Invariant-Based Production Control Reviewed: Mixing Hierarchical and Heterarchical Control in Flexible Job Shop Environments

Henning Blunck(✉) and Julia Bendul

School of Mathematics and Logistics, Jacobs University Bremen gGmbH, 28759 Bremen, Germany
{h.blunck,j.bendul}@jacobs-university.de

Abstract. We are interested in the interplay of hierarchical and heterarchical control to reduce myopic behavior in a setting where central planning establishes relaxed schedules and distributed control is applied to make remaining decisions at runtime. We therefore pick up an idea introduced by Bongaerts et al. [4] to generate *invariants*, relaxed schedules, as constraints on distributed production control.

We apply this concept to the Flexible Job Shop Scheduling Problem (FJSSP), represented as disjunct graphs, introduce a measure to quantify the "tightness" of invariants, constrains the set of local decision heuristics that can be applied in such setting and present a simulation implementation, based on standard problem instances and optimization models with initial results. They validate the proposed measure and highlighting the need for further investigation of the interplay between problem structure and achieved performance.

Keywords: Production control · Invariants · Scheduling · Distributed decision making

1 Introduction

Responding to increasing complexity in dynamics in manufacturing systems, the distribution of control capacity and authority has been investigated as an alternative control scheme for production planning and control (PPC) systems [11]. While "first-generation" distributed PPC systems allowed for no coordination between decision making entities, subsequent extensions of the concept gradually increasingly promoted the idea of mixing centralized and distributed decision making in the control hierarchy [11,27].

This raises the question of finding the optimal "mix" of hierarchical and distributed control approaches as a function of the controlled system, order characteristics and decision preferences. The idea of using relaxed schedules (invariants) as a vehicle to mix both control paradigms has been proposed in [4], but so far not been experimentally investigated (c.f. Sec. 2.1) as a vehicle to establish semi-hierarchical control systems.

V. Mařík et al. (Eds.): HoloMAS 2015, LNAI 9266, pp. 96–107, 2015.
DOI: 10.1007/978-3-319-22867-9_9

The remainder of this article is structured as follows: Section 2 reviews the relevant concepts and previous findings. Section 3 describes the approach to invariant creation and implementation used in this contribution. First results are presented in Section 4 and the findings as well as ideas for future research are briefly discussed in Section 5.

2 Definitions and Literature Review

2.1 Balancing Hierarchical and Heterarchical Control

It is commonly believed that the optimal control architecture is a function of the system to be controlled and the decision preferences applied. As a rule of thumb, (highly) hierarchical systems allow long term optimality in calm planning environments and low flexibility production systems while (highly) heterarchical systems enable short term optimization in turbulent environments and controlled systems that allow alternative process paths [11,19,27].

Both Philipp et al. [20] and Zambrano Rey et al. [29] hypothesize that the system performance, when plotted as a function of control heterarchy, follows a curved shape with a global optimum attained for a mixture of hierarchical and heterarchical control, where the advantages of better responsiveness and short term optimization of distributed control systems can be harvested, while minimizing the amount of myopia induced by distributed decision making. The identification and reduction of myopia in distributed control settings has lately received increased attention from the works of Zambrano Rey et al. [29,30].

Reasons for myopic decision making include

- the decomposition of the original planning problem [21, Ch.2]
- selfish actors (known in Game-Theory as *Cost of Anarchy* [12]),
- decision making based on local information only [c.f.e.g. 17,24,27], and
- time-constraints on decision making [30].

Existing simulation studies on the combination of hierarchical and distributed production control differ in their approach: Scholz-Reiter et al. [25] constrained the set of parallel worksystems where scheduling decisions are made locally. Mönch and Drießel [18] change the information and planning horizon. Mediating agents are proposed and investigated e.g. in [6,8]. In [29], supervisor agents are given enhanced decision making time and computing power to perform simulation-optimization to attain better scheduling decisions. Grundstein et al. [14] investigate the combination of central scheduling and autonomous production control by investigating the interdependend between order release method, local decision making heuristic, and production performance, without finding conclusive relationships.

To the best of our knowledge, we are the first to quantitatively investigate the idea of an invariant-based mixing of hierarchical and heterarchical control, as proposed in [4]. The proposed model extends the idea of [25] in that scheduling decisions can be removed at any point in the schedule (not necessarily spatially confined to particular worksystems) and is different to the other contributions mentioned above in that elements of hierarchical and heterarchical coordination are not working simultaneously but subsequently, constituting a constructional distributed decision making system [24, Ch.1.1].

