Flow Time vs Work In Progress: Benchmark of Algorithms for the Online-Time Job Shop Scheduling Problem

Erwann ESTEVE[1*], Laurie BOVEROUX[2] and Quentin LOUVEAUX[2]

[1*]École des Ponts ParisTech, 8 Av. Blaise Pascal, Champs-sur-Marne, 77420, France.
[2]Institut Montefiore, Université de Liège, Allée de la Découverte 10, Liège, 4000, Belgique.

*Corresponding author(s). E-mail(s): erwannesteve@gmail.com;
Contributing authors: Laurie.Boveroux@uliege.be;
Q.Louveaux@uliege.be;

Abstract

The online-time Job Shop Scheduling Problem provides a useful model for real-time scheduling in industrial settings. In this problem, jobs with precedence constraints arrive according to a Poisson process, and resolution algorithms must generate a schedule at any given time. Two of the most insightful efficiency criteria to evaluate a schedule in an industrial context are flow time and work in progress, respectively the duration from a job's arrival to its completion and the number of jobs being processed at any time. This paper aims to compare the performance for these two criteria of solution algorithms, notably rescheduling algorithms and priority dispatching rules. In order to enhance rescheduling policies performances, a new objective function, termed the "fee-reward objective function", is implemented in order to find a balance between both flow time and work in progress. The studied algorithms using the fee-reward objective function show significantly better performances on the work in progress criterion and achieve comparable or even superior results on the flow time criterion relative to the best priority dispatching rules and rescheduling algorithms using the makespan objective function.

Keywords: Job Shop Scheduling, Online, Flow time, Work In Progress

1 Introduction

In manufacturing and service industries, scheduling is an essential tool for optimizing productivity and enhancing operational efficiency. In practice, industries often receive a continuous stream of tasks to schedule. Depending on the nature of the tasks, they may be divisible into successive operations, each requiring the use of resources such as machines. In the latter case, the scheduling problem encountered is similar to the Job Shop Scheduling Problem (JSSP): the tasks are called jobs, whose operations require the use of a particular machine. This stream of tasks can be modeled as an online-time version of JSSP.

The classical offline JSSP involves solving a fully known instance. For the online-time JSSP, each job is associated with an arrival date. The solution involves providing a schedule for the current system at any point in time, without information about future arrivals. Various strategies exist to solve an online-time JSSP: it may involve a simple priority dispatching rule such as First In First Out (FIFO) or be based on optimal scheduling for the currently known instance, according to an objective function.

This paper investigates the performance of various strategies for solving the JSSP with a focus on two secondary performance criteria: flow time and outstanding time. Flow time is defined as the duration from a job's arrival to its completion, providing insights into the system's efficiency in meeting customer demands. Outstanding time measures the period from the initiation of the job's first operation to the completion of its last, which serves as an indicator of the amount of fixed assets in use. This indicator is closely related to the work in progress (WIP) of the system, which is at any point in time the number of jobs which have been started but are not yet completed. This dual perspective on performance highlights the trade-offs in scheduling decisions, where optimizing for one criterion may affect the other.

In light of these challenges, we propose a novel fee-reward objective function tailored to maximize profits while accounting for both flow and outstanding times. This study aims to evaluate the impact of various scheduling heuristics on the established performance criteria. Ultimately, our findings aim to provide practitioners and researchers with actionable insights into the effectiveness of these heuristics, guiding the selection of appropriate scheduling policies based on specific operational priorities.

The remainder of this paper is organized as follows. Section 2 introduces the framework of study and defines technical terms. Section 3 analyses the results of the simulations.

2 Framework

2.1 Problem definition

The offline JSSP is defined by a finite set J of n_J jobs and a finite set M of n_M machines. Each job j consists of a set of n_j operations $o_1, o_2, \ldots, o_{n_j}$ which need to be processed in a specific order. An operation requires the use of a machine and is processed for p_o units of time. The machine is unavailable for other operations while an operation is being processed. For our purpose, recirculation is allowed: several operations of the same job may require the same machine. According to the job scheduling

triplet notation (Graham et al. (1979)), the offline JSSP referred to in this article is: $J_n|rcrc|\gamma$, with γ being the objective function to minimize.

Additionally, in the online-time problem, each job has an arrival time a_j. A solution to the online-time version is a schedule of the jobs, but decisions have to be made at any point during the simulation, with information only about the current and past states of the system. Preemption of operations is not allowed, however the schedule of future operations can be changed. Therefore, decisions must be made with uncertainty about future events, and an optimal schedule for the currently known and available jobs and machines can become obsolete due to new arrivals, possibly leading to a globally suboptimal schedule. The job scheduling triplet notation for this article's problem is $J_n|online - time, rcrc|\gamma$.

In this article, a distinction is made between objectives, policies, and strategies in the online-time version of the problem. Objectives are values to be minimized for a given scheduling problem. They are used in exact methods to compute schedule when the instance is fully known (offline JSSP). Policies refer to the approach chosen to tackle the online aspect of the problem. For example, the REPLAN policy determines according to an objective the best schedule at any point in time based on the currently known jobs and machines. Policies can be categorized in two types: proactive policies, which maintain and adjust a schedule when new jobs arrive, and completely reactive policies, which do not require a pre-generated schedule and make decisions at the machine level in real-time.

