HAL
open science

A comparison of priority rules for the job shop scheduling problem under different flow time- and tardiness-related objective functions

Veronique Sels, Mario Vanhoucke, Nele Gheysen

To cite this version:

Veronique Sels, Mario Vanhoucke, Nele Gheysen. A comparison of priority rules for the job shop scheduling problem under different flow time- and tardiness-related objective functions. International Journal of Production Research, 2011, pp.1. 10.1080/00207543.2011.611539 . hal-00728900

HAL Id: hal-00728900

https://hal.science/hal-00728900

Submitted on 7 Sep 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A comparison of priority rules for the job shop scheduling problem under different flow time- and tardiness-related objective functions

Journal:	*International Journal of Production Research*
Manuscript ID:	TPRS-2010-IJPR-0999.R2
Manuscript Type:	Original Manuscript
Date Submitted by the Author:	09-Jun-2011
Complete List of Authors:	Sels, Veronique; Ghent University, Dept of Mgmt Info, Ops Mgmt and Tech Policy Vanhoucke, Mario; Ghent University, Dept of Mgmt Info, Ops Mgmt and Tech Policy Gheysen, Nele
Keywords:	JOB SHOP, FLEXIBLE JOB SHOP, JOB SHOP SCHEDULING
Keywords (user):	

SCHOLARONE™
Manuscripts

A comparison of priority rules for the job shop scheduling problem under different flow time- and tardiness-related objective functions

Veronique Sels[1], Nele Gheysen[1], and Mario Vanhoucke[1,2,3]

[1]Faculty of Economics and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Gent (Belgium), veronique.sels@ugent.be), mario.vanhoucke@ugent.be

[2]Operations and Technology Management Centre, Vlerick Leuven Gent Management School, Reep 1, 9000 Gent (Belgium)

[3]Department of Management Science and Innovation, University College London, Gower Street, London WC1E 6BT (United Kingdom)

For Peer Review Only

Abstract

In this paper, a comparison and validation of various priority rules for the job shop scheduling problem under different objective functions is made. In a first computational experiment, 30 priority rules from literature are used to schedule job shop problems under two flow time-related and three tardiness-related objectives. Based on this comparative study, the priority rules are extended to 13 combined scheduling rules in order to improve the performance of the currently best-known rules from literature. Moreover, the best performing priority rules on each of these five objective functions are combined into hybrid priority rules in order to be able to optimize various objectives at the same time. In a second part of the computational experiment, the robustness on the relative ranking of the performance quality is checked for the various priority rules when applied on larger problem instances, on the extension of multiple machines possibilities per job as well as on the introduction of sequence-dependent setup times. Moreover, the influence of dynamic arrivals of jobs has also been investigated to check the robustness on the relative ranking of the performance quality between static and dynamic job arrivals. The results of the computational experiments are presented and critical remarks and future research avenues are suggested.

Keywords: Job shop scheduling, priority rules, setup times, dynamic job shop, flexible job shop

1 Introduction

The job shop scheduling problem is a well-known optimization problem in literature. The problem can be briefly described as follows. There are a number of jobs that need to be processed on a number of machines. The jobs visit each of the machines in a predefined sequence. The goal is to schedule the jobs on the machines, respecting the constraints that each machine can only process one job at a time and that preemption of the jobs is not allowed, while minimizing one or more measures of performance. The classical job-shop problem, where the objective is to minimize the makespan ($J||C_{max}$), is a NP-hard combinatorial problem. Since its practical importance, numerous exact as well as heuristic solution procedures have been developed in the past four decades. These solution procedures are able to find optimal or near-optimal solutions, although the required computational cost can increase tremendously if the problem size grows. In practice, priority rules (in the static case) or dispatching rules (in the dynamic case) have often been applied to overcome these computational costs. Such rules inspect the waiting jobs and select the job with the highest priority to be processed next. Although these rules are not able to outperform the previous mentioned exact and heuristic solution procedures, they are frequently used because of their ease of implementation and their low time complexity (Ho and Tay, 2005). As a result, despite the recent developments in computer science, leading to a vast amount of meta-heuristic solution approaches to solve challenging problems such as the job shop problem, we believe that the use of simple and straightforward priority rules still deserves a place in the academic literature and practical scheduling environments. Indeed, priority rules are often used as sub-methods in more advanced (meta-) heuristic solution approaches (e.g. Chiang and Fu (2008)). Moreover, due to practical reasons (size of the problem, complexity of the scheduling environment, lack of scheduling software), simple and easy-to-understand priority rules are often a matter of degree in real scheduling environments (Jayamohan and Rajendran, 2000). Finally, due to the dynamic nature of jobs arriving at the company, simple priority rules, which act as dispatching rules, can easily determine which jobs get priority during the manufacturing process.

The contribution of this paper is fourfold: First, the paper gives an overview of a large set of priority rules and a validation of their performance on five different objective functions used for solving the job shop scheduling problem. It is, to the best of our knowledge, the most complete study on priority rules for static and dynamic job shop scheduling problems and builds further upon results available in the academic literature. Second, the study

For Peer Review Only

also strives to combine different priority rules, inspired by the evaluation and validation on various objectives (considered separately or simultaneously), in order to construct hybrid priority rules that outperform the currently best-known priority rules from literature. This choice is inspired by the findings of Barman (1997), Holthaus and Rajendran (1997) and Rajendran and Holthaus (1999), who observed that no single priority rule has been found to perform well on all flow time- and tardiness-related objectives and that the combination of priority rules can overcome this limitation (Subramaniam et al., 2000a; Dominic et al., 2004; Tay and Ho, 2008). Thirdly, the results of all priority rules are checked on their robustness on the relative ranking of the performance quality for different extensions of the traditional job shop problem. Finally, the study extends the static job shop scheduling problem to a dynamic environment, where not all jobs are available at the start of the scheduling period. In doing so, the priority rules will be used as dynamic dispatching rules, and the results of their dynamic performance will be compared with their static solution quality.

The outline of this paper can be summarized as follows: Section 2 gives the job shop problem formulation under two flow time-related and three tardiness-related objective functions. Section 3 presents the results of the computational experiments. The experiments consist of three main tests. In a first experiment, the relative performance of various priority rules is measured for the five objective functions used to solve the job shop problem. In a second set of experiments, the traditional job shop scheduling problem is extended with different features and the relative rankings of the priority rules obtained during the first test experiment are re-evaluated and critically analyzed along the different job shop extensions. In Section 4, a third experiment has been set up, which extends the static experiment to dynamic job arrivals in order to investigate whether the relative performance of the priority rules remains relatively robust when they act as dynamic dispatching rules. Section 5 draws overall conclusions and gives future research avenues.

2 Job shop problem

The job shop scheduling problem (JSP) is a well-known optimization problem, which can be formulated as follows. A set of N jobs (index $j = 1, 2, ..., n$) needs to be processed on a set of M different machines (index $i = 1, 2, ..., m$). The processing of a job j on a machine i is referred to as operation O_{ij}, which has a known deterministic processing time p_{ij}. Each job visits each machine exactly once according to its own predefined sequence

For Peer Review Only

and preemption of the jobs is not allowed. The machines can process only one operation at a time and a job cannot be processed by two machines at the same time. The objective is to schedule each operation on the machines, taking the precedence constraints into account such that a certain objective Z is minimized.

The traditional job shop scheduling problem aims at an optimal solution with a minimal makespan C_{max}. The problem can be represented by $J||C_{max}$ using the classification scheme of Graham et al. (1979) and is known to be NP-hard (Garey et al., 1976). It can be mathematically represented as follows (Pinedo, 2002): Let the variable S_{ij} denote the starting time of operation O_{ij}. Let set O denote the set of all operations O_{ij} and set A the set of all precedence constraints, i.e. $O_{ij} \rightarrow O_{hj}$, which requires job j to be processed on machine i before it is processed on machine h.

$$\text{minimize} \quad C_{max}$$

$$S_{hj} - S_{ij} \geq p_{ij} \qquad \forall O_{ij} \rightarrow O_{hj} \in A \tag{1}$$

$$C_{max} - S_{ij} \geq p_{ij} \qquad \forall O_{ij} \in O \tag{2}$$

$$S_{ij} - S_{ik} \geq p_{ik} \vee S_{ik} - S_{ij} \geq p_{ij} \qquad \forall O_{ik}, O_{ij}, i = 1, ..., m \tag{3}$$

$$S_{ij} \geq 0 \qquad \forall O_{ij} \in O \tag{4}$$

The first set of constraints (1) ensures that operation O_{hj} cannot start before operation O_{ij} is completed. The second constraint (2) defines the makespan value. The disjunctive constraints of (3) ensure that a precedence relation exists among operations of different jobs that have to be processed on the same machine. Constraint (4) assures that the starting times of all operations are non-negative integers.