2.2 The Flexible Job Shop Scheduling Problem

The Flexible Job Shop Scheduling Problem (FJSSP) extends the classical Job Shop Scheduling problem by relaxing the a priori assignment of operations to worksystems [5]. The FJSSP is particularly interesting for the sequential application of hierarchical and heterarchical control, since it combines the allocation (assigning operations to machines) and sequencing (determining a sequence of operations on each machine) sub-problems, that are often dealt with separately in hierarchical production planning systems [10]. It is prone to gain attention with the rise of flexible manufacturing systems (FMS).

Out of the large number of test-instances published for the FJSSP. We use here a total of 393 test-instances from [1,2,5,9,13,16]. As elaborated in [2], the test-instances are not only different in size (number of jobs, operations and machines) and level of machine flexibility, but arose from different considerations and with different analysis intentions in mind. For a first analysis, the test-instances by Hurink et al. [16] and Dauzère-Pérès and Paulli [9] were both generated from JSP instances by gradually increasing operation flexibility [2] (c.f. table footnotes). We hence distinguish 11 problem groups for our analysis.

Table 1. Test-Instances used in this publication. Notes summarized from [2].

Source	# Instances	Notes
Brandimarte [5]	10	medium degree of machine-flexibility
Hurink et al. [16]	$66 \cdot 4$	4 series with increasing processing flexibility[a], processing times independent of machine
Dauzère-Pérès and Paulli [9]	$6 \cdot 3$	6 different setups, each with 3 levels of machine flexibility.[b] Slightly different processing times across machines
Chambers and Barnes [1]	21	obtained from JSP problems by replicating machines acc. to different heuristics
Behnke and Geiger [2]	60	"Similar" machines are grouped into workcenters
Fattahi et al. [13]	20	Randomly generated, medium-sized problems

[a] EDATA: Few operations with $\geq$ 1 possible machine, RDATA: Most operations assignable to >1 machine, and VDATA: All operations with >1 possible machines.

[b] Probability of a machine being assignable to a given operation set to $0.1, 0.3, 0.5$ respectively.

3 Model

3.1 Background: Flexible Job-Shop Scheduling on Graphs

Disjunct (or mixed) graphs were introduced as a representation of scheduling problems by [23] and are widely used to represent scheduling problems [3].

Let $G(V, A, E)$ be a mixed graph, consisting of the set of nodes (V), and sets of undirected (E) and directed (A) edges. We denote by $\tilde{G} = (V, A)$ the *directed* subgraph of G, composed of the same node set, but only the directed edges. A directed edge $A \rightarrow B$ indicates a precedence relationship $A \prec B$ in that operation B cannot be started until A has finished (Finish-Start Constraint). The set of directed edges in a scheduling graph can naturally be subdivided into a set of technical and environmental constraints A_T (that are constraints on the planning process) and scheduling decisions A_S [4].

While the classical disjunct graph formulation assumes a solved assignment problem (operations are assigned to one worksystem each), it can be extended to represent a more complex situations [c.f. 10], by assigning a processing time matrix P to every operation, indicating if and in which time, an operation can be processed at a given worksystem. If in the initial graph, an undirected edge exists between any two operations that can be processed on the same worksystem [10]. A FJSSP then is solved feasibly if and only if (1) every operation has been assigned to one worksystem, (2) there exists a directed path $\in \tilde{G}$ between any two operations assigned to the same worksystem and (3) $\tilde{G}$ is acyclic [5,10].

The term invariant in natural and computer sciences describes a statement or property whose value is unaltered during an applied transformation. For the domain of FJS-scheduling, a schedule-invariant is understood as a subset of all assignment and sequencing decisions (directed edges $\in A_S$, c.f. [4]) that have to be observed (remain unchanged) during production control transformation of the scheduling graph.

