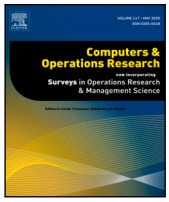
Computers & Operations Research 157 (2023) 106237

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



Maximizing the service level on the makespan in the stochastic flexible job-shop scheduling problem

Mario Flores-Gómez [a,*], Valeria Borodin [a], Stéphane Dauzère-Pérès [a,b]

[a] Mines Saint-Etienne, Univ Clermont Auvergne, CNRS, UMR 6158 LIMOS CMP, Department of Manufacturing Sciences and Logistics, Gardanne, France
[b] Department of Accounting and Operations Management, BI Norwegian Business School, Oslo, Norway

ARTICLE INFO

Keywords:
Scheduling
Flexible job-shop
Stochastic
Makespan
Service level
Tabu search
Monte Carlo sampling-based approximation

ABSTRACT

This paper considers the flexible job-shop scheduling problem with stochastic processing times. To find a sequence insensitive to shop floor disturbances, the available probabilistic information related to the variability of processing times is taken into account by maximizing the *makespan service level* for a given deadline. This corresponds to the probability of the makespan to be smaller than a given threshold. After showing why this criterion makes sense compared to minimizing the average makespan, a solution approach relying on a tabu search and a Monte Carlo sampling-based approximation is presented. Then, new instances are generated by extending the deterministic benchmark instances. Extensive computational experiments are conducted to evaluate the relevance of the makespan service level and the performance of the proposed solution method. The drawbacks of a number of reference scenarios, including worst-case and best-case scenarios, in addressing effectively the problem under study are presented. A numerical analysis is also performed to compare the scope of the proposed criterion against the minimization of the expected makespan. The accuracy of the proposed solutions induced by the hyper-parameters of the Monte Carlo approximation is explicitly analyzed.

1. Introduction

The Flexible Job-shop Scheduling Problem (FJSP) is one of the most studied scheduling problems, found in many practical contexts where multiple resources can perform the same operation. This is the case when identical resources are available or when resources are flexible and can process different types of operations. The additional complexity associated with the assignment of operations to resources makes the FJSP significantly more difficult to solve than the classical Job-shop Scheduling Problem (JSP). However, the flexibility of the resources allows classical criteria such as the makespan (i.e., the maximum completion time of all the jobs) to be reduced compared to the JSP. One of the reasons is that an optimal assignment of operations to resources contributes to better balancing the workload on the resources. This is particularly important when resources are unrelated, i.e., when the processing time of an operation can vary from one resource to another. In this case, it might for instance be sounder to avoid the assignment of an operation to the slowest capable resource.

We believe that another factor to be considered when an operation is assigned to a resource is related to the variability or the limited control of the processing times. For instance, in industrial contexts such as semiconductor manufacturing, the processing times of operations of new products are often more fluctuating (i.e., less controlled) than the processing times of operations of mature products (Karabuk, 2001). In this case, it could be risky to assign all the operations with the less controlled processing times to the same machine. Also, the processing time of an operation might be more difficult to control if the operation is assigned to a new machine or to a machine that rarely processes the operation, than to an older machine or to a machine that very often processes the operation (Çatay et al., 2003). Considering average processing times as deterministic processing times does not capture the impact of the variability of processing times on the quality of a schedule.

In this paper, we assume that the probability distributions of the processing times can be estimated and are thus known, and we are interested in solving a Stochastic FJSP (SFJSP). This problem is even more difficult to solve than the FJSP as it makes the decision-making process significantly more difficult with the uncertainty factor regarding the processing times of the operations. The SFJSP can be found in many industrial environments, being often subject to various sources of variability induced by machines (e.g., breakdowns), job releases, and/or processing times (e.g., complex multi-chamber machines, complex processing routes) (see e.g., Jamrus et al. (2018)). Furthermore, since the problem is no longer deterministic, the classical deterministic criteria become out of context, and an adaptation is required. Typically,

* Corresponding author.
E-mail addresses: mario.flores@emse.fr (M. Flores-Gómez), valeria.borodin@emse.fr (V. Borodin), dauzere-peres@emse.fr (S. Dauzère-Pérès).

https://doi.org/10.1016/j.cor.2023.106237
Received 9 August 2022; Received in revised form 24 February 2023; Accepted 31 March 2023
Available online 8 April 2023

in stochastic scheduling research, the counterpart of a deterministic criterion is its mathematical expectation (Mahdavi et al., 2010; Zhang et al., 2012). Instead, we focus on the maximization of the makespan service level, as proposed by Dauzère-Pérès et al. (2005, 2013). We believe, this notion is relevant and, to the best of our knowledge, has been rarely studied in the scheduling-related literature (see Section 3.2). It corresponds to the probability of a classical criterion to be smaller (or larger) than a given value. In this paper, the *makespan service level* is considered, which is the probability that all the jobs are completed before an allowed time limit. In real-life applications, this limit corresponds to a specific scheduling horizon. The service level of other classical criteria could be also studied such as the sum of the completion times or the maximum tardiness (Dauzère-Pérès et al., 2013).

This paper extends the solution approach proposed by Flores Gómez et al. (2021) to solve the SFJSP in the following ways: **(i)** The exploration of the neighborhood during the execution of the tabu search algorithm is enhanced by considering a more efficient move evaluation procedure, **(ii)** The accuracy of the proposed solutions induced by the sampling size in a crude Monte Carlo sampling-based approximation is explicitly evaluated based on existing statistical results, **(iii)** Several levers allowing us to enhance the efficiency of the Monte Carlo sampling-based approximation are empirically investigated, **(iv)** Extensive numerical experiments are conducted to highlight the relevance of the makespan service level against a number of reference scenarios and the minimization of the expected makespan.

The paper is organized as follows. The literature on the deterministic and stochastic flexible job-shop scheduling problems is reviewed in Section 2. The problem under study is formalized and motivated in Section 3. A solution approach is presented in Section 4, which combines a tabu search approach with a Monte Carlo sampling-based approximation. Stochastic FJSP instances are generated from classical deterministic instances in Section 5. Then, extensive computational experiments are discussed in Section 6 to assess the relevance of the makespan service level. Additional computational experiments are analyzed in Section 7 to tune the hyper-parameters of the Monte Carlo approximation. In Section 8, we conclude and discuss some perspectives of this research with a view to improving the solution approach and to applying the notion of service level to other scheduling criteria than the makespan.

2. Literature review

As previously stated, given the large spectrum of associated real-life applications, the FJSP is a very popular scheduling problem, both in academic research and in practical contexts. The FJSP has been abundantly studied in the literature, and new approaches are continuously being proposed or extensions of the problem being studied. In this section, a brief literature review is proposed, on both the FJSP and its stochastic version, the SFJSP.

2.1. Flexible job-shop scheduling problem

The Flexible Job-shop Scheduling Problem (FJSP) is a generalization of the classical job-shop problem, with the difference that operations can be processed on several machines. Brucker and Schlie (1990) were the first to address the FJSP with only two jobs. Ever since, many papers have been published proposing different solution approaches for this problem. In a recent survey, Chaudhry and Khan (2016) realized that up to 2016, most of the published research dedicated to the FJSP (almost 75%) are focused on the optimization of the value of the makespan C_{max} or a combination of C_{max} and another objective function. Brandimarte (1993) decomposed the problem into an assignment problem and a scheduling problem, and proposed a hierarchical algorithm. Paulli (1995) proposed a very similar approach for the *Flexible Manufacturing System Scheduling* problem, which is basically the FJSP with a constraint limiting the maximal number of jobs that can be processed at the same time. In (Hurink et al., 1994), a more integrated approach using tabu search is suggested. Dauzère-Pérès and Paulli (1994) proposed a new procedure for the reassignment of operations. All these publications decompose the FJSP into two subproblems. Dauzère-Pérès and Paulli (1997) proposed an integrated approach for the FJSP, called multiprocessor job-shop scheduling problem in the paper, using tabu search. The approach relies on a disjunctive graph representation of the problem and a connected neighborhood structure introduced in (Dauzère-Péres, 1994). Besides the exploitation of the structural properties of the problem, the authors proposed several lower bounds on the makespan C_{max} to accelerate the neighborhood exploration. In (García-León et al., 2015), a tabu search using the neighborhood structure from (Dauzère-Péres, 1994) in the disjunctive graph and novel conditions are presented to optimize any regular criterion. This work is extended to multi-objective optimization by García-León et al. (2019). In (Shen et al., 2018), a Mixed Integer Linear Program (MILP) for the FJSP with sequence-dependent setup times is presented that can only solve small instances. In addition, the feasibility conditions in (Dauzère-Pérès and Paulli, 1997) are revisited and improved. Recent works on extensions of the FJSP to batching constraints and other realistic constraints or objectives can be found in (Knopp et al., 2017) and (Tamssaouet et al., 2022).

Several other techniques than tabu search have been used to solve different versions of the FJSP. For instance, Kacem et al. (2002) proposed two different methods, including an evolutionary algorithm, to deal jointly with the assignment and job-shop scheduling problems. Rossi and Dini (2007) used an Ant Colony Optimization to solve the FJSP with sequence-dependent setup and transportation times, and operation lag times. Xing et al. (2009) proposed a simulation model for a multi-objective version of the FJSP to minimize the makespan, the workload of all the machines, and the workload of critical machines. Yazdani et al. (2010) proposed a heuristic based on the exploration of a parallel variable neighborhood to increase the diversification during the exploration of the search space based on shake and local search procedures. A complete survey on different approaches used for the FJSP is conducted by Chaudhry and Khan (2016). A survey of solution approaches implemented for complex job-shop scheduling problems in wafer fabrication is presented by Mönch et al. (2011). A commonly used heuristic to solve a variant of the FJSP is the shifting bottleneck heuristic, which decomposes the job-shop scheduling problem into a set of smaller scheduling sub-problems related to parallel machines (Adams et al., 1988). Genetic algorithms (GA) have become increasingly popular to solve the FJSP. Çaliş and Bulkan (2015) pointed out that more than 25% of published research related to the FJSP involved the implementation of GAs. One of these approaches is presented in (Mönch et al., 2007). Pezzella et al. (2008) developed a GA-based approach to solve a resource-constrained operation-machine assignment problem and the FJSP while optimizing the makespan. Gao et al. (2008) proposed a hybrid GA to minimize the makespan, the maximal machine workload, and the total workload. An extensive survey on the evolution and implementation of GA-based techniques for the FJSP can be found in (Amjad et al., 2018).

2.2. Stochastic job-shop and flexible job-shop scheduling problems

In the literature, the stochastic version of the classical job-shop scheduling problem has been significantly less addressed than its deterministic counterpart, even if real-life problems usually include stochastic parameters. In (Horng et al., 2012), an evolutionary algorithm that embeds an evolutionary strategy in ordinal optimization is proposed to solve the stochastic job-shop scheduling problem while minimizing the expected sum of storage expenses and tardiness penalties. The approach uses simulation to select a fixed number of roughly good schedules as a starting point, aiming to reduce the search space at every step. For the SFJSP, Mokhtari and Dadgar (2015) proposed a

simulation model for an FJSP with machine maintenance operations, where processing times and due dates are stochastic and machine failure rates are time-varying. A MILP model is introduced with the objective of minimizing the number of tardy jobs and a minimum total availability constraint, and a simulation–optimization framework based on a simulated annealing optimizer and Monte Carlo sampling-based approximation is proposed to solve the problem. In (Mahdavi et al., 2010), a simulation-based decision support system controlling an SFJSP manufacturing system is described. It uses the results from a real-time simulator to identify opportunities for incremental improvements of the performance criteria, built around the supervisory control theory based on discrete-event simulation. Also based on simulations (Monte Carlo) but combined with the second stage of a two-stage particle swarm optimization algorithm, the model presented in (Zhang et al., 2012) aims at minimizing the expected total weighted tardiness. In none of the papers cited above, the service level is used to evaluate the quality of a solution. In general, the expected value of the processing times can be used to transform the stochastic problem into a deterministic one.

2.3. Service level in scheduling problems

Very few papers have used the notion of service level in scheduling problems. To the best of our knowledge, Golenko-Ginzburg et al. (1995) is the first who used the *delivery performance* notion as the probability of a given job to be completed on time (i.e., to respect the due date). Two heuristics based on pairwise job comparison, to either maximize the weighted sum of jobs' delivery performances or to constrain them to a minimal value while minimizing the makespan, were proposed. Each heuristic is based on comparing the jobs that are available to be executed on the available machine at any point in time. These comparisons are carried out by calculating each job's delivery performance. Both methods are tested on a 10×5 (10 jobs, 5 machines) JSP instance. Daniels and Carrillo (1997) defined, for a single machine scheduling problem, the β-robust scheduling problem (β-RSP) as identifying the schedule with the maximum likelihood of achieving flow time performance no greater than a given target level. Exact and heuristic solution approaches are proposed to obtain the best possible schedule when the processing times of jobs are independent random variables. The authors demonstrate that finding such a schedule is NP-hard.

The term *service level*, used in inventory management, was first coined in scheduling in (Dauzère-Pérès et al., 2005, 2013). The authors provide a general context of the notion and several examples based on an instance of the flow-shop scheduling problem subject to random processing times following several probabilistic laws. They also propose two different methods to quantify the service level. The notion of β-robustness was extended by Beck and Wilson (2007) to deal with the probabilistic JSP while exploring the notion of α-makespan. If the α-makespan of a solution S is lower than or equal to a value D, then there is at least a $1-\alpha$ probability that the makespan of S is lower than or equal to D. They propose a number of techniques combining Monte Carlo sampling-based approximation with different solution approaches such as constraint programming and tabu search. They also prove that, when carefully defined, the deterministic JSP can be used as a lower bound for the probabilistic JSP (as defined in their paper). Wu et al. (2009) approach the β-RSP with a constraint programming model explicitly representing the uncertainty and robustness as input parameters and objectives, by developing explicit representations of the task uncertainty and enabling the uncertainty to propagate. Three models are developed (primal, dual, and hybrid) and the effect of dominance rules in the search space was analyzed. All references above used the notion of service level, although not necessarily under that name.

As the conducted literature review shows, there is much work to be done in several directions: **(i)** Defining key performance indicators to quantify to what extent a sequence is insensitive to disturbances and how uncertainties impact the quality of sequences, **(ii)** Proposing stochastic extensions of scheduling criteria and explaining their scope, **(iii)** Modeling uncertainty in a low-cost manner given the intrinsic complexity of the FJSP, etc. From an industrial point of view, the efforts dedicated to handling uncertainties in scheduling problems are still insufficient to support their automation in manufacturing processes. By contributing to filling these gaps, this paper applies the notion of service level on the makespan in the FJSP subject to uncertain processing times. New instances are generated by extending the deterministic benchmark instances to discuss the relevance of the makespan service level against the expected makespan and the makespan corresponding to a number of reference scenarios. Based on the existing background, a solution approach relying on a tabu search and a Monte Carlo sampling-based approximation is presented, while considering explicitly the tuning of its hyper-parameters.

3. Problem description

As an extension of the JSP, the FJSP is thus NP-hard in the strong sense (Garey et al., 1976). The random nature of processing times adds another level of complexity to the deterministic FJSP, making the Stochastic FJSP at least as hard to solve.

