
Organic Computing

Doctoral Dissertation Colloquium 2020

S. Tomforde, C. Krupitzer (Editors)






This document – excluding quotations and otherwise identified parts – is licensed under the Creative Commons Attribution-Share Alike 4.0 International License (CC BY-SA 4.0: <https://creativecommons.org/licenses/by-sa/4.0/>)

Bibliographic information published by Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at <http://dnb.dnb.de>.

ISBN 978-3-7376-0945-6

DOI: <https://doi.org/doi:10.17170/kobra-202103173535>

© 2021, kassel university press, Kassel
<https://kup.uni-kassel.de>

kassel
university 
press

Printing Shop: Print Management Logistics Solutions, Kassel
Printed in Germany

Research Challenges in Adaptive Production Systems

Martin Neumayer

Institute for Software & Systems Engineering, University of Augsburg, Germany
neumayer@isse.de

Abstract. In times of personalised products, fluctuating demands and ever-increasing complexity in hard- and software, production systems crave for flexibility and robustness. Self-organisation can help to achieve these goals as self-organising systems autonomously monitor themselves and their environment and adapt to changes observed. Despite extensive study, researchers have hardly addressed some aspects of self-organising production systems. Therefore, we identify three areas to contribute to the vision of self-organising production systems: We plan to extend product descriptions to be more realistic. We further intend to investigate extensions to dynamic scheduling in self-organising production systems. Lastly, we present an approach to avoid deadlocks in self-organising production systems that handle multiple types of products at once.

Keywords: Self-organisation, Production systems, Manufacturing systems, Autonomous systems.

3.1 Introduction

This section introduces the paradigms of organic computing and adaptive systems. It further motivates the application of these paradigms to the manufacturing domain. Lastly, it covers previous work on a special class of adaptive systems, so-called product-flow systems to conclude with open research questions that have hardly been discussed in previous work.

3.1.1 Organic Computing and Adaptive Systems

Organic Computing [40] is an initiative that aims to develop technical systems that exhibit life-like properties, often found in biological systems. Most prominently these life-like properties include robustness and flexibility against disturbances [39]. To achieve these properties, Organic Computing systems observe their environment and adapt autonomously to changes observed by manipulating their environment accordingly. This involves a paradigm shift: Instead of human engineers taking decisions at design time, we are now facing adaptive systems deciding at runtime [23].

To fulfil these requirements, Organic Computing systems are designed to feature self-* properties, including self-configuration and self-organisation. Self-configuration is the ability of a system to change its parameters according to user goals. Self-organisation describes systems autonomously changing their structure to accomplish higher-level goals [40].

3.1.2 Motivation

The vision of applying the paradigm of adaptive systems to manufacturing is long-standing, with publications dating back to the nineties [24]. Since then, the topic has gained additional traction, as the manufacturing domain experiences a shift from mass production to producing customised and even individual products. This shift is accompanied by volatile markets and fluctuating demand. At the same time, production systems consist of many increasingly complex and interconnected hard- and software components. To adapt to these new circumstances, manufacturers focus on gaining flexibility and robustness instead of solely increasing throughput. Adaptive manufacturing systems offer a way to gain these properties:

1. **Robustness:** Adaptive production systems can deal with partial breakdowns by detecting faults and finding new paths of production at runtime.
2. **Flexibility:** Adaptive production systems offer flexibility in terms of the products manufactured and their quantity. As long as the needed capabilities for a new product exist in the system, agents in the system can find new paths applying the methods mentioned above. Adaptive production systems also enable flexibility in terms of the objectives pursued, such as high throughput or low energy consumption.

3.1.3 Background

One way of reaching robustness and flexibility for a special class of systems has been explored in previous work [14, 26, 35], so-called *resource-flow systems* or *product-flow systems*¹. Product-flow systems contain *agents* dispatching, transporting, processing or collecting products. *Storages* are agents dispatching and collecting products. Agents, transporting products from one processing agent to another, are referred to as *autonomous guided vehicles* (AGVs). *Processing agents* may offer several *capabilities* to process a product, such as drilling. A *task*, the blueprint on how to manufacture a product, is described as a sequence of capabilities. Matching the capabilities and transports needed to manufacture a product and the capabilities offered by the agents is termed *reconfiguration*. Reconfiguration can be seen as a form of task allocation [6] or as a scheduling subproblem. The problem of reconfiguration is formulated as a Constraint Satisfaction Problem (CSP) [3]. This CSP can then be solved at runtime in different ways, e.g., centrally using a constraint solver [25] or through coalition

¹ In contrast to previous work, we prefer the notion of product-flow systems as the word resource is ambiguous in the manufacturing domain: It can serve as a term for a machine as well as for a product [6].

formation [26]. We denote the result of reconfiguration, i.e., the product's path through production, as *product flow*.