Completely reactive policies include priority dispatching rules such as FIFO (where each machine selects the waiting job with the earliest arrival time a_j). Proactive policies include REPLAN (see Section 2.5 for more details on the policies considered). For proactive policies, an objective function is necessary to compute the schedule: different objectives result in different schedules and, consequently, different behaviors.

A strategy is defined as a solution algorithm for the online-time JSSP problem, which may either be a completely reactive policy or a combination of a proactive policy and an objective. Thus, FIFO, REPLAN with a makespan objective, and REPLAN with a total completion time objective represent three distinct strategies.

2.2 Literature review

Online scheduling, and particularly the online-time version of JSSP, emerges around the year 2000, following research on online algorithmic in the 90s. The most common criterion seems to be the competitive ratio, defined as the ratio between the score of an algorithm for a criterion and that of the optimal solution of the offline problem, with criteria such as makespan (Kimbrel and Saia (2000)) or flow time (Divakaran and Saks (2011)). Average-case analysis, as in this paper, is uncommon but used in situations where a reasonable approximation of the input distribution is known, such as when job arrivals follow a Poisson distribution (Caceres et al. (1998)).

The two most common criteria remain makespan and flow time. The so-called outstanding time in this article is not directly studied: in some papers, there is confusion with flow time when the usual definition of the latter is not relevant (Felbecker (1980)). However, more papers have studied the waiting time (Ostermeier (2022)), which is the difference between flow time and outstanding time.

The study of online scheduling algorithms in the 2000s mainly focused on priority dispatching rules such as Shortest Remaining Processing Time (SRPT) and FIFO (Pruhs et al. (2004)). These algorithms had the benefit of providing some results about their competitive ratio (Torng and McCullough (2008)). A few papers using modern resolution methods for the offline JSSP have discussed application in an online-time environment (Zhiming and Chunwei (2000)).

2.3 Criteria

This paper focuses on two criteria to evaluate schedules computed by different strategies: flow time and outstanding time. Flow time is defined as the duration between a job's arrival and its completion:

$$FlowTime_j = C_j - a_j,$$

where C_j and a_j are, respectively, the completion time and arrival time of job j. This criterion is widely used in both literature and industrial practice (Ostermeier and Deuse (2024), Albers (2009)). The outstanding time is defined in this article as the duration between a job's start and its completion:

$$OutstandingTime_j = C_j - S_j,$$

where C_j and S_j are the completion time and start time of job j, respectively. Note that in some literature, this criterion is sometimes referred to as flow time when $C_j - a_j$ is not considered useful (Felbecker (1980)). It is worth noting that the two criteria differ only by the waiting time. This paper also briefly makes use of the work in progress (WIP) criterion. WIP is a value associated to the system at a point of time t and is equal to the number of jobs at time t having their initial operation started or completed but their last operation not completed yet.

Following F. Ostermeier and J. Deuse's supply-chain-based classification of scheduling objectives (Ostermeier and Deuse (2024)), both outstanding time and flow time are considered manufacturer-related objectives. Flow time is widely recognized as an effective objective to reduce lead and response times for customers Dessouky et al. (1995)(. On the other hand, outstanding time contributes to reducing capital lockup and minimizing work in progress inventories (Geiger (2015)).

The outstanding time criterion has been preferred to the WIP criterion for several reasons. The outstanding time is a value associated to jobs, as for the flow time. On the other hand, WIP is associated to a moment of time. We prefer using two criteria associated with jobs, making comparison between the two criteria easier through the use of cumulative distribution function performance profiles (see Section 3.2). Also, there is no way to our knowledge to implement WIP as a variable in a mixed integer linear program for the JSSP with disjunctive formulation, which is used in rescheduling policies. Other formulations such as time indexed formulations allow WIP implementation, but solve JSSP much slower (Unlu and Mason (2010)). Nonetheless, a short comparison of algorithms with WIP is available in Section 3.7.

Note that, for a given offline instance, the minimization of the sum of flow times is equivalent to the total completion time objective. The minimization of the flow time could therefore be used as an objective function. However, this is not the case for the sum of outstanding times. Using this objective function can lead to undesirable schedules, where each job is processed one after another. The outstanding time is minimized for this schedule, being for each job the sum of its operation processing times. Such schedules process only one job at a time, therefore the WIP is also minimized (among non-idling schedules).