Next to the makespan objective, various other objectives have been proposed in literature and studied by different researchers over the past decennia. Time-based objectives are commonly considered in the literature because these measures of performance are surrogate measures for cost-based measures of performance (Blackstone et al., 1982; Rajendran and Holthaus, 1999; Holthaus and Rajendran, 2000). To be more precise, flow time-related and tardiness-related objectives are often minimized since the associated inventory and tardiness costs are assumed to be directly proportionate to the time periods of flow time and tardiness, respectively (Rajendran and Holthaus, 1999; Jayamohan and Rajendran, 2004). Moreover, in a more customer-oriented and highly competitive environment, which characterizes the business environment of the last decennia, tardiness-related performance measures become crucial in measuring the true performance of companies with respect to overall customer service.

For Peer Review Only

The current paper has a focus on the validation, comparison and improvement of both flow time-related and tardiness-related performance measures, trying to optimize them separately as well as simultaneously. The two flow time-related and three tardiness-related classes of performance measures result in different versions of the job shop scheduling problem. These two classes result in five objective functions studied and are summarized along the following lines:

- Flow time-based performance measures:
  - Makespan ($C_{max} = \max_j C_j$)
  - Mean flow time ($\overline{F} = \frac{1}{n}\sum_{j=1}^{n} F_j$, with $F_j = C_j - r_j$)
- Tardiness-based performance measures:
  - Proportion of tardy jobs ($\%T = \frac{\sum_{j=1}^{n}\delta(T_j)}{n}$, with $\delta(T_j) = 1$ if $C_j > d_j$, 0 otherwise)
  - Mean tardiness ($\overline{T} = \frac{1}{n}\sum_{j=1}^{n} T_j$ with $T_j = \max(0, C_j - d_j)$)
  - Maximum tardiness ($T_{max} = \max_j T_j$)

In the remainder of this paper, an overview will be given of various priority rules used to solve the five objective functions for the job shop scheduling problem. In the next section, these priority rules will be combined to search for further improvements and will be sometimes linked to other objective functions for which they were originally not constructed.

3 Computational experiments

In this section, various priority rules taken from literature as well as combined versions will be tested for the five objective functions using data instances from literature. These instances are sometimes extended with randomly generated extensions such as due dates and setup times. In Section 3.1, the approach to compare, validate and improve the priority rules to solve different variants of the static job shop scheduling problem will be outlined. In Section 3.2, the computational results for all existing and newly developed priority rules on the five objective functions will be displayed and discussed. In the remaining subsections, these results are compared with various extensions of the traditional job shop problem. More precisely, Section 3.3 will rerun all experiments on larger problem instances. Section

For Peer Review Only

3.4 adds multiple machine possibilities to the job shop problem in order to solve the so-called flexible job shop scheduling problem. Finally, in Section 3.5 the job shop scheduling problem is extended with sequence-dependent setup times. As already mentioned, this section studies the scheduling of job shops under static assumptions, where the arrival times of the jobs are externally given and static. In Section 4, we will extend our study to the dynamic job shop scheduling problem.

3.1 Approach

This section briefly gives an overview on the approach taken to set up the various experiments. More precisely, it describes the different data instances from literature used throughout the experiments and the required adjustments for the extended job shop scheduling problem. A short discussion on how these priority rules are evaluated and compared is given as well.

Data instances from literature: Table 1 displays the different datasets used in the computational experiments of the following sections. Each time the size is mentioned, a reference to the section where the instances are used and possible extensions generated to the data where necessary. The data used in Section 3.2 to validate and compare the priority rules on the traditional job shop scheduling problem contains 82 instances, i.e. the abz- (Adams et al., 1988), ft- (Fisher and Thompson, 1963), la- (Lawrence, 1984), orb- (Applegate and Cook, 1991), swv- (Storer et al., 1992) and yn- (Yamada and Nakano, 1992) instances, as given in Table 1. In order to test the impact of larger instances on the performance of the priority rules (Section 3.3), 80 instances defined by Taillard (1993) are used with sizes up to 100x20. Section 3.4 uses the data as defined by Hurink et al. (1994), which consist of the abz-, ft-, la- and orb- instances, as well as the car-instances defined by Carlier (1978). Hurink et al. (1994) created four different sets (*sdata*, *edata*, *rdata* and *vdata*), in which these instances have a different flexibility factor. This flexibility factor determines to how many different machines an operation can be assigned. The *sdata*-set contains the standard job shop instances where each operation can be assigned to exactly only one machine. In the *edata*-set, a few number of operations can be assigned to different machines, while in the *rdata*-set, most operations can be assigned to a few number of different machines. The *vdata*-set contains the most flexible instances, where each operation can be assigned to many different machines.

For Peer Review Only

Extensions made on the data instances: In order to test whether the priority rules show a similar performance on the job shop problem with setup times (Section 3.5), the 82 instances have been extended with sequence-dependent setup times (s_{ij}) by randomly generating an nxn matrix with setup times generated from the interval [p_{ij}^{min},p_{ij}^{max}], where p_{ij}^{min} is the minimal and p_{ij}^{max} the maximal processing time of the problem instance.

All instances have been extended by due dates when the due date based priority rules are tested. To that purpose, due dates (d_j) are assigned to jobs by applying the commonly used total work content (TWK) method of Blackstone et al. (1982), where the due date is set as a multiple of its total work content of that job. This can be easily done by summing the processing times of all operations for each job, and multiplying this summation with a certain allowance factor value (c) that represents the due date tightness. The values for the allowance factor have been experimentally set such that approximately x% of the jobs will finish later than their due date (Jensen et al., 1995; Russell et al., 1987). To that purpose, the job shop instances have been scheduled with the quick and easy SPT rule applicable to various objective functions for different values of the allowance factor. Each time, the percentage of jobs that finish late is reported. Computational experience has shown that this percentage varies between 100% and 45% when the allowance factor has been set to 1 and 2, respectively. In the remainder of the paper, results will be reported for an allowance factor equal to 1.9, which corresponds to approximately 50% of the jobs that are expected to finish late. Other allowance factor values have shown no significant differences throughout the experiments and therefore, separate results are not reported.

Evaluation: In order to measure the quality of the various priority rules, we have not chosen to report an absolute validation in comparison with the optimal or best-known solution (for data from literature) or from a lower bound (for generated data). The reason is that the absolute performance of priority rules for the job shop problem has little value since other more advanced and computationally more demanding procedures such as meta-heuristics lead to much better solutions. Moreover, for some of the objective functions tested in this paper, neither best-known solution nor lower bounds are available. Instead, we have opted for a ranking of the priority rules based on their relative performance in comparison with the best performing priority rule for each of the five objective functions. Therefore, each priority rule's performance is measured as the percentage deviation $\%Dev$ from the best performing priority rule, as follows:

For Peer Review Only

Table 1: Data instances used for the computational experiments

Testset (# instances)	**Size** (nxm)	**Section(s)**	**Extensions**
abz (5)	10x10, 20x15	3.2, 3.4, 3.5 & 4	due dates, setup times
ft (3)	6x6, 10x10, 20x5	3.2, 3.4, 3.5 & 4	due dates, setup times
la (40)	10x5, 15x5, 20x5, 10x10, 15x10, 20x10, 30x10, 15x15	3.2, 3.4, 3.5 & 4	due dates, setup times
orb (10)	10x10	3.2, 3.4, 3.5 & 4	due dates, setup times
swv (20)	20x10, 20x15, 50x10	3.2, 3.5 & 4	due dates, setup times
yn (4)	20x20	3.2, 3.5 & 4	due dates, setup times
Taillard (80)	15x15, 20x15, 20x20, 30x15, 30x20, 50x15, 50x20, 100x20	3.3	due dates
car (8)	11x5, 13x4, 12x5, 14x4, 10x6, 8x9, 7x7, 8x8	3.4	due dates

For Peer Review Only

$$\%Dev = \frac{Z^X - Z^B}{Z^B} \tag{5}$$

where Z^X is the average objective function value over all (82) instances obtained by the priority rule under evaluation and Z^B the average objective function value over the same instances obtained by the best priority rule used to solve the job shop problem with the objective function under study.

3.2 Comparative results

In this section, the relative performance of commonly used priority rules to solve the job shop scheduling problem is evaluated based on their average deviation from the best performing rule for each of the five objective functions. The abbreviations of the priority rules are given in the first column of Table 2. The relative performances are calculated according to equation (5). For more information on these abbreviations and the construction of the combined rules, we refer to the appendix at the end of this paper. The table contains two blocks separated by a horizontal line, and two classes (underlines and non-underlined numbers) in the first block, classifying the priority rules in three different categories, as follows:

- Commonly used priority rules: This block contains a set of priority rules for which it has been shown in literature that they perform well on the objective function under study. These priority rules are displayed above the horizontal line and are underlined. The best performing rule from this list has a value of 0.00 (i.e. $Z^X = Z^B$), and all other priority rules in this column are compared against this 0.00 value.

