CIRP Annals - Manufacturing Technology 59 (2010) 465–468

Contents lists available at ScienceDirect

CIRP Annals - Manufacturing Technology

journal homepage: http://ees.elsevier.com/cirp/default.asp

# Dynamic flexible flow shop problems—Scheduling heuristics vs. autonomous control

B. Scholz-Reiter (2)*, H. Rekersbrink, M. Görges

*Department of Planning and Control of Production Systems, University of Bremen and BIBA, Hochschulring 20, 28359 Bremen, Germany*

ARTICLE INFO

*Keywords:*
Production
Simulation
Autonomous control

ABSTRACT

Flexible flow shop problems are well known and common tasks. Two practice-oriented extensions are unrelated parallel machines on the one hand and dynamic aspects in terms of distributed release times on the other hand. Triggered by the growing complexity of logistics systems, the paradigm of central planning is being shifted to decentralized autonomous control. This work focuses on two different autonomous control methods applied to flexible flow shop problems with both of the extensions mentioned above. In order to evaluate these methods, various common scheduling heuristics and one genetic algorithm are taken as references. Problem instances are defined and solved by autonomous control and the reference algorithms. The results of this evaluation are shown and discussed.



## 1. Introduction

Modern manufacturing systems are exposed to an increasingly dynamic and volatile environment. Therefore, the ability to cope with dynamic effects becomes more and more essential to manufacturing companies. The concept of autonomous control promotes distributed and flexible handling of dynamic complexity due to decentralized decision making of local system elements. It also intends to improve the logistic performance of production systems [1]. By contrast, centralized production planning methods are superior in well-defined situations with all information fixed in advance. But they may not be able to adapt to changes in an appropriate manner [2]. This paper addresses the area of transition between static and dynamic conditions in order to derive application ranges and boundaries of centralized planning and autonomous control methods concerning the impact of dynamics. The analysis focuses on the logistic performance of two different autonomous control methods and of twelve centralized scheduling algorithms in a flexible flow shop (FFS) environment. The FFS structure used for the analysis is sketched in Fig. 1 (see [3] also).

A practice-oriented extension is the introduction of unrelated parallel machines with setup times. The literature provides several examples of this problem class and its application in real production systems [4]. Common scheduling heuristics for the FFS with unrelated parallel machines and sequence dependent setup times are taken as reference to benchmark two types of autonomous control methods. The evaluation includes instances with different configurations concerning the number of stages and the number of machines per stage as a parameter of structural complexity. Furthermore, different degrees of dynamics are modeled for each instance and are represented by the incoming workload. In the static situation all jobs are released at the same time, whereas dynamic situations are characterized by distributed inter-arrival times of jobs. In this context, scenarios of different complexity and dynamics will be used to evaluate the following hypothesis: 'While centralized production planning is best for relatively simple and static situations, autonomous control is best for complex and dynamic environments.' The investigation on the dynamic performance of autonomously and conventionally planned production systems is still an active research area [1]. However, a comparative study concerning centralized and autonomous control in FFS problems is missing so far.

## 2. Autonomous control methods

Autonomous cooperating logistic processes are characterized by a shift of decision-making capabilities from the system layer to its elements. This allows single intelligent logistic objects to make and execute decisions according to their own objectives [5]. These objects may either be physical objects like machines or immaterial objects like production orders or jobs. Due to modern information and communication technologies, these objects are able to interact with others in order to gather information about current local system states [1]. On this basis, the objects make decentralized decisions. This aims at improving the achievement of the logistic targets by flexible handling of dynamics and complexity. Existing models of autonomous control have already provided promising results. Previous work confirmed that autonomous control can help increase the logistic performance and robustness of production systems [6,7]. Further studies of a real data based manufacturing system showed that performance of different autonomous control methods depends on the application scenario and on the prioritization of logistic targets [7]. In order to evaluate the logistic performance of autonomous control in a FFS environment, two different autonomous control methods are implemented. These methods can be classified in two categories: *local information methods* and *information discovery methods*. Local information methods gather and process only local information. Contrary to

* Corresponding author.

