plans without transport consideration, have been created and analysed. The agent system can cope with both procedures. Here as well the plans created by the MAS were feasible in a sense of all physical constraints being matched (blocking of machines during pickup, no setup during waiting times etc.).

6 Summary and outlook

Even though there is abundant research on scheduling in general as well as agent-based systems in manufacturing control, there is still a lack of practical applicability of

many approaches [20]. This contribution is intended as a further step towards practical application of agent technology by first incorporating human traits and constraints within the algorithm by formulating the plan stability as an important criterion. Second, it tries to solve the problem of creating feasible plans for the integrated production and transportation problem that on the one hand meet all physical restrictions of the manufacturing scenario and on the other hand provide accurate information about pickup times and possible dates to start the manufacturing operations. For this, the concept of the time increment is developed that dynamically adjusts already scheduled orders to display an exact picture of possible operations. The inherent flexibility of the agent-based approach enables it to cope with different process plans as well as varying resource-dependent and dynamic process times. It also incorporates sequence and location-dependent setup times as well as the already described requirements of buffer scarcity and heterarchical architecture. The approach is especially suitable for applications where resources are added and withdrawn on short notice, for example, due to reconfigurations or even in case of connection issues as the algorithm does not depend on completeness during the coordination process. The decentralised architecture and applied heuristics make the calculation time very robust to large-scale applications that do not have a central entity to supervise the scheduling process. The approach can also be applied for scheduling within frozen zones, as it does not change any scheduled operations. These characteristics, as well as the ability to exactly schedule the transport operations within a factory, renders the approach a suitable means for controlling production and transport processes in an integrated manner – especially for small-scale production sites where transportation operations can be a bottleneck.

To summarize, the agent approach provides feasible and practically applicable solutions for production scheduling, i. e., as described in the Application Scenario “Order Controlled Production” of the Platform Industry 4.0 [4], [36].

A current drawback of the approach, as pointed out in Section 4, is that the adherence to the plan (according to Requirement 3) does not always result in desired outcomes. If, for example, an operation can be postponed without interrupting the production process, this should be incorporated into the solution-finding process. Furthermore, additional analyses need to be conducted regarding the algorithm's scalability for other scenarios as well as the effects of wrong assumptions by the agents, e. g., regarding the ETT or pickup time and different algorithms

of price calculation. Further research will look at the applicability of the algorithm to rescheduling problems that incorporate the handling of different disturbances as well as fine-tuning the plan stability criterion to achieve better results. Another aspect of further research will be the detailed analysis of the approach's performance in comparison to state of the art dispatching rules and other control algorithms as well as a statistical analysis of the different factors that influence the duration of the coordination process.

Funding: This work has been supported by the German Federal Ministry of Economy and Energy in the frame of the ZIM Programme under grant no. FKZ 16KN069722. All responsibility for the content remains with the authors.