3.1.4 Research Questions

Despite these promising characteristics and ongoing research, some issues have hardly been discussed in previous work. Therefore, we plan to contribute to the vision of adaptive and self-organising production systems, especially product-flow systems. Our research is guided by three partially interconnected questions.

How to ensure wide applicability? Previous work [26, 35] shows that adaptive production systems can be implemented. However, there are still limitations, e.g., avoiding deadlocks while supporting multiple types of products at a time [37] or allowing task descriptions beyond ordered sequences of capabilities [28]. We plan to extend previous work to overcome these limitations and therefore ensure applicability.

How to ensure performance and scalability? Overcoming limitations such as deadlocks and simplified task descriptions might increase complexity. Thus, we have to re-evaluate the methods used, also considering the scalability necessary for practical application. Concrete research questions subsumed by this main question are whether the constraint-based approach is still suitable and whether decentralisation in the sense of distributed constraint optimisation can increase performance and scalability.

How to achieve openness for human intervention? Lastly, being open for human intervention is one integral feature of organic computing systems [33, 36, 40]. However, this aspect has hardly been studied in the context of adaptive production systems. Therefore, we want to investigate the role of humans in adaptive production systems: How can humans intervene and pose new constraints to adaptive production systems? Is operation according to user-given constraints opposed to performance? Or can human expertise help to relax problems?

From these research questions we derive three research challenges in Section 3.2: Section 3.2.1 discusses the shortcomings of modelling tasks as a sequence of capabilities and presents our planned contributions to address the problem. We cover approaches towards the problem of dynamic scheduling in Section 3.2.2. In Section 3.2.3, we examine the problem of deadlocks in self-organising production systems and briefly summarise an approach to avoid deadlocks in adaptive production systems manufacturing multiple types of products at once. Section 3.3 concludes this paper.

3.2 Research Challenges

This section presents the identified challenges in greater detail. The description of the individual problems adheres to the following structure: Related work, our (planned) contribution and plans to evaluate our contribution.

3.2.1 Realistic Descriptions of Task and Capabilities

3.2.1.1 Related Work

Several publications describe a task as an ordered sequence of capabilities that are executed one after another, altering one particular product [26, 41, 45]. This modelling of a task contradicts practice in many areas of application [28]. E.g., in the furniture industry wooden panels are sawn into several workpieces that are machined individually and later assembled to make up the final product [28, 38].

Keddis et al. [17] refer to splitting an intermediate product or raw material into several as a fork task. A fork task also splits the production process into two parallel processes. In contrast, a synchronisation step synchronises two or more parallel processes. Furthermore, there are cases where capabilities can be replaced by other capabilities (selective tasks) or executed in arbitrary order [17]. Qiao et al. describe similar structures in [32]. Figure 3.1 visualises the different task structures mentioned.

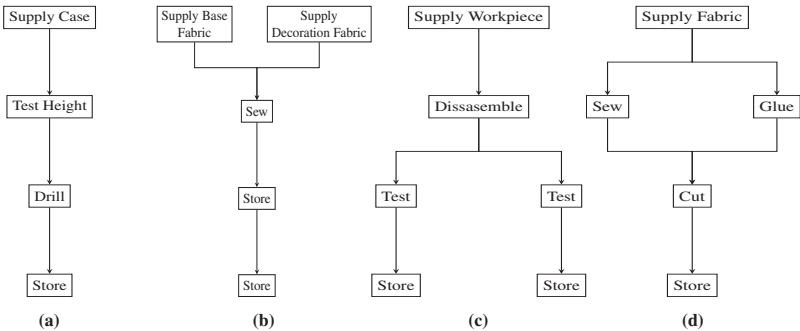


Fig. 3.1. Visualisation of different task structures according to [17]: (a) Sequential task, (b) synchronisation task, (c) fork task, (d) selective task.