0007-8506/$ – see front matter © 2010 CIRP.
doi:10.1016/j.cirp.2010.03.030

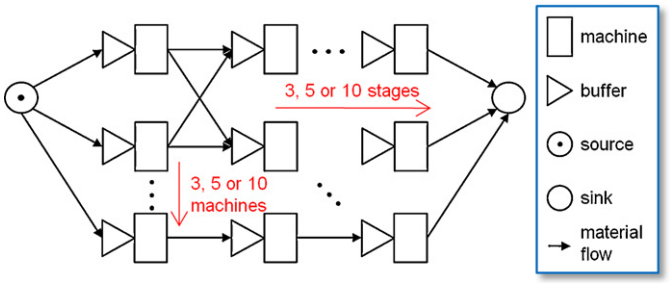


**Fig. 1.** FFS scenarios used for the evaluation.

this, information discovery methods collect information from other objects. This discovery does not cover the whole system, but is directed to information that is relevant for the actual decision.

### 2.1. Local information methods

Local information methods enable jobs to decide about further processing steps. Jobs using one of these methods only gather local information about states of direct succeeding buffers and machines. Several local information methods based on different decision strategies have been designed for manufacturing purposes. These strategies can be classified in rational, bounded rational and combined strategies [8]. Rational methods predict future system states and make decisions. Biologically inspired methods belong to the bounded rational strategies and use aggregated data from past events.

The *Queue Length Estimator* (QLE) method is modeled here as a representative of rational local information methods. Jobs using the QLE method compare the buffer levels of each production line at a certain stage and estimate their actual workload. In order to reduce throughput time, jobs will choose the machine with the lowest workload [6]. The comparison is triggered whenever a job is finished on a machine. At this point the QLE method estimates the workload of each machine and its corresponding buffer for the next stage. Thereby, the QLE considers the expected processing and setup times of all waiting jobs.

### 2.2. Information discovery method

The *Distributed Logistics Routing Protocol* (DLRP) was developed and presented earlier as an information discovery method. Originally, the DLRP was developed for the field of transport logistics (see [9]) and it is now being transferred to production logistics.

The DLRP for production defines two logistic object types: machine objects and job objects. Both decide and execute on their own. The basic concepts for the DLRP as an information discovery method are adapted from routing algorithms that are used in wireless ad hoc communication networks, where routes have to be found in dynamically changing topologies. With the DLRP, job objects are able to make a routing through the production environment.

Due to its complexity, it is not possible to give a detailed description of the developed protocol here. But the fundamentals of the protocol can shortly be described as follows (see [9,10] for details). When a job enters the system, it needs a route through the production system. It sends a route request to the next machine, which fills it with necessary information and sends it ahead to all possible successive machines. This is repeated until the last production step has been reached. The last machine sends back the collected information as a route reply. The job receives several route alternatives by this discovery scheme. After its route decision, the job disannounces old routes and announces the new routes. For the system to have a higher degree of freedom, each job decides on multiple desired routes. Together with the route announcement, a lot of information can be passed to the relevant machines, like production times, setup times, probabilities, urgencies, etc. The jobs do a rerouting after every production step to update their decision base. Because of the ongoing processes, this scheme leads to a continuous cooperative structure – at any time there is enough information for a decision. Jobs decide their preferred routes and machines decide on their setup plan, the dispatching and the next machine for every job that leaves.

In the simulations described below, the decisions of the objects are made as follows: The route decision of the jobs is based on the expected completion time as stated in the received route alternatives with some preference on already announced routes. Two routes are announced with different preference values. The decision for the next machine is based on the preference for the different route announcements. In order to minimize the sum of setup times for a machine, the dispatching is based on the shortest setup time for the next job.

## 3. Scheduling heuristics

The FFS scheduling problem itself is NP-hard (see e.g. [11]). Optimal solutions can only be calculated in feasible time for small problem instances with few jobs. Calculating optimal solutions for larger *dynamic* flow shop scheduling problems takes far too long, not to mention unrelated parallel machines.