2.4 Objective functions

Objective functions serve as criteria for rescheduling policies. Two of the most common are the makespan, which is the time of completion of the last job,

$$C_{max} = \max_{j \in J}(C_j),$$

and the Total Weighted Completion Time (TWCT), which is the weighted sum of job completion times:

$$TWCT = \sum_{j \in J} \omega_j C_j.$$

However, neither of these explicitly seeks to minimize outstanding time. Here, we propose a new objective function: for each job, there is an associated fee to be "paid" at the start and a reward upon completion. In the context of companies seeking to maximize profit, the objective can be expressed as:

$$\text{Maximize} \sum_{j \in J} \frac{R_j}{(1+r)^{C_j}} - \frac{F_j}{(1+r)^{S_j}},$$

where, J represents the set of jobs, S_j and C_j are the starting and completion times of job j, F_j and R_j denote the respective fee and reward for job j, and r is a discount rate. This function is nonlinear, but the discount rate is typically low over the problem's time horizon (e.g., a 5% annual rate corresponds to a 0.4% monthly rate). Therefore, we can consider the first-order Taylor expansion at 0:

$$\text{Maximize} \sum_{j \in J} r(-R_j C_j + F_j S_j).$$

This problem is equivalent (in terms of solutions) to

$$\text{Minimize} \sum_{j \in J} R_j C_j - F_j S_j.$$

Fees and rewards can be chosen based on real-life data (e.g., raw material costs and selling prices). However, this data is highly case-dependent. In this paper, we assume

constant fees and rewards across all jobs. The problem then simplifies to

$$\text{Minimize} \sum_{j \in J} C_j - \frac{F}{R} S_j. \tag{1}$$

Note that this objective simplifies to minimizing the total completion time when $F = 0$, and to minimizing the outstanding time when $F = R$. The ratio $\frac{F}{R}$ determines the relative weight assigned to these two criteria.

2.5 Policies

As stated in 2.1, we define policies as the method used to determine which operation process at any given time, and can either be completely reactive or proactive. Among completely reactive policies, priority dispatching rules (PDR) are used to select the next job to be processed from a set of jobs awaiting at a machine that becomes free. This paper analyses three PDR performances:

- First In, First Out (FIFO): Machines process the operation of the waiting job with the earliest arrival time a_j
- Shortest Remaining Processing Time (SRPT): Machines process the operation of the waiting job with the shortest remaining processing time.
- Shortest Operation Processing Time (SOPT): Machines process the operation with the shortest individual processing time.

PDRs are useful as a benchmark for rescheduling algorithms and do not require an objective function.

Proactive policies include rescheduling algorithms, which generate a new schedule at each job arrival based on a specific method and an objective function. If a job has started but not yet finished, its remaining operations are treated as an "opened" job that needs to be rescheduled, while jobs that have not started are considered "unopened." The considered rescheduling algorithms are:

- REPLAN: A single schedule is created using both opened and unopened jobs.
- WEIGHTED REPLAN (with a weight factor ω): The weight of all opened jobs is multiplied by $\omega > 1$, and a schedule is created considering both opened and unopened jobs. If $\omega = 1$, this policy is equivalent to REPLAN. This policy is only meaningful for objectives that assign weights to jobs (such as Weighted Total Completion Time or the Fee-Reward objective).
- FILLING REPLAN: A schedule is first computed for the opened jobs only, and this schedule then acts as a constraint for scheduling all jobs.

For all rescheduling policies, if the fee-reward objective is used, the fee for an opened job is set to 0, as the fee was already "paid" when the job was opened.

Rescheduling algorithms require to find a schedule minimizing an objective with a specific method. All rescheduling algorithms tested in this article solve a version $J_n|rcrc, r_J, r_M|\gamma$ with a certain set of jobs (e.g. all unfinished jobs, unopened jobs...) to create their schedule. Therefore, a method is implemented to solve instances of the offline JSSP. This method is based on Mixed Integer Programming (MIP) with

a disjunctive formulation (Manne (1960)), which has proven to be among the most effective MIP formulation for the JSSP (Ku and Beck (2016)).

Most of the time, the JSSP instances are quite small (less than ten jobs) and the optimal solution can be found in a few seconds. However, the simulation of an online-time JSSP instance requires to solve hundreds of offline JSSP instances. Therefore, a time limit was implemented at 60 seconds for the subproblems, enabling the MIP to find an optimal solution in most cases, but also to find good solutions in reasonable time for bigger instances. In order to improve the quality of non-optimal solutions, Relaxation Induced Neighborhood Search (RINS) is used, which have proven to be effective for scheduling problems (Danna et al. (2005)). The RINS heuristic is called every 100 nodes and the sub-MIP node limit is set at 1000.

3 Computational results

In this section we analyze the performance of different strategies with respect to the flow time and outstanding time criteria. After presenting the simulation procedure, we start by analyzing the performance of some priority dispatching rules. Then, we observe the influence of parameters associated with the fee-reward objectives (fee-reward ratio $\frac{F}{R}$) and WEIGHTED REPLAN (weight ω). Finally, the best performing strategies compete with each other for the two criteria, with also a short comparison for the WIP.

3.1 Simulation Procedure

The simulation aims to proceed as follows: over a given duration, jobs arrive at a specific rate, referred to as intensity. The interruption of an ongoing operation is prohibited, and unfinished jobs at the end of the simulation remain unfinished. Note that the set of unfinished jobs at the end may vary from one simulation to another.