3.1. Stochastic flexible job-shop scheduling problem

The FJSP aims at scheduling a set of operations $\mathcal{O}$ partitioned into a set of jobs $\mathcal{J}$ on a set of machines $\mathcal{M}$. The operations of each job j are grouped in set $\mathcal{O}_j$ ($\bigcup_{j\in\mathcal{J}} \mathcal{O}_j = \mathcal{O}$), and have to be processed in a specific order (routing). A machine can only perform one operation at a time, and cannot be interrupted once the operation starts, i.e., preemption is not allowed. Unlike the JSP, each operation i can be processed on any machine in a given subset of machines $\mathcal{M}_i$ in $\mathcal{M}$ in the FJSP. The processing time p_{ik} of operation i may vary depending on the machine k to which i is assigned (i.e., processing times are machine-dependent). Solving the FJSP requires determining both an assignment of the operations to the machines and a sequence of operations on the machines. The objective is to optimize a scheduling criterion, the most common one being the makespan or C_{max}, which is the maximum completion time of all operations. Contrary to the FJSP, the SFJSP includes random processing times described via known probability distributions. Let each random variable ξ_{ik} modeling the processing time of operation i on machine k be discrete and defined on a finite support $[c_{ik}, d_{ik}]$. In practical settings, an operation takes a minimum time ($c_\bullet$) to be completed on a machine and does not take more than a maximum time ($d_\bullet$) under normal operating conditions (Jacobs et al., 2003). By normal operating conditions, we mean that only short stoppages of the machine are considered, but not major disturbances (e.g., machine breakdown, unexpected job arrivals, shortage of raw materials). The latter usually entails a rescheduling problem (Ghaleb et al., 2020). We want the proposed assignment and sequence of operations to be able to withstand small variations of processing times without requiring them to be updated. Processing times are assumed to be independent random variables.

3.2. Notion of makespan service level

In this paper, we extend the study on the relevance of the notion of *makespan service level* as defined in (Dauzère-Pérès et al., 2013, 2005) and in the continuity of our work in (Flores Gómez et al., 2021). To better understand the practicality of the service level in scheduling, it is important to highlight that this notion is frequently employed in inventory management theory. It is often used to determine the safety stock, which is the stock level below which a new command has to be started, in order to have a higher likelihood of satisfying the stochastic demand. This stock level is called the order point, which is generally computed in two different ways, either by minimizing an estimated global cost in the system at hand (backorder cost, order cost, holding cost) or by satisfying a given service level (Silver et al., 1998).

The notion of service level in scheduling applications represents the probability that a criterion is lesser (or greater) than or equal to a given value. If the used performance measure is the makespan, i.e., the date of completion of all jobs in the system, then the makespan service level is the probability of finishing the execution of all jobs before a given point in time. It is thus possible to consider the service level of any criterion in deterministic scheduling. Dauzère-Pérès et al. (2005, 2013) observed that optimizing a service level on a scheduling criterion leads to sequences of good quality with a high probability and, therefore, optimizes the robustness of the solution. As in inventory theory, the notion of service level is quite easy to be interpreted. In particular, in the case of the makespan, it makes more sense to maximize the chance of finishing all jobs before a given deadline than to minimize the mean completion time of all jobs, especially from an industrial standpoint when it is important to know how many hours the operator of a given machine would be needed.

As the processing times are random variables, the makespan of a sequence S is also a random variable. In the literature, the mathematical expectation is usually optimized. A common industry practice we are aware of is to set the values of the processing times as their expected values and to solve a classical FJSP. We propose to study the service level $\alpha(S,T)$ on the makespan, i.e., the probability that the makespan of sequence S is lower than or equal to a given threshold, denoted by T. The makespan service level is formally defined below:

$$\alpha(S,T) = \mathbb{P}(C_{max}(S,\xi) \leq T),$$

where ξ is a multivariate random variable of dimension n.

To evaluate the service level of a sequence, a set of scenarios Ω is generated, and an algorithm based on Monte Carlo sampling-based approximation is implemented. The resulting estimated service level is denoted $\alpha(S,T,\Omega)$. The proposed solution approach is based on the tabu search of Dauzère-Pérès and Paulli (1997) and includes a Monte Carlo sampling-based procedure to represent and deal with uncertainties. The results of the first computational experiments as well as an outline of the approach can be found in (Flores Gómez et al., 2021). A more detailed overview of the approach is given in Section 4.

3.3. Reference scenarios

To assess the relevance of the notion of service level on the makespan, let us consider the following reference scenarios:

- **Average processing times:** Let $\bar{p}$ be the scenario where processing times take the mean value given their probability distributions.
- **Worst-case and best-case scenarios:** Under pessimistic (resp. optimistic) settings, a common practice to handle uncertainties is to use the worst-case (resp. best-case) scenario, where all random variables take their largest (smallest) values (see e.g., Birge and Louveaux (2011)).
  Let ω_q be the scenario where every random processing time takes the value of $c_\bullet + (d_\bullet - c_\bullet)q$. Hence, $q = 1$ for the worst-case scenario (ω_1) and $q = 0$ and for the best-case scenario.
  For every operation i and every machine k, let $p_{ik}(\omega)$ be the realization of the processing time of i when executed by k in scenario ω, such that:

  $$p_{ik}(\omega_0) \leq p_{ik}(\omega) \leq p_{ik}(\omega_1), \quad \forall \omega \in \Omega.$$

  Let $C_{max}(S,\omega)$ be the makespan of sequence S with the processing times in scenario ω. Then:

  $$C_{max}(S,\omega_0) \leq C_{max}(S,\omega) \leq C_{max}(S,\omega_1), \quad \forall \omega \in \Omega.$$

3.4. Illustrative example

For illustration purposes, consider the instance with 3 jobs and 3 machines in Table 1. The processing times of operations 4 and 5 of job 2 are random variables following a four-parameter Beta distribution function $f^{\bullet}_{\beta}(c,d,\mu,\sigma)$, where c and d are the limits of the definition domain, μ represents the mean and σ denotes the standard deviation.

Consider the processing times of operations 4 and 5 of job 2 approximated by their means values $\bar{p}$. The two feasible sequences shown in Fig. 1 only differ by the position of operations 5 and 7, which respectively belong to jobs 2 and 3. Sequence π_1 has a makespan of 80, while the makespan of sequence π_2 is equal to 90. A common decision driven by the minimization of the makespan based on $\bar{p}$ would dictate that π_1 is a more suitable sequence.

Fig. 2 shows sequences π_1 and π_2 when the processing times of the operations of job 2 take the values in the best-case scenario ω_0. Both sequences have the same makespan. However, when the processing times are approximated by the worst-case scenario, the makespan is significantly larger for sequence π_1, $C_{max}(\pi_1,\omega_1) = 130$, than for sequence π_2, $C_{max}(\pi_2,\omega_1) = 110$, as shown in Fig. 3. Sequence π_2 is thus better than sequence π_1 in the worst-case scenario. Also, the jobs in sequence π_2 have a 100% probability of being completed before 110, i.e., the service level $\alpha(\pi_2,110)$ is equal to 1, whereas this is not the case for sequence π_1.

For illustration purposes, consider Fig. 4 showing the service levels of two sequences π_1 and π_2 as a function of threshold T. Two main remarks can be made:

- The service level is specific to a sequence, not to an instance.
- The service level depends on threshold T. In real-life settings, T is usually fixed. Depending on the context, T may represent a week, a working day, or a worker's shift.

Maximizing the service level in the SFJSP means finding a sequence of operations with the largest service level among all feasible sequences. From Fig. 4, it is possible to see that there is a probability equal to 0.5 of completing the jobs before $T = 85$ using sequence π_1, whereas this probability is almost zero when using sequence π_2. However, if T is equal to 95, then the service level of sequence π_2 is almost 100%, while the service level of sequence π_1 is around 70%. Furthermore, there is a probability equal to 0.99 that all jobs are completed before $T = 118$ for both sequences. The impact of the shape of the probability density functions of the processing times on the service level is analyzed in (Flores Gómez et al., 2021) for different values of threshold T.

4. Solution approach

The works in (Dauzère-Pérès et al., 2005, 2013) and (Beck and Wilson, 2007) are the closest to ours. In (Dauzère-Pérès et al., 2005, 2013), a framework on the notion of service level and solution approaches are thoroughly explained using different distribution laws and an illustrative example. Beck and Wilson (2007) consider the probabilistic JSP, where assigning operations to machines is not part of the decision process, and most of the proposed approaches are based on branch and bound schemes. A tabu search (Gendrau and Potvin, 2010) is also presented, with a time limit as a stopping criterion, and an estimation of the probabilistic makespan of the sequences found in the tabu search. The main difference between the current paper and (Beck and Wilson, 2007) lies in the way the problem is posed. Beck and Wilson (2007) aim to find *as small a value of D as possible* such that there is a sequence S with a high $\alpha(S,D)$. In this paper, we do not focus on minimizing threshold T, as it is considered as an input of the problem.

Table 1
Data of an illustrative example with 3 jobs and 3 machines.

Job	Number of operations	Operation	{Machine, $\overline{p}$}	Random variables $f^{\star}_{\beta}(c,d,\mu,\sigma)$
1	3	1	{1,30}, {2,30}	–
		2	{2,40}, {3,20}	–
		3	{2,20}	–
2	2	4	{2,30}	$f^{4,2}_{\beta}(12,60,30,15)$
		5	{1,20}, {3,20}	$f^{5,1}_{\beta}(10,40,20,5)$, $f^{5,3}_{\beta}(10,40,20,5)$
3	2	6	{3,40}	–
		7	{1,30}	–

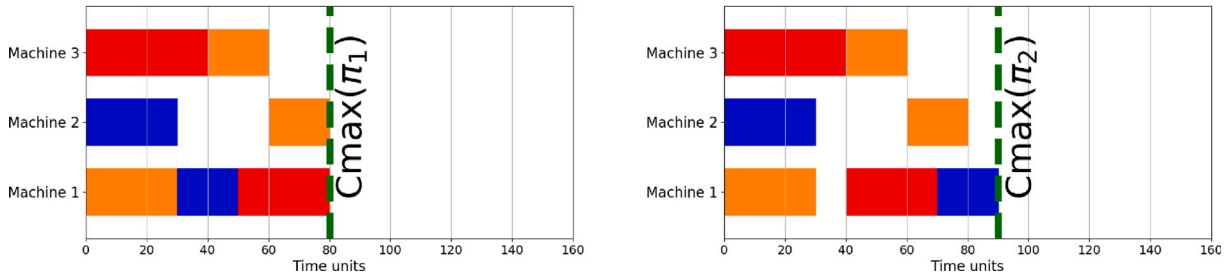


Fig. 1. Two sequences π_1 and π_2 corresponding to scenario $\bar{p}$.

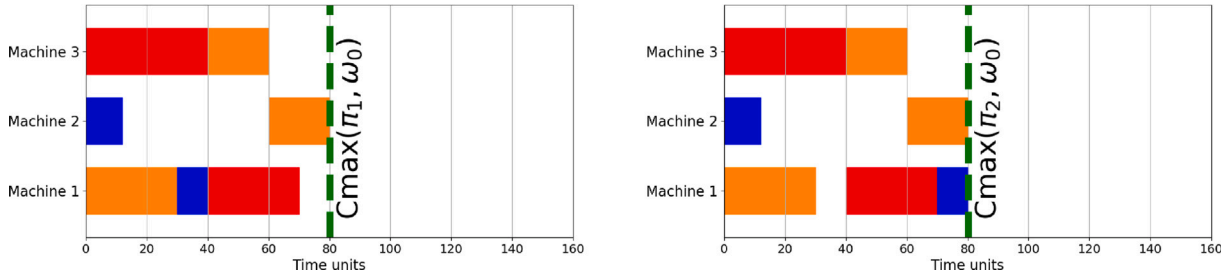


Fig. 2. Sequences π_1 and π_2 for the best-case scenario ω_0.

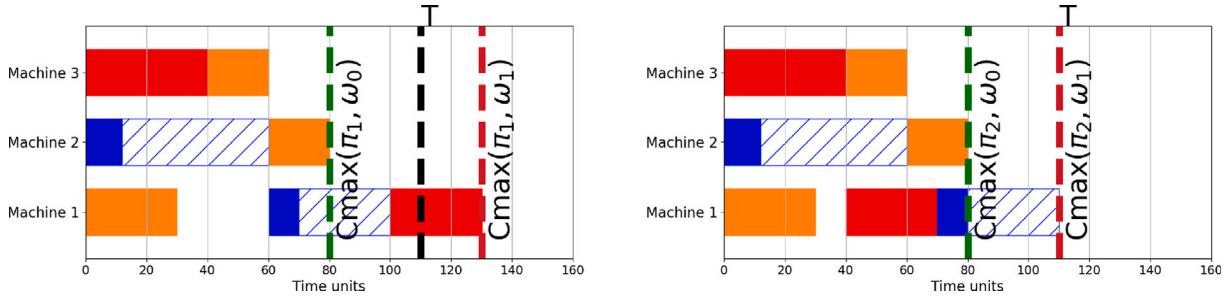


Fig. 3. Sequences π_1 and π_2 for the worst-case scenario ω_1.

4.1. Solving the deterministic flexible job-shop scheduling problem

As previously stated, the deterministic FJSP is NP-hard in the strong sense, and exact approaches, mainly mathematical models solved by a standard solver, can only handle small instances. The stochastic FJSP being even more difficult, we propose a heuristic approach, which is based on the tabu search proposed in (Dauzère-Pérès and Paulli, 1997) for the deterministic FJSP. This tabu search relies on an extension, introduced in (Dauzère-Péres, 1994) and illustrated in Fig. 5, of the disjunctive graph model for the JSP (see Roy and Sussmann (1964)). In a disjunctive graph $G = (N, A, E)$, set N includes the nodes associated with operations $\mathcal{O}$ and two virtual nodes 0 and $*$ ($\forall k \in \mathcal{M}, p_{0k} = p_{*k} = 0$ and $\mathcal{M}_0 = \mathcal{M}_* = \emptyset$). A is the set of conjunctive arcs modeling the routing of operations, and E is the set of disjunctive arcs modeling the potential assignment and sequencing of the operations on the machines.

In the disjunctive graph, the weight of an arc is the processing time of the operation from which the arc starts. The processing times are machine-dependent, thus the lengths of two arcs between two operations might be different if the two operations can be both assigned to two common machines. In the example illustrated in Fig. 5, there are 3 jobs to be processed on 3 machines. Operations 1 and 2 belong to the first job, operations 3, 4, and 5 to the second job, and operations 6, 7, and 8 to the third job. Operations 2, 3, 4, 7, and 8 can only be processed on a single machine ($\mathcal{M}_3 = \mathcal{M}_8 = \{1\}, \mathcal{M}_4 = \{2\}$ and $\mathcal{M}_2 = \mathcal{M}_7 = \{3\}$), whereas operations 1, 5 and 6 may be processed on two machines ($\mathcal{M}_1 = \{2,3\}, \mathcal{M}_5 = \{1,3\}$ and $\mathcal{M}_6 = \{1,2\}$). Let us denote by S the selection of conjunctive arcs in E, where the disjunctive arcs are replaced by conjunctive arcs modeling the assignment of each operation to a machine as well as the sequencing of operations on the machines. Each feasible selection S corresponds to a feasible sequence for the

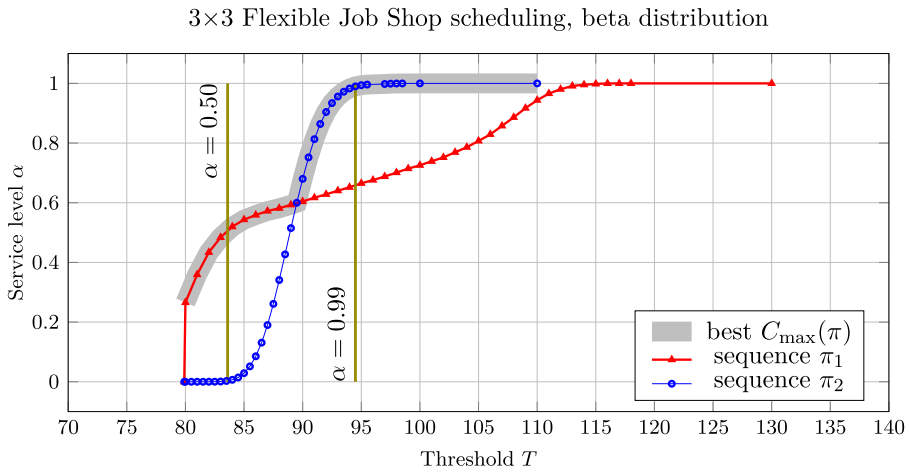


Fig. 4. Service level of sequences π_1 and π_2 as a function of threshold T.