In order to find approximate solutions, many heuristic methods have been developed for the classical FFS problem and for the dynamic FFS problem with parallel machines. Usually, the total problem is divided into a sequencing problem and an assignment problem [4]. Both problems are successively solved. At the beginning, a sequence for the first stage is determined by an adapted sequencing algorithm. After that, a scheme, consisting of sorting algorithms and algorithms that select the best solution out of different alternatives, assigns jobs to machines and creates the complete schedule for all stages.

Jungwattanakit et al. [12] made large computational studies to evaluate this algorithmic scheme with different sequencing heuristics. They used the following sequencing algorithms for the first stage: PAL (a slope index heuristic by Palmer), CDS (a best choice heuristic by Campbell, Dudek and Smith), GUP (a slope index heuristic by Gupta), DAN (a heuristic by Dannenbring), NEH (a constructive heuristic by Nawaz, Enscore and Ham). In addition to these heuristics, Jungwattanakit et al. created a *genetic algorithm* (GAL). This algorithm takes the sequence for the first stage as the genome and puts the results of all heuristics into the initial population.

All of these scheduling heuristics (SCD) try to minimize the makespan ($C_{\max}$) of a given problem instance. However, they naturally have some shortcomings in dynamic environments where arrival of jobs is distributed in time. Therefore, we modified these scheduling heuristics to include a *rolling planning horizon*, where the original job sequence is split chronologically into parts with 25 jobs and then these subinstances are solved. In the remainder of this paper, these scheduling heuristics with rolling planning horizon (rSCD) are referred to rPAL, rCDS, rGUP, rDAN, rNEH and rGAL.

## 4. Problem description and instances

In order to obtain comparability for this evaluation, an existing problem formulation from Jungwattanakit et al. [12] was chosen. It defines a FFS problem with unrelated parallel machines and sequence dependent setup times. In addition to this problem formulation, job types were also defined.

Consider a FFS system as sketched in Fig. 1. There are $T$ stages and $M^t$ unrelated machines at every stage $t$. All job types have different processing times $p^t_{m,s}$ on the different machines and they have different sequence dependent setup times $s^t_{m,u,s}$. The completion time $C_j$ and the throughput time $T_j$ are defined for the entire network at the end of the last stage $t = T$.

Parameters and variables:

$J$ number of jobs
$S$ number of job types (indices $s$ and $u$)

| | | 3 machines | | 5 machines | | 10 machines | |
|---|---|---|---|---|---|---|---|
| | | $\lambda$ = st | $\lambda$ =0.5 | $\lambda$ = st | $\lambda$ =0.5 | $\lambda$ = st | $\lambda$ =0.5 |
| 3 stages | best for $C_{max}$ | SCD | SCD | SCD | SCD | SCD | AC |
| | best for TPT | SCD | SCD | SCD | AC | SCD | AC |
| 5 stages | best for $C_{max}$ | SCD | SCD | SCD | SCD | SCD | AC |
| | best for TPT | SCD | AC | SCD | SCD | SCD | AC |
| 10 stages | best for $C_{max}$ | SCD | SCD | SCD | SCD | SCD | SCD |
| | best for TPT | SCD | SCD | SCD | SCD | SCD | SCD |

SCD = scheduling heuristic, AC = autonomous control

**Fig. 2.** Best performers for static and nearly static scenarios.

$T$ number of stages
$M^t$ number of parallel machines at stage $t$
$r_j$ release time of job $j$
$s^t_{m,u,s}$ setup time from job type u to s at machine m at stage $t$
$ps^t_s$ standard processing time of job type sat stage $t$
$\upsilon^\tau_\mu$ relative speed of machine $m$ at stage $t$
$p^t_{m,s}$ processing time of $s$ on $m$ at $t$, where $p^t_{m,s} = ps^t_s/\upsilon^\tau_\mu$
$C_j$ completion time of job $j$
$C_{\max}$ makespan, where $C_{\max} = \max(C_j)$
$T_j$ throughput time of job $j$; where $T_j = C_j - r_j$
TPT mean throughput time; where TPT = mean($T_j$)