An instance of the online-time JSSP can be described as a list of jobs with associated arrival times. We create instances with the following procedure :

1. A set of 1002 jobs in a 10-machine environment is generated from real industrial data.
2. At each time step t, the number of jobs x arriving into the system at t is determined by drawing from a Poisson random variable. The use of a Poisson distribution reflects the discretization of a Poisson process to model job arrivals.
3. x jobs are randomly selected from the set of jobs, and are added to the instance with arrival date t.

3.2 Strategy comparison

In order to compare strategies based on outstanding time and flow time (which are values assigned to each completed job), we use cumulative distribution function performance profiles (referred in this article as performance profiles).

For a set of solutions provided by different strategies and a given criterion, the performance profile plots the proportion of jobs in the strategy's solution whose criterion is at most X times its best value across all strategies' solutions, where X is

the value on the x-axis. For each job, its performance is compared to the results from other strategies, and the job's performance ratio is defined as:

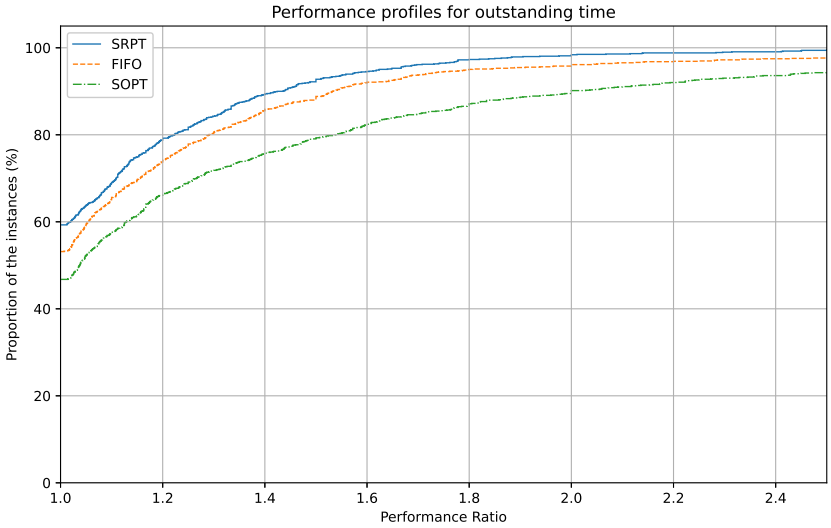
$$PerformanceRatio_j = \frac{Criterion_j}{\min_{j \in J}(Criterion_J)}.$$

The value of a strategy's curve on the performance profiles at x represents the proportion of jobs in that strategy's solution with a performance ratio less than x. Note that the performance profiles curve is a non-decreasing step function defined over $[1, +\infty[$ and taking values in $[0, 1]$. performance profiles have the advantage of being independent of job size, unlike measures such as the average of the criteria.

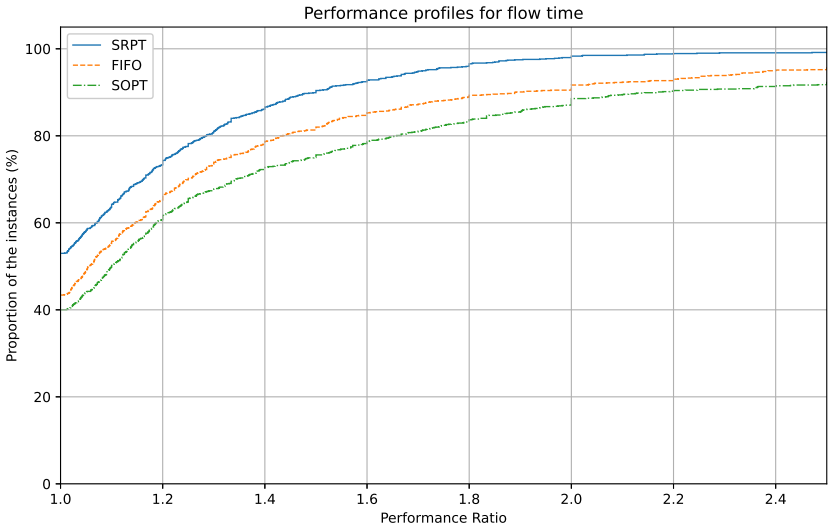
In this paper, a strategy A is said to dominate strategy B if the curve of A lies above that of B for both the outstanding time and flow time criteria.

In order to avoid border anomalies, comparisons are made only on jobs whose arrival time falls between two specified time points. The first point is chosen to exclude the transient period at the beginning, and the second to minimize the number of unfinished jobs. However, some unfinished jobs may remain for certain heuristics and need to be removed on a case-by-case basis, as otherwise their performance ratios cannot be evaluated. These removed jobs account for less than 1% of the total number of jobs. To mitigate bias introduced by the randomness of a particular instance, heuristics are tested on 10 different instances of the online-time problem. The sets of jobs from these solutions are then merged, and performance profiles are plotted from the merged sets.