- Other well-known priority rules: This block consists of priority rules from literature that are known to perform well on other objective functions than the objective function under study, but also seem to perform well on that objective function. These priority rules are displayed above the horizontal line and are not underlined. Negative values denote priority rules that perform better than the best performing rule known from literature (the rule with the 0.00 value).

- Newly developed priority rules: This block contains priority rules that are combined versions of the priority rules from literature in order to make further improvements. This block is inspired by the remarks of Dominic et al. (2004) who concluded that

For Peer Review Only

there is still room for improvements by combining good performing priority rules from literature. These priority rules are displayed in the bottom block of the table. The best performing rule is indicated in bold.

The results of the comparative study are displayed in Table 2 for each of the five objective functions. It is worth noting the similar comparative studies has been made by other authors, such as Waikar et al. (1995), Holthaus and Rajendran (1997), Rajendran and Holthaus (1999), Holthaus and Rajendran (2000), Jayamohan and Rajendran (2000), Jayamohan and Rajendran (2004), Dominic et al. (2004) and Chiang and Fu (2007), but with a slightly different focus or limited scope compared to our full comparison on the five objective functions. The results can be briefly summarized per objective function along the following paragraphs.

Makespan (Z_1): The second column of Table 2 displays the rankings on the makespan objective function and shows that the newly developed FDD/MWKR priority rule performs on average 2.31% better than the currently best performing OPFSLK/PT priority rule known in literature (Jayamohan and Rajendran, 2000). The CR+SPT rule, which has been originally developed by Anderson and Nyirenda (1990) to solve tardiness-related objectives, is able to outperform the OPFSLK/PT priority rule as well, as shown by the negative value -0.33. It does, however, not outperform the newly combined FDD/MWKR rule.

Mean flow time (Z_2): The mean flow time results are shown in the third column of Table 2 and indicate that the newly developed 2PT+LWKR+FDD priority rule outperforms all other rules and leads to an improvement of 0.59% compared to the currently best performing LWKR+SPT priority rule known in literature (Dominic et al., 2004). This newly developed priority rule contains information about both the processing times (flow time-related objectives) and the due dates (tardiness-related objectives), such that this priority rule is an ideal candidate to perform well on various objective functions at the same time, as discussed at the end of this section.

Proportion of tardy jobs (Z_3): The fourth column of Table 2 shows the results for the proportion of tardy jobs and reveal that the newly developed AVPRO+SPT+LWKR priority rule has a 2% improvement compared to the best performing LWKR rule in literature (Muhlemann et al., 1982). The best performing rules in the list again contain both processing time and due date information. It is interesting to note that the rankings for the

For Peer Review Only

Table 2: Performance of priority rules for the 5 objective functions (%Dev)

Rule	Makespan (Z_1)	Mean flow time (Z_2)	Proportion of tardy jobs (Z_3)	Mean tardiness (Z_4)	Maximum tardiness (Z_5)
2PT+WINQ+NPT	3.80	3.44	8.29	6.05	17.89
ATC	8.25	8.73	16.42	36.35	64.86
AVPRO	9.83	1.77	2.54	-0.58	14.57
COVERT	5.76	10.40	22.06	31.69	50.75
CR	0.60	29.59	44.63	114.22	57.43
CR+SPT	-0.33	5.71	13.97	15.07	17.18
EDD	9.55	1.92	3.43	0.00	14.27
FDD	3.34	27.57	43.41	102.67	24.42
FIFO	6.54	30.18	43.20	117.85	63.03
LIFO	7.27	9.75	21.59	35.39	63.55
LPT	14.93	27.41	41.53	110.51	103.05
LWKR	11.90	1.01	0.00	-0.11	34.48
LWKR+SPT	11.40	0.00	-1.33	-6.21	25.96
MCOVERT	2.22	9.72	35.96	26.56	9.37
MOD	1.68	4.07	10.26	8.80	21.26
MOPNR	4.10	35.87	49.44	142.19	64.87
MWKR	0.55	32.02	47.08	127.44	67.54
OPFSLK/PT	0.00	9.65	23.77	26.33	13.04
RR	1.92	5.38	12.57	13.89	23.33
Slack	4.15	13.97	41.97	50.59	12.38
Slack/OPN	2.92	23.69	29.33	87.28	21.03
Slack/RPT+SPT	1.93	3.94	11.21	7.75	19.53
SPT	1.68	4.07	10.26	8.80	21.26
SPT+PW	4.77	3.13	4.42	8.95	33.71
SPT+PW+FDD	1.66	6.76	16.72	16.85	24.84
SPT+WINQ	0.64	4.14	11.07	9.66	19.09
SPT+WINQ+NPT+WSL	3.77	6.73	15.10	19.54	1.86
SPT+WINQ+SL	1.39	7.99	19.66	25.91	0.00
SPT/MWKR	3.88	12.89	23.23	49.83	51.86
WINQ	7.26	9.32	20.50	36.51	62.50
2PT+LWKR	10.73	-0.21	1.07	-6.97	24.79
2PT+LWKR+EDD	10.50	0.12	-1.49	-6.85	20.80
2PT+LWKR+FDD	7.59	**-0.59**	-0.56	**-10.61**	9.21
2PT+LWKR+Slack	8.53	-0.20	0.69	-9.91	11.19
AVPRO+SPT+LWKR	10.83	0.14	**-2.00**	-5.67	27.13
EDD/MOPNR	0.46	22.28	32.82	80.71	26.16
FDD/MWKR	**-2.31**	20.91	37.26	79.24	45.37
LWKR+MOD	11.40	0.00	-1.33	-6.21	25.96
SPT+LWKR+Slack	9.19	0.15	-0.43	-7.98	11.49
SPT+PW+LWKR	10.46	0.74	-0.05	-2.02	34.67
SPT+Slack	2.59	8.39	13.91	21.46	**-3.31**
SPT+WINQ+LWKR	10.89	0.12	-0.83	-5.47	23.90
SPT+WINQ+Slack	2.45	8.64	13.13	22.99	-3.14

For Peer Review Only

best results show strong similarities with the performance for the mean flow time objective. In addition, eight of the newly developed rules were able to outperform the priority rules known in literature.

Mean tardiness (Z_4): This objective has been widely studied in literature and hence, the fifth column of Table 2 serves, more than the other columns, as an integrated comparative overview to support the fragmentary and sometimes conflicting results in literature on the performance on the mean tardiness objective. Again, the results show that the 2PT+LWKR+FDD priority performs best on the objective under study, leading to a 10.61% improvement compared to the best performing EDD rule (Baker and Bertrand, 1981). It has been shown earlier that this combined priority rule also has an improved performance on the mean flow time objective.

Maximum tardiness (Z_5): While the results for the mean tardiness were rather incomplete and often contradictory, the results on the maximum tardiness objective were more clear in literature. The best performing priority rule in literature is the SPT+WINQ+SL rule (Holthaus and Rajendran, 1997; Jayamohan and Rajendran, 2000). The SL part of this rule measures the deviation between a job's due date and its completion time and is a negative number when the completion time is greater than or equal to the due date, and zero otherwise. The last column of Table 2 displays comparative results for the maximum tardiness objective and illustrate that incorporating negative and/or positive slack values (Slack) instead of only focusing on non-positive slack values (SL) leads to better solutions. The Slack priority rule differs from the SL rule in its calculation of positive slack values in case the job's completion time is scheduled earlier than its due date. The results show that this simple extension leads to improvement up to 3.14% for the newly developed SPT+WINQ+Slack rule and up to 3.31% for the SPT+Slack rule.

All objectives Inspired by the critical remarks of Holthaus and Rajendran (1997) and many other authors, who stimulate the development of priority rules that perform relatively well on a set of objective functions rather than on a single objective, the performance of the priority rules on multiple objectives has been evaluated in the following way. While the rankings of the priority rules contain relevant information about the performance on a particular objective function, they do not hold information about the relative deviations between the different priority rules. Therefore, we have calculated the performance of each

For Peer Review Only

priority rule per objective function as a percentage equal to $1 - \%Dev$ with $\%Dev$ given in equation (5). In doing so, the best performing priority rule on a particular objective function gets a value 1 (or 100%) and the others show a lower performance, which clearly expresses the relative deviation from the best performing rule. These values are summarized in the body of Table 3, which shows the five best performing priority rules on the five objective functions under study. Next, the column 'Avg.' contains the average of these percentages and expresses the performance of the priority rules for the five objective functions. Finally, the column 'StDev' displays the standard deviation and expresses the variability of the priority rule's performance along the five objective functions.