The problem instances for this evaluation were chosen very close to the so called 'large size problems' defined in [12]: the number of jobs $J$ is 250. The number of job types $S$ is set to 10, they are uniformly distributed. The number of stages $T$ and the number of parallel machines $M^t$ can be 3, 5 or 10. The standard processing times $ps^t_s$ are integers uniformly distributed in the interval [1 50]. The relative speeds $\upsilon^\tau_\mu$ are uniformly distributed in [0.7 1.3]. The setup times $s^t_{m,u,s}$ are integers uniformly distributed in [1 10]. The release times $r_j$ are all 0 (static, referred as 'st') or a Poisson process with $\lambda$ jobs per time unit. The value of $\lambda$ can be st, 0.5, 0.1, 0.075, 0.05, 0.025. Lambda denotes dynamic aspects of the workload and covers the whole range from overloaded to underloaded situations.

With 3 variants of $T$, 3 variants of $M^t$ and 6 variants of $r$, there are 54 different scenarios. Each scenario has 5 instances with different random numbers. The results below are given as a mean over these 5 instances.

## 5. Results

Due to the size of this evaluation, the key findings of the study are presented here using examples. Aside from the makespan, the throughput time (TPT) is one of the most important logistic measures for the performance of a system. For planning situations, the makespan is the more important value, while throughput time is more important for control situations with a continuous workload. The results are, therefore, presented for both indicators.

For the static and nearly static situation ($\lambda$ = st and $\lambda$ = 0.5), the results are clear. Fig. 2 shows the type of the algorithm which performed best for a particular combination of $\lambda$, number of machines and number of stages. The SCD methods perform best in all static scenarios for both performance measures TPT and $C_{\max}$. In the nearly static situation, the autonomous control (AC) methods perform better in some cases, although SCD and AC are very close in these cases. Fig. 3 shows examples of this behavior: with decreasing $\lambda$ the AC methods become more advantageous.

Fig. 3 shows the results of the GAL, the rGAL and both autonomous control methods (QLE and DLRP) for the 5 stages, 5 machines scenarios. The results for $C_{\max}$ and TPT are normalized to a theoretical lower bound calculated for every single instance. This normalization takes the lowest possible production time, the lowest possible setup time and the last release time into account.

Fig. 3 confirms that the GAL is the best choice for the static problem. It leads to the lowest $C_{\max}$ and TPT. However, rGAL performs better in a more dynamic environment. These rolling horizon heuristics are especially made for dynamic environments with continuous workloads and naturally perform better in terms of throughput time. In the interval between $\lambda$ = 0.5 and $\lambda$ = 0.1 the rGAL curve crosses the curve of the GAL for both performance measures. But the decision for SCD or rSCD is not clear in all other scenarios. Sometimes, the intersections of $C_{\max}$ and TPT are at different lambdas. Additionally, the workload of a real company is not fixed but fluctuates around a working level. So the control method for realistic situations must be suitable for a wide range of dynamic degrees.