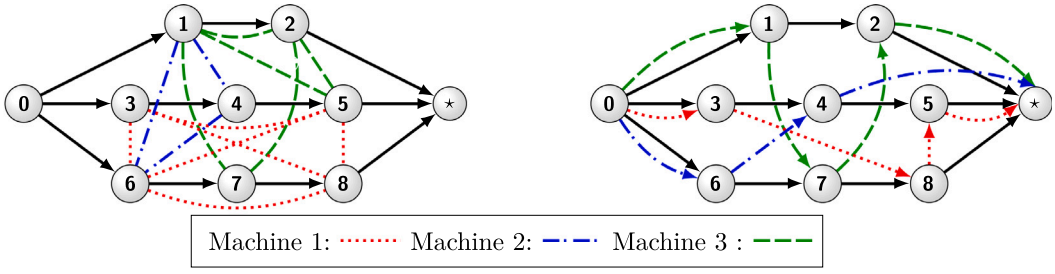


Fig. 5. Example of a disjunctive graph and a conjunctive graph for FJSP with 3 jobs and 3 machines.

scheduling problem, which can be modeled as a conjunctive graph $G' = (N, A, S)$. The sequence is feasible if and only if G' is a directed acyclic graph.

Using this disjunctive graph representation, a neighborhood structure is introduced and exploited to find the best solution possible. A neighbor sequence corresponds to moving an operation i sequenced between operations s and t on the machine assigned to i in the conjunctive graph of the current solution, between operations s' and t'. This move is performed by deleting the conjunctive arcs (s, i) and (i, t), adding the conjunctive arc (s, t), deleting the conjunctive arc (s', t') and adding the conjunctive arcs (s', i) and (i, t'). If operations s' and t' are assigned to another machine than operations s and t, then operation i is reassigned, otherwise, i is "only" resequenced on the same machine. To ensure that a cycle is not created in the resulting graph before moving an operation, the sufficient feasibility conditions proposed in (Dauzère-Pérès and Paulli, 1997) are used in this paper. A lower bound is also introduced by Dauzère-Pérès and Paulli (1997) to evaluate the quality of a move. This lower bound is used for the SFJSP in Sections 4.4 and 4.5.

The *length* of a path in the conjunctive graph G' is the sum of the weights of the arcs on the path. Let $L(i, i')$ be the *longest path* between operations i and i' (= 0 if there is no path), $r_i = L(0, i)$ denotes the release date (head) of i and $q_i = L(i, *) - p_{ia(i)}$ its delivery time (tail), where operation i has been assigned to machine $a(i) \in \mathcal{M}_i$. A longest path between virtual nodes 0 and $*$ is called a *critical path*, and its length $r_* = L(0, *)$ is the value of the makespan of the sequence associated with G'.

4.2. Initial solution

In general, the initial solution obtained as proposed in (Dauzère-Pérès and Paulli, 1997) is rather poor. It is a simple two-step heuristic, separating the assignment and the sequencing of the operations on the machines. We propose a new approach based on the largest tail priority rule. Since in the beginning, no operation is placed, the tail definition that is usually employed is not accurate since we cannot compute the longest path between a given node and the virtual node $*$ until all operations have been placed. Therefore, we only take into account the operations in the same job (predecessor of i in the job routing $p(i)$ and the follower of i in the job routing $f(i)$) and establish the initial tail as the sum of the smaller processing times of the subsequent operations in the job routing (followers). If operation i is the last in the routing of its associated job, then $f(i) = *$.

4.3. Handling uncertainty

Approximation of multivariate probability integrals is a hard problem in general (Szántai, 2009). As naturally emerges when dealing with probabilistic knowledge in stochastic programming (Homem-de-Mello and Bayraksan, 2014), let us estimate the service level of a given sequence S via a Monte Carlo sampling-based approximation procedure. Each operation with a stochastic processing time ξ is sampled according to its probability distribution a fixed number of times to generate a set of scenarios, denoted by Ω. Hence, every processing time in scenario $\omega \in \Omega$ is a realization of the corresponding random variable. The makespan of sequence S (where every operation i is assigned to and sequenced on machine $a(i)$) is thus computed for the processing times in each scenario ω, and denoted by $C_{max}(S, \omega)$. For a given threshold T and sequence S, the empirical service level is determined as follows:

$$\hat{\mathbb{P}}\big(C_{\max}(S, \xi) \le T\big) = \frac{1}{|\Omega|} \sum_{\omega \in \Omega} \mathbb{I}\big(C_{\max}(S, \omega) \le T\big) \tag{1}$$

where $\mathbb{I}(\bullet)$ represents the indicator function on set Ω, which takes 1 if $C_{\max}(S, \omega) \le T$, and 0 otherwise, $\forall \omega \in \Omega$.

In return for their simplicity, sampling-based approaches require a trade-off between the quality of estimation and the involved computational times. The literature related to probabilistic-constrained

problems provides statistics-based results indicating how large should be the sample size to guarantee, with a probability of at least $1-\beta$, that the solution provided by the scenario-based probabilistic problem remains feasible for the true problem with a probability at least $1-\epsilon$ (Campi and Calafiore, 2004; Calafiore and Campi, 2005; Luedtke and Ahmed, 2008). For the general case, Luedtke and Ahmed (2008) derived the following statistical estimates of the sampling size when probabilistic constraints involve random parameters on the right-hand side for finite and discrete probability distributions:

$$N \geq \frac{1}{2(\epsilon-\gamma)^2}\ln\frac{1}{\beta} + \frac{n}{2(\epsilon-\gamma)^2}\ln U \tag{2}$$

where U^n represents the size of the definition domain of the n-variate probability distribution.

To reduce the computational time, two strategies to select a subset of scenarios $\Omega' \subset \Omega$ are also presented and compared in (Flores Gómez et al., 2021): **(i)** A single random selection of subset Ω' when initializing the tabu search, and **(ii)** A random selection of subset Ω' at each iteration of the tabu search. Computational experiments show that no strategy dominates the other one. In what follows, the first strategy is arbitrarily used.

4.4. Maximizing the makespan service level

Solving the SFJSP, in our case, means finding a feasible sequence with the maximum service level. Relying on (Dauzère-Pérès and Paulli, 1997), we propose a tabu search approach to explore a large number of feasible sequences. An overview of the proposed approach can be found in Algorithm 1.

Algorithm 1 Given threshold T and set of scenarios Ω, maximize service level $\alpha(S,T,\Omega)$

1: Find initial sequence S_{init} and compute $\alpha(S_{init},T,\Omega)$ (see Section 4.2)
2: $S^* = S = S_{init}$, $it = 0$, $N_{limit} = 10,000$
3: **while** $(it < N_{limit})$ and $(\alpha(S,T,\Omega) < 1)$ **do**
4: Search neighborhood of S to find sequence S' with largest $\alpha(S',T,\Omega)$ (see Section 4.5)
5: **if** (multiple equivalent sequences) **then**
6: Select S' with smallest $\mathbb{E}(S',\Omega)$ (see Section 4.5)
7: **end if**
8: $S \leftarrow S'$
9: Compute $\alpha(S,T,\Omega)$
10: **if** $\alpha(S,T,\Omega) > \alpha(S^*,T,\Omega)$ **then**
11: $S^* = S$, $it = 0$
12: **else**
13: $it = it + 1$
14: **end if**
15: **end while**

Solution S_{init} referred in Step 1 is obtained by applying a fast constructive priority-rule-based heuristic (see Section 4.2) and the tabu search of Dauzère-Pérès and Paulli (1997) minimizing the makespan based on $\bar{p}$. Note that the tabu search of Dauzère-Pérès and Paulli (1997) uses a lower bound on the makespan to evaluate sequences during the search. We use a similar strategy to evaluate the makespan service level of every sequence in the neighborhood of a given sequence. The idea is to compute an upper bound on the makespan service level of such sequences as explained thoroughly in the next section.

The current sequence is replaced by the solution in the neighborhood with the largest estimated service level (Step 9 of Algorithm 1). Depending on the neighborhood and the quality of the move evaluation using set Ω, it is possible that no sequence S' has an estimated service level that is strictly positive or that more than one sequence is estimated to have a largest service level. When this happens, we propose to discriminate the sequences in the neighborhood by computing the mathematical expectation of a lower bound on the makespan detailed in the next section.

4.5. Estimating the makespan service level

In (Dauzère-Pérès and Paulli, 1997), the makespan of potential makespan-improving sequences is estimated using a lower bound ($LB_{C_{max}}$) on the makespan (see Theorem 5 in (Dauzère-Pérès and Paulli, 1997)). The sequences of the neighborhood are classified using an upper bound ($UB_{C_{max}}$) on the length of the critical path within the potential sequence by comparing it to the makespan of the current sequence. If $UB_{C_{max}} < C_{max}$, then the corresponding sequence is guaranteed to have a smaller makespan. In (Flores Gómez et al., 2021), we estimated the makespan service level of the sequences in the neighborhood by computing $LB_{C_{max}}(S,\omega), \forall\omega \in \Omega$, and counting the number of scenarios for which the lower bound is smaller than or equal to T. This provides an upper bound on the makespan service level of each sequence in the neighborhood:

$$UB_{\hat{\mathbb{P}}\left(C_{\max}(S,\xi)\leq T\right)} = \frac{1}{|\Omega|}\sum_{\omega\in\Omega}\mathbb{I}\left(LB_{C_{max}}(S,\omega)\leq T\right) \tag{3}$$

Since $LB_{C_{max}}(S,\bullet)$ is a lower bound of $C_{max}(S,\bullet)$, the true value of $C_{max}(S,\omega)$ could be larger than T even if $LB_{C_{max}}(S,\omega) \leq T$, $\forall\omega \in \Omega$. This could lead to rather poor estimations of the makespan service level. In addition to the upper bound of the makespan service level (3), we propose to apply the upper bound $UB_{C_{max}}(S,\bullet)$ on the makespan proposed in (Dauzère-Pérès and Paulli, 1997) (see Theorem 5 in Dauzère-Pérès and Paulli (1997)) to calculate a lower bound on the makespan service level, as follows:

$$LB_{\hat{\mathbb{P}}\left(C_{\max}(S,\xi)\leq T\right)} = \frac{1}{|\Omega|}\sum_{\omega\in\Omega}\mathbb{I}\left(UB_{C_{max}}(S,\omega)\leq T\right) \tag{4}$$

To guide the decision-making process described in Algorithm 1, the lower bound of the makespan service level (4) is used. When more than one sequence in the neighborhood has the largest estimated makespan service level (even if it is 0), the neighbor with the smallest $\mathbb{E}(S,\Omega) = \frac{1}{|\Omega|}\sum_{\omega\in\Omega} LB_{C_{max}}(S,\omega)$ is selected.

5. Generation of new instances

New instances for the SFJSP were required for our experiments. We extended the benchmark instances of Hurink et al. (1994) and Dauzère-Pérès and Paulli (1997), by considering the processing times of all operations in a given job $j \in \mathcal{J}$ as random variables ξ_{ik}. To do this, the following settings are used:

- The mean values $\mu = \bar{p}_{ik}, \forall i \in \mathcal{O}_j, \forall k \in \mathcal{M}_i$, where $\bar{p}_{ik}$ corresponds to the original processing time in the benchmark instances.
- The standard deviation $\sigma = 0.15\mu$, unless indicated otherwise.
- The finite definition interval $[c,d]$ of each random variable is defined asymmetrically compared to μ as follows: $c = \mu - 0.2\mu$, $d = \mu + 0.8\mu$. This setting allows distributions with long tails to be generated, where abnormal occurrences of processing times are far from the central (normal) part of the distribution.

In this paper, we consider that the random processing times follow the beta distribution (Marshall and Olkin, 2007), which depends on four parameters: c and d (limits of the definition domain), $a > 0$ (shape) and $b > 0$ (scale). The Probability Distribution Function (PDF) of the beta distribution is recalled below:

$$f_\beta(x|a,b,c,d) = \frac{(x-c)^{a-1}(d-x)^{b-1}}{B(a,b)(d-c)^{a+b-1}},$$

where $B(a,b)$ is the beta function defined as:

$$B(a,b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)},$$

Table 2
Characteristics of instances from Dauzère-Pérès and Paulli (1997).

Instance	$\|\mathcal{J}\|$	$\|\mathcal{M}\|$	Operations per job	$C_{max}(S_{init}, \bar{p})$
01a	10	5	[15,...,25]	2655
02a				2291
03a				2255
04a				2633
05a				2282
06a				2245
07a	15	8	[15,...,25]	2481
08a				2120
09a				2098
10a				2440
11a				2125
12a				2063
13a	20	10	[15,...,25]	2439
14a				2211
15a				2189
16a				2422
17a				2182
18a				2171

where Γ is the Gamma function ($\Gamma(1) = 1$ and $\Gamma(n) = (n-1)!, \forall n > 1$) and parameters a and b ($a, b > 0$) can be expressed as function of μ, σ, c, d as follows:

$$a(\mu', \sigma') = \frac{\mu'^2(1-\mu')}{\sigma'^2} - \mu',$$

$$b(\mu', \sigma', a) = \frac{\mu'(1-\mu')}{\sigma'^2} - 1 - a,$$

where $\mu' = (\mu - c)/d$ and $\sigma' = \sigma/d$.

By virtue of the definition of the beta distribution, and given the values of c, d and σ, parameters a and b can be negative depending on the value of μ. The beta PDF is only defined for $a > 0$ and $b > 0$. Whenever this case arises, we ensure the Beta distribution function is well-defined, by setting the parameters as follows: $a = 13, b = 26, c = max = \{1, \mu - 0.5\}, d = \mu + 2.5$.

Based on the continuous probability distributions described above, we consider that the true available probability distributions of stochastic processing times correspond to their discrete counterparts. To discretize the continuous probability distributions, a rounding-based discretization procedure of the definition interval is used. The set of scenarios Ω is generated for every job $j \in \mathcal{J}$ as described in Section 4.3.

The characteristics of the benchmark instances of Dauzère-Pérès and Paulli (1997) and Hurink et al. (1994) can be found in Tables 2 and 3. To construct the instances in Table 2, a probabilistic parameter $\ell \in \{0.1, 0.3, 0.5\}$ has been used to generate the list of machines capable to process a given operation. In other words, parameter ℓ corresponds to three levels of flexibility. Hence, instances 01a, 02a, and 03a are only differentiated by the value of ℓ. The instances in Table 3 consider different levels of flexibility associated with the average number of machines in the subsets $\mathcal{M}_i$. In the *edata* instances, few operations may be assigned to more than one machine whereas, in the *rdata* instances, this is the case for most operations. In the *vdata* instances, all operations can be assigned to more than one machine.

The value in Column "$C_{max}(S_{init}, \bar{p})$" is the makespan of sequence S_{init} determined as described in Sections 4.1–4.2, and $\bar{p}$ is the scenario defined by the processing times of the corresponding original instances. Independently of the experiments, sequence S_{init} is always the same for one particular instance as well as $C_{max}(S_{init}, \bar{p})$.

6. Computational experiments

Extensive numerical experiments have been conducted to investigate empirically the properties of the sequences obtained by maximizing the makespan service level in the framework of the SFJSP, and to evaluate the performance of the proposed solution approach. More specifically, the empirical study is performed as follows:

- The relevance of the makespan service level as an optimization criterion for the SFJSP is discussed compared to **(i)** Solving the FJSP instantiated with a number of reference scenarios (see Section 6.2), and **(ii)** Minimizing the expected makespan (see Section 6.3).
- Given the high computational cost induced by the Monte Carlo sampling-based approximation of uncertain processing times, several levers allowing its efficiency to be enhanced are investigated in Section 7.

In the computational experiments of this paper, a single job has random processing times. This is for example the case when a new product is introduced in a factory, and its processing times on the different machines are not stabilized yet. The algorithms were implemented in C++, and the experiments run on an INTEL® Core™ i7-9700 CPU @ 3.00 GHz with 32 RAM.