The modelling of capabilities has to be more realistic as well. E.g., stating that an agent can perform the capability ‘drill’ does not satisfy the need for practical application. A realistic capability description encompasses parameters describing the material, geometry, and process [17]. A description of the materials used is needed to determine whether an agent can perform the required capability: An agent might be able to drill a piece of wood, while the same agent might not be able to drill a piece of metal. Alike, a description of the product’s geometry is needed to check whether an agent can handle and execute a capability on a product. Due to specific grippers or fixtures, geometry may prevent the execution of a capability. Lastly, process-related information is needed. In our drilling example, we might need to know the exact position, depth, and diameter of the hole. Process-related information should also contain auxiliary materials, such as screws if needed. Depending on the process, related information can take different forms. Therefore, flexible data structures are required.

The increase of capabilities combined with the variety of data needed to describe a task might also turn the modelling of tasks into a tedious and error-prone duty. Here another challenge arises: Generating valid task descriptions from user input [31] such as 3D models. E.g., Lau et al. demonstrate how 3D models of furniture can automatically be split into parts and connectors, using formal grammars [19].

3.2.1.2 Contribution

Based on the related work presented, we identify the following areas of contribution:

1. Survey of descriptions: While the authors in [17] present a solution for the realistic description of task and capabilities, they also state that there might be other methods, e.g., the Business Process Model and Notation (BPMN). A survey will help to compare different approaches and identify the advantages and disadvantages of the approaches.
2. Implementation: After comparing different approaches, we will implement one or several promising approaches for realistic task and capability description. The implementation should include a user interface to create task descriptions, as well as suitable data structures. Using a graph-based structure seems promising.
3. Finding suitable approaches to task allocation and product routing: Differentiating between capabilities with different parameters will lead to an increase in overall capabilities. Together with a realistic description of tasks, the problem of task allocation might turn out more complex. We will have to re-evaluate the constraint-based approach and compare it to other approaches to clarify, which approaches are best suited to these requirements.
4. Generating task descriptions from user input: The approach of Lau et al. [19] seems like a first step in this direction. We plan to reimplement and extend the approach to generate a task description from the parts and connectors.

3.2.1.3 Evaluation

We plan to evaluate the contributions on a showcase basis, i.e., we will provide some showcase products, possibly from the furniture domain, and check if the implemented task and capability description can capture these products. Further, we can compare different methods of task allocation with the provided descriptions, e.g., in terms of runtime. Finally, using a 3D model of the showcase product, we can verify that a valid task description can be generated automatically.

3.2.2 Dynamic Scheduling

3.2.2.1 Related Work - Traditional Scheduling Approaches

Controlling production facilities is a well-studied subject. Researchers have been studying job shop scheduling problems (JSSP) as an NP-hard combinatorial optimisation problem since the 1950s. In a job shop, there is a finite set of products or jobs to

be processed on a finite set of machines. Every product might have a different task, comprised of a set of capabilities. These capabilities must be performed in the given order. Every machine is specialised for its operation, i.e., it offers only one capability. Also, machines can only process one product at a time without the possibility of preemption [5].

With the advent of flexible and reconfigurable manufacturing systems, where machines can perform different capabilities [18], the focus of research has extended to the flexible job shop scheduling problem (FJSSP). Here a machine may offer several capabilities, but switching between capabilities requires a setup time. Thus, an FJSSP can be divided into two subproblems [5, 41]:

1. Assignment of operations to suitable machines.
2. Sequencing of operations on all selected machines to obtain a schedule.

We additionally focus on the subproblem of routing and transporting the products to the machines selected in the assignment. Research on the classical FJSSP often neglects this aspect [5]. The notion of job shop scheduling problems with transportation resources [29] extends the JSSP by a set of identical vehicles that can transport any product. Whenever a product changes from one machine to another, a vehicle must be scheduled to do the transport. Transportation times depend on the machines involved [29].

The goal of all problem variants is to produce a feasible schedule that includes all products or jobs. Furthermore, this schedule should minimise (or maximise) one or several predefined objectives, such as the overall makespan, tardiness, lateness or machine workload, considering transportation and setup times [5]. Recently, objectives considering the environmental impact, e.g., energy consumption, are becoming increasingly relevant in scheduling [22].

Chaudhry and Khan reviewed the techniques used to solve FJSSP problems in [5] to conclude that most of the studied journal contributions devised hybrid techniques (35%) or some form of evolutionary algorithm (24%), e.g., genetic algorithms, differential evolution or learning classifier systems. The authors define hybrid techniques as techniques that combine one or several (meta-) heuristics to benefit from their strengths [5]. About 10% of the authors used deterministic heuristics, while tabu search was used in 6% of the cited papers. Other techniques include integer/linear programming and mathematical programming, as well as nature-inspired algorithms such as particle swarm optimisation, simulated annealing, ant colony optimisation, or artificial bee colony [5]. Scott et al. investigated whether human expertise can help to solve hard optimisation problems such as routing or scheduling [34]. In human-computer optimisation, humans and computers collaborate, e.g., a user specifies a search space that the computer then explores. Scott et al. conclude that human expertise can indeed help to manage the usage of computational resources in optimisation [34].