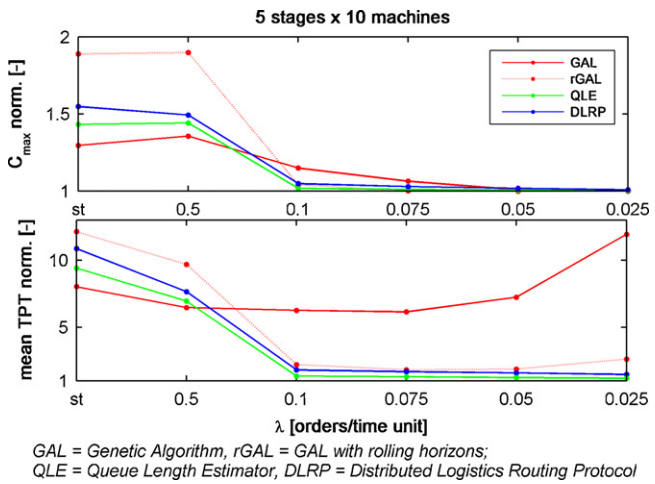


**Fig. 3.** Results for 5 stages and 5 machines.

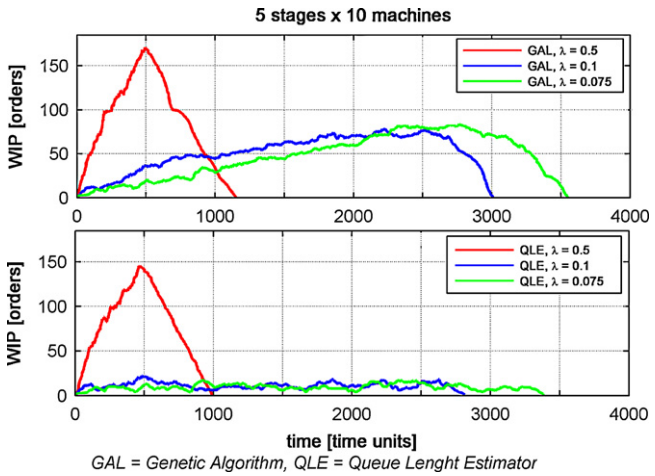


**Fig. 4.** Examples for WIP over time for 5 stages and 10 machines.

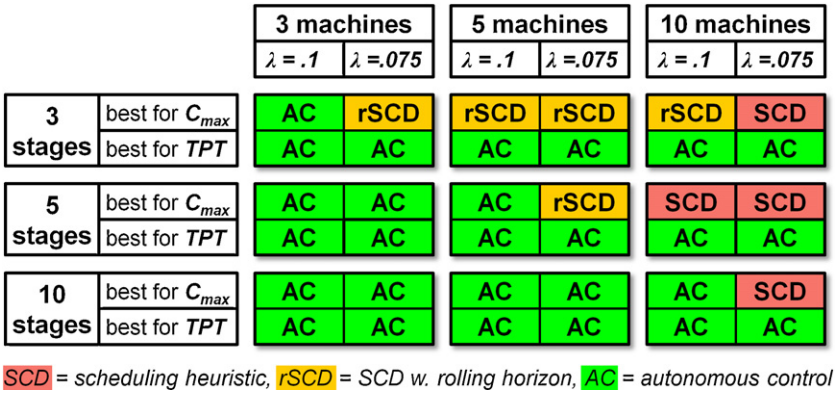


**Fig. 5.** Best performers for capacity-near workloads.

Fig. 3 shows that there is a TPT transition from λ = 0.5 to λ = 0.1. One can say that reasonable workloads for the scenarios used are around λ = 0.1, and the system is overloaded to the left of 0.1, with more jobs entering the system than the system can process.

From this point of view, Fig. 3 indicates that both of the AC methods lead to systems with a higher capacity than the SCD or rSCD methods. Furthermore, all methods except GAL are near optimal values with a very low workload.

Fig. 4 gives a more detailed view on the differences between SCD an AC. It shows the WIP over time for 5 stages and 10 machines – exemplarily for GAL and QLE. For the overloaded situation (λ = 0.5), both methods naturally build up high WIP levels. For more dynamic situations (λ = 0.1 and λ = 0.075), both methods have a similar makespan performance, but the GAL leads to significantly higher WIP than the QLE method. Because of the increasing WIP, one can say that the system with GAL is still somewhat overloaded. SCD methods in general focus on minimizing setup times, but this leads to unsuitable sequences for continuous workloads, high WIP and high TPT values. The rolling horizon methods reduce the TPT values for dynamic scenarios, but the basic problem remains.

As noted above, reasonable workloads for the scenarios used are around λ = 0.1. Therefore, Fig. 5 shows the aggregated results for λ = 0.1 and λ = 0.075. Fig. 5 confirms that the situation for realistic workloads is not as clear as for static ones. In terms of TPT, the AC methods show the best performance for all scenarios. But the rGAL values are close to the AC ones in the 'upper right corner' of the table (see also Fig. 3). Concerning $C_{max}$ one can say that more parallel machines and fewer stages favor scheduling methods in general. Within these methods, more parallel machines seem to favor classic SCD, while fewer stages seem to favor rSCD methods. AC methods seem to be best when there are fewer parallel machines than stages.