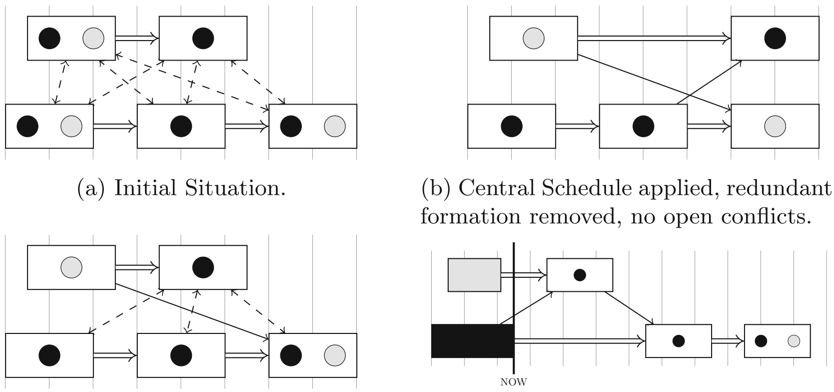
3.2 General Setup

Our simulation model is based on a graph as described in Sec. 3.1 that represents the test-instances introduced in Sec. 2.2.

The model is executed as follows: After representing the scheduling problem as a graph (Fig. 1a), we start by applying a centrally computed solution (Section 3.3) to the problem graph (Fig. 1b), which we then relax (Sec. 3.5) to an invariant by removing scheduling decisions (Fig. 1c). The invariant is handed over to a controller, defined by the agent class assigned to operations and worksystems respectively (c.f. Sec. 3.6) that then makes the resulting decisions at runtime, based on local information (Fig. 1d). The controller assigns agents to the operations (nodes in the graph) and worksystems and initiates the simulation. Once an operation starts processing, an actual processing time is calculated following a truncated[1] Normal distribution with the planned processing time μ as expected

[1] We do not allow negative processing times. However, even with the highest standard deviations investigated here, such event has a probability of $\approx 0.1\%$.

value and a standard deviation of $CV \cdot \mu$. While other performance measures are possible, we focus on minimizing the makespan (maximum Lateness $L_{max}(\sigma)$) in this contribution. Since the schedule is *left-justified*, i.e. operations start as early as possible without changing operation sequence, not as late as possible, the results will not minimize e.g. Work in Process (WIP). We measure ratio of the CP-solution and the makespan attained including process time variability and distributed control as the *relative Performance* of the semi-hierarchical control structure (c.f. Sec. 4).



(a) Initial Situation.

(b) Central Schedule applied, redundant information removed, no open conflicts.

(c) Invariant, created by deleting the worksystem assignment of the bottom right operation and the sequencing constraints around the second operations of both jobs.

(d) Snapshot of the distributed production control solving the sequencing problem based on FIFO as actual processing times are realized.

Fig. 1. Stylized visualization of the approach taken here. Scheduling problem with two machines, two jobs and a total of five operations. Rectangles represent operations and their planned processing times. Circles represent assignable worksystems. Filled operations are already underway and actual processing times ($\sim \mathcal{N}(\mu, CV \cdot \mu)$) are known. Solid arcs represent precedence constraints introduced during scheduling. Double arrows indicate technical precedence constraints. Dashed edges represent still open sequencing decisions between operations.

3.3 Implementation of Hierarchical Planning

To emulate a hierarchically derived production schedule, we use the Constraint Programming FJSSP model, shipped with the popular optimization suite IBM CPLEX Studio 12.6. The model has (with minor adjustments) been used to find optimal solutions to some previously unsolved problem-instances[2] [2, 22]. So we may assume both broad availability and competitive performance of the model and solver.

[2] The same we use here, c.f. Sec. 2.2.

Each problem instance was computed for a maximum of 20 minutes on a UNIX-machine with an Intel Xeon quad-core processor, 2.8 GHz and 3 GB of RAM and the best (shortest makespan) solution attained in this time was considered the hierarchical production schedule for this instance.

3.4 Removing Redundant Scheduling Decisions

Removing redundant edges from the scheduling graph has been considered important before for the application of various scheduling heuristics [5], but is gains even higher importance in the context of invariant-based scheduling: To effectively measure and compare the degree to which the original schedule was relaxed, it is necessary to remove redundant information from the mixed graph, so that any further relaxation does in fact open feasible decision alternatives. To this end, we replace a an arc $a \in A_S$ with an undirected edge[3], if and only if there exist a directed path between start- and end-node of a in $\tilde{G} \setminus \{a\}$[10].