3.3 Comparison of priority dispatching rules



(a) Outstanding time



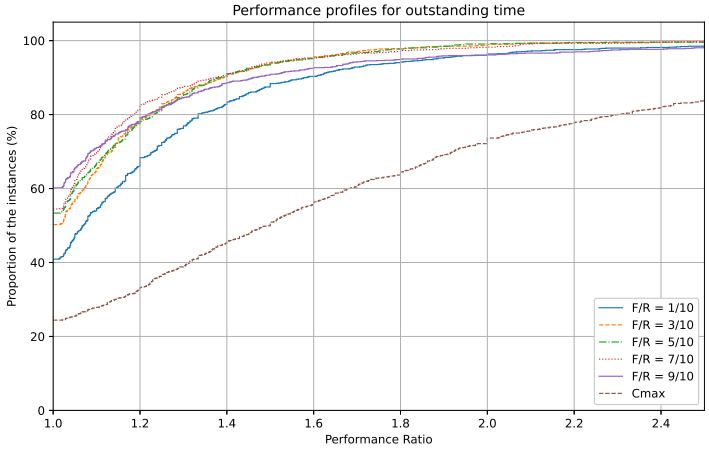
(b) Flow time

Fig. 1: Performance profiles of simple dispatching rules

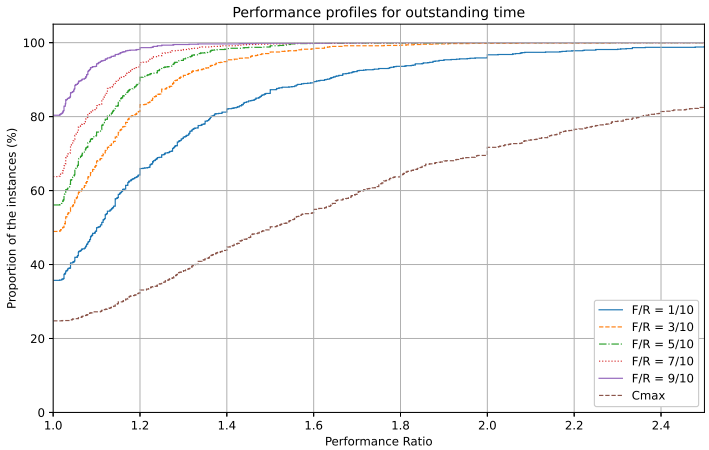
Figure 1 displays the performances of the three different priority dispatching rules that we consider: SRPT, FIFO and SOPT. As SRPT dominates the other PDRs on both criteria, we only keep SRPT as a PDR for strategies comparison in the remainder of this section.

3.4 Performance of the objective function in rescheduling strategies

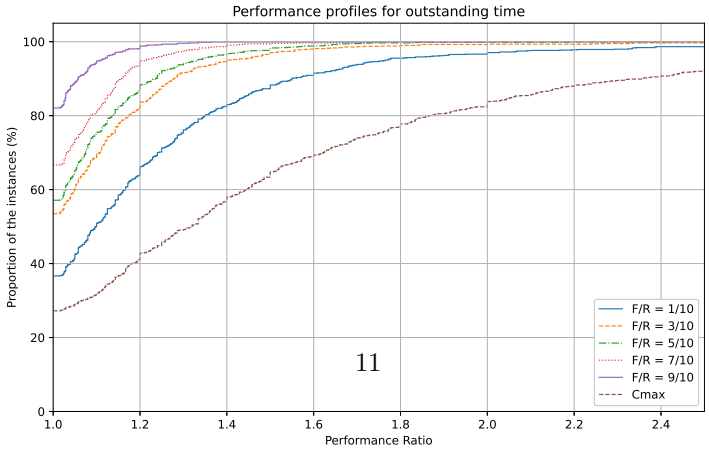
The makespan and the fee reward objectives are compared for different rescheduling policies (REPLAN, WEIGHTED REPLAN and FILLING REPLAN). Figure 2 and 3 respectively show the results for outstanding time and flow time.