References

1. P. Leitão: Agent-based distributed manufacturing control: A state-of-the-art survey. *Engineering Applications of Artificial Intelligence*, Vol. 22 (7), 2009, pp. 979–991.
2. P. Leitão, N. Rodrigues, J. Barbosa, C. Turrin, A. Pagani: Intelligent products: The grace experience. *Control Engineering Practice*, Vol. 42, 2015, S. 95–105.
3. T. Bauernhansl, M. ten Hompel, B. Vogel-Heuser: *Industrie 4.0 in Produktion, Automatisierung und Logistik – Anwendung, Technologien, Migration*. Springer, Wiesbaden, 2017.
4. *Plattform Industrie 4.0: Platform Industrie 4.0 Aspects of the Research Roadmap in Application Scenarios I4.0*. Berlin, Germany, 2016.
5. F. Thiesse, E. Fleisch: On the value of location information to lot scheduling in complex manufacturing processes. *International Journal of Production Economics*, Vol. 112 (2), 2008, pp. 532–547.
6. P. Nyhuis, J. Mayer et al. (Hrsg.): *Aktuellen Herausforderungen der Produktionsplanung und -steuerung mittels Industrie 4.0 begegnen: Studienergebnisse*. Garbsen: PZH Verlag, October 2016.
7. G. Schuh, P. Nyhuis, C. Reuter, A. Hauptvogel, S. Schmitz, J. Nywlt, F. Brambring, F. Schulte, J. Hansen: *Produktionsdaten als Enabler für Industrie 4.0: Gemeinsame Studie der produktionstechnischen Institute IFA, IPMT, Fraunhofer IWU und WZL*. *wt Werkstattstechnik online*, Vol. 105 (4), 2015, pp. 200–203.
8. I.N. Pujawan, A.U. Smart: Factors affecting schedule instability in manufacturing companies. *International Journal of Production Research*, Vol. 50 (8), 2012, pp. 2252–2266.
9. W. Weng, S. Fujimura: Flexible Flow Shop Scheduling by Intelligent Multi-agents. In: Ormandjieva (Hrsg.): *Eighth ACIS International Conference on Software Engineering Research, Management and Applications (SERA)*, 2010: 24–26 May 2010, Montreal, Canada; proceedings. Piscataway, NJ: IEEE, pp. 113–120, 2010.
10. J.-S. Liu, K. P. Sycara: Multiagent Coordination in Tightly Coupled Task Scheduling. In: Tokoro (Hrsg.): *Proc. Second International Conference on Multi-Agent Systems*, Dec 10–13, 1996, Kyoto, Japan. Menlo Park, Calif.: AAAI Press, 1996.
11. A. Mascis, D. Pacciarelli: Job-shop scheduling with blocking and no-wait constraints. *European Journal of Operational Research*, Vol. 143 (3), 2002, pp. 498–517.
12. F. Gehlhoff, H. Nabizada, A. Fay: Optimization of multi-agent auctioning processes in flexible production networks. In: *IEEE 17th International Conference of Industrial Informatics INDIN 2019*, July 22–25, 2019, Helsinki, Finland.
13. G. Bencheikh, A. Letouzey, X. Desforges: Scheduling of production and maintenance activities using multi-agents system. In: *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*.
14. A. Fazlirad, R.W. Brennan: Multiagent Manufacturing Scheduling: An Updated State of the Art Review. In: Vogel-Heuser (Hrsg.): *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*: 20–24 Aug. 2018. Piscataway, NJ: IEEE, S. 722–729, 2018.
15. M. Paolucci, R. Sacile: *Agent-based manufacturing and control systems: New agile manufacturing solutions for achieving peak performance*. Boca Raton Fla.: CRC Press, 2005.
16. M. I.L. Pinedo: *Scheduling*. New York, NY: Springer New York, 2008.
17. W. Shen, L. Wang, Q. Hao: Agent-based distributed manufacturing process planning and scheduling: A state-of-the-art survey. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, Vol. 36 (4), 2006, pp. 563–577.
18. J. Sgall: Woeginger, Fiat (Hrsg.): *Online Algorithms: The State of the Art*. Vol. 144. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998 (Lecture notes in computer science Online algorithms).
19. N.R. Jennings, S. Bussmann: Agent-based control systems: Why are they suited to engineering complex systems? *IEEE Control Systems*, Vol. 23 (3), 2003, S. 61–73.
20. M. Caridi, S. Cavalieri: Multi-agent systems in production planning and control: An overview. *Production Planning & Control*, Vol. 15 (2), 2004, pp. 106–118.
21. FIPA – Foundation for Intelligent Physical Agents, <http://www.fipa.org/repository/standardspecs.html>.
22. G.Y.-J. Lin, J.J. Solberg: Integrated Shop Floor Control Using Autonomous Agents. *IIE Transactions*, Vol. 24 (3), 1992, pp. 57–71.
23. K. Kuiss, H. Pierreval, N. Mebarki: Using multi-agent architecture in FMS for dynamic scheduling. *Journal of Intelligent Manufacturing*, Vol. 8, 1997.
24. P. Sousa, C. Ramos: A dynamic scheduling holon for manufacturing orders. *Journal of Intelligent Manufacturing*, Vol. 9 (2), 1998, pp. 107–112.
25. P. Leitão, F. Restivo: A holonic approach to dynamic manufacturing scheduling. *Robotics and Computer-Integrated Manufacturing*, Vol. 24 (5), 2008, pp. 625–634.
26. P. Leitão, F. Casais, F. Restivo: Holonic Manufacturing Control: A Practical Implementation. In: Camarinha-Matos (Hrsg.): *Emerging solutions for future manufacturing systems: IFIP TC 5/WG 5.5 sixth IFIP International Conference on Information Technology for Balanced Automation Systems in Manufacturing and Services*, 27–29 September 2004, Vienna, Austria. New York: Springer Science & Business Media, Vol. 159, pp. 33–44, 2005.