3.5 Invariant Creation and Assessment

For this initial investigation we define two schedule-relaxation-heuristics, each of which is applied to $\alpha \in [0, 1]$ of all operations (we call α the schedule removal degree).[4] We investigate two simple relaxation heuristics:

Removing the sequencing information. By keeping machine assignments proposed by the initial scheduling but removing the sequencing decisions (removing the added directed edges from the graph) for α of all operations. For $\alpha = 1$, we have created the related JSP problem, thus a natural hierarchical decomposition found in the FJSSP [5, Ch.3] and also currently present in many hierarchical PPC systems.

Resetting Operations. We entirely reset (delete scheduled constraints and worksystem-assignment) for α of all operations. For $\alpha = 1$, we attain the original FJSSP problem.

In addition to measures suggested e.g. in [7], we propose to measure the degree of freedom preserved by an invariant as the share of schedulable (i.e. not technologically constrained) edges in the graph for which the invariant does not prescribe an orientation (i.e. there does not exist a directed path between the two ends of the undirected edge) and the operations could still be assigned to the same worksystem (i.e. a precedence decision might become necessary).

Note that in a disjunct graph representation as described in Sec. 3.1 with removed redundant edges (Sec. 3.4), also the relaxation of machine assignments will lead to such conflicting edges.

---

[3] Note that we do not delete technological constraints from $\tilde{G}$.

[4] This is a first attempt to create invariants of different "tightness" but not the only way to attain them. Relaxation heuristics do not need to be node-based.

3.6 Distributed Production Controller

As stated before, the FJSSP is particular in that any valid production schedule has to solve both the assignment and sequencing subproblems. In the research on dispatching rule based FJS-scheduling, a "shortest queue length" heuristic seems to be commonly applied to solve the assignment problem [c.f. 26].

A QLE-control can be implemented in a mixed-graph by giving the operation agent the authority to choose a worksystem (by querying all possible worksystems for the current queue length, i.e. the expected finishing time of the last operation) and assigning itself to worksystem i where $i := \arg\min_{i \in W}\{q_i + p_i\}$ where W is the set of possible worksystems, q_i is the queue length of worksystem i and p_i is the processing time of the operation on worksystem i. The worksystem agents of the chosen worksystem forms an precedence constraint from the last entry in their queue to the newly assigned operation (i.e. implements a FIFO strategy).

Building upon which, we define two controllers which implement two different sequencing heuristics:

A *FIFO*-Controller processes the operations at the worksystems in the sequence of assignment (i.e. in the sequence in which their respective last predecessor was finished).

A *LRPT-Controller* sequences operations by decreasing remaining workcontent (LRPT: Longest Remaining Processing Time). Where ≥ 1 worksystem is possible, the average processing time over all possible worksystems is assumed in this contribution, expressing no prior belief concerning the upcoming assignment decision.

3.7 Distributed Control Heuristics that Guarantee Valid Schedules

The FIFO- and LRPT- Production controller (Sec. 3.6) implement a decision logic on the side of the operations and worksystems respectively that is based on local information.

However, to avoid forming a directed cycle (and hence an invalid schedule) global information about the existence of paths is required.[5] It is hence easy to imagine a situation in which an agent decides to form a precedence constraint, closing a directed cycle on the graph (c.f. Fig. 2), a problem particular to invariant-based scheduling problems.[6]

Following [15], we can investigate the dynamic on the graph, the observable result of the interplay of worksystem and operation agents, as the combination of (1) a *neighborhood assessment strategy*, applied by an entity to make a decision within its decision space (forming edges, committing to worksystems, ...) and

[5] With such information, a node would not be allowed to form a precedence constraint $A \prec B$ if there exist a path $B, \ldots, A$ in the graph.

[6] The problem could only be averted by updating the entire graph after every scheduling decision (i.e. converting edges into arcs, if there exists a directed path). This however would require central coordination and high computational effort.

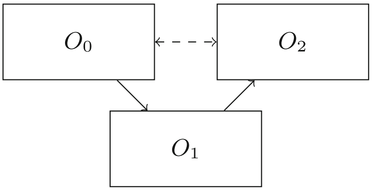


Fig. 2. Stylized invariant-based scheduling problem with two scheduling decisions imposed by the invariant (solid, directed arcs) and a remaining scheduling decision (dashed edge). Given their limited information horizon, Operations O_0 and O_2 are not aware of the path existing between them. A decision by either of the two, to form a precedence relationship $O_2 \prec O_0$ would now form a directed cycle and hence an invalid schedule.