(a) REPLAN

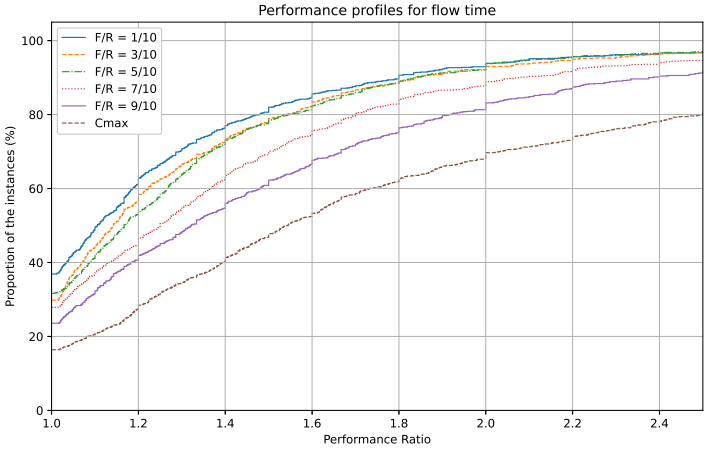


(b) WEIGHTED REPLAN with opened weight factor $\omega = 2$

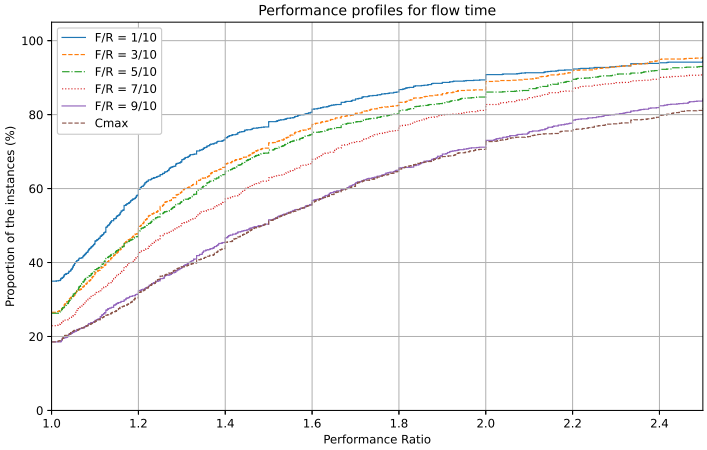


(c) FILLING REPLAN

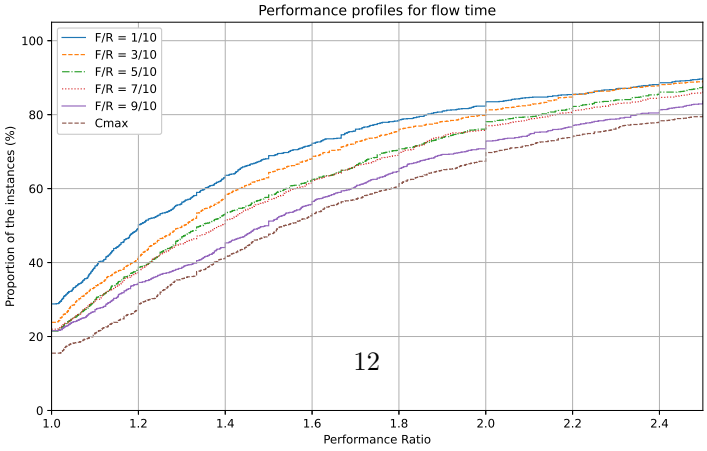
Fig. 2: Performance profiles for outstanding time of policies with various objectives



(a) REPLAN



(b) WEIGHTED REPLAN with opened weight factor $\omega = 2$



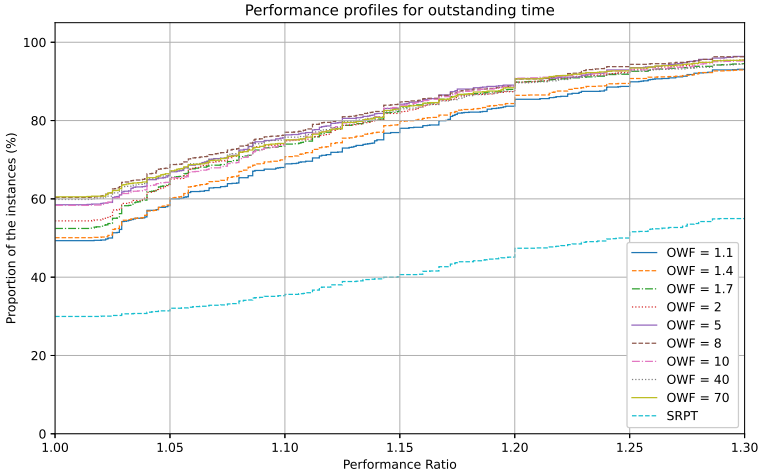
(c) FILLING REPLAN

Fig. 3: Performance profiles for flow time of policies with various objectives

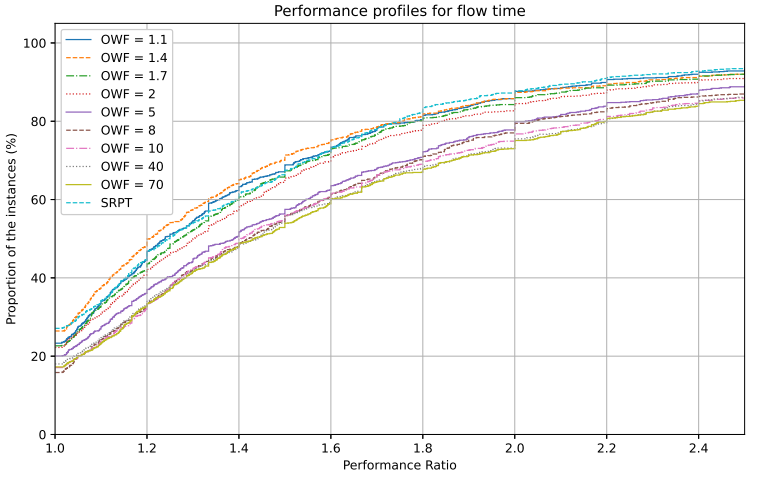
Across all policies, the makespan scores significantly worse results than the fee-reward objective for both criteria. It also seems that the higher the fee-reward ratio $\frac{F}{R}$ is, the lower the outstanding times and the higher the flow times are.