Table 3: Top 5 of the priority rules that perform best on the five objective functions ($1 - \%Dev$)

Priority rule	Z_1	Z_2	Z_3	Z_4	Z_5	Avg.	StDev.
2PT+LWKR+FDD	100.00	89.87	98.54	100.00	87.05	95.09	6.16
2PT+LWKR+Slack	99.60	88.90	97.26	99.21	85.00	93.99	6.63
SPT+LWKR+Slack	99.25	88.23	98.40	97.05	84.70	93.53	6.61
2PT+LWKR+EDD !	99.27	86.89	99.49	95.79	75.07	91.30	10.41
SPT+WINQ+LWKR	99.28	86.49	98.81	94.25	71.86	90.14	11.44

The table clearly shows that the five best performing priority rules are all newly developed priority rules that mainly consist of combinations of well-known rules from literature. The 2PT+LWKR+FDD priority rule shows the best performance and contains both information about the processing times of the jobs (flow time-related objectives) as well as due date information (tardiness-related objectives). Moreover, since two of the top five rules contain the Slack value, it confirms the statement made earlier that it is often beneficial to incorporate negative and/or positive slack values (Slack) instead of only focusing on non-positive slack values (SL).

3.3 Large instances

In literature, there is a no general agreement on the machine and job size thresholds of a job shop that determines the complexity of a job shop instance. Holthaus and Rajendran (1997) state that variations in job shop sizes do not significantly affect the relative performance of priority rules and that a job shop with six machines is adequate enough to represent the complex structure of a job shop. Jain and Meeran (1999), however, argue that a job

For Peer Review Only

shop larger than 15x15 can be considered as complex and that its complexity increases as the number of jobs relative to the number of machines increases (i.e. $n \leq 2.5m$). These, and other statements in literature have led to a computational experiment that compares the relative performance of the priority rules of the previous sections with their relative performance on larger data set instances. Consequently, this section tests whether larger sized instances lead to similar or completely different results than the findings of Section 3.2. To that purpose, the 80 job shop instances of Taillard (1993) are used in the computational experiments of which the results are given in Figure 1. The left column of Figure 1 shows the robustness on the relative ranking of the performance quality of the priority rules for the five objective function classes (one for each row) for the large-sized job shop data compared to the smaller instances of Section 3.2. The X-axis of the table shows the different priority rules ranked in decreasing order of performance (based on Table 2). The solid line shows the uniform increasing ranking of the different priority rules for the 82 job shop instances used in Section 3.2. The dotted line shows the fluctuating ranking of these priority rules when used to solve the larger instances. Consequently, the closer these two lines lie to each other, the more robust the priority rules are for size of the problem instances.

The results displayed in the 5 graphs of Figure 1 can be summarized as follows: The graphs for the *mean flow time* and *mean tardiness* objective functions show that the ranking of the priority rules stays more or less the same for larger job shop instances and that the best performing rule (2PT+LWKR+FDD) still outperform all other priority rules. The results for the *proportion of tardy jobs* objective function show that the rankings are somewhat different than for the smaller job shop instances and that the 2PT+LWKR priority rule now performs best. The *makespan* and *maximum tardiness* objective function results, however, show large differences in rankings. Results on the makespan objective show large fluctuations and reveal that the (simple) MWKR, CR and MOPNR priority rules, originally ranked on number 5, 6 and 21, are now the three best performing rules. The maximum tardiness graph also shows large differences, but the four best performing priority rules of the smaller job shop instances remain the best on larger job shop instances.

3.4 Flexible job shop problem

The flexible job shop problem is an extension of the traditional job shop problem and allows an operation to be processed by a pre-defined set of machines. Although the problem has not been investigated as widely as the classical job shop problem, quite a number of solution

For Peer Review Only

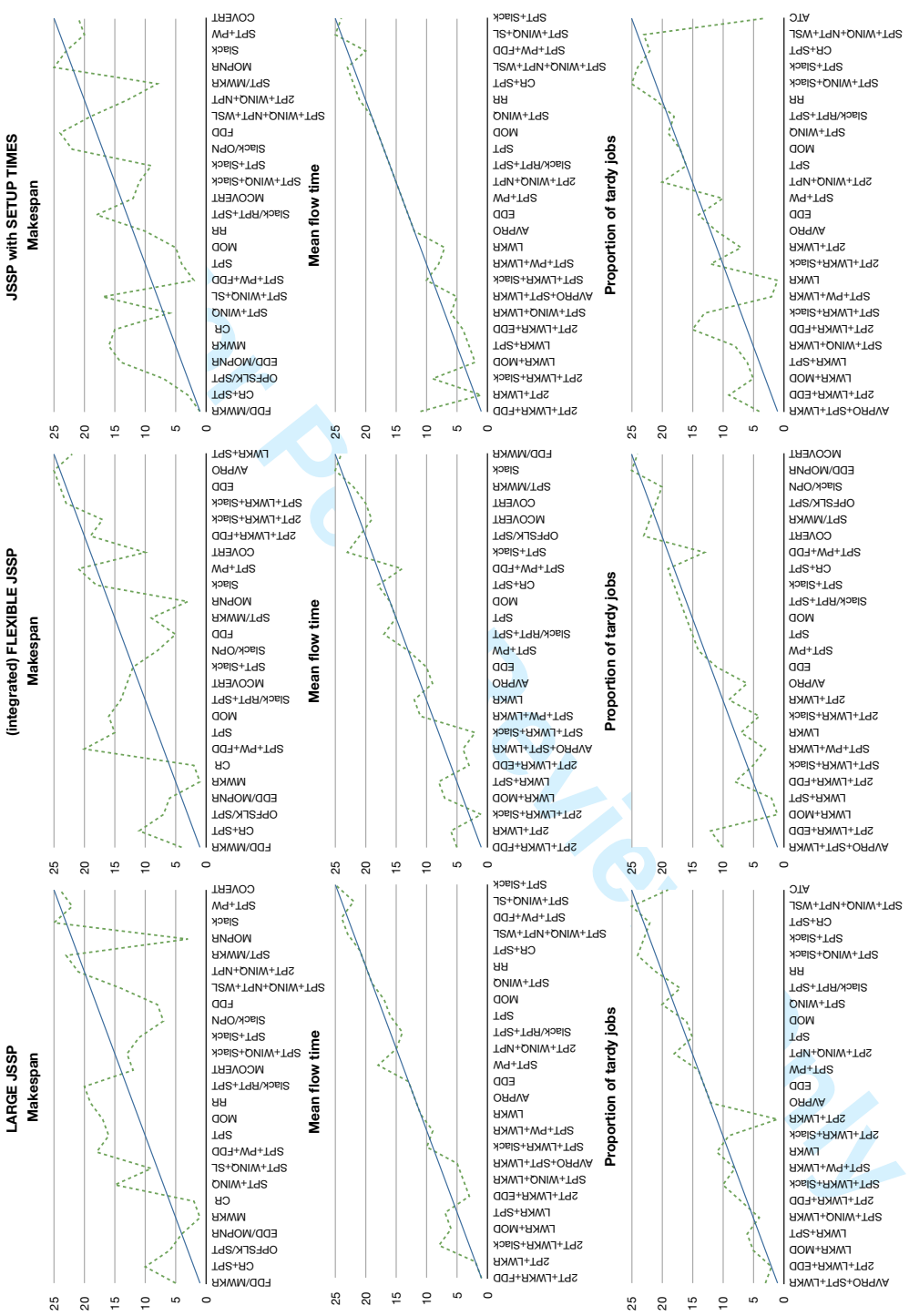

Figure 1: Comparison of priority rule rankings for the traditional job shop problem and three extensions (left: large JSSP; middle: (integrated) flexible JSSP; right: JSSP with setup times)