6.1. Design of experiments

Numerical experiments have been conducted according to the following experimental plan oriented towards two main directions:

- *Evaluating the relevance of maximizing service level*
  1. Run the tabu search approach as described in Sections 4.1–4.2 for every instance to determine an initial solution S_{init} based on $\bar{p}$.
  2. Randomly generate a set Ω_{5000} of 5,000 scenarios divided in 5 batches of 1000 scenarios Ω^b_{1000}, $b \in \{1, 2, \dots, 5\}$. Samples of realizations of the random variables are combined to create the set of equiprobable scenarios Ω.
  3. Run the tabu search approach as described in Sections 4.1–4.2 for reference scenarios $\omega_q, \forall q \in \{0, 0.5, 0.75, 0.95, 1\}$ (see Section 6.2). Evaluate $\alpha(S, T, \Omega_{5000})$ for each resulting sequence S.
  4. Determine sequence $S^{\mathbb{E}}$ that minimizes $\mathbb{E}(C_{max}(S))$ for a given subset Ω_{500} (Section 7.1) using the tabu search approach and analyze (see Section 6.3):
     (a) $\mathbb{E}(C_{max}(S_{init}))$ and $\mathbb{E}(C_{max}(S^{\mathbb{E}}))$,
     (b) $\alpha(S^*, T, \Omega_{5000})$ and $\alpha(S^{\mathbb{E}}, T, \Omega_{5000})$,
     (c) Computational times.
- *Monte Carlo sampling-based approximation: Hyper-parameter tuning and performance analysis*
  1. Determine S^* that maximizes $\alpha(S^*, T, \Omega_s)$ for $\Omega_{b'} \subseteq \Omega^1_{1000}$, $\forall b' \in \{250, 500, 750, 1000\}$ using Algorithm 1.
  2. Evaluate the impact of $|\Omega_{b'}|$ on (see Section 7.1):
     (a) The quality of the estimated makespan service level,
     (b) $\alpha(S_{init}, T, \Omega_s)$ and $\alpha(S^*, T, \Omega)$,
     (c) Computational times.
  3. Analyze the impact of the number of random variables per instance and of the size of $\Omega_{b'}$ (Section 7.1.3).
  4. Determine S^* that maximizes $\alpha(S^*, T, \Omega^b_{500}), \forall \Omega^b_{500} \subset \Omega$, $b \in \{1, \dots, 10\}$ using Algorithm 1, and analyze (Section 7.2):
     (a) The accuracy of the evaluation of $\alpha(S_{init}, T, \Omega^b_{500})$ for different subsets $\Omega^b_{500} \subset \Omega$,
     (b) $\alpha(S^*, T, \Omega^b_{500}), \forall \Omega^b_{500} \subset \Omega$.

6.2. Solving the FJSP based on reference scenarios

As introduced in Section 3.3, let ω_q be a reference scenario, where every random processing time is instantiated with a value in the

Table 3
Characteristics of instances from Hurink et al. (1994).

Instance	$\|\mathcal{J}\|$	$\|\mathcal{M}\|$	Operations per job	Average $\|\mathcal{M}_i\|$			$C_{max}(S_{init}, \bar{p})$		
				edata	rdata	vdata	edata	rdata	vdata
mt06	6	6	6	1.15	2	3	55	47	47
mt10	10	10	10	1.15	2	5	923	698	655
mt20	20	5	5	1.15	2	2.5	1157	1025	1023
la01							621	582	576
la02							662	537	532
la03	10	5	5	1.15	2	2.5	568	482	480
la04							590	509	506
la05							503	463	465
la06							833	802	800
la07							778	755	752
la08	15	5	5	1.15	2	2.5	848	769	767
la09							892	858	855
la10							866	807	806
la11							1118	1074	1074
la12							960	939	937
la13	20	5	5	1.15	2	2.5	1053	1040	1039
la14							1127	1073	1071
la15							1136	1093	1090
la16							918	717	717
la17							739	646	646
la18	10	10	10	1.15	2	5	871	669	663
la19							826	704	617
la20							875	756	756
la21							1089	873	819
la22							937	795	751
la23	15	10	10	1.15	2	5	993	882	834
la24							964	842	789
la25							985	820	764
la26							1195	1103	1064
la27							1271	1125	1097
la28	20	10	10	1.15	2	5	1240	1115	1078
la29							1243	1018	1006
la30							1299	1121	1081
la31							1616	1546	1526
la32							1735	1677	1663
la33	30	10	10	1.15	2	5	1616	1517	1504
la34							1664	1554	1541
la35							1741	1570	1554
la36							1234	1053	948
la37							1446	1109	986
la38	15	15	15	1.15	2	7.5	1228	979	943
la39							1266	1042	922
la40							1213	987	955

Table 4
Minimization of makespan for reference scenarios and instances from Dauzère-Pérès and Paulli (1997). $C_{max} = \frac{1}{|\mathcal{J}|}\sum_{\mathcal{J}} C_{max}(S,\omega), \alpha = \frac{1}{|\mathcal{J}|}\sum_{\mathcal{J}} \alpha(S,T,\Omega)$ (in %), $T = C_{max}(S_{init}, \bar{p})$.

Inst.	S_0, ω_0		$S_{0.5}, \omega_{0.5}$		$S_{0.75}, \omega_{0.75}$		$S_{0.95}, \omega_{0.95}$		S_1, ω_1		$S_{init}, \bar{p}$	
	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α
01a	2507.9	**100.0**	2646.1	90.0	2734.2	60.0	2789.4	35.0	2817.5	18.0	2655.0	90.0
02a	2228.8	66.0	2333.6	3.0	2396.2	0.0	2446.1	0.0	2463.7	0.0	2291.0	**80.0**
03a	2200.2	55.0	2310.1	0.0	2365.3	0.0	2414.8	0.0	2429.3	0.0	2255.0	**77.0**
04a	2527.4	**95.0**	2653.1	70.0	2727.5	20.0	2795.6	20.0	2826.7	10.0	2633.0	83.0
05a	2212.3	**77.0**	2322.1	0.0	2382.5	0.0	2433.8	0.0	2452.9	0.0	2282.0	**77.0**
06a	2180.3	**83.0**	2292.0	0.0	2346.9	0.0	2390.4	0.0	2405.7	0.0	2245.0	76.0
07a	2432.5	73.0	2528.4	13.0	2596.7	7.0	2641.1	0.0	2658.9	0.0	2481.0	**80.0**
08a	2079.9	43.0	2144.9	0.0	2187.5	0.0	2236.9	0.0	2259.3	0.0	2120.0	**79.0**
09a	2060.1	29.0	2129.1	0.0	2163.6	0.0	2214.4	0.0	2237.5	0.0	2098.0	**55.0**
10a	2478.6	6.0	2554.4	0.0	2615.6	0.0	2649.7	0.0	2694.5	0.0	2440.0	**79.0**
11a	2078.6	**68.0**	2148.3	0.0	2187.7	0.0	2223.8	0.0	2245.0	0.0	2125.0	61.0
12a	2032.3	25.0	2098.3	0.0	2134.3	0.0	2171.4	0.0	2190.1	0.0	2063.0	**61.0**
13a	2316.2	**100.0**	2378.6	**100.0**	2413.4	85.0	2444.3	75.0	2463.7	40.0	2439.0	85.0
14a	2179.0	25.0	2232.1	5.0	2261.6	0.0	2287.2	0.0	2302.0	0.0	2211.0	**64.0**
15a	2162.4	13.0	2214.2	0.0	2244.0	0.0	2272.2	0.0	2288.4	0.0	2189.0	**62.0**
16a	2319.6	**100.0**	2383.9	**100.0**	2408.8	90.0	2450.8	30.0	2460.2	20.0	2422.0	81.0
17a	2157.8	23.0	2213.2	0.0	2242.2	0.0	2264.6	0.0	2273.8	0.0	2182.0	**67.0**
18a	2143.1	23.0	2192.4	0.0	2220.7	0.0	2243.9	0.0	2255.4	0.0	2171.0	**63.0**

Table 5
Minimization of makespan for reference scenarios for some instances from Hurink et al. (1994). Three rows per job (*edata, rdata* and *vdata*). $C_{max} = \frac{1}{|\mathcal{J}|}\sum_{\mathcal{J}} C_{max}(S,\omega), \alpha = \frac{1}{|\mathcal{J}|}\sum_{\mathcal{J}} \alpha(S,T,\Omega)$ (in %), $T = C_{max}(S_{init},\bar{p})$.

Inst.	S_0,ω_0		$S_{0.5},\omega_{0.5}$		$S_{0.75},\omega_{0.75}$		$S_{0.95},\omega_{0.95}$		S_1,ω_1		$S_{init},\bar{p}$	
	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α
	53.2	**47.0**	57.3	40.0	60.5	25.0	63.5	16.0	66.0	14.0	55.0	45.0
mt06	46.5	0.0	50.3	46.0	52.7	17.0	55.5	17.0	59.0	0.0	47.0	**58.0**
	45.7	16.0	49.2	**83.0**	52.0	50.0	54.8	50.0	58.5	33.0	47.0	69.0
	893.4	73.0	930.1	69.0	964.6	10.0	1014.7	0.0	1025.3	0.0	923.0	**89.0**
mt10	681.3	58.0	738.0	10.0	819.2	0.0	897.7	0.0	923.8	0.0	698.0	**82.0**
	649.2	25.0	705.7	30.0	797.3	30.0	885.2	0.0	915.3	0.0	655.0	**95.0**
	1163.6	13.0	1185.9	5.0	1207.2	0.0	1218.3	0.0	1219.4	5.0	1157.0	**83.0**
mt20	1015.4	32.0	1040.6	0.0	1053.4	0.0	1063.0	0.0	1066.4	0.0	1025.0	**79.0**
	1014.4	22.0	1039.8	0.0	1052.6	0.0	1062.2	0.0	1065.4	0.0	1023.0	**63.0**
	598.1	**89.0**	628.6	59.0	649.7	28.0	670.3	19.0	676.8	9.0	621.0	80.0
la01	565.6	59.0	593.4	0.0	608.6	0.0	628.3	0.0	635.4	0.0	582.0	**87.0**
	562.8	52.0	591.3	0.0	607.9	0.0	626.3	0.0	633.2	0.0	576.0	**80.0**
	822.8	58.0	850.9	33.0	866.8	11.0	881.0	11.0	884.6	0.0	833.0	**71.0**
la06	791.3	35.0	818.1	0.0	831.0	0.0	840.9	0.0	844.6	0.0	802.0	**73.0**
	791.2	24.0	817.2	0.0	830.1	0.0	840.7	0.0	844.1	0.0	800.0	**64.0**
	1095.0	**93.0**	1122.6	72.0	1136.9	30.0	1149.2	5.0	1151.6	5.0	1118.0	79.0
la11	1062.7	40.0	1089.2	1.0	1102.8	0.0	1112.8	0.0	1116.0	0.0	1074.0	**67.0**
	1062.4	35.0	1089.4	0.0	1102.2	0.0	1112.8	0.0	1115.6	0.0	1074.0	**76.0**
	881.2	82.0	927.9	68.0	971.2	10.0	1022.1	10.0	1041.2	19.0	918.0	**84.0**
la16	714.3	5.0	770.6	10.0	854.2	0.0	935.9	0.0	961.7	0.0	717.0	**92.0**
	711.2	24.0	763.5	50.0	840.3	40.0	928.4	0.0	959.1	0.0	717.0	**94.0**
	1048.6	**91.0**	1084.5	80.0	1102.1	47.0	1132.6	27.0	1139.5	7.0	1089.0	85.0
la21	847.1	**83.0**	881.7	20.0	926.3	0.0	985.0	0.0	1003.3	0.0	873.0	73.0
	809.4	7.0	845.1	0.0	898.9	0.0	963.5	0.0	982.7	0.0	819.0	**78.0**
	1145.7	**98.0**	1175.1	95.0	1198.2	65.0	1210.8	55.0	1221.3	20.0	1195.0	75.0
la26	1078.8	69.0	1103.7	65.0	1118.2	5.0	1136.0	0.0	1145.1	0.0	1103.0	**74.0**
	1051.0	16.0	1076.6	0.0	1090.0	0.0	1111.4	0.0	1120.1	0.0	1064.0	**69.0**
	1561.9	**100.0**	1587.5	**100.0**	1601.8	97.0	1613.5	80.0	1615.4	63.0	1616.0	71.0
la31	1521.8	**80.0**	1546.3	67.0	1559.6	0.0	1569.3	0.0	1572.0	0.0	1546.0	67.0
	1514.6	12.0	1539.1	0.0	1551.7	0.0	1561.8	0.0	1564.4	0.0	1526.0	**57.0**
	1164.5	**99.0**	1221.1	80.0	1297.1	40.0	1402.8	13.0	1434.2	7.0	1234.0	91.0
la36	1030.0	54.0	1100.5	40.0	1228.0	0.0	1358.0	0.0	1402.6	0.0	1053.0	**79.0**
	946.2	2.0	1051.8	40.0	1204.4	0.0	1357.7	0.0	1402.6	0.0	948.0	**85.0**

range of its definition domain $[c,d]$ corresponding to $c+(d-c)q, q \in \{0, 0.5, 0.75, 0.95, 1\}$.

For every instance, the processing times of the operations from the job considered stochastic take the values in the reference scenarios $\omega_{\bullet}$, one job at a time. A sequence was obtained using the deterministic tabu search, and its makespan was computed based on the processing times specified by $\omega_{\bullet}$. The makespan service level was also estimated based on the set of scenarios $\Omega = \Omega_{5000}$ and $T = C_{max}(S_{init},\bar{p})$.

The average results on all jobs for all instances from Dauzère-Pérès and Paulli (1997) are presented in Table 4, and the results for some instances from Hurink et al. (1994) in Table 5. The average results for all instances are given in Tables B.15, B.16 and B.17 in the supplementary material. Tables 4 and 5 show the average makespan and the average makespan service level obtained for every reference scenario ω_q (columns "S_q,ω_q"), where S_q is the sequence obtained when using scenario ω_q. With the objective of avoiding bias due to the computation of the initial solution, for a given instance and a given job in that instance, the initial sequence is the same regardless of the scenario ω_q. The makespan and the average makespan service level of S_{init} are also given as reference in Column "$S_{init},\bar{p}$". The best service levels are in bold.

Tables 4 and 5 show that replacing random variables with reference scenarios does not guarantee to obtain a sequence with an acceptable service level. In particular, Column "S_1,ω_1" shows rather poor results for almost every instance, thus indicating that the use of the worst-case scenario is not adapted to absorb small shop floor perturbations impacting the processing times. This is also the case for Column "$S_{0.95},\omega_{0.95}$". Hence, being too conservative, the worst-case scenario leads to poor makespan service levels.

In Table 6, the values of the makespan and the makespan service level are presented for each level of flexibility (*edata, rdata, vdata*) and for every job in instance *mt10* from Hurink et al. (1994). In addition to the previous observations, note that no sequence has a strictly positive service level when the worst-case scenario ω_1 is considered (Column "S_1,ω_1"). This is most likely related to the fact that the upper bound in the definition interval of every random processing time is $d = \mu + 0.8\mu$ (i.e., 80 percent larger than the mean value). Although there is no direct correlation between the makespan and the makespan service level as shown Section 3.4, the minimization of the makespan based on large $p_i(\omega_1)$ is insensitive to sequences respecting T for scenarios $\omega \in \Omega$.

Let us focus our attention on scenarios $\omega_{0.5}$ and $\omega_{0.75}$, where every random processing time takes the values $\mu + 0.3\mu$ and $\mu + 0.55\mu$ respectively. There are cases for which the sequence has a strictly positive service level, in particular for the most flexible *vdata* instances. This observation illustrates that, given the instance and its flexibility, the operations of some jobs are more critical than others.

In the remaining computational experiments of the paper, Ω_{500} is used to estimate online the makespan service level, unless indicated otherwise. For the offline evaluation of the sequences, the whole set Ω is used. The estimation quality for each subset is presented in Table 12 in Section 7.1.3.