27. J. Fischer, M. Marcos, B. Vogel-Heuser: Model-based development of a multi-agent system for controlling material flow systems. at – Automatisierungstechnik, Vol. 66 (5), 2018, S. 438–448.
28. L.-C. Wang, S.-K. Lin: A multi-agent based agile manufacturing planning and control system. Computers & Industrial Engineering, Vol. 57 (2), 2009, pp. 620–640.
29. I. Badr, F. Schmitt, P. Göhner: Integrating Transportation Scheduling with Production Scheduling for FMS: An Agent-Based Approach. In: IEEE International Symposium on Industrial Electronics (ISIE), 2010: Bari, Italy, 4–7 July 2010. Piscataway, NJ: IEEE, pp. 3539–3544, 2010.
30. O.J. Shukla, G. Soni, R. Kumar, S. A: An agent-based architecture for production scheduling in dynamic job-shop manufacturing system. at – Automatisierungstechnik, Vol. 66 (6), 2018, S. 492–502.
31. M. Klein, A. Löcklin, N. Jazdi, M. Weyrich: A negotiation based approach for agent based production scheduling. Procedia Manufacturing, Vol. 17, 2018, S. 334–341.
32. R.G. Smith: The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. IEEE Transactions on Computers, Vol. C-29, (12), 1980, pp. 1104–1118.
33. P. Leitão, An agile and adaptive holonic architecture for manufacturing control, PhD thesis, Univ. of Porto, Portugal, 2004.
34. F. Bellifemine, G. Caire, D. Greenwood: Developing Multi-Agent Systems with JADE, Chichester 2007.
35. J. Sundermeier, F. Gehlhoff, A. Fay: Development of a simulation model to analyze the performance of decentral rescheduling algorithms in production systems. In: C. Deatcu, T. Schramm, K. Zobel (Hrsg.): Tagungsband ASIM 2018, 24. Symposium Simulationstechnik, ARGESIM Report Nr. 56, S. 275–283, 04.–05. October 2018, Hamburg.
36. A. Fay, F. Gehlhoff, M. Seitz, B. Vogel-Heuser, H. Baumgärtel, C. Diedrich, A. Lüder, T. Schöler, G. Sutschet, R. Verbeet: VDI-Statu Report Agents for the realisation of Industrie 4.0, Juli 2019. Platform Industrie 4.0.

Bionotes

Felix Gehlhoff

Institute of Automation Technology,
Helmut-Schmidt-Universität / Universität
der Bundeswehr Hamburg, Holstenhofweg
85, 22043 Hamburg, Germany
felix.gehlhoff@hsu-hh.de

M. Sc. Felix Gehlhoff (born 1988) graduated in Industrial Engineering and Business Management (Wirtschaftsingenieurwesen) in 2017. He is a research assistant at the Institute of Automation Technology at Helmut Schmidt University. His main research interests are agent-based control solutions for production and logistic systems.

Alexander Fay

Institute of Automation Technology,
Helmut-Schmidt-Universität / Universität
der Bundeswehr Hamburg, Holstenhofweg
85, 22043 Hamburg, Germany
alexander.fay@hsu-hh.de

Prof. Dr.-Ing. Alexander Fay (born 1970) is Director of the Institute of Automation Technology at Helmut Schmidt University Hamburg. His main research interests are models, methods, and tools for the efficient engineering of distributed automation systems. Prof. Fay is member of the board of the German association for Measurement and Automation (VDI-/VDE-GMA).