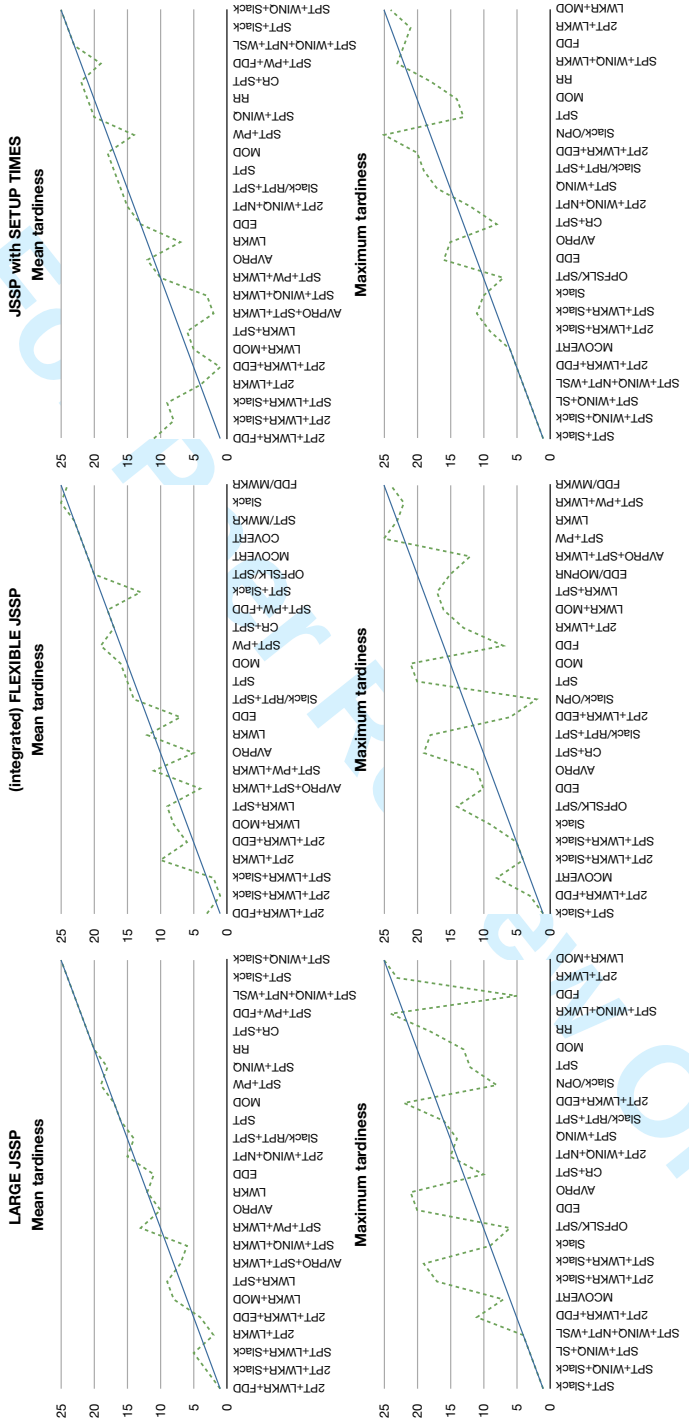

Figure 1: Comparison of priority rule rankings for the traditional job shop problem and three extensions (continued)

For Review Only

approaches are available in literature. Brandimarte (1993) and Dauzère-Pérès and Paulli (1997) classify the solution procedures to solve the flexible job shop scheduling problem into two categories: a set of hierarchical and a set of integrated solution procedures. While the hierarchical solution procedures split the problem in two separate sub-problems, i.e. first a machine assignment problem followed by a job scheduling problem, the integrated methods make no distinction while solving the problem. In the integrated approach, the eligible job operations are assigned to the first available machine. Due to the inherent complexity of the problem, most solution procedures put a clear focus on meta-heuristic search optimization, and hence, little or no research is available on the use of (static and/or dynamic) priority rules to solve the flexible job shop problem, mainly done by Kim (1990), Chen and Luh (2003), Subramaniam et al. (2000b), Ho and Tay (2005) and Tay and Ho (2008).

Table 4 displays comparative results between the hierarchical and integrated approaches tested on the most flexible *vdata* of Hurink et al. (1994) as mentioned in Section 3.1. The results obtained by using the hierarchical solution approach use the "Lowest Average Process Time (LAP)" rule of Subramaniam et al. (2000b) and the "Shortest Queue (SQ)" and the "Least Work in Queue (LW)" rules of Chen and Luh (2003) during the machine assignment step. The values in the columns 'Hierarchical' and 'Integrated' are the average objective function values over all instances of the *vdata*-set taken over all priority rules for the FJSP under the corresponding objective function. The column '%Difference' gives the relative deviation of the average objective function values obtained by the best hierarchical ($\overline{Z}^H$), i.e. the LW-approach, and the integrated ($\overline{Z}^I$) approach according to the formula:

$$\%\text{Difference} = \frac{\overline{Z}^H - \overline{Z}^I}{\overline{Z}^I} \tag{6}$$

The table shows that the LW-rule is the best machine assignment rule in the hierarchical method and that the integrated method always outperforms the hierarchical methods for all objective functions, as shown by the positive percentage difference. It should be noted that this percentage difference is larger for the tardiness-related priority rules in comparison to the flow time-related priority rules, which can probably be explained by the lack of due date information in the machine assignment rules. A drawback of the integrated methods, however, could be that some priority rules cannot be used since they require a single machine per operation. For this reason, the priority rule list on the x-axis of Figure 1 is slightly different for the flexible job shop column (column 2) than for the other columns. Indeed, priority rules that cannot be used to solve the (integrated) flexible variant of the

For Peer Review Only

problem have been taken out from the list, and hence, the 25 best performing priority rules consist of the 25 best rules of Table 2 that can be used for both the traditional and the (integrated) flexible job shop problem.

Table 4: Comparative results between the hierarchical and the integrated method(s)

	Hierarchical			**Integrated**	**%Difference**
Objective	**LW**	**SQ**	**LAP**	-	$(Z^H - Z^I)/Z^I$
Makespan	1,584	1,698	3,150	1,526	3.80%
Mean flow time	1,252	1,270	1,889	1,074	16.58%
Proportion of tardy jobs	0.28	0.29	0.64	0.19	48.81%
Mean tardiness	103	114	580	68	51.02%
Maximum tardiness	276	320	1444	237	16.65%

Waikar et al. (1995) assumed that the performance of priority rules for the traditional job shop problem would not be affected by the addition of alternative machines for each operation, which is the subject of the computational tests of this section. To that purpose, the *vdata*-set proposed by Hurink et al. (1994), containing the most flexible instances, are used. Computational experiments on the other sets revealed that the degree of flexibility does not have an influence on the relative performance of the priority rules. The middle column of Figure 1 shows the comparative priority rule rankings used to solve the traditional as well as the flexible job shop problem for each of the five objective functions. The graphs show that the relative performance of the priority rules to solve the problems under the *mean flow time*, *mean tardiness* and *proportion of tardy jobs* objectives does not differ much from their performance on the classical job shop scheduling problem, which partly confirms the conjecture of Waikar et al. (1995). However, the best performing rule for the traditional job shop problem is not always the best one for its flexible variant. The two graphs for the *makespan* and *maximum tardiness* objective functions (again) show more fluctuations and illustrate that the Slack/OPN rule with its low quality ranking for the traditional job shop problem has an excellent performance for the flexible job shop problem.

3.5 Setup times

In this section we extend the classical job shop scheduling problem with sequence-dependent setup times, as also studied by Cheung and Zhou (2001) and Artigues et al. (2005). We

For Peer Review Only

examine whether the best performing priority rules are still successful on the extended problem or if some more effective rules can be developed.

The right column of Figure 1 shows the comparative priority rule rankings used to solve the job shop problem with and without sequence-dependent setup times for each objective function. The graphs for the *makespan* and *maximum tardiness* objectives again show large fluctuations, although their best performing priority rules, FDD/MWKR and SPT+Slack, respectively, are still the best performing rules when sequence-dependent setup times are incorporated. For the three other objectives, there is a relatively high variation in rankings for the top 10 best performing rules, but this variation declines over the remaining priority rules. An exception is the ATC rule for the *proportion of tardy jobs* objective, which performs remarkably well for the problem with setup times.

In order to find priority rules that outperform the currently best-known rules, each rule has been extended with the SS (shortest setup time) rule and the best found results have been summarized in Table 5. The column 'Priority rule' gives the best performing priority rule for each of the objective functions extended with the SS-rule. The column with label 'Result' shows the average objective function values obtained by this rule taken over all data instances and the column labeled with '%Difference' shows the relative improvement of the best performing rules mentioned in column 'priority rule' with and without the SS rule extension. The column '% Improvement' gives the average improvement of these five best performing priority rules relative to all other rules of table 2 that have been extended with the SS rule. The table shows that the best-known priority rule, as found in Section 3.2, is still the best one when extending it with the SS rule, except for the *proportion of tardy jobs* objective. Consequently, although the graphs in Figure 1 show that the best performing rules for the job shop problem are not always the best when sequence-dependent setup times are incorporated, the extension of these rules with the SS rule put them back on the first place for four of the five objective classes. The table shows that the extension with the SS rule leads to further improvements up to 14%. Only the job shop problem with the *proportion of tardy jobs* objective function can be better solved with the LWKR rule without incorporation of the SS rule. Obviously, no percentage difference value is reported.

4 Dynamic job arrivals

In the previous sections, it was assumed that jobs arrive at the beginning of the scheduling period and hence all information is available at the start of the scheduling period.