Due to the complexity, researchers often tackle JSSP variants by splitting the problem into the aforementioned subproblems and solving them one after another [45]. Researchers also assume a deterministic environment [4] and omit complex constraints, e.g., regarding uncertain processing, transportation or setup times, maintenance, or machine breakdown to facilitate the problems [5]. Another method to

relax the problem is to decrease the time horizon of the schedule [45]. However, beyond these simplifications, manufacturing systems are characterised by unpredictable events and disturbances [21, 30, 45]. Therefore, authors doubt whether centralised approaches can cope with the dynamic and sometimes even chaotic nature of production systems [6, 45] and provide the required flexibility [21].

3.2.2.2 Related Work - Dynamic Scheduling through Self-Organisation

Instead of computing a schedule upfront using global knowledge, research in adaptive production systems has focused on solving the problems of assignment, sequencing, and routing through the interaction of the involved agents. This leads to a different focus: From finding an optimal to finding a dynamic schedule [30, 42]. In return, researchers hope to achieve greater robustness, flexibility and scalability.

Different authors [13, 20, 42, 45] have devised potential field approaches to solve the assignment and routing subproblems and guide products through production: On the one hand, machines send out potential fields to attract empty vehicles or vehicles carrying products. On the other hand, vehicles sense the attraction fields sent out by machines, decide for one and move towards it. Figure 3.2 shows the local interaction between vehicles or AGVs and machines in [13].

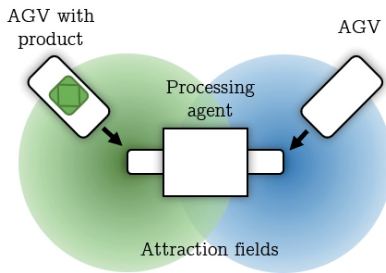


Fig. 3.2. Local interaction between a processing agent and AGVs, adapted from [13]: The input buffer on the left sends out a potential field to attract AGVs carrying products, while the output buffer on the right emits a potential field to attract empty AGVs that remove products from the output buffer.

While the approaches are conceptually similar, they differ in many details: Attraction fields can encode a simple enumeration of product types [13] or include more complex concepts and constraints such as product size, quality of service, availability and workload of machines [42, 45]. Routing can take place on a fixed graph that represents routes of a shuttle system [20, 45] or a general two-dimensional space [13, 42]. Lastly, the control of attraction fields can be hardcoded [42] or learned, e.g., by reinforcement learning [13].

The potential field approach exhibits strong self-organisation, as it requires no central control [10]. However, quantitative analysis is challenging due to its dynamic

nature [42]. Therefore, researchers resort to an experiment-based analysis: They measure objectives during simulation [13] or produce a schedule by running a simulation [45]. This schedule or data is afterwards analysed in terms of optimality. The experiment-based analysis does not allow for behavioural guarantees, which are indispensable for production systems.

The Restore Invariant Approach (RIA) [26] tries to fill this gap by specifying and enforcing a corridor of correct behaviour. Correct behaviour includes a feasible assignment of machines and correct routing of products. Sequencing of products is not part of the behavioural corridor. Instead, it arises as an emergent property. The agents monitor the corridor to ensure that the agent that detects a violation starts a reconfiguration. The purpose of reconfiguration is bringing the system back into the corridor. Reconfiguration can be centralised [25] or partly decentralised using coalition formation [35]. In the centralised variant, a central controller collects information about all agents and can then solve the problems of assignment and routing by applying constraint solving or a genetic algorithm. In the decentral reconfiguration, the agent noticing the violation (leader) forms a coalition with its neighbouring agents. The leader then tries to solve the problems of assignment and routing using the information from its neighbours. If the leader can't solve the problem, he enlarges the coalition and re-tries to solve the problem until he finds a solution [26]. A verified result checker then reviews the found solution before the leader distributes it among the agents in the coalition. The verified result checker together with the verification of the functional system allows guaranteeing that the system behaves as intended [26,27].