(2) a *temporal organization strategy*, which defines in which order entities take decisions. Using this notation, we can outline a set of constraints that guarantee valid schedules, given a valid invariant.[7]

Theorem 1. *A dynamic Ω is guaranteed to create a valid, non-preemptive schedule on a mixed graph representing any valid invariant, if the temporal organization strategy only allows an operation O to form scheduling decisions*

1. *once all immediate predecessors (in-neighbors in $\tilde{G}$) have been completed and*
2. *if the end of the directed edge points at an operation on which processing has not started yet.*

Proof. Condition (1) guarantees through transitive closure that all (not just the immediate) predecessors of an operation O (i.e. operations from which a directed path to O exists) have been completed, before O can form a scheduling constraint. Condition (2) then guarantees that no such new constraint can close a directed cycle in $\tilde{G}$. □

Note that in particular, all distributed control systems that are based on dispatching rules and hence also the two controllers investigated here (c.f. Sec. 3.6), satisfy above conditions.

4 Initial Results

4.1 Validation of Invariant Assessment Measure

Fig. 3 shows the measure of invariant tightness discussed in Sec. 3.5. The proposed measure for invariant flexibility shows to serve two important purposes: (1) It does in fact measure the existence of decision alternatives (it grows with α) and (2) it distinguished the problem sets, highlighting the different degree sequencing and assignment decisions are present in the problem.

[7] Note that the conditions outlined here are sufficient, but not necessary and other conditions might be found.

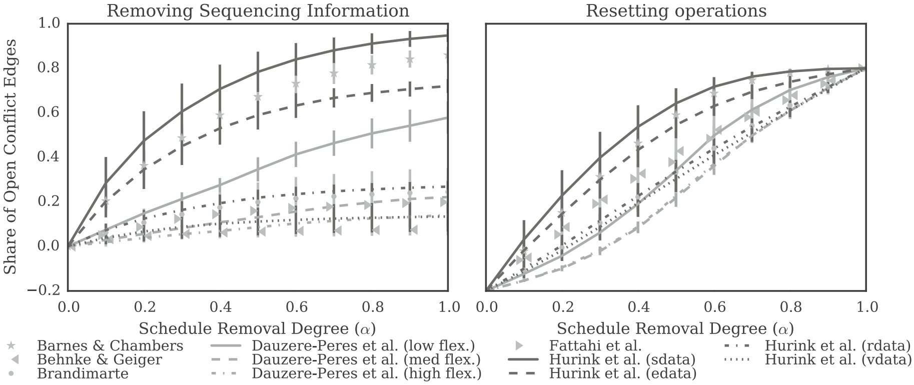


Fig. 3. Share of open conflicts (c.f. Sec. 3.5) for both investigated relaxation heuristics. Over all instance groups

4.2 Impact on Performance

We measure the relative performance of the combination of an invariant and a controller as the ratio of the makespan calculated in Sec. 3.3 and the makespan attained with this configuration. Over all instance-sets, relaxation heuristics and removal degrees, the LRPT-Controller outperforms the FIFO-controller.[8]

Non-surprisingly, test-instances with high machine flexibility fare significantly better when given decision alternatives at runtime (eventually outperforming pure hierarchical planning). Notably, a mix of hierarchical and distributed control (as considered optimal in [20,29], c.f. Sec. 2.1) consistently fares worse then pure approaches with worst relative performance figures attained for *alpha* between 0.25 and 0.5.

We believe that this result comes courtesy of the random schedule relaxation policies applied here. With only some, probably incoherent, operations free to make autonomous decisions, any deviation from the previously determined schedule causes more harm than good because the disturbance created by the local decision making of one operation does lead to the emergence of new sub-schedules but fails to integrate with the framework still imposed around it. A successful schedule, given stochastic processing times, does not seem to be accessible through simple, undirected neighborhood search from the best known solution to the deterministic problem.

The key to successful heterarchical production control then, it appears, lies not in giving autonomy to single operations (as investigated here), but rather to substructures in the scheduling graph. In particular, we should be able to replicate the results from [25], by only deleting sequencing and assignment decisions between parallel servers. A natural extension of the concept of parallel worksystems would be one of clusters of worksystems or operations, like suggested in [28]. A purely random based "neighborhood search" approach, starting from

[8] Which comes courtesy of our focus on total makespan.