For Peer Review Only

Table 5: Comparative results for the job shop problem with sequence-dependent setup times

Objective	Priority rule	Result	%Difference	%Improvement
Makespan	FDD/MWKR+SS	2,305	11.78%	11.44%
Mean flow time	2PT+LWKR+FDD+SS	1,629	5.36%	6.10%
Proportion of tardy jobs	LWKR	0,8215	n.a.	10.38%
Mean tardiness	2PT+LWKR+FDD+SS	711	8.89%	14.10%
Maximum tardiness	SPT+Slack+SS	1,363	14.14%	12.61%

For Peer Review Only

This section extends this study and assumes a job shop scheduling model where jobs arrive dynamically. In doing so, the priority rules can be used as dispatching rules, which dynamically assigns jobs to the partial schedule upon arrival. In most research papers, arrival patterns are generated from an exponential (Waikar et al., 1995; Holthaus and Rajendran, 1997; Rajendran and Holthaus, 1999; Jayamohan and Rajendran, 2000; Dominic et al., 2004) or a Poisson distribution (Vepsalainen and Morton, 1987; Barman, 1997; Taner et al., 2003), where the distribution parameters are linked to the machine utilization levels. However, in their paper, Waikar et al. (1995) argue that the job shop performance is relatively unaffected by the type of arrival time distribution, but instead, the average number of jobs arriving per time unit is important since it affects the shop load level. Therefore, the experiment on the traditional job shop problem (cf. Section 3.2) has been repeated for the 5 objective functions under 5 arrival pattern versions defined by the expected number of jobs arriving per time unit. Figure 2 shows 5 graphs for the five objective functions where the arrival patters $\mathrm{AP}_i, i = 1, ..., 5$ have been defined as follows:

- AP_1 = STA: A static arrival pattern where all jobs are available at the beginning of the scheduling period. This is the arrival pattern scenario used in the previous sections and will be used as benchmark scenario for comparison purpose of the other dynamic arrival patterns.

- AP_2 = EQL: The job arrivals are equally spread over a time interval to assure that the jobs arrive at regular time intervals throughout the time horizon. In the computational experiments, the interval has been set to the minimum of the sum of the processing times of all operations for each job, i.e. $\min_{\forall i} \sum_{\forall j} p_{ij}$.

- AP_3 = INC: The jobs arrive in an increasing rate along the predefined time window to assure that little to no jobs arrive at the start of the scheduling horizon, but an

increasing amount of jobs arrive as time goes by.

- AP_4 = DEC: The jobs arrive in an decreasing rate along the predefined time window to model an opposite behavior compared to the previous scenario.
- AP_5 = RAN: The jobs arrive randomly along the predefined time window.

The x-axis of Figure 2 shows the 5 different arrival pattern scenarios. The y-axis shows the relative ranking of the top 6 best priority rules for the static job shop scheduling problem (AP_1) as given in Table 2 with a 1 the best performing rule, a 2 the second best rule, etc.... The ranking might differ for other arrival patterns, as shown in the figure. In order to facilitate readability, the priority rules are connected by lines, and hence, the more lines crossing each other, the less robust the relative ranking of the priority rules performance is for different arrival patterns.

The figure shows that the presence of dynamic job arrivals does not always lead to the same relative ranking on the performance quality of the priority rules compared to the static job shop scheduling problem. The job shop scheduling problem with *makespan* and *maximum tardiness* objectives are relatively robust as both graphs show little to no variation for the best performing priority rules over all arrival pattern scenarios. Only the second best performing rule for the makespan objective (CR+SPT) has a larger variation and is therefore considered not to be robust for different arrival patterns. However, the three other objectives show a much lower degree of robustness for various dynamic arrival patterns. The *mean flow time* and *mean tardiness* objectives show some degree of robustness. More precisely, the second best priority rule (2PT+LWKR) for the mean flow time and best performing priority rule (2PT+LWKR+FDD) for the mean tardiness perform best on all other dynamic arrival patterns, while the other priority rules show a higher variability. The problem with the *proportion of tardy jobs* objective cannot be regarded as being robust. Although the five best performing priority rules on the static job shop scheduling problem still perform reasonable well over different arrival patterns, there is a higher degree of variation between the arrival patterns.

It is worth noting that the two objective functions where a relatively high (*dynamic*) robustness has been found in Figure 2 (makespan and maximum tardiness) are those where the fluctuations are rather high in Figure 1, i.e. their *static* robustness is rather low. A similar but opposite effect is found for the three other objectives: a rather high degree of static robustness is accompanied by a rather low degree of dynamic robustness.

For Peer Review Only

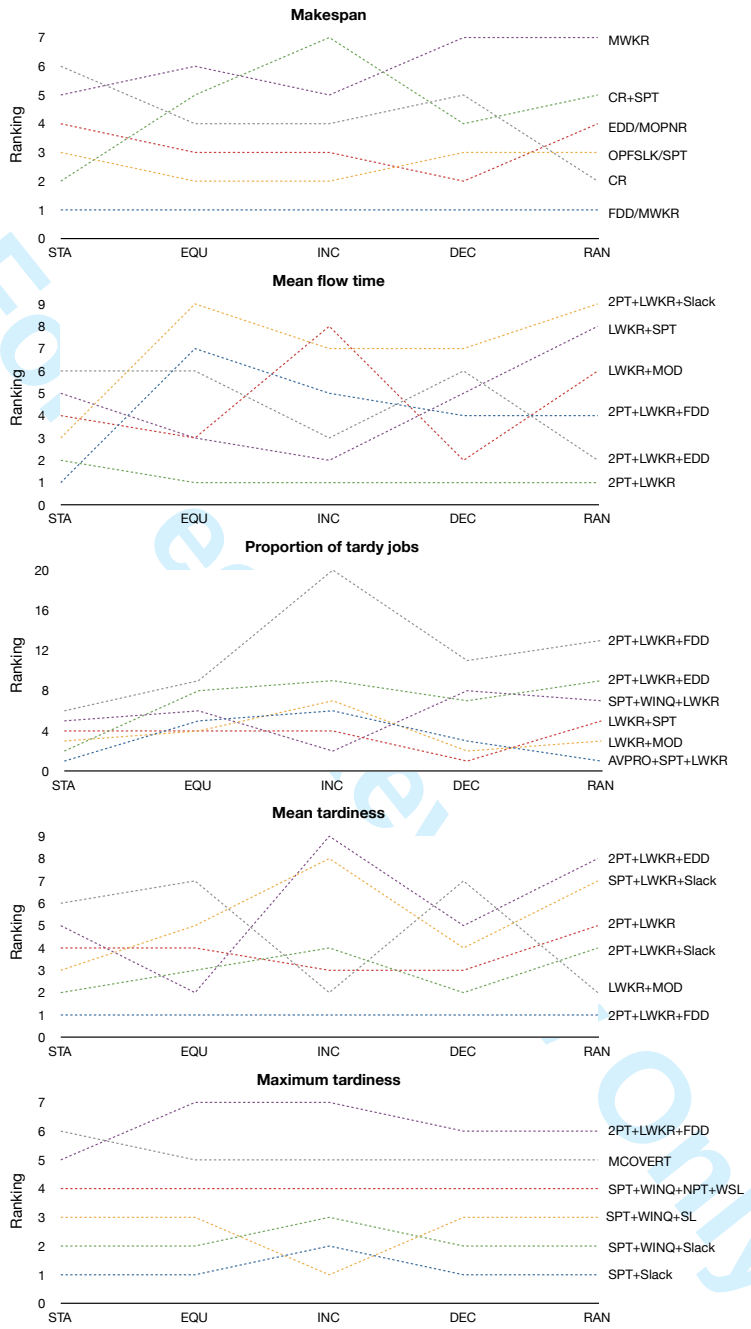

Figure 2: Robustness of the top 6 priority rules for different arrival patterns

5 Conclusions

This paper discussed an extensive comparison of different priority rules to solve the job shop scheduling problem under five objective functions, which can be classified in two main categories. Two flow time-related objectives, the makespan and the mean flow time, and three tardiness-related objectives, the proportion of tardy jobs, the mean tardiness and the maximum tardiness, have been used during various computational experiments.

A first computational experiment has been set up to determine the relative ranking of the best performing priority rules. Some rules are taken from literature and are known to perform well on the particular objective under study, while others are known to perform well on other objectives than the one under study. Based on this knowledge, priority rules have been combined to construct new hybrid priority rules that outperform the currently best-know rules from literature. The experiments have been extended with a second set of experiments to other job shop environments. More precisely, the robustness on the quality and relative ranking of the priority rules has been tested on larger job shop problem instances, on the introduction of alternative machines per operation as occurs in the flexible job shop as well as on the presence of sequence-dependent setup times. In a final set of computational experiments, the robustness of the relative ranking of the priority rules' performance quality is tested against dynamic versions of the job shop scheduling problem, where not all jobs are assumed to be available at the beginning of the scheduling period.