3.2.2.3 Contribution

Building upon previous and related work, there are several areas of contribution:

1. The first area of contribution is related to the realistic description of tasks and capabilities presented in Section 3.2.1. We plan to investigate how these realistic descriptions affect finding a solution towards the assignment and product routing in the context of the RIA. We assume that the realistic descriptions will increase the complexity of the problems. Thus, we plan to re-evaluate the use of constraint solving and genetic algorithms in comparison to other optimisation or learning methods.
2. The second area of contribution is concerned with comparing the mechanisms presented before: Can the different mechanisms profit from another? E.g., can we get rid of partly centralised control of the coalition leader in the RIA to achieve strong self-organisation as seen in the potential field approach? As a concrete contribution, we plan to implement and evaluate a reconfiguration mechanism based on distributed constraint optimisation.
3. Third, we plan to examine the use of machine learning techniques for dynamic scheduling. One exemplary use case is predictive maintenance. Researchers already use machine learning algorithms to predict machine or component failure successfully [8]. However, often effective countermeasures besides human intervention are missing. The combination of dynamic scheduling and machine

learning seems promising, as products could be re-assigned and rerouted autonomously in case of imminent failure.

4. Lastly, we plan to investigate the role of humans in dynamic scheduling: Can human expertise help to solve the problem, like in Scott et al.? Can human-computer cooperation help to build understanding and trust in the solution found? We further want to answer the following questions: How can humans intervene and post new constraints? Do those constraints oppose performance and scalability?

3.2.2.4 Evaluation

We plan to evaluate our contribution in comparative studies, where we compare two variants, e.g., with and without realistic task descriptions, in a given scenario. These comparative studies allow us to measure and compare the relevant attributes, e.g., runtime and solution quality. The evaluation should also cover different problem sizes, e.g., number of agents or number of products, to draw conclusions about scalability. Scenarios might also include disturbances, i.e., component or agent failure, to quantify changes in robustness.

3.2.3 Dealing with Deadlocks

Deadlocks are situations where two or more agents are waiting for another to finish in a way that no one ever finishes [9]. The risk of deadlocks in production systems is well-known, and as deadlocks may halt production, they are also heavily studied [1, 16, 37]. Consider the motivating example in Figure 3.3 that demonstrates how a simple cyclic arrangement can lead to a deadlock. Cyclic arrangements concerning multiple tasks are also possible and might be even harder to detect locally.

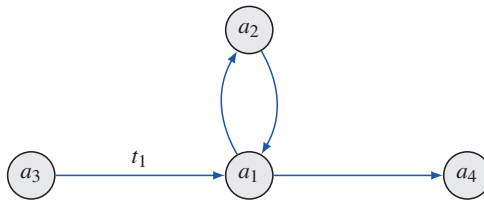


Fig. 3.3. Cyclic arrangement of two agents a_1 and a_2 . The arrows denote the product flow of task t_1 : a_1 receives products from a_3 , processes them and hands them over to a_2 . After processing at a_2 , a_1 receives the products again and applies another capability before handing them over to a_4 . If a_1 accepts a product from a_3 while a_2 also holds a product a deadlock emerges.

3.2.3.1 Related Work

As Figure 3.3 suggests, deadlocks in manufacturing systems are caused by cycles. Specifically, Wysk et al. proof that the following two conditions must be met for a deadlock to occur [7, 44]:

1. There has to be at least one cycle in the product flow.
2. Each agent in the cycle has to be occupied by a product.

To deal with deadlocks, researchers devised a variety of methods, including Petri nets [1, 43] that restrict the agent's actions to prevent deadlocks. Event-based approaches [11, 12] use global knowledge to detect cycles and decide on save transactions. However, as both methods require global knowledge or control, they are not suitable for distributed systems.

Distributed cycle detection algorithms, such as the one presented in [2], detect cycles by passing messages between the agents. Messages are forwarded until they return to their sender or they reach the end of the system and cannot be forwarded any further. This algorithm allows determining whether an agent is in a cycle. Though, it does not provide additional information, such as the cycle's size, which is essential for avoiding deadlocks in a distributed manner.

Lastly, we directly build upon the work of Steghöfer et al. [37]. In their work, the authors present a decentralised deadlock avoidance approach based on message passing. However, dealing with multiple types of products is left as future work.