6.3. Maximizing service level versus minimizing mathematical expectation

A very common practice when considering stochasticity in scheduling problems is to minimize the mathematical expectation of a given criterion. This is because of the interesting properties of the expectation, but also because of the relatively easy process of adapting

Table 6
Minimization of makespan for reference scenarios for instance *mt*10 from Hurink et al. (1994). Three rows per job (*edata, rdata* and *vdata*). $C_{max} = C_{max}(S,\omega), \alpha = \alpha(S,T,\Omega), T = C_{max}(S_{init},\bar{p})$.

Job	S_0,ω_0		$S_{0.5},\omega_{0.5}$		$S_{0.75},\omega_{0.75}$		$S_{0.95},\omega_{0.95}$		S_1,ω_1		$S_{init},\bar{p}$	
	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α	C_{max}	α
	885.0	**99.0**	932.0	0.0	950.0	0.0	965.0	0.0	983.0	0.0	923.0	88.0
1	686.0	52.0	692.0	**100.0**	726.0	0.0	737.0	0.0	754.0	0.0	698.0	77.0
	655.0	42.0	655.0	**100.0**	655.0	**100.0**	682.0	0.0	706.0	0.0	655.0	**100.0**
	901.0	30.0	929.0	**99.0**	961.0	0.0	998.0	0.0	999.0	0.0	923.0	94.0
2	686.0	**95.0**	709.0	0.0	784.0	0.0	883.0	0.0	914.0	0.0	698.0	84.0
	655.0	3.0	660.0	0.0	784.0	0.0	883.0	0.0	914.0	0.0	655.0	**90.0**
	870.0	70.0	926.0	**100.0**	1007.0	0.0	1055.0	0.0	1076.0	0.0	923.0	94.0
3	655.0	57.0	769.0	0.0	875.0	0.0	987.0	0.0	1020.0	0.0	698.0	**90.0**
	655.0	36.0	737.0	0.0	875.0	0.0	987.0	0.0	1020.0	0.0	655.0	**98.0**
	897.0	83.0	953.0	**88.0**	1014.0	0.0	1138.0	0.0	1175.0	0.0	923.0	84.0
4	672.0	3.0	849.0	0.0	1009.0	0.0	1138.0	0.0	1175.0	0.0	698.0	**94.0**
	597.0	46.0	849.0	0.0	1009.0	0.0	1138.0	0.0	1175.0	0.0	655.0	**90.0**
	897.0	97.0	910.0	**100.0**	942.0	0.0	944.0	0.0	972.0	0.0	923.0	94.0
5	686.0	**96.0**	713.0	0.0	731.0	0.0	739.0	0.0	739.0	0.0	698.0	82.0
	655.0	80.0	655.0	**100.0**	655.0	**100.0**	678.0	0.0	702.0	0.0	655.0	94.0
	894.0	95.0	930.0	0.0	929.0	**98.0**	989.0	0.0	961.0	0.0	923.0	79.0
6	681.0	48.0	709.0	0.0	773.0	0.0	858.0	0.0	889.0	0.0	698.0	**70.0**
	655.0	0.0	677.0	0.0	762.0	0.0	858.0	0.0	889.0	0.0	655.0	**95.0**
	898.0	68.0	925.0	**100.0**	965.0	0.0	1000.0	0.0	998.0	0.0	923.0	82.0
7	686.0	**90.0**	705.0	0.0	716.0	0.0	729.0	0.0	744.0	0.0	698.0	82.0
	655.0	36.0	655.0	**100.0**	655.0	**100.0**	720.0	0.0	744.0	0.0	655.0	**100.0**
	899.0	91.0	920.0	**100.0**	947.0	0.0	996.0	0.0	1013.0	0.0	923.0	94.0
8	686.0	1.0	733.0	0.0	828.0	0.0	934.0	0.0	965.0	0.0	698.0	**82.0**
	655.0	7.0	697.0	0.0	828.0	0.0	934.0	0.0	965.0	0.0	655.0	**94.0**
	890.0	58.0	949.0	0.0	958.0	0.0	1066.0	0.0	1070.0	0.0	923.0	**95.0**
9	689.0	69.0	773.0	0.0	919.0	0.0	1036.0	0.0	1070.0	0.0	698.0	**88.0**
	655.0	1.0	773.0	0.0	919.0	0.0	1036.0	0.0	1070.0	0.0	655.0	**89.0**
	903.0	39.0	927.0	**99.0**	973.0	0.0	996.0	0.0	1006.0	0.0	923.0	85.0
10	686.0	73.0	728.0	0.0	831.0	0.0	936.0	0.0	968.0	0.0	698.0	**74.0**
	655.0	0.0	699.0	0.0	831.0	0.0	936.0	0.0	968.0	0.0	655.0	**97.0**

deterministic methods to solve the associated stochastic scheduling problem. In this section, we focus on the estimation of the mathematical expectation of the makespan of sequence S denoted by $\mathbb{E}(S,\Omega)$, which corresponds to $\mathbb{E}(C_{max}(S,\Omega)) = \frac{\sum_{\omega\in\Omega} C_{max}(S,\omega)}{|\Omega|}$.

Algorithm 2 Given set of scenarios Ω, minimize $\mathbb{E}(S,\Omega)$

```
Find initial sequence S_init and compute E(S_init, Ω)
S^E = S = S_init, it = 0, N_limit = 10,000
while it < N_limit do
    Classify the neighborhood of S as explained in (Dauzère-Pérès and Paulli, 1997)
    Choose non-tabu S' minimizing the estimation on E(S', Ω)
    S ← S'
    Compute E(S, Ω)
    if E(S, Ω) < E(S^E, Ω) then
        S^E = S, it = 0
    else
        it = it + 1
    end if
end while
```

A tabu search procedure that minimizes $\mathbb{E}(S,\Omega)$ is proposed in Algorithm 2. This algorithm was then executed using online the subset of scenarios Ω_{500}. Let $S^{\mathbb{E}}$ be the obtained solution by Algorithm 2 for every job in every instance. Let us compare the solutions obtained by maximizing the makespan service level and minimizing the expected makespan in terms of the following indicators:

- Mean mathematical expectation per instance: $\overline{\mathbb{E}}(S^{\mathbb{E}}) = \frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} \mathbb{E}(S^{\mathbb{E}},\Omega)$.
- Gap between initial and optimized sequences: $\Delta_{\mathbb{E}} = \frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} [\mathbb{E}(S_{init},\Omega) - \mathbb{E}(S^{\mathbb{E}},\Omega)]$. There is no need to apply the absolute value of the difference between $\mathbb{E}(S_{init},\Omega)$ and $\mathbb{E}(S^{\mathbb{E}},\Omega)$, since $\mathbb{E}(S_{init},\Omega) \geq \mathbb{E}(S^{\mathbb{E}},\Omega)$ by definition.
- Mean gap between the makespan service levels associated with S^* (Algorithm 1) and $S^{\mathbb{E}}$: $\Delta_1 = \frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} |\alpha(S^{\mathbb{E}},T,\Omega) - \alpha(S^*,T,\Omega)|$.
- Average computational time: $\overline{CPU} = \frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}}$ CPU.
- Average gap $\Delta_{CPU} = \frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} |\text{CPU}(S^*) - \text{CPU}(S^{\mathbb{E}})|$, where CPU($S^{\mathbb{E}}$) (resp. CPU($S^*$)) is the computational time of Algorithm 2 (resp. Algorithm 1).

Two counters are introduced in this section to track the following occurrences:

- $N_{\alpha(S^{\mathbb{E}},S^*)}$: Number of times per instance that $\alpha(S^{\mathbb{E}},T,\Omega) > \alpha(S^*,T,\Omega)$. Note that, even if the search process is not guided by the maximization of the makespan service level, the value of $\alpha(S^{\mathbb{E}},T,\Omega)$ can be larger than $\alpha(S^*,T,\Omega)$ for some jobs.
- $N_{CPU(S^{\mathbb{E}},S^*)}$: Number of times per instance that CPU($S^{\mathbb{E}}$) < CPU(S^*).

The results of the experiments for all studied instances are presented in Table 7 and in Tables B.18, B.19 and B.20 in the supplementary material. To better analyze the metrics, in particular $\Delta_{\mathbb{E}}$ and Δ_1, two columns are added to each table to point out the minimal and maximal values over the entire set of jobs, for $\mathbb{E}(S_{init},\Omega) - \mathbb{E}(S^{\mathbb{E}},\Omega)$ and $\alpha(S^{\mathbb{E}},T,\Omega) - \alpha(S^*,T,\Omega)$, respectively. For $\alpha(S^{\mathbb{E}},T,\Omega) - \alpha(S^*,T,\Omega)$, a negative value means that the makespan service level of S^* is larger than the makespan service level estimated for $S^{\mathbb{E}}$.

In columns "$\overline{\mathbb{E}}(S^{\mathbb{E}})$" of Table 7 (resp. Tables B.18, B.19 and B.20 in the supplementary material), we present the mean over all jobs of the sequences $S^{\mathbb{E}}$ determined via Algorithm 2. When compared to columns

Table 7
Minimization of $\mathbb{E}(S,\Omega_{500})$ for instances from Dauzère-Pérès and Paulli (1997). Δ_1 in percentage. $\overline{CPU}$ and Δ_{CPU} in seconds. $\mathbb{E}(S)=\mathbb{E}(S,\Omega)$, $\alpha(S)=\alpha(S,T,\Omega)$, $N_\alpha = N_{\alpha(S^{\mathbb{E}},S^*)}$, $N_{CPU} = N_{CPU(S^{\mathbb{E}},S^*)}$.

Inst.	$\lvert\mathcal{J}\rvert$	$\overline{\mathbb{E}}(S^{\mathbb{E}})$	$\Delta_{\mathbb{E}}$	$\mathbb{E}(S_{init})-\mathbb{E}(S^{\mathbb{E}})$		Δ_1	$\alpha(S^{\mathbb{E}})-\alpha(S^*)$		N_α	$\overline{CPU}$	Δ_{CPU}	N_{CPU}
				$min_{\mathcal{J}}$	$max_{\mathcal{J}}$		$min_{\mathcal{J}}$	$max_{\mathcal{J}}$				
01a	10	2641.7	2.0	0.0	9.0	5.8	−15.3	−0.7	0	92.9	80.8	10
02a		2286.6	0.2	0.0	0.5	19.1	−31.4	−5.9	0	463.9	242.7	1
03a		2252.8	0.2	0.0	0.5	22.1	−30.7	−10.3	0	496.4	167.2	2
04a		2624.1	1.5	0.0	8.7	6.4	−14.1	4.2	1	173.2	52.3	1
05a		2273.7	4.5	0.6	8.7	13.8	−21.9	−5.0	0	221.8	139.6	1
06a		2240.3	2.3	0.0	18.1	21.0	−35.4	−1.4	0	371.4	129.5	4
07a	15	2476.4	0.9	0.0	7.9	2.0	−8.1	13.4	1	331.8	129.4	0
08a		2118.3	0.2	0.0	0.4	0.5	−6.8	0.0	0	686.9	87.5	1
09a		2100.1	0.2	0.0	0.2	44.1	−53.6	−25.4	0	1148.4	496.6	9
10a		2437.9	0.1	0.0	0.2	12.8	−22.0	−0.9	0	330.5	66.4	3
11a		2126.7	0.1	0.0	0.5	38.2	−49.1	−29.8	0	758.4	436.5	2
12a		2064.6	0.1	0.0	0.3	37.4	−47.5	−22.9	0	2446.4	1285.2	0
13a	20	2436.4	0.1	0.0	0.2	1.6	−9.2	0.0	0	1264.7	797.4	0
14a		2211.7	0.5	0.0	7.9	25.7	−47.7	−5.6	0	3952.0	2519.9	0
15a		2190.5	0.2	0.0	0.1	33.1	−46.7	−21.5	0	8280.6	5869.2	0
16a		2412.5	7.3	4.4	11.5	8.7	−3.9	16.5	19	2069.5	1654.5	0
17a		2182.7	0.1	0.0	0.2	31.2	−51.7	−20.0	0	4547.8	2878.2	1
18a		2171.5	0.8	0.0	2.4	31.4	−45.3	−16.5	0	6179.3	4379.4	0

Table 8
Minimization of $\mathbb{E}(S,\Omega_{500})$ for instance $04a$ from Dauzère-Pérès and Paulli (1997). $\alpha(S)=\alpha(S,T,\Omega)$. $\alpha(S_{init})$ and $\alpha(S^{\mathbb{E}})$ in %. $T = C_{max}(S_{init},\bar{p}) = 2633$.

Job	$C_{max}(S^{\mathbb{E}},\bar{p})$	$\mathbb{E}(S_{init},\Omega)$	$\mathbb{E}(S^{\mathbb{E}},\Omega)$	$\alpha(S_{init})$	$\alpha(S^{\mathbb{E}})$	$\alpha(S^*)$	CPU (in seconds)
1	2636	2632.5	2630.8	69.0	74.0	**85.3**	162.7
2	2633	2626.6	2626.6	79.1	79.1	**88.6**	175.1
3	2633	2628.1	2628.1	89.2	89.2	**92.5**	180.2
4	2636	2618.3	2618.1	89.1	88.9	**91.1**	162.8
5	2633	2625.2	2625.2	85.4	85.4	**86.2**	178.7
6	2633	2629.6	2629.6	76.3	76.3	**85.6**	178.8
7	2620	2622.7	2613.9	86.1	**97.5**	93.3	204.6
8	2633	2628.7	2628.7	82.5	82.5	**96.7**	180.7
9	2636	2619.5	2619.4	89.9	89.4	**94.0**	164.0
10	2636	2623.3	2620.0	87.4	90.9	**95.1**	144.6

"$C_{max}(S_{init},\bar{p})$" in Tables 2 and 3, the values of $\mathbb{E}(S^{\mathbb{E}})$ are consistently smaller than the makespan of S_{init} computed based on $\bar{p}$. However, this does not systematically imply that $C_{max}(S^{\mathbb{E}},\bar{p}) < C_{max}(S_{init},\bar{p})$. The mathematical expectation $\mathbb{E}(S,\Omega)$ does not provide any direct indication about the value of $C_{max}(S,\bar{p})$. Being out of the scope of this study, $C_{max}(S,\bar{p})$ is not explicitly given in the tables, but a tendency is given by the columns corresponding to $\Delta_{\mathbb{E}}$ ("$\Delta_{\mathbb{E}}$") showing the mean improvement of the mathematical expectation between S_{init} and $S^{\mathbb{E}}$, as well as their maximal values. Let us take for example the fourth row of Table 7: The value of $\overline{\mathbb{E}}(C_{max}(S^{\mathbb{E}},\Omega)) = 2624.1$ is smaller than 2633 with an average variation $\Delta_{\mathbb{E}} = 1.5$. When job 10 is considered stochastic, the gap $\mathbb{E}(S_{init},\Omega) - \mathbb{E}(S^{\mathbb{E}},\Omega) = 3.3$ as shown in Table 8, but $C_{max}(S^{\mathbb{E}},\bar{p}) > C_{max}(S_{init},\bar{p})$. Furthermore, $\mathbb{E}(S_{init},\Omega) < C_{max}(S_{init},\bar{p})$ when job 10 is stochastic. As for the makespan and the makespan service level, the mathematical expectation of C_{max} is not correlated to the makespan.

$\overline{CPU}$, resp. Δ_{CPU}, indicates the average CPU time of Algorithm 2, resp. the mean absolute gap of the CPU time of Algorithm 2 compared to the CPU time of Algorithm 1. Following our experiments, we observed that the execution times of every iteration for both algorithms are equivalent on average. $N_{CPU(S^{\mathbb{E}},S^*)}$ also indicates the number of times per instance that CPU($S^{\mathbb{E}}$) < CPU(S^*). If $N_{CPU(S^{\mathbb{E}},S^*)}$ is close to the number of jobs, it means that Algorithm 2 does not perform well on that particular instance.