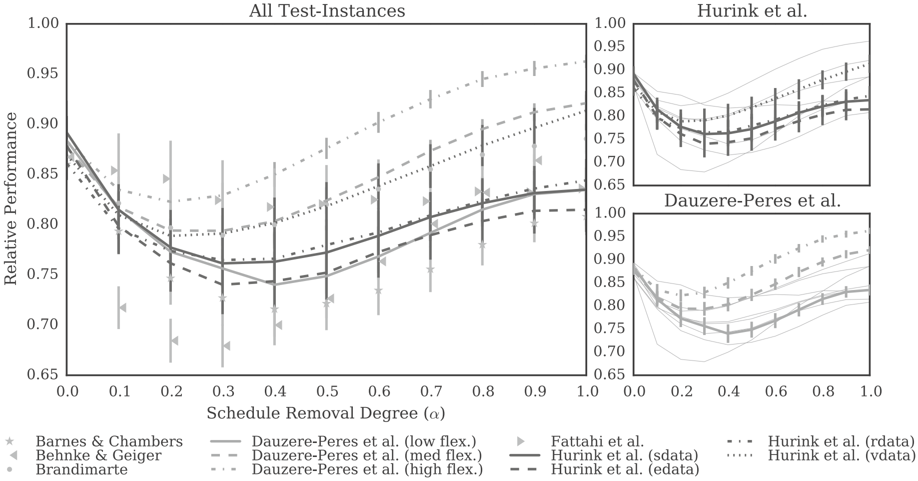


Fig. 4. Relative Performance for all (left) and instance sets with increasing flexibility [9,16] (right). Node-Reset relaxation heuristic, LRPT-Controller, $CV = 0.3$.

the deterministic solution, is apparently not a good approach to finding a good solution for the stochastic production control problem.

5 Discussion and Future Research

We have proposed an implementation of invariant-based scheduling for flexible job shops and proposed a measure to quantify invariants. We have shown that distributed decision heuristics have to be constrained in order to generate valid schedules. Besides demonstrating the validity of our approach, our simulation experiments shows that, in order to achieve competitive performance, the process of schedule relaxation and distributed control has to consider the particularities of the scheduling problem and the subsequent distributed control approach.

Future research should hence seek to identify problem parameters in combination with relaxation and control heuristics that perform particularly well. The strict representation of the problem as a network thereby allows to use network science methods and concepts (like motifs) to be applied in defining such relaxation approaches. Ideally, this would allow future scheduling systems to intentionally relax the set of constraints such that production control can work best, establishing an anticipation relationship, found in many distributed decision making systems [24, Ch.1.3], but not in today's PPC systems.

References

1. Barnes, J.W., Chambers, J.B.: Solving the job shop scheduling problem with tabu search. IIE Transactions **27**(2), 257–263 (1995)
2. Behnke, D., Geiger, M.J.: Test instances for the flexible job shop scheduling problem with work centers. Tech. Rep. 12–01-01, Helmut-Schmidt Universität der Bundeswehr Hamburg, Lehrstuhl für Betriebswirtschaftslehre, insbes. Logistik-Management (May 2012), http://edoc.sub.uni-hamburg.de/hsu/volltexte/2012/2982/
3. Błażewicz, J., Pesch, E., Sterna, M.: The disjunctive graph machine representation of the job shop scheduling problem. European Journal of Operational Research **127**(2), 317–331 (2000)
4. Bongaerts, L., Monostori, L., McFarlane, D., Kádár, B.: Hierarchy in distributed shop floor control. Computers in Industry **43**(2), 123–137 (2000)
5. Brandimarte, P.: Routing and scheduling in a flexible job shop by tabu search. Annals of Operations Research **41**(3), 157–183 (1993)
6. Brennan, R.W.: Performance comparison and analysis of reactive and planning-based control architectures for manufacturing. Robotics and Computer-Integrated Manufacturing **16**(2–3), 191–200 (2000)
7. Brennan, R.W., Norrie, D.H.: Metrics for evaluating distributed manufacturing control systems. Computers in Industry **51**(2), 225–235 (2003). virtual Enterprise Management
8. Cavalieri, S., Garetti, M., Macchi, M., Taisch, M.: An experimental benchmarking of two multi-agent architectures for production scheduling and control. Computers in Industry **43**(2), 139–152 (2000)
9. Dauzère-Pérès, S., Paulli, J.: An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search. Annals of Operations Research **70**(1–4), 281–306 (1997)
10. Dauzère-Pérès, S., Roux, W., Lasserre, J.: Multi-resource shop scheduling with resource flexibility. European Journal of Operational Research **107**(2), 289–305 (1998)
11. Dilts, D., Boyd, N., Whorms, H.: The evolution of control architectures for automated manufacturing systems. Journal of Manufacturing Systems **10**(1), 79–93 (1991)
12. Dubey, P.: Inefficiency of nash equilibria. Mathematics of Operations Research **11**(1), 1–8 (1986)
13. Fattahi, P., Saidi Mehrabad, M., Jolai, F.: Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. Journal of Intelligent Manufacturing **18**(3), 331–342 (2007)
14. Grundstein, S., Schukraft, S., Scholz-Reiter, B., Freitag, M.: Coupling order release methods with autonomous control methods – an assessment of potentials by literature review and discrete event simulation. International Journal of Production Management and Engineering 3(1), 43 (2015)
15. Hadzhiev, B., Windt, K., Bergholz, W., Hütt, M.T.: A model of graph coloring dynamics with attention waves and strategic waiting. Advances in Complex Systems **12**(6), 549–564 (2009)
16. Hurink, J., Jurisch, B., Thole, M.: Tabu search for the job-shop scheduling problem with multi-purpose machines. Operations-Research-Spektrum 15(4), 205–215 (1994)