3.2.3.2 Contribution

Thus, in [15], we present a decentral approach to avoid deadlocks in production systems that handle multiple tasks at once. We refrain from generally averting cycles, as this results in a loss of flexibility. Instead, we rely on the aforementioned theoretical insight of Wysk et al. To prevent that each agent is occupied by a product, we employ a two-step procedure [15]:

1. Cycle detection: Whenever the configuration of the system changes, e.g., due to a new type of product or the (partial) failure of an agent, agents send out messages to detect cyclic arrangements. Cycles are then stored alongside the number of products that are allowed to enter.
2. Enforcing the limits for products in cycles: When production resumes, agents keep track of the number of products that are currently in each cycle. The agents that are entrances and exits of the cycles enforce the limits calculated in cycle detection by coordinating through message-passing.

3.2.3.3 Evaluation

To evaluate our approach experimentally, we run several simulations with different configurations and measure the number of deadlocks encountered, the runtime needed, and the number of messages sent. Additionally, we calculate the system's throughput

by dividing the number of manufactured products by the runtime. Our results suggest that our approach effectively avoids deadlocks in the configurations considered. Furthermore, our approach outperforms a simple conservative locking algorithm in terms of message overhead and runtime. Therefore, systems using our approach can realise higher throughput compared to systems using the conservative locking algorithm [15].

3.2.3.4 Future Work

Despite the encouraging results, some challenges remain: First, our experimental evaluation does not formally prove the deadlock avoiding property of our algorithm. Therefore, we strive for formal proof confirming our experimental results. Additionally, the experimental configurations only cover a small set of agents. To ensure the scalability of our approach, we plan to conduct experiments with a larger number of agents and also investigate the message overhead in a formal way. Lastly, we plan to examine whether adding soft constraints that favour solutions without cycles to our constraint model can relax the problem.

3.3 Conclusion

In this paper, we summarise research questions in adaptive production systems. Namely, we suggest using more realistic task descriptions, including structures such as selective tasks, forks, and synchronisations. Further, we plan to extend capability descriptions to contain material-, geometry-, and process-related information. The effects of elaborating task and capability descriptions on the problem of task allocation have to be studied. We further plan to direct research to automatically generating task descriptions from user input, as manually creating task descriptions becomes more complex and error-prone.

In times of fluctuating markets, dynamic control is another key-issue for adaptive production systems. We present different approaches towards the problem of dynamic scheduling and propose to take advantage of the combination of the different concepts. We intend to allow realistic task structures and human intervention in dynamic scheduling. We further plan to integrate machine learning techniques into adaptive production systems to benefit from the rapid progress in this area. Combining machine learning and self-organisation allows to detect failures beforehand and offer countermeasures such as rerouting products. Therefore, the combination may further increase the robustness of adaptive production systems.

Another problem in flexibly linked, decentral production systems with multiple tasks is dealing with deadlocks. To handle both, the decentral nature of adaptive systems as well as many products at a time, we present a message-based deadlock avoidance approach in [15]. Our experimental evaluation shows that the approach avoids deadlocks in several realistic system configurations with reasonable message overhead. However, evaluating the scalability of our approach in larger configurations, as well as formally proving the deadlock-avoiding property remains as future work.