Performance indicator Δ_1 measures the mean absolute difference between the makespan service levels of sequences $S^{\mathbb{E}}$ and S^* determined using Algorithm 1. This helps to compare the effectiveness of both methods, in particular given the smoothing effect of the mean between normal and abnormal processing times. To provide a better understanding of the performance of Algorithms 1 and 2, two columns with the minimal and maximal values are added to each table ($min_{\mathcal{J}}$ and $\max_{\mathcal{J}}$), and $N_{\alpha(S^{\mathbb{E}},S^*)}$, the number of times that $\alpha(S^{\mathbb{E}},T,\Omega)$ is larger than $\alpha(S^*,T,\Omega)$. In other words, $N_{\alpha(S^{\mathbb{E}},S^*)}$ indicates how many times Algorithm 2 outperforms Algorithm 1 in terms of the makespan service level. Because of the fact that the goal of Algorithm 2 is not the maximization of α, $N_{\alpha(S^{\mathbb{E}},S^*)}$ represents a very small set of the jobs (regularly empty) for most instances. This phenomenon is also reflected by the values in columns "$max_{\mathcal{J}}$", which, when negative, presents a smaller makespan service level for $S^{\mathbb{E}}$ for all jobs in that particular instance. However, there are specific instances for which this is not the case, in particular for the least flexible instances (*edata* instances from Hurink et al. (1994) and 04a, 07a and 16a from Dauzère-Pérès and Paulli (1997)). Let us take for example the seventh row of Table 8, Algorithm 2 outperforms Algorithm 1, but there is a significant improvement gap between $\alpha(S^*)$ and $\alpha(S_{init})$. The fact that $\alpha(S^{\mathbb{E}}) - \alpha(S_{init})$ is large could be explained by an insufficient number of iterations of the tabu search. Furthermore, when Jobs 4 and 9 are stochastic, there is even a slight decrease in the makespan service level despite the improvement of the mathematical expectation. In terms of the relevance of the makespan service level, in the rare situations where $\alpha(S^{\mathbb{E}},T,\Omega) > \alpha(S^*,T,\Omega)$, there is a very high probability of completing all jobs before the deadline, which does not happen systematically when minimizing the mathematical expectation, as shown by the results in this section.

7. Monte Carlo sampling-based approximation: Hyper-parameter tuning and performance analysis

Monte Carlo based-methods are very general and have many practical applications, including in the field of stochastic programming

(Homem-de-Mello and Bayraksan, 2014). The principle behind Monte Carlo sampling-based approximation is relatively simple. For example, in the case of $\mathbb{E}(C_{max}(S))$, a crude Monte Carlo estimate is an average of $C_{max}(S,\omega)$, $\forall\omega \in \Omega$. The simplicity of Monte Carlo based-methods comes at an expensive price in terms of computational time. Since the makespan is a random variable in the context of the problem under study with a given threshold, it is a known result that the variance of this estimate is $\frac{1}{|\Omega|}\mathbb{V}(C_{max}(S))$ and the root mean square error is $O(|\Omega|^{-\frac{1}{2}})$, meaning an accuracy of ϵ requires $|\Omega| = O(\epsilon^{-2})$ scenarios (Hammersley and Handscomb, 1964; Giles, 2015). In other words, the smaller the error, the larger the set of scenarios required by the Monte Carlo sampling-based approximation.

This section is dedicated to discussing the levers allowing us to conciliate the efficiency and effectiveness of the Monte Carlo sampling-based approximation used in the tabu search approach. In Section 7.1, we first study to which extent the sample size of Ω can be reduced without degrading the solution quality. To enhance the competitiveness of the proposed approach, we analyze the sensitivity of the makespan service level estimation, by distinguishing the set of scenarios used to evaluate the quality of moves (*online* or *a priori* evaluation) from the set of scenarios used to check the quality of best-found solutions (*offline* or *a posteriori* evaluation) in Section 7.2.

7.1. Approximation error induced by reducing the sample size

One major drawback of Algorithm 1 is the time consumed to evaluate the service level of moves during the search, which depends on the size of the set of scenarios Ω. To study the impact of the sample size, several runs of Algorithm 1 were conducted with different subsets Ω_s of sizes $s \in \{250, 500, 750, 1000\}$. Every subset Ω_s contains the first s scenarios in set Ω, such that:

$$\Omega_{250} \subset \Omega_{500} \subset \Omega_{750} \subset \Omega_{1000} \subseteq \Omega^1_{1000} \subset \Omega$$

The estimation quality of each subset can be found in Table 12. In this section, let both the online and offline evaluation of candidate sequences be based on the same set $\Omega_s \subset \Omega$, and S^* be the sequence found by Algorithm 1 when one of the jobs in $\mathcal{J}$ is stochastic. Consider the following performance indicators provided in Table 9 and Tables B.21, B.22 and B.23 in the supplementary material for all instances:

- *Error of estimation* $\frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} |\alpha(S^*,T,\Omega) - \alpha(S^*,T,\Omega_s)|$: This performance indicator reflects the impact of reducing the sample size during the entire search process on average over the set of jobs against Ω. Note that the sample size plays a crucial role since, at every iteration of Algorithm 1, the whole neighborhood of the current sequence is evaluated to identify the move with the best-estimated makespan service level (see Section 4.4).
- *Improvement gap*: $\frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} [\alpha(S^*,T,\Omega) - \alpha(S_{init},T,\Omega_s)]$. This performance indicator measures the gap between the initial makespan service level $\alpha(S_{init},T,\Omega_s)$ (approximated on Ω_s) and the final makespan service level $\alpha(S^*,T,\Omega)$ (approximated on Ω). When this gap is negative (in the worst case), it is set to zero since $\alpha(S^*,T,\Omega) = \alpha(S_{init},T,\Omega)$, thus the relative value of the difference is sufficient.
- *Average computational times* $\overline{CPU}$ involved by Algorithm 1 over jobs in every instance. In Columns "$\overline{CPU}$" of Table 9 and Tables B.21, B.22 and B.23 in the supplementary material, aside from few exceptions, there is a clear correlation between the size of Ω_s and the computational time of Algorithm 1, as it takes less time to run when the smaller subsets are used to approximate the makespan service level.

7.1.1. On the error of estimation

Generally, the smaller $\frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} |\alpha(S^*,T,\Omega) - \alpha(S^*,T,\Omega_s)|$ the better. To be more specific, an estimation error close to 0 means that the performed reduction of the sample size does not influence the quality of the estimation of the makespan service level. As observed in Table 9 in Columns "$\frac{100\%}{|\mathcal{J}|}\sum_{\mathcal{J}} |\alpha(S^*,\Omega) - \alpha(S^*,\Omega_s)|$", this is never the case. In fact, since the size of Ω_s in three of four cases is smaller than the order of magnitude of Ω (10^3 versus 10^4), there is an additional significant digit in the evaluation based on the full set of scenarios, which almost guarantees a non-zero error. However, in general, the average absolute error is smaller than 6%, which could indicate that for all subsets Ω_s, there is a $\pm 3\%$ margin of error against Ω. This interval decreases with the increase of the reduced sample size, and thus the quality of estimation improves as shown in Fig. 6, where the dispersion of points in the box-plot tightens as the number of scenarios in subset Ω_s increases. In Fig. 6, 10 evaluations (one per job in instances $mt10-edata$ and $mt10-vdata$) are plotted per subset Ω_s. The dispersion of the points follows the general observation, both in terms of wideness with respect to the sample size and of the error margin for the less flexible instance, and mostly for the more flexible instance, with the exception of Ω_{500}.

7.1.2. On the improvement gap

Let us now focus on the improvement gap. The larger $\frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} [\alpha(S^*,T,\Omega) - \alpha(S_{init},T,\Omega_s)]$, the more the proposed solution approach is effective. A large value of service level maximization indicates that using the mean values of the processing times to find a sequence minimizing the makespan is not as effective for the SFJSP as maximizing the makespan service level. The improvement of $\alpha(S^*,T,\Omega)$ over $\alpha(S_{init},T,\Omega_s)$ is set to zero when negative. The results presented in Column "$\frac{100\%}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} [\alpha(S^*,T,\Omega) - \alpha(S_{init},T,\Omega_s)]$" of Table 9 and Tables B.21, B.22 and B.23 (supplementary material) show improvement for every considered sample size and every instance with the exception of instances $la14-edata$ and $la30-edata$ from Hurink et al. (1994). However, there is not a clear tendency related to the impact of the sample size on the improvement gap. As shown in Fig. 7, the improvement gap varies within a similar range for this particular instance ($[0, 10.4]$) $\forall\Omega_s$, which confirms the general observation. It is important to highlight that the estimation error is consistent with the improvement gap since the makespan service levels of S_{init} and S^* are not estimated using the same set of scenarios.

Let us pay particular attention to the approximation quality induced by set Ω_{500} used in Section 6. In Table 10 and Tables B.24, B.25 and B.26 (supplementary material), the performance indicators are calculated for subsets Ω_{500} and Ω_{1000}. The error margins and the maximal values of improvement are provided for all jobs.

The results show that the makespan service levels calculated based on Ω_{500} and Ω_{1000} are very similar. Although, in general, the values of $\alpha(S^*,T,\Omega) - \alpha(S^*,T,\Omega_s)$ are closer to 0 for Ω_{1000} than for Ω_{500} as shown in Columns "$min_{\mathcal{J}}$" and "$max_{\mathcal{J}}$", and confirmed in Columns "$\frac{100\%}{|\mathcal{J}|}\sum_{\mathcal{J}} |\alpha(S^*,\Omega) - \alpha(S^*,\Omega_s)|$", the difference between the results obtained with Ω_{500} and Ω_{1000} is rarely larger than 1%. Meanwhile, the average improvement in computational time is substantial for every instance as shown in Column "$\overline{CPU}$". The improvement gap is also similarly wide regardless of the size of the subset used during the search, both on average and for the maximal value, as indicated by Columns "$\frac{100\%}{|\mathcal{J}|}\sum_{\mathcal{J}} [\alpha(S^*,\Omega) - \alpha(S_{init},\Omega_s)]$" and "$max_{\mathcal{J}}$". Due to the number of experiments and the small difference in the results obtained when using these two subsets, the use of Ω_{500} to evaluate online the candidate sequence and of Ω to evaluate offline the final sequence are justified.

7.1.3. Quantifying the impact of the dimensionality of the uncertainty on the sample size

As indicated by the results in Tables 9 and 10 as well as in Tables B.21, B.22 B.23, B.24, B.25 and B.26 in the supplementary material, the performance of the proposed approach vary very differently from one

Table 9
Results for subsets of scenarios Ω_s, $s \in \{250, 500, 750, 1000\}$ for instances from Dauzère-Pérès and Paulli (1997). $\alpha(S, \Omega) = \alpha(S, T, \Omega)$.

Inst.	$\frac{100\%}{\lvert\mathcal{J}\rvert}\sum_{\mathcal{J}}\lvert\alpha(S^*,\Omega)-\alpha(S^*,\Omega_s)\rvert$				$\frac{100\%}{\lvert\mathcal{J}\rvert}\sum_{\mathcal{J}}[\alpha(S^*,\Omega)-\alpha(S_{init},\Omega_s)]$				$\overline{CPU}$ (s)			
	Ω_{250}	Ω_{500}	Ω_{750}	Ω_{1000}	Ω_{250}	Ω_{500}	Ω_{750}	Ω_{1000}	Ω_{250}	Ω_{500}	Ω_{750}	Ω_{1000}
01a	1.1	0.6	0.4	0.3	6.7	6.8	7.0	6.7	99.5	173.6	129.7	236.0
02a	0.7	0.6	0.4	0.2	19.2	19.2	19.4	19.6	93.3	241.9	349.8	680.4
03a	1.3	0.6	0.4	0.2	22.8	22.3	22.1	22.1	245.2	523.9	383.9	864.2
04a	2.1	1.4	0.7	0.6	8.6	7.3	8.7	8.6	107.2	132.1	168.8	219.8
05a	0.9	0.5	0.3	0.1	22.9	23.8	23.9	23.6	91.0	88.6	96.1	398.5
06a	0.7	0.7	0.3	0.2	23.8	22.8	23.7	23.4	376.4	360.4	568.7	1107.9
07a	2.2	1.6	1.1	1.0	0.3	0.5	2.3	0.9	154.3	202.3	211.8	268.5
08a	2.0	1.1	1.1	1.0	0.5	0.5	0.5	0.4	340.7	601.3	381.4	455.5
09a	1.6	0.6	0.3	0.3	44.9	45.5	44.7	45.1	644.6	1298.1	1239.1	1649.6
10a	1.2	1.0	0.8	0.6	13.1	13.0	12.5	12.8	163.3	314.7	230.8	266.8
11a	0.9	0.6	0.3	0.3	39.4	38.1	38.4	38.3	239.3	393.1	415.3	677.1
12a	1.3	0.7	0.5	0.4	37.6	37.7	37.5	37.2	720.8	1161.2	1103.6	1560.9
13a	1.5	0.9	0.7	0.6	0.7	1.7	0.8	0.6	275.3	467.3	378.1	473.5
14a	1.9	1.0	0.7	0.5	25.5	27.3	24.2	26.6	991.8	1432.1	1246.8	1591.7
15a	1.3	0.9	0.6	0.5	33.4	34.3	32.0	32.5	1615.6	2411.4	2393.4	2798.4
16a	2.1	1.1	1.1	0.8	5.0	6.0	4.8	5.4	274.6	415.1	360.1	436.5
17a	1.5	0.7	0.5	0.5	31.4	33.2	31.6	28.7	1093.9	1869.5	1919.4	2513.1
18a	1.5	0.9	0.6	0.4	35.2	36.2	35.5	35.6	1286.6	1799.9	2311.2	2674.0

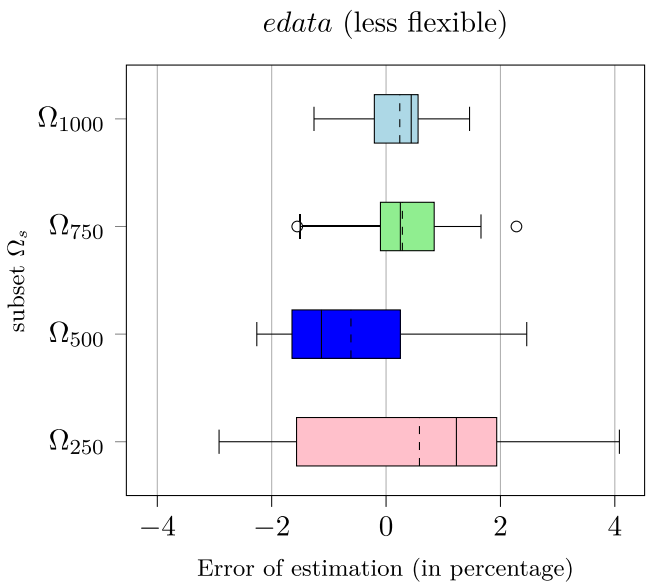
Table 10
Results for instances from Dauzère-Pérès and Paulli (1997): Ω_{500} *vs.* Ω_{1000}. $\alpha(S, \Omega) = \alpha(S, T, \Omega)$.