3.5 Influence of the weight factor ω on WEIGHTED REPLAN performances

To evaluate the influence of ω, Figure 4 shows the performance profiles for different values of ω with $\frac{F}{R} = \frac{2}{5}$. For comparison, the curve of Shortest Remaining Processing Time is also displayed.



(a) Outstanding time



(b) Flow time

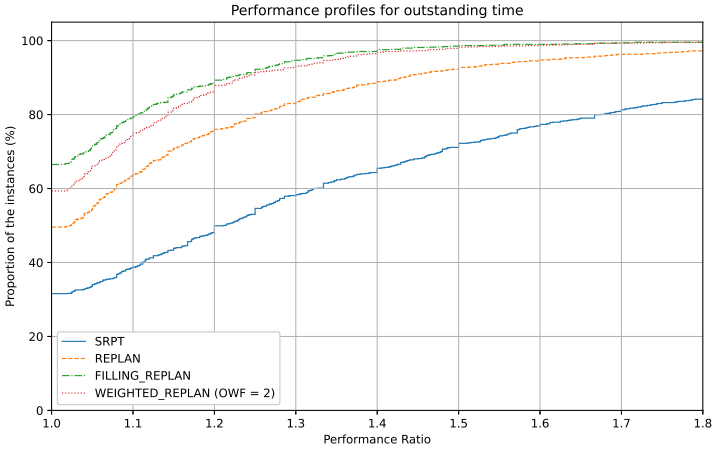
Fig. 4: Performance profiles of WEIGHTED REPLAN with various weight factors ω (OWF)

Regarding outstanding times, WEIGHTED REPLAN significantly outperforms SRPT, with its performance improving as ω increases. In terms of flow times, WEIGHTED REPLAN's performance ranges from slightly worse than SRPT at higher

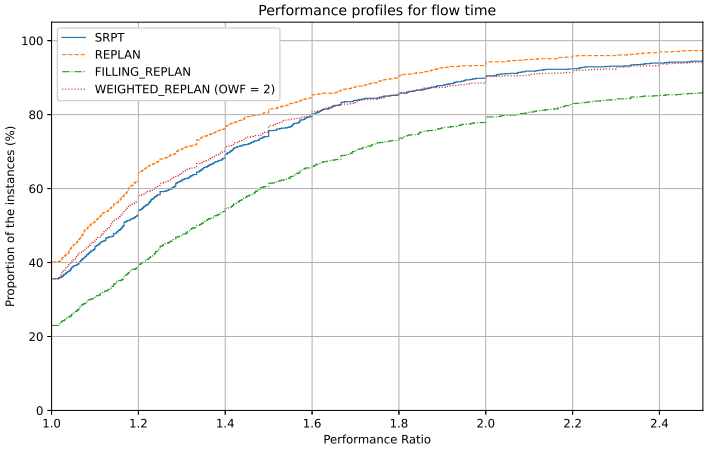
ω values to equivalent performance at lower ω values. Additionally, decreasing the fee-reward ratio $\frac{F}{R}$ can enhance flow times, however this comes at the cost of increased outstanding times.

3.6 Comparison of strategies

The graph below shows the performance profiles for SRPT and the rescheduling policies with the fee-reward objective for $\frac{F}{R} = \frac{1}{10}$. This ratio is chosen to be very low, as SRPT performs more competitively against the other policies in terms of flow times, and lower ratio increase rescheduling policies performances on flow times.



(a) Outstanding time



(b) Flow time

Fig. 5: Comparison of strategies for $\frac{F}{R} = \frac{1}{10}$

As seen in Figure 5, all rescheduling strategies perform much better than SRPT for the outstanding time criterion, with the best results coming from FILLING REPLAN, followed by WEIGHTED REPLAN. For flow times, SRPT and WEIGHTED REPLAN perform similarly, REPLAN is strictly better, and FILLING REPLAN is strictly worse. Therefore, REPLAN with a fee-reward ratio of $\frac{1}{10}$ dominates SRPT.

3.7 Performances of strategies on the work in progress

As explained, WIP is positively related to outstanding time. Figure 6 shows an histogram displaying the proportion of time (frequency) of a value of the WIP during the simulation for the previously described strategies. All parameters are kept the same ($\frac{F}{R} = \frac{1}{10}$, $\omega = 2$).