References

1. Banaszak, Z.A., Krogh, B.H.: Deadlock avoidance in flexible manufacturing systems with concurrently competing process flows. *IEEE Transactions on robotics and automation* 6(6), 724–734 (1990)
2. Boukerche, A., Tropper, C.: A distributed graph algorithm for the detection of local cycles and knots. *IEEE Transactions on Parallel and Distributed Systems* 9(8), 748–757 (1998)
3. Brailsford, S.C., Potts, C.N., Smith, B.M.: Constraint satisfaction problems: Algorithms and applications. *European Journal of Operational Research* 119(3), 557 – 581 (1999)
4. Chaari, T., Chaabane, S., Aissani, N., Trentesaux, D.: Scheduling under uncertainty: Survey and research directions. In: 2014 International Conference on Advanced Logistics and Transport (ICALT). pp. 229–234 (2014)
5. Chaudhry, I.A., Khan, A.A.: A research survey: review of flexible job shop scheduling techniques. *International Transactions in Operational Research* 23(3), 551–591 (2016)
6. Chevalyere, Y., Endriss, U., Lang, J., Dunne, P., Lemaitre, M., Maudet, N., Padget, J., Phelps, S., Rodriguez-Aguilar, J., Sousa, P.: Issues in multiagent resource allocation. *Informatica* 30, 3–31 (2006)
7. Cho, H., Kumaran, T., Wysk, R.A.: Graph-theoretic deadlock detection and resolution for flexible manufacturing systems. *IEEE Transactions on Robotics and Automation* 11(3), 413–421 (1995)
8. Cline, B., Niculescu, R.S., Huffman, D., Deckel, B.: Predictive maintenance applications for machine learning. In: 2017 Annual Reliability and Maintainability Symposium (RAMS). pp. 1–7 (2017)
9. Coffman, E.G., Elphick, M., Shoshani, A.: System deadlocks. *ACM Computing Surveys (CSUR)* 3(2), 67–78 (1971)
10. Di Marzo Serugendo, G., Gleizes, M.P., Karageorgos, A.: Self-organization in multi-agent systems. *Knowledge Engineering Review* 20(2), 165–189 (2005)
11. Fanti, M.P., Maione, B., Mascolo, S., Turchiano, A.: Event-based feedback control for deadlock avoidance in flexible production systems. *IEEE Transactions on Robotics and Automation* 13(3), 347–363 (1997)
12. Fanti, M.P., Zhou, M.: Deadlock control methods in automated manufacturing systems. *IEEE Transactions on systems, man, and cybernetics-part A: systems and humans* 34(1), 5–22 (2004)
13. Fujii, N., Hatono, I., Ueda, K.: Reinforcement learning approach to self-organization in a biological manufacturing system framework. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture* 218(6), 667–673 (2004)
14. Güdemann, M., Ortmeier, F., Reif, W.: Formal modeling and verification of systems with self-x properties. In: Yang, L.T., Jin, H., Ma, J., Ungerer, T. (eds.) *Autonomic and Trusted Computing*. pp. 38–47. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
15. Hirsch, J., Neumayer, M., Ponsar, H., Kosak, O., Reif, W.: Deadlock avoidance for multiple tasks in a self-organizing production cell. In: 2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS). pp. 178–187 (2020)
16. Hu, H., Li, Z.: Local and global deadlock prevention policies for resource allocation systems using partially generated reachability graphs. *Computers & Industrial Engineering* 57(4), 1168 – 1181 (2009)
17. Keddis, N., Kainz, G., Zoitl, A., Knoll, A.: Modeling production workflows in a mass customization era. In: 2015 IEEE International Conference on Industrial Technology (ICIT). pp. 1901–1906. IEEE (2015)
18. Koren, Y.: *The global manufacturing revolution: product-process-business integration and reconfigurable systems*. John Wiley & Sons (2010)

19. Lau, M., Ohgawara, A., Mitani, J., Igarashi, T.: Converting 3d furniture models to fabricatable parts and connectors. *ACM Trans. Graph.* 30(4) (Jul 2011)
20. Leitão, P., Barbosa, J., Trentesaux, D.: Bio-inspired multi-agent systems for reconfigurable manufacturing systems. *Engineering Applications of Artificial Intelligence* 25(5), 934–944 (2012)
21. Leitão, P.: Agent-based distributed manufacturing control: A state-of-the-art survey. *Engineering Applications of Artificial Intelligence* 22(7), 979 – 991 (2009), distributed Control of Production Systems
22. May, G., Stahl, B., Taisch, M., Prabhu, V.: Multi-objective genetic algorithm for energy-efficient job shop scheduling. *International Journal of Production Research* 53(23), 7071–7089 (2015)
23. Müller-Schloer, C., Tomforde, S.: *Organic Computing - Technical Systems for Survival in the Real World.* Birkhäuser (2017), <https://doi.org/10.1007/978-3-319-68477-2>
24. N. Kubota, T. Fukuda, F. Arai, K. Shimojima (eds.): Genetic algorithm with age structure and its application to self-organizing manufacturing system: ETFA '94. 1994 IEEE Symposium on Emerging Technologies and Factory Automation. (SEIKEN) Symposium) -Novel Disciplines for the Next Century- Proceedings (1994)
25. Nafz, F., Ortmeier, F., Seebach, H., Steghöfer, J.P., Reif, W.: A universal self-organization mechanism for role-based organic computing systems. In: González Nieto, J., Reif, W., Wang, G., Indulska, J. (eds.) *Autonomic and Trusted Computing.* pp. 17–31. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
26. Nafz, F., Seebach, H., Steghöfer, J.P., Anders, G., Reif, W.: *Constraining Self-organisation Through Corridors of Correct Behaviour: The Restore Invariant Approach,* pp. 79–93. Springer Basel, Basel (2011)
27. Nafz, F., Seebach, H., Steghöfer, J.P., Bäuml, S., Reif, W.: A formal framework for compositional verification of organic computing systems. In: Xie, B., Branke, J., Sadjadi, S.M., Zhang, D., Zhou, X. (eds.) *Autonomic and Trusted Computing.* pp. 17–31. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
28. Neumayer, M.: Towards realistic task and capability descriptions in self-organizing production systems. In: 2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C). pp. 234–236 (2020)
29. Nouri, H.E., Driss, O.B., Ghédira, K.: A classification schema for the job shop scheduling problem with transportation resources: State-of-the-art review. In: Silhavy, R., Senkerik, R., Oplatkova, Z.K., Silhavy, P., Prokopova, Z. (eds.) *Artificial Intelligence Perspectives in Intelligent Systems.* pp. 1–11. Springer International Publishing, Cham (2016)
30. Ouelhadj, D., Petrovic, S.: A survey of dynamic scheduling in manufacturing systems. *Journal of scheduling* 12(4), 417 (2009)
31. Pfrommer, J., Schleipen, M., Beyerer, J.: Pprs: Production skills and their relation to product, process, and resource. In: 2013 IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA). pp. 1–4 (2013)
32. Qiao, L., Kao, S., Zhang, Y.: Manufacturing process modelling using process specification language. *The International Journal of Advanced Manufacturing Technology* 55(5-8), 549–563 (2011)
33. Schmeck, H.: Organic computing - a new vision for distributed embedded systems. In: Eighth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'05). pp. 201–203 (2005)
34. Scott, S.D., Lesh, N., Klau, G.W.: Investigating human-computer optimization. In: Proceedings of the SIGCHI conference on Human factors in computing systems. pp. 155–162 (2002)