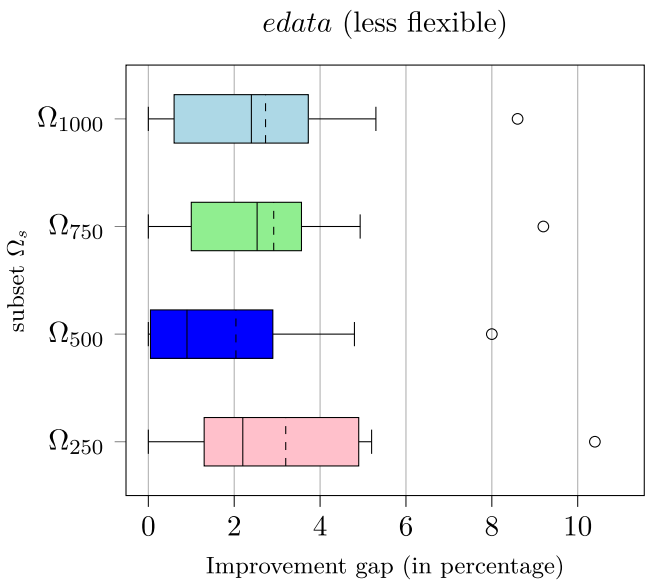
Inst.	$\frac{100\%}{\lvert\mathcal{J}\rvert}\sum_{\mathcal{J}}\lvert\alpha(S^*,\Omega)-\alpha(S^*,\Omega_s)\rvert$						$\frac{100\%}{\lvert\mathcal{J}\rvert}\sum_{\mathcal{J}}[\alpha(S^*,\Omega)-\alpha(S_{init},\Omega_s)]$				$\overline{CPU}$ (s)	
	Ω_{500}			Ω_{1000}			Ω_{500}		Ω_{1000}		Ω_{500}	Ω_{1000}
		$min_{\mathcal{J}}$	$\max_{\mathcal{J}}$		$min_{\mathcal{J}}$	$\max_{\mathcal{J}}$		$max_{\mathcal{J}}$		$max_{\mathcal{J}}$		
01a	0.5	−1.2	1.6	0.3	−0.5	1.0	6.5	16.0	6.4	15.7	158.2	215.2
02a	0.5	−1.4	0.5	0.2	−0.6	0.0	17.0	32.8	17.3	30.2	237.9	636.2
03a	0.5	−1.3	0.1	0.2	−0.5	0.1	19.1	32.4	19.1	32.1	465.2	710.6
04a	1.2	−2.5	2.2	0.5	−1.6	0.9	5.9	14.6	7.0	15.6	120.9	198.1
05a	0.5	−1.4	−0.0	0.1	−0.3	0.0	21.4	41.8	21.2	39.6	63.5	349.1
06a	0.6	−0.9	2.1	0.2	−0.4	0.2	21.4	35.6	21.8	35.0	345.7	969.3
07a	1.4	−2.2	3.9	0.9	−2.2	2.1	0.5	2.4	0.3	8.4	190.1	237.7
08a	1.0	−1.4	4.8	1.0	−2.1	2.3	0.5	7.8	0.4	6.6	578.4	426.2
09a	0.6	−1.0	0.1	0.3	−0.8	0.4	42.0	56.0	41.7	54.8	1199.1	1499.1
10a	0.8	−1.2	3.1	0.6	−1.3	1.4	12.1	21.4	11.8	21.6	297.3	249.0
11a	0.6	−1.1	−0.2	0.3	−0.5	−0.0	35.7	47.6	35.8	48.0	386.0	627.5
12a	0.7	−1.3	−0.2	0.4	−1.0	0.2	35.5	49.2	35.1	49.4	1042.3	1360.9
13a	0.9	−2.8	2.5	0.5	−1.5	1.5	1.1	11.4	0.6	3.0	461.6	451.4
14a	0.9	−4.6	1.0	0.4	−1.3	0.7	26.8	47.0	26.1	41.5	1324.2	1538.9
15a	0.9	−2.0	1.3	0.5	−1.4	0.7	32.0	48.4	30.1	46.2	2392.3	2580.8
16a	1.0	−2.3	2.9	0.8	−1.9	1.9	5.3	12.8	4.8	13.1	391.7	396.1
17a	0.7	−1.8	−0.0	0.4	−1.1	2.0	31.6	53.6	28.3	50.9	1809.8	2457.1
18a	0.9	−2.0	−0.3	0.3	−1.7	0.1	34.9	50.8	34.3	49.6	1786.4	2626.6

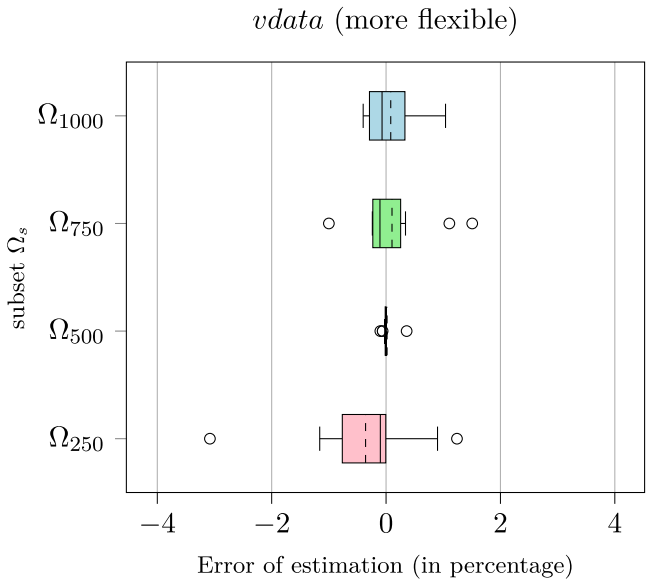
instance to another. This could be related to the fact that the number of considered random variables n is not the same for every instance, as it depends on the job that is stochastic. Also, and differently than in stochastic JSP, where there is a single machine per operation and thus only one random variable per operation, in the SFJSP the number of random variables depends on the considered level of flexibility. Therefore, a study on the impact of the sample size on the representativity of each random variable for each instance was conducted. The three different levels of flexibility were considered separately for instances from Dauzère-Pérès and Paulli (1997) (parameter $\ell \in \{0.1, 0.3, 0.5\}$) and from Hurink et al. (1994) (*edata, rdata, vdata*). For a given level of flexibility, all instances (one for each job and each instance) are sorted in non-increasing order according to the number of random variables: 90 per level of flexibility for instances from Dauzère-Pérès and Paulli (1997) and 711 per level of flexibility for instances from Hurink et al. (1994). The associated relationships are illustrated in Figures B.11 and B.12 (supplementary material).

Since the instances are sorted in non-increasing order of the number of random variables n, the number of scenarios per random variable decreases when n increases and the sample size drops regardless of the considered level of flexibility. However, the steepness of the evolution of the descent is more significant for the most flexible instances because, as stated earlier, a larger level of flexibility implies a larger number of processing times in the benchmark instances and thus a larger number of random variables in the instances. For example, for the jobs with more operations (left to right in the axis "Instances") in the least flexible instances ($\ell = 0.1$ and *edata*), the number of scenarios per random variable when $|\Omega_\bullet| = 500$ ($\Omega_\bullet \subset \Omega$) is between 150 and 200 (axis "$\frac{Nb\_scenarios}{Nb\_variables}$"). This number is divided by a factor of ten which is significant. This difference is even more extreme for the most flexible cases ($\ell = 0.5$ and *vdata*) since the number of scenarios per random variable decreases to between 40 and 50 when $|\Omega_\bullet| = 5000$ which is not the subset used in the search process, thus reducing significantly the representativity of each random variable in the sample size.

These observations are also confirmed by the impact of the sample size s on the approximation error of the makespan service level quantified by $\alpha(S_{init}, T, \Omega) - \alpha(S_{init}, T, \Omega_s)$, $\forall s \in \{100, 200, \ldots, 4800, 4900\}$, where Ω_s includes the first s scenarios of Ω. The empirical results for $\alpha(S_{init}, T, \Omega) - \alpha(S_{init}, T, \Omega_s)$ are presented in Fig. 8 depending on the number of scenarios s ("$|\Omega_s|$"), and for every instance (sorted in non-decreasing order of the number of random variables). The three different levels of flexibility were considered separately for instances from Dauzère-Pérès and Paulli (1997) ($\ell \in \{0.1, 0.3, 0.5\}$).







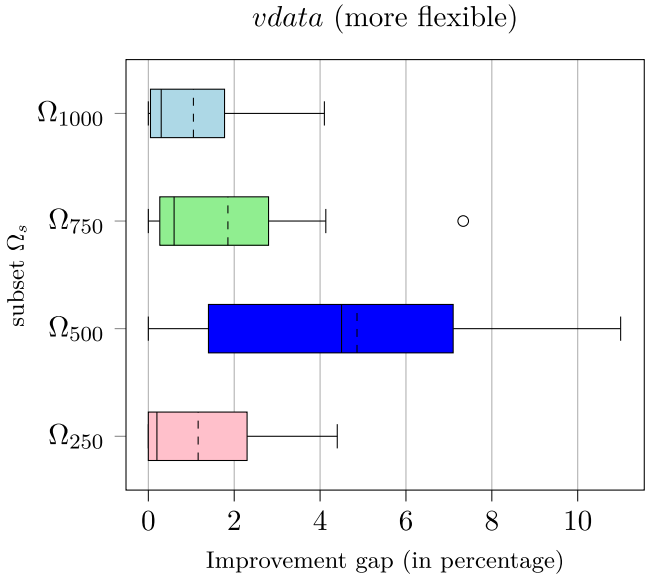


Fig. 6. Box-plot associated with the error of estimation (in %) for each subset Ω_s for instances $mt10-edata$ and $mt10-vdata$ from Hurink et al. (1994). One point per job.

Fig. 7. Box-plot representing the dispersion of values of the improvement gap (in %) for each subset Ω_s for instances $mt10-edata$ and $mt10-vdata$ from Hurink et al. (1994). One point per job.

Consider the plot corresponding to the least flexible instances ($\ell = 0.1$) in Fig. 8. This figure shows that the gap is quite large for the small subsets Ω_s for all instances in general, but the ones with more random variables (left to right in the axis "Instances") present larger estimation errors of $\alpha(S_{init})$. This observation also applies to the other levels of flexibility. Since more flexibility implies more random variables per instance, there is a clear difference in the convergence of the approximation error that requires more scenarios for more flexible instances.

The number of random variables per instance is equal to the sum of the size of subsets of the machines capable to execute each operation ($\sum_{i \in O_j} |\mathcal{M}_i|$) for every operation in a given job j. It depends on the level of flexibility of the instance. However, for the evaluation of the makespan service level of a particular sequence, only one random variable per operation is considered, since every operation i is already assigned to a machine in $\mathcal{M}_i$ as in a classical JSP. An overview of the number of random variables per instance is presented in Table 11, after the assignment of the operations to the machines, ranging between 5 and 25 for all instances for all flexibility levels. During the exploration of the neighborhood, the number of random variables after the machine assignment to operations: **(i)** is the same when evaluating a sequence, where the moved operation stays in the same machine, and **(ii)** increases by one when evaluating a sequence, where the moved operation changes from one machine to another. In this case, $\gamma = 0$ in Eq. (2) since it is not a known value. The approximation quality of the makespan service level is calculated via Eq. (5) as proposed by Luedtke and Ahmed (2008) and presented in Table 12 for different sample sizes and $\forall n \in \{5, 6, \ldots, 26\}$ with $\beta = 0.01$.

$$\epsilon \geq \sqrt{\frac{1}{2N}\left(\ln\frac{1}{\beta} + n \ln U\right)} \tag{5}$$

7.2. Approximation error induced by a leave-one-out online–offline evaluation scheme

As previously specified, the uncertainty in our approach is represented via a set of randomly generated scenarios. To assess the online

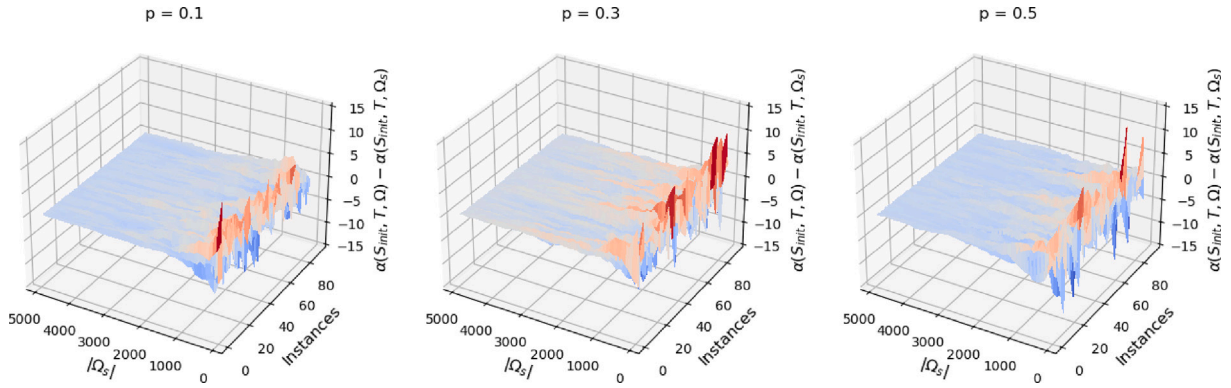


Fig. 8. Estimation error $\alpha(S_{init}, T, \Omega) - \alpha(S_{init}, T, \Omega_s)$ depending on the sample size and the number of random variables for instances from Dauzère-Pérès and Paulli (1997).

Table 11
Number of random variables per flexibility level before and after the assignment of operations to machines.

	Hurink et al. (1994)		Dauzère-Pérès and Paulli (1997)	
	FJSP	JSP	FJSP	JSP
Low flexibility	[5,20]	[5,15]	[15,34]	[15,25]
Medium flexibility	[6,36]	[5,15]	[22,74]	[15,25]
High flexibility	[8,114]	[5,15]	[33,125]	[15,25]

evaluation of the makespan service level, 10 runs of Algorithm 1 are conducted using 10 mutually-exclusive subsets $\Omega^b_{500} \subset \Omega$, $b \in \{1, \dots, 10\}$. Each subset Ω^b_{500} represents the bth batch of 500 scenarios in Ω. Since $|\Omega| = 5000$, there are 10 such subsets in Ω. Set Ω_{500} used in the previous sections corresponds to Ω^1_{500}. The final sequence obtained by Algorithm 1 with Ω^b_{500} is denoted by $S^*_{\Omega^b_{500}}$. To cross-evaluate the impact of the sampling set used online in Algorithm 1, the following performance indicators are considered:

- *Estimation error of S_{init} induced by the leave-one-out evaluation scheme* $\frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} |\alpha(S_{init}, T, \Omega) - \alpha(S_{init}, T, \Omega^b_{500})|$,
- *Improvement gap* $\frac{1}{|\mathcal{J}|}\sum_{job\in\mathcal{J}} [\alpha(S^*_{\Omega^b_{500}}, T, \Omega) - \alpha(S_{init}, T, \Omega)]$ based on an online–offline move evaluation scheme. The offline makespan service level estimation is done using the whole set Ω.

The aforementioned indicators are presented in Tables 13–14 and in Tables B.27–B.32 in the supplementary material.

In general, the average estimation error induced by the leave-one-out evaluation scheme of S_{init} stay in a similar range for all subsets Ω^b_{500}, as shown in Table 13, and Tables B.27, B.28 and B.29 (supplementary material). Note that the mean compensates for extreme values for certain instances. In this sense, consider the dispersion of $\alpha(S_{init}, T, \Omega) - \alpha(S_{init}, T, \Omega^b_{500}), \forall b \in \{1, \dots, 10\}$ for instance $mt10 - edata$ from Hurink et al. (1994) in Fig. 10, which varies between −3% and 3%. The only exceptions correspond to Ω^1_{500} and Ω^9_{500} for $mt10 - edata$. The evaluation of $\alpha(S_{init})$ based on these sets is ±4% far compared to Ω, at least for one job. The random generation of scenarios is evidenced by the different values of the gaps. However, the range of the dispersion is very consistent regardless of the considered subset. In Columns "$min_{\mathcal{J}}, b$" and "$max_{\mathcal{J}}, b$", the extreme estimation errors found for all batches for all jobs are shown jointly with the corresponding (different) batches b. Although these values show systematically a large range, they seldom correspond to the same batch, meaning the dispersion for every subset Ω^b_{500} is tighter than $[min_{\mathcal{J}}, max_{\mathcal{J}}]$.