### 5.1. Local information vs. information discovery methods

Fig. 3 shows a good example of the performance differences between the QLE and the DLRP autonomous control methods. In nearly all scenarios, both methods have similar values, but QLE performs better than the DLRP. The DLRP is designed for situations in which decisions have long range consequences. In terms of decision consequences, the FFS problem is simple because one routing decision does not affect later decisions. For this environment, the DLRP seems to be too complex and the simpler decisions made by the QLE method seem to be more effective. DLRP may be preferable in complex scenarios such as production networks with geographically dispersed plants.

## 6. Conclusion

The hypothesis stated at the outset may be refined by the results of the study. Within the range of the used scenarios, centralized production planning using scheduling heuristics appears to be best for static situations regardless of system complexity. The complexity can be split into the two dimensions: parallel machines and stages. While autonomous control is best for many stages and dynamic environments, many parallel machines promote centralized production planning. This lies in the different problem solving strategies of the two approaches. The central strategy takes all data and tries to satisfy all constraints at once, which has disadvantages with changing situations and with too many constraints. The autonomous strategy in contrast divides the problem into subproblems and solves them successively, which has disadvantages for well-defined problems.

Next research steps for the evaluation of autonomous control in general are to study extensions like machine breakdowns and rush orders, as well as general problem formulations like job shop or open shop problems. Another step will be the combination of transport and production logistics e.g. in production networks with geographically dispersed plants, where decisions have long range consequences.

## Acknowledgements

This research is funded by the German Research Foundation (DFG) as a part of the Collaborative Research Centre 637 "Autonomous Cooperating Logistic Processes: A paradigm Shift and its Limitations".

## References

[1] Windt K, Böse F, Philipp T (2008) Autonomy in Production Logistics – Identification, Characterisation and Application. *International Journal of Robotics and CIM* 24/4:572–578.
[2] Kim J-H, Duffie NA (2004) Backlog Control Design for a Closed Loop PPC System. *CIRP Annals-Manufacturing Technology* 53/1:357–360.
[3] Quadt D, Kuhn H (2007) A Taxonomy of Flexible Flow Line Scheduling Procedures. *European Journal of Operational Research* 178/3:686–698.
[4] Allahverdi A, Ng CT, Cheng TCE, Kovalyov M (2008) A Survey of Scheduling Problems with Setup Times or Costs. *European Journal of Operational Research* 187/3:985–1032.
[5] Windt K, Hülsmann M (2007) Approaches to Methods of Autonomous Cooperation and Control and Examples for the Material Flow Layer. in Hülsmann M, Windt K, (Eds.) *Understanding Autonomous Cooperation & Control – The Impact of Autonomy on Management, Information, Communication, and Material Flow*. Springer, Berlin, pp. 295–301.
[6] Scholz-Reiter B, Freitag M, de Beer C, Jagalski T (2005) Modelling and Analysis of Autonomous Shop Floor Control. *Proceedings of 38th CIRP International Seminar on Manufacturing Systems*, Florianopolis, Brazil, CD-ROM, .
[7] Scholz-Reiter B, Görges M, Philipp T (2009) Autonomously Controlled Production Systems – Influence of Autonomous Control Level on Logistic Performance. *CIRP Annals-Manufacturing Technology* 58/1:395–398.
[8] Windt K, Becker T (2009) Applying Autonomous Control Methods in Different Logistic Processes – A Comparison by Using an Autonomous Control Application Matrix. *Proceedings of the 17th Mediterranean Conference on Control and Automation*, Thessaloniki, Greece, .
[9] Rekersbrink H, Makuschewitz T, Scholz-Reiter B (2008) A Distributed Routing Concept for Vehicle Routing Problems. *Logistics Research* 1/1:45–52.
[10] Scholz-Reiter B, Rekersbrink H, Freitag M (2006) Internet Routing Protocols as an Autonomous Control Approach for Transport Networks. *Proceedings of the 5th CIRP ICME Seminar*, 341–345.
[11] Garey MR, Johnson DS (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco.
[12] Jungwattanakit J, Reodecha M, Chaovalitwongse P, Werner F (2005) An Evaluation of Sequencing Heuristics for Flexible Flowshop Scheduling Problems with Unrelated Parallel Machines and Dual Criteria, Preprint series: 05-28 (MSC: 90B35) Otto-von-Guericke-Universität Magdeburg Germany.