35. Seebach, H., Nafz, F., Steghöfer, J.P., Reif, W.: How to Design and Implement Self-organising Resource-Flow Systems, pp. 145–161. Springer Basel, Basel (2011)
36. Steghöfer, J.P., Kiefhaber, R., Leichtenstern, K., Bernard, Y., Klejnowski, L., Reif, W., Ungerer, T., André, E., Hähner, J., Müller-Schloer, C.: Trustworthy organic computing systems: Challenges and perspectives. In: Xie, B., Branke, J., Sadjadi, S.M., Zhang, D., Zhou, X. (eds.) *Autonomic and Trusted Computing*. pp. 62–76. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
37. Steghöfer, J.P., Mandrekar, P., Nafz, F., Seebach, H., Reif, W.: On deadlocks and fairness in self-organizing resource-flow systems. In: Müller-Schloer, C., Karl, W., Yehia, S. (eds.) *Architecture of Computing Systems - ARCS 2010*. pp. 87–100. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
38. Tippayawong, K.Y., Prapasirisulee, T.: Productivity enhancement in a wood furniture manufacturing factory by improving work procedures and plant layout. *Recent Advances in Manufacturing Engineering* pp. 30–34 (2011)
39. Tomforde, S., Kantert, J., Müller-Schloer, C., Bödel, S., Sick, B.: Comparing the effects of disturbances in self-adaptive systems - A generalised approach for the quantification of robustness. *Trans. Comput. Collect. Intell.* 28, 193–220 (2018)
40. Tomforde, S., Sick, B., Müller-Schloer, C.: Organic computing in the spotlight. arXiv preprint arXiv:1701.08125 (2017)
41. Trentesaux, D., Pach, C., Bekrar, A., Sallez, Y., Berger, T., Bonte, T., Leitão, P., Barbosa, J.: Benchmarking flexible job-shop scheduling and control systems. *Control Engineering Practice* 21(9), 1204–1225 (2013)
42. Vaario, J., Ueda, K.: An emergent modelling method for dynamic scheduling. *Journal of Intelligent Manufacturing* 9(2), 129–140 (1998)
43. Wu, N., Zhou, M.: Modeling and deadlock avoidance of automated manufacturing systems with multiple automated guided vehicles. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 35(6), 1193–1202 (2005)
44. Wysk, R.A., Yang, N.S., Joshi, S.: Detection of deadlocks in flexible manufacturing cells. *IEEE Transactions on robotics and automation* 7(6), 853–859 (1991)
45. Zbib, N., Pach, C., Sallez, Y., Trentesaux, D.: Heterarchical production control in manufacturing systems using the potential fields concept. *Journal of Intelligent Manufacturing* 23(5), 1649–1670 (2012)