The results of the three experiments can be briefly summarized as follows. First, it has been shown that combined versions of priority rules from literature perform better on all objective functions in comparison to the single priority rules separately. Secondly, it has been shown that robustness of the performance quality for the different priority rules, when extending the traditional job shop problem, depends on the objective function. Overall, it has been shown that the mean flow time, mean tardiness and proportion of tardy jobs are more robust objective functions than the makespan and maximum tardiness objectives. Moreover, experiments have also confirmed that an integrated optimization approach outperforms the hierarchical approach when solving the flexible job shop problem. The incorporation of sequence-dependent setup times has verified that the best performing rules for the traditional job shop problem still perform the best when they are extended with the SS rule on four of the five objective classes. Finally, the extension of static job arrivals in the job shop problem to dynamic job arrivals have resulted in test experiments where the robustness shows an opposite behavior compared to the static experiments. Indeed,

For Peer Review Only

while the three objective, mean flow time, mean tardiness and proportion of tardy jobs, have shown a relatively high degree of robustness during the static experiments, they fail to report a high robustness degree on the dynamic job arrival experiments. The makespan and maximum tardiness objective, on the contrary, show an opposite behavior.

6 Acknowledgements

We acknowledge the support given by the 'Bijzonder Onderzoeks Fonds (BOF), Vlaanderen, Belgium' under contract number BOF07/24j/034.

References

Adams, J., Balas, E., and Zawack, D. (1988). The shifting bottleneck procedure for job shop scheduling. *Management Science*, 34:391–401.

Anderson, E. and Nyirenda, J. (1990). Two new rules to minimize tardiness in a job shop. *International Journal of Production Research*, 28:2277–2292.

Applegate, D. and Cook, W. (1991). A computational study of the job-shop scheduling problem. *ORSA Jounal on Computing*, 3:149–156.

Artigues, C., Lopez, P., and Ayache, P. (2005). Schedule generation schemes for the job-shop problem with sequence-dependent setup times: Dominance properties and computational analysis. *Annals of Operations Research*, 138:21–52.

Baker, K. (1984). Sequencing rules and due-date assignments in a job shop. *Management Science*, 30:1093–1104.

Baker, K. and Bertrand, J. (1981). An investigation of due-date assignment rules with constrained tightness. *Journal of Operations Management*, 1:109–120.

Barman, S. (1997). Simple priority rule combinations: an approach to improve both flow time and tardiness. *International Journal of Production Research*, 35:2857–2870.

Blackstone, JR, J., Phillips, D., and Hogg, G. (1982). A state-of-the-art survey of dispatching rules for manufacturing job shop operations. *International Journal of Production Research*, 20:27–45.

For Peer Review Only

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research*, 41:157–183.

Carlier, J. (1978). Ordonnancements a contraintes disjonctives. *Operations Research*, 12:331–351.

Chen, H. and Luh, P. (2003). An alternative framework to Lagrangian relaxation approach for job shop scheduling. *European Journal of Operational Research*, 149:499–512.

Cheung, W. and Zhou, H. (2001). Using genetic algorithms and heuristics for job shop scheduling with sequence-dependent setup times. *Annals of Operations Research*, 107:65–81.

Chiang, T. and Fu, L. (2007). Using dispatching rules for job shop scheduling with due-date based objectives. *International Journal of Production Research*, 45:3245–3262.

Chiang, T. and Fu, L. (2008). A rule-centric memetic algorithm to minimize the number of tardy jobs in the job shop. *International Journal of Production Research*, 46:6913–6931.

Dauzère-Pérès, S. and Paulli, J. (1997). An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search. *Annals of Operations Research*, 70:281–306.

Dominic, P., Kaliyamoorthy, S., and Saravana Kumar, M. (2004). Efficient dispatching rules for dynamic job shop scheduling. *International Journal of Advanced Manufacturing Technology*, 24:70–75.

Fisher, H. and Thompson, G. (1963). *Industrial Scheduling*, chapter Probabilistic learning combinations of local job-shop scheduling rules, pages 225–251. Prentice-Hall, Inc.

Garey, M. R., Johnson, D. S., and Seth, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1:117–129.

Graham, R., Lawler, E., Lenstra, J., and Rinnooy Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, 5:287–326.

Ho, N. and Tay, J. (2005). Evolving dispatching rules for solving the flexible job-shop problem. *Proceedings of the IEEE congress on evolutionary computing, CEC*, pages 2848–2855.

For Peer Review Only

Holthaus, O. and Rajendran, C. (1997). Efficient dispatching rules for scheduling in a job shop. *International Journal of Production Economics*, 48:87–105.

Holthaus, O. and Rajendran, C. (2000). Efficient jobshop dispatching rules: Further developments. *Production Planning & Control*, 11:171–178.

Hurink, J., Jurisch, B., and Thole, M. (1994). Tabu search for the job-shop scheduling problem with multi-purpose machines. *OR Spektrum*, 15:205–215.

Jain, A. and Meeran, S. (1999). Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research*, 113:390–434.

Jayamohan, M. and Rajendran, C. (2000). New dispatching rules for shop scheduling: A step forward/backward. *International Journal of Production Research*, 38:563–586.

Jayamohan, M. and Rajendran, C. (2004). Development and analysis of cost-based dispatching rules for job shop scheduling. *European Journal of Operational Research*, 157:307–321.

Jensen, J. B., Philipoom, P. R., and Malhotra, M. K. (1995). Evaluation of scheduling rules with commensurate customer priorities in job shops. *Journal of Operations Management*, 13:213–228.

Kim, Y.-D. (1990). A comparison of dispatching rules for job shops with multiple identical jobs and alternative routeings. *International Journal of Production Research*, 28:953–962.

Lawrence, S. (1984). Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques. *Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA*.

Mizrak, P. and Bayhan, G. M. (2006). Comparative study of dispatching rules in a real-life job shop environment. *Applied Artificial Intelligence*, 20:585–607.

Muhlemann, A., Lockett, A., and Farn, C.-K. (1982). Job shop scheduling heuristics and frequency of scheduling. *International Journal of Production Research*, 20:227–241.

Panwalkar, S. and Iskander, W. (1977). A survey of scheduling rules. *Operations Research*, 25:45–61.

For Peer Review Only

Pinedo, M. (2002). *Scheduling - Theory, Algorithms, and Systems (Second Edition)*. Prentice-Hall, Upper Saddle River, New Jersey.

Rajendran, C. and Holthaus, O. (1999). A comparative study of dispatching rules in dynamic flowshops and jobshops. *European Journal of Operational Research*, 116:156–170.

Russell, R., Dar-El, E., and Taylor, B. (1987). A comparative analysis of the covert job sequencing rule using various shop performance measures. *International Journal of Production Research*, 25:1523–1540.

Storer, R., Wu, S., and Vaccari, R. (1992). New search spaces for sequencing problems with application to job shop scheduling. *Management Science*, 38:1495–1509.

Subramaniam, V., Lee, G., Hong, G., Wong, Y., and Ramesh, T. (2000a). Dynamic selection of dispatching rules for job shop scheduling. *Production Planning & Control*, 11:73–81.

Subramaniam, V., Lee, G., Ramesh, T., Hong, G., and Wong, Y. (2000b). Machine selection rules in a dynamic job shop. *International Journal of Advanced Manufacturing Technology*, 16:902–908.

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research*, 64:278–285.

Taner, M., Hodgson, T., King, R., and Thoney, K. (2003). Satisfying due-dates in an job shop with sequence-dependent family setups. *International Journal of Production Research*, 41:4153–4169.

Tay, J. and Ho, N. (2008). Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems. *Computers & Industrial Engineering*, 54:453–473.

Vepsalainen, A. and Morton, T. (1987). Priority rules for job shops with weighted tardiness costs. *Management Science*, 33:1035–1047.

Waikar, A., Sarker, B., and Lal, A. (1995). A comparative study of some priority dispatching rules under different shop loads. *Production Planning & Control*, 6:301–310.

For Peer Review Only

Yamada, T. and Nakano, R. (1992). A genetic algorithm applicable to large-scale job-shop problems. *Proceedings of the Second International Conference on Parallel Problem Solving from Nature*, pages 281–290.

Appendix A

In this appendix, we give more information on the priority rules used in the paper (cf. table 2). In Section A.1, the abbreviations used are outlined. Section A.2 list the single priority rules together with their mathematical formulation. For this overview, we based ourselves on the papers of, among others, Panwalkar and Iskander (1977), Blackstone et al. (1982), Baker (1984), Russell et al. (1987), Vepsalainen and Morton (1987), Anderson and Nyirenda (1990), Waikar et al. (1995), Holthaus and Rajendran (1997), Rajendran and Holthaus (1999), Jayamohan and Rajendran (2000), Holthaus and Rajendran (2000), Dominic et al. (2004), Mizrak and Bayhan (2006) and Chiang and Fu (2007). Finally, in Section A.3, more information is given on the combination of the single rules.