17. Lin, G.Y.J., Solberg, J.J.: Effectiveness of flexible routing control. International Journal of Flexible Manufacturing Systems **3**(3–4), 189–211 (1991)
18. Mönch, L., Drießel, R.: A distributed shifting bottleneck heuristic for complex job shops. Computers & Industrial Engineering **49**(3), 363–380 (2005)
19. Ouelhadj, D., Petrovic, S.: A survey of dynamic scheduling in manufacturing systems. Journal of Scheduling **12**(4), 417–431 (2009)
20. Philipp, T., Böse, F., Windt, K.: Evaluation of autonomously controlled logistic processes. In: Proceedings of 5th CIRP International Seminar on Intelligent Computation in Manufacturing Engineering, Ischia, Italy, pp. 347–352 (2006)
21. Pochet, Y., Wolsey, L.A.: Production Planning by Mixed Integer Programming. Springer Series in Operations Research and Financial Engineering. Springer, New York (2006)
22. Puget, J.F.: Solving flexible job shop scheduling problems (November 2013). https://www.ibm.com/developerworks/community/blogs/jfp/entry/solving_flexible_job_shop_scheduling_problems?lang=en (accessed: March 10, 2015)
23. Roy, B., Sussmann, B.: Les problèmes d'ordonnancement avec contraintes disjonctives. Note DS 9 (1964)
24. Schneeweiß, C.: Distributed decision making, 2 edn. Springer (2003)
25. Scholz-Reiter, B., Görges, M., Philipp, T.: Autonomously controlled production systems - influence of autonomous control level on logistic performance. CIRP Annals - Manufacturing Technology **58**(1), 395–398 (2009)
26. Tay, J.C., Ho, N.B.: Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems. Computers & Industrial Engineering **54**(3), 453–473 (2008)
27. Trentesaux, D.: Distributed control of production systems. Engineering Applications of Artificial Intelligence **22**(7), 971–978 (2009). distributed Control of Production Systems
28. Vrabič, R., Husejnagić, D., Butala, P.: Discovering autonomous structures within complex networks of work systems. CIRP Annals - Manufacturing Technology **61**(1), 423–426 (2012)
29. Zambrano Rey, G., Bonte, T., Prabhu, V., Trentesaux, D.: Reducing myopic behavior in fms control: A semi-heterarchical simulationoptimization approach. Simulation Modelling Practice and Theory **46**, 53–75 (2014). simulation-Optimization of Complex Systems: Methods and Applications
30. Zambrano Rey, G., Pach, C., Aissani, N., Bekrar, A., Berger, T., Trentesaux, D.: The control of myopic behavior in semi-heterarchical production systems: A holonic framework. Engineering Applications of Artificial Intelligence **26**(2), 800–817 (2013)