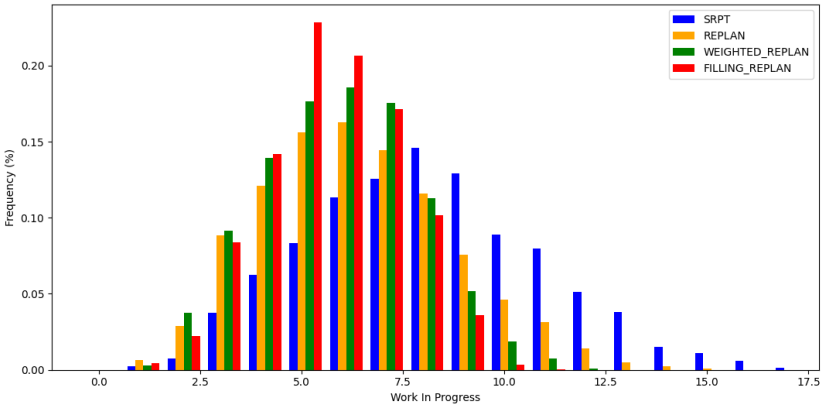


Fig. 6: Distribution of WIP during simulations

On average, worse performances for outstanding time is translated in higher WIP: SRPT demonstrate worse performance than the other strategies both in terms of average and maximum value. However, some differences can be noticed among rescheduling policies: FILLING REPLAN and WEIGHTED REPLAN see the frequences of high WIP decrease quickly compared to REPLAN, which results in lower maximum WIP (respectively 10 and 12 against 15 for REPLAN and 17 for SRPT). If the maximum amount of WIP is limited (because of, for example, a limited amount of space in a factory), FILLING REPLAN and WEIGHTED REPLAN may be preferable.

4 Conclusion

We have looked at the performance for the outstanding time and the flow time criteria of different algorithms solving the online-time JSSP, which is a convenient way to model real-time scheduling in industrial settings. The results show that the choice of policy plays a crucial role in scheduling performance. For rescheduling policies, the performances highly depend on the selected objective: the fee-reward objective has significantly outperformed the makespan, even though the latter is commonly used in the research field. Priority dispatching rules such as shortest remaining processing time are able to deliver good results in minimizing flow times but exhibit weaker performance in minimizing outstanding times. Rescheduling policies, particularly REPLAN and

WEIGHTED REPLAN, provide better overall results than priority dispatching rules for both criteria. Moreover, these algorithms allow the users to adjust the trade-off between outstanding time and flow time by tuning parameters such as the fee-reward ratio $\frac{F}{R}$ and the weight factor ω.

In practice, the flow time is useful to evaluate how fast jobs are done: shorter flow times mean being able to serve costumer quickly, which improve their satisfaction. On the other hand, the outstanding time does not directly impact the costumer: longer outstanding times can induce confusion for agent and technical difficulties. The rescheduling policies presented in this paper are particularly adapted for situation where low flow time is required and where long outstanding time can cause issue, such as in factories.

References

Albers, S.: Online scheduling. In: Introduction to Scheduling, pp. 71–98. CRC Press, ??? (2009)

Caceres, R., Douglis, F., Feldmann, A., Glass, G., Rabinovich, M.: Web proxy caching: The devil is in the details. ACM SIGMETRICS Performance Evaluation Review **26**(3), 11–15 (1998)

Dessouky, M.I., Moray, N., Kijowski, B.: Taxonomy of scheduling systems as a basis for the study of strategic behavior. Human Factors **37**(3), 443–472 (1995)

Danna, E., Rothberg, E., Pape, C.L.: Exploring relaxation induced neighborhoods to improve mip solutions. Mathematical Programming **102**, 71–90 (2005)

Divakaran, S., Saks, M.: An online algorithm for a problem in scheduling with set-ups and release times. Algorithmica **60**(2), 301–315 (2011) https://doi.org/10.1007/s00453-009-9337-9

Felbecker, O.: Ein beitrag zur reihenfolgeplanung bei mehrprodukt-linienfertigung. doctoralthesis, RWTH Aachen University (1980)

Geiger, M.J.: Multikriterielle Ablaufplanung. Springer, ??? (2015)

Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.G.K.: Optimization and approximation in deterministic sequencing and scheduling: a survey. Proceedings of the Advanced Research Institute on Discrete Optimization and Systems Applications of the Systems Science Panel of NATO and of the Discrete Optimization Symposium (5), 287–326 (1979)

Ku, W.-Y., Beck, J.C.: Mixed integer programming models for job shop scheduling: A computational analysis. Computers & Operations Research **73**, 165–173 (2016)

Kimbrel, T., Saia, J.: On-line and off-line preemptive two-machine job shop scheduling. Journal of Scheduling **3**(6), 355–364 (2000)

Manne, A.S.: On the job-shop scheduling problem. Operations research **8**(2), 219–223 (1960)

Ostermeier, F.F., Deuse, J.: A review and classification of scheduling objectives in unpaced flow shops for discrete manufacturing. Journal of Scheduling **27**(1), 29–49 (2024)

Ostermeier, F.F.: On the trade-offs between scheduling objectives for unpaced mixed-model assembly lines. International Journal of Production Research **60**(3), 866–893 (2022)

Pruhs, K., Sgall, J., Torng, E.: Online scheduling. (2004)

Torng, E., McCullough, J.: Srpt optimally utilizes faster machines to minimize flow time. ACM Transactions on Algorithms (TALG) **5**(1), 1–25 (2008)

Unlu, Y., Mason, S.J.: Evaluation of mixed integer programming formulations for non-preemptive parallel machine scheduling problems. Computers & Industrial Engineering **58**(4), 785–800 (2010)

Zhiming, W., Chunwei, Z.: Genetic algorithm approach to job shop scheduling and its use in real-time cases. International Journal of Computer Integrated Manufacturing **13**(5), 422–429 (2000) https://doi.org/10.1080/09511920050117919 https://doi.org/10.1080/09511920050117919