A.1 Abbreviations

In order to explain the abbreviations used, an example of a disjunctive graph for a JSSP with three machines and three jobs is given in figure 3. In this example, each job j (j = 1, 2 or 3) has three operations to be performed on one of the machines i (i = 1, 2 or 3), which is denoted as O_{ij}. The index q is used to refer to the qth operation of a job. For example, operation O_{22} is the first operation of job 2 on machine 2, indicated by $j = 2$, $i = 2$ and $q = 1$.

In section A.2, we calculate the priority value (Z_j) for each job j at time τ. We assume that this corresponds with the current operation q to be performed on machine i. The

For Peer Review Only

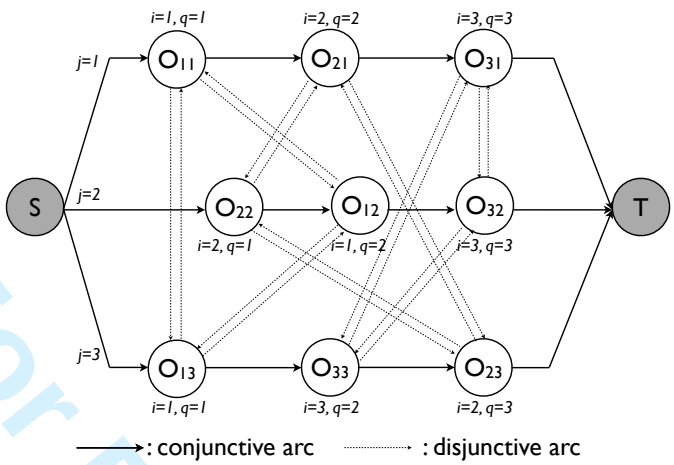

Figure 3: Disjunctive graph representation of a 3×3 job shop

abbreviations used can be summarized along the following lines:

j = job index;

i = machine index;

q = operation index;

$O(j)$ = total number of operations of job j;

Z_j = priority index of job j at time τ;

τ = current time at which the dispatching decision is to be made;

p'_{qj} = processing time of the qth operation of job j, which corresponds to a certain p_{ij}-value according to the machine i on which operation q is to be processed;

$\overline{P}$ = average processing time of all jobs waiting in line;

$P_{q+1,j}$ = total processing time of all jobs in the queue of the next operation $(q+1)$ of job j;

C_{qj} = completion time of the qth operation of job j;

d_j = due date of job j;

c = due date allowance factor;

r_j = release (arrival) time of job j on the shop floor;

r_{ij} = release (arrival) time of job j at the current machine i;

s_{ij} = setup time of operation O_{ij} of job j on machine i;

η_i = utilization level of machine i;

k = exponential look-ahead parameter;

W_{qj} = expected waiting time of the qth operation of job j;

W_j = total expected waiting time of job j for all unscheduled operations $(> q) = \sum_{t=q+1}^{O(j)} W_{tj}$;

$W_{q+1,j}$ = expected waiting time of job j at the machine of its next operation $(q+1)$;

For Peer Review Only

A.2 Single priority rules

1. 2PT = *Double Processing Time*

$$\min \quad Z_j = 2p'_{qj}$$

2. ATC = *Apparent Tardiness Cost*

$$\max \quad Z_j = (\frac{1}{p'_{qj}}) \left\{ \exp \left\{ -\max \left(\frac{d_j - \sum_{t=q+1}^{O(j)} (W_{tj} + p'_{tj}) - \tau - p'_{qj}}{(k\,\overline{P})}, 0 \right) \right\} \right\}$$

3. AVPRO = *Average Processing time per Operation*

$$\min \quad Z_j = \sum_{t=1}^{O(j)} \frac{p'_{tj}}{O(j)}$$

4. COVERT = *Cost Over Time*

$$\max \quad Z_j = \begin{cases} \frac{1}{p'_{qj}} \frac{(W_j - s_j)}{W_j} & \text{if} \quad 0 \leq s_j < W_j \\ 0 \quad \text{if} \quad s_j \geq W_j \\ \frac{1}{p'_{qj}} \quad \text{if} \quad s_j < 0 \end{cases}$$

5. CR = *Critical Ratio*

$$\min \quad Z_j = \frac{d_j - \tau}{\sum_{t=q}^{O(j)} p'_{tj}}$$

6. EDD = *Earliest Due Date*

$$\min \quad Z_j = d_j$$

7. FDD = *Flow Due Date*

$$\min \quad Z_j = r_j + \sum_{t=1}^{q} p'_{tj}$$

8. FIFO = *First in First out*

$$\min \quad Z_j = r_{ij}$$

For Peer Review Only

9. LIFO = *Last in First out*

$$\max \quad Z_j = r_{ij}$$

10. LPT = *Longest Processing Time*

$$\max \quad Z_j = p'_{qj}$$

11. LWKR = *Least Work Remaining*

$$\min \quad Z_j = \sum_{t=q}^{O(j)} p'_{tj}$$

12. MCOVERT = *Modified Cost Over Time*

$$\max \quad Z_j = \begin{cases} \frac{1}{p'_{qj}} \frac{(W_j - s_j)}{W_j} & \text{if} \quad 0 \leq s_j < W_j \\ 0 \quad \text{if} \quad s_j \geq W_j \\ \frac{-s_j}{p'_{qj}} \quad \text{if} \quad s_j < 0 \end{cases}$$

13. MOD = *Modified Operational Due date*

$$\min \quad Z_j = \max(r_j + c \sum_{t=1}^{q} p'_{tj}; \tau + p'_{qj})$$

14. MOPNR = *Most Operations Remaining*

$$\max \quad Z_j = O(j) - q + 1$$

15. MWKR = *Most Work Remaining*

$$\max \quad Z_j = \sum_{t=q}^{O(j)} p'_{tj}$$

16. NPT = *Next Processing Time*

$$\min \quad Z_j = p'_{q+1,j}$$

For Peer Review Only

17. OPFSLK/PT = *Operational Flow Slack per Processing Time*

$$\max \quad Z_j = \frac{\max(\tau + p'_{qj} - (r_j + \sum_{t=1}^{q} p'_{tj}); 0)}{p'_{qj}}$$

18. PW = *Process Waiting time*

$$\min \quad Z_j = \tau - C_{q-1,j}$$

19. RR = *Raghu and Rajendran*

$$\min \quad Z_j = \frac{(s_j \ \exp(-\eta_i) \ p'_{qj})}{\sum_{t=q}^{O(j)} p'_{tj}} + \exp(\eta_i) p'_{qj} + W_{q+1,j}$$

20. SL = *negative slack*

$$\min \quad Z_j = \min(s_j; 0)$$

21. Slack = *slack*

$$\min \quad Z_j = s_j$$

22. Slack/OPN = *Slack per Remaining Operations*

$$\min \quad Z_j = \begin{cases} \frac{s_j}{O(j)-q+1} & \text{if} \quad s_j \geq 0 \\ s_j \times (O(j) - q + 1) & \text{if} \quad s_j < 0 \end{cases}$$

23. Slack/RPT = *Slack per Remaining Processing Time*

$$\min \quad Z_j = \frac{s_j}{\sum_{t=q}^{O(j)} p'_{tj}}$$

24. SPT = *Shortest Processing Time*

$$\min \quad Z_j = p'_{qj}$$

25. SPT/MWKR = *Shortest Processing Time per Work Remaining*

$$\min \quad Z_j = \frac{p'_{qj}}{\sum_{t=q}^{O(j)} p'_{tj}}$$

For Peer Review Only

26. SS = *Shortest Setup time*

$$\min \quad Z_j = s_{ij}$$

27. WINQ = *Work In Next Queue*

$$\min \quad Z_j = P_{q+1,j}$$

28. WSL = *Waiting Slack*

$$\min \quad Z_j = \min(s'_j; 0)$$

$$\text{with} \quad s'_j = d_j - (\tau + P_{q+1,j} + \sum_{t=q}^{O(j)} p'_{tj})$$

A.3 Combined priority rules

The combined priority rules (with "+"-sign) are obtained by summing up the different priority values. For example, the priority value for the SPT +WINQ+ SL-rule (Holthaus and Rajendran, 1997) is an additive combination of the processing time of the job, the work content of all jobs in the queue of the next operation of the job and its non-positive slack value and is obtained according to following formula:

$$\min \quad Z_j = p'_{qj} + P_{q+1,j} + \min(s_j; 0)$$

By combining priority rules, several job characteristics are considered simultaneously when developing priorities. As such, the overall shop performance can be improved. Some other combined rules discussed in literature and used in table 2 are SPT+WINQ (Holthaus and Rajendran, 1997), 2PT+WINQ+NPT, SPT+WINQ+NPT+WSL (Holthaus and Rajendran, 2000), CR+SPT, Slack/RPT+ SPT (Anderson and Nyirenda, 1990) LWKR+SPT (Dominic et al., 2004), SPT+PW and SPT+PW+FDD (Jayamohan and Rajendran, 2000). Our newly develop priority rules follow the same combination principle.

For Peer Review Only