The results based on the online–offline evaluation scheme are presented in Table 14 and in Tables B.30, B.31 and B.32 (supplementary material). Column "$min_{\mathcal{J}}, b$" (resp. "$max_{\mathcal{J}}, b$") presents the smallest (resp. largest) improvement $\forall b \in \{1, \dots, 10\}$ and the associated argument (batch b) per job and instance. If the minimum improvement gap is equal to 0, batch b in Column "$min_{\mathcal{J}}, b$" corresponds to the first batch detected. The more the flexibility level (i.e., parameter ℓ) of instances, the more the improvement gap as shown in Table 14. Only instances 07a, 10a, 13a, and 16a ($\ell = 0.1$) present $min_{\mathcal{J}}, b = 0$. Furthermore,

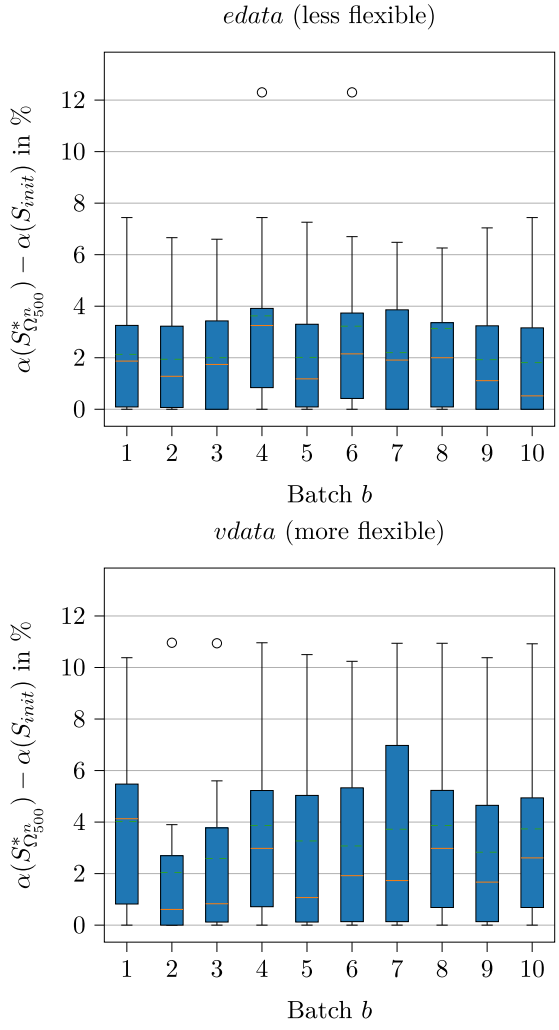


Fig. 9. Box-plot associated with the improvement gap (in %) for subset $\Omega^b_{500} \subset \Omega, b \in \{1, \dots, 10\}$ for instances $mt10 - edata$ and $mt10 - vdata$ from Hurink et al. (1994). One point per job. $\alpha(S, \Omega) = \alpha(S, T, \Omega), T = \{923, 653\}$.

Table 12
Estimation quality for every subset Ω_s depending on the number of random variables and sample size s.

# Random variables	5	6	7	8	9	10	11	12	13	14	15
Ω_{250}	0.25	0.27	0.29	0.31	0.33	0.35	0.36	0.38	0.39	0.40	0.42
Ω_{500}	0.19	0.20	0.22	0.23	0.25	0.26	0.27	0.28	0.29	0.30	0.31
Ω_{750}	0.16	0.17	0.18	0.20	0.21	0.22	0.23	0.24	0.25	0.25	0.26
Ω_{1000}	0.14	0.15	0.16	0.17	0.18	0.19	0.20	0.21	0.22	0.23	0.23
Ω	0.07	0.07	0.08	0.09	0.09	0.09	0.10	0.10	0.11	0.11	0.12
# Random variables	16	17	18	19	20	21	22	23	24	25	26
Ω_{250}	0.43	0.44	0.46	0.47	0.48	0.49	0.50	0.51	0.52	0.53	0.54
Ω_{500}	0.32	0.33	0.34	0.35	0.36	0.37	0.38	0.38	0.39	0.40	0.41
Ω_{750}	0.27	0.28	0.29	0.29	0.30	0.31	0.32	0.32	0.33	0.34	0.34
Ω_{1000}	0.24	0.25	0.25	0.26	0.27	0.27	0.28	0.29	0.29	0.30	0.30
Ω	0.12	0.12	0.13	0.13	0.13	0.14	0.14	0.14	0.14	0.15	0.15

Table 13
Estimation error of S_{init} induced by leave-one-out online evaluation with $\Omega^b_{500} \subset \Omega, \forall b \in \{1,\dots,10\}$ against Ω for instances from Dauzère-Pérès and Paulli (1997). $\alpha(\Omega) = \alpha(S_{init}, T, \Omega)$.

Inst.	$\frac{100\%}{\lvert\mathcal{J}\rvert}\sum_{\mathcal{J}} \lvert\alpha(S_{init},T,\Omega) - \alpha(S_{init},T,\Omega^b_{500})\rvert$										$\alpha(\Omega) - \alpha(\Omega^b_{500})$ (%)	
	$b=1$	$b=2$	$b=3$	$b=4$	$b=5$	$b=6$	$b=7$	$b=8$	$b=9$	$b=10$	$min_{\mathcal{J}}, b$	$max_{\mathcal{J}}, b$
01a	1.3	0.7	1.4	1.4	1.2	1.3	1.1	1.0	0.9	1.0	−5.1, 4	3.7, 3
02a	1.4	1.5	1.3	1.0	1.7	1.6	1.2	1.1	1.4	1.6	−3.9, 10	6.4, 5
03a	1.6	1.7	1.3	1.8	1.9	1.6	2.1	1.0	1.3	1.3	−3.5, 2	4.9, 1
04a	1.2	1.0	1.6	1.2	1.2	0.7	1.3	1.1	0.8	1.1	−3.2, 3	4.2, 1
05a	1.9	0.8	2.5	1.4	1.1	1.4	1.5	2.0	1.2	1.9	−5.2, 6	5.0, 8
06a	1.8	1.3	1.3	1.3	0.6	1.2	1.7	1.1	1.2	1.2	−3.9, 1	4.2, 3
07a	1.5	1.4	1.1	1.4	1.7	1.5	1.8	1.3	1.0	1.5	−6.2, 7	4.4, 5
08a	1.2	1.4	1.2	0.9	2.0	1.3	1.3	0.9	1.7	1.2	−4.9, 5	4.1, 7
09a	2.2	1.6	1.4	1.6	1.8	1.8	1.8	1.6	1.6	1.9	−6.5, 6	5.6, 8
10a	1.7	1.2	1.3	1.3	1.7	1.6	1.7	1.5	1.5	1.1	−5.0, 7	4.4, 8
11a	1.6	1.6	1.4	2.0	1.4	1.9	1.5	1.3	1.4	1.8	−4.8, 10	4.4, 2
12a	1.5	1.9	2.1	1.8	1.4	1.5	1.7	1.8	1.5	1.8	−5.9, 3	5.8, 9
13a	1.0	1.5	1.3	0.8	0.7	1.3	1.1	1.3	1.1	1.7	−3.8, 8	3.9, 10
14a	1.8	1.0	1.4	1.9	1.1	1.9	1.9	1.8	1.5	1.4	−5.4, 6	5.0, 6
15a	1.6	1.7	1.6	1.5	1.6	1.7	1.5	1.5	1.6	1.5	−4.2, 3	5.2, 9
16a	1.5	1.0	1.5	1.0	1.4	1.3	1.2	1.0	0.9	1.3	−4.2, 3	3.5, 1
17a	1.8	1.8	2.2	1.3	1.5	1.3	1.4	1.6	1.4	1.4	−4.4, 3	3.9, 2
18a	1.7	1.7	1.4	1.6	2.0	1.9	1.8	1.6	2.0	1.9	−5.9, 5	5.0, 5

Table 14
Improvement gap between initial and optimized sequences based on subsets $\Omega^b_{500} \subset \Omega$, $b \in \{1,\dots,10\}$ for instances from Dauzère-Pérès and Paulli (1997). $\alpha(S) = \alpha(S,T,\Omega)$.

Inst.	$\frac{100\%}{\lvert\mathcal{J}\rvert}\sum_{\mathcal{J}}[\alpha(S^*_{\Omega^b_{500}},T,\Omega) - \alpha(S_{init},T,\Omega)]$										$\alpha(S^*_{\Omega^b_{500}}) - \alpha(S_{init})$ (%)	
	$b=1$	$b=2$	$b=3$	$b=4$	$b=5$	$b=6$	$b=7$	$b=8$	$b=9$	$b=10$	$min_{\mathcal{J}}, b$	$max_{\mathcal{J}}, b$
01a	7.6	7.2	7.4	7.9	6.9	6.8	7.8	8.0	7.5	8.0	0.5, 1	16.5, 7
02a	17.6	16.6	17.4	18.0	18.1	18.1	17.4	17.4	18.2	18.1	3.7, 5	31.2, 10
03a	20.0	20.1	20.1	20.3	20.0	20.3	20.1	20.2	20.2	20.2	7.7, 7	29.8, 4
04a	6.9	9.3	8.0	8.0	8.0	8.0	8.0	6.9	8.0	6.5	0.5, 1	30.1, 2
05a	19.2	19.8	19.9	19.2	19.6	19.9	19.5	19.8	21.2	20.6	3.0, 7	35.7, 1
06a	21.5	21.7	21.6	21.7	21.0	21.6	21.5	21.7	21.1	21.2	11.2, 10	34.7, 3
07a	0.3	0.6	1.2	0.5	0.2	0.6	1.1	0.7	0.6	0.3	0.0, 1	8.7, 3
08a	15.3	16.3	15.7	17.0	15.7	17.2	16.0	16.3	17.1	16.4	4.4, 1	24.9, 6
09a	39.7	39.3	39.7	39.6	39.6	39.5	39.7	39.4	39.4	39.5	21.7, 5	52.2, 2
10a	11.8	11.9	11.3	11.8	11.9	11.4	11.9	11.5	11.7	11.7	0.0, 2	24.2, 3
11a	36.1	36.1	36.2	35.9	36.2	36.2	36.0	36.2	36.2	36.0	27.1, 10	45.3, 10
12a	35.6	35.6	35.3	35.4	35.3	35.5	35.5	35.6	35.7	35.5	21.2, 5	45.8, 5
13a	0.7	0.6	0.1	0.8	0.6	0.5	0.4	0.6	0.7	0.7	0.0, 1	5.2, 4
14a	25.6	25.0	25.4	24.1	22.4	23.7	22.3	25.9	21.9	25.9	3.8, 9	44.6, 6
15a	31.7	32.3	32.3	33.2	31.4	30.6	32.9	32.9	33.7	31.2	11.4, 8	44.8, 1
16a	6.2	7.5	6.3	7.5	6.4	7.4	7.0	7.4	6.4	7.0	0.0, 1	22.7, 7
17a	28.0	27.5	27.5	25.7	27.7	28.1	27.8	26.6	26.4	27.1	10.6, 9	40.9, 7
18a	33.7	33.5	33.6	33.7	33.5	33.5	33.6	33.4	33.6	33.7	19.8, 4	46.0, 2

the values in column "$max_{\mathcal{J}}, b$" are always larger for the instances with $\ell = 0.3$ and $\ell = 0.5$. The improvement gap is very consistent on average regardless of the subset Ω^b_{500} as shown in Fig. 9 regardless of the level of flexibility, where $\alpha(S^*_{\Omega^b_{500}}, T, \Omega) - \alpha(S_{init}, T, \Omega)$ is plotted for all batches for all jobs for instances $mt10{-}edata$ and $mt10{-}vdata$. The improvement is between 0% and 8%, except for $b = 4$ and $b = 6$ for $mt10 - edata$, and between 0% and 12% in the case of $mt10 - vdata$.

8. Conclusions and perspectives

In this paper, we maximize the makespan service level in the stochastic flexible job-shop scheduling problem. The relevance of the studied optimization criterion is explicitly shown compared to **(i)** the minimization of the expected makespan, and **(ii)** the minimization

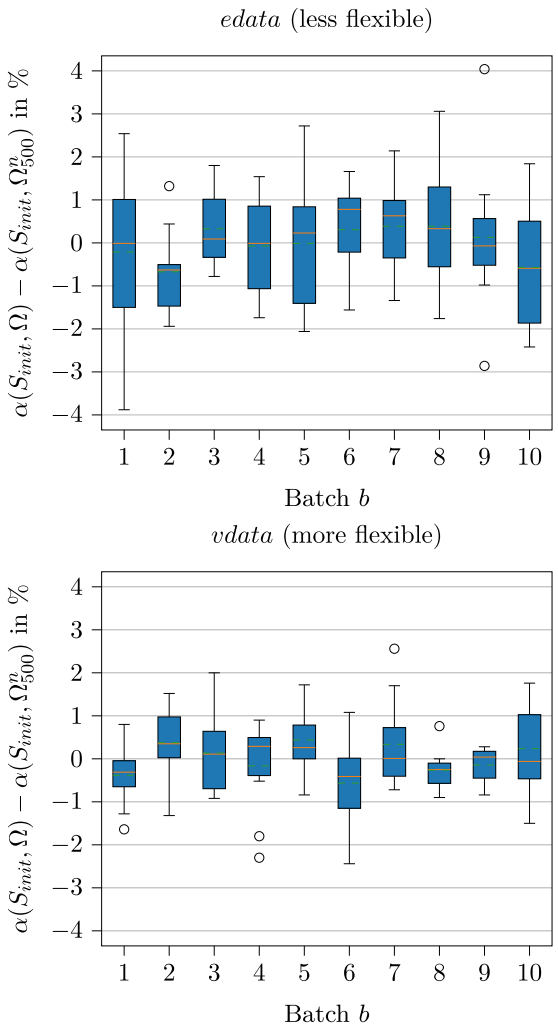


Fig. 10. Box-plot associated with the estimation error (in %) for subset $\Omega^b_{500} \subset \Omega$, $\forall b \in \{1,\dots,10\}$ for instances $mt10-edata$ and $mt10-vdata$ from Hurink et al. (1994). One point per job. $\alpha(S,\Omega)=\alpha(S,T,\Omega), T=\{923,653\}$.

of the makespan for several reference scenarios. To solve the considered optimization problems, a tabu search approach is combined with a Monte Carlo sampling-based approximation when dealing with uncertainty. New randomly generated instances are proposed for the stochastic flexible job-shop scheduling problem. To overcome the expensive computational cost specific to Monte-Carlo approximations, extensive numerical experiments have been conducted to study the impact of critical hyper-parameters while keeping particular attention to the quality of the proposed sequences. The main managerial implications are:

- Solving the SFJSP by approximating the random processing times with reference statistical summaries (min, max, mean) does not necessarily lead to a sequence with an acceptable service level.
- Considering the worst-case scenario (occurring under abnormal conditions) is not adapted to absorb small shop-floor perturbations impacting the processing times. Being too conservative, the worst-case scenario leads to poor makespan service levels.
- Given the smoothing effect of the mean, minimizing the expected makespan may lead to over-conservative sequences with respect to processing times under normal operating conditions, in particular in the case of long-tail frequency distributions. This does not necessarily lead to deadline-compatible sequences.
- As long as the deadline is consistent with the processing of jobs under normal operating conditions, maximizing the makespan service level leads to deadline-compatible solutions while neglecting the tails of the frequency distributions of processing times.

The following perspectives are being investigated. Even if a study on the dimensionality of the uncertainty and its impact depending on the instance flexibility was presented for a fixed number of scenarios for all instances, other advanced sampling-based techniques could be applied to represent uncertainties accurately. The random fashion in which the scenarios are generated is being replaced by a more efficient generation based on the discretization of all random variables and the comparability of scenarios. This should lead to new approaches for the optimization of the makespan service level while integrating some properties of the problem at hand. Although an important goal of this paper is to show the relevance of the notion of makespan service level in stochastic scheduling, it would be interesting to compare the presented approach with other known solution approaches such as simulated annealing and genetic algorithms. The notion of critical jobs in the stochastic version of the FJSP is also an interesting subject, as it could help to design more efficient methods and from a practical standpoint, could lead to more robust schedules. Similar problems to the one presented in this paper are also being targeted, for example, the consideration of a resource as the source of uncertainty (new machine in the workshop) rather than the operations of a given job. Also, minimizing the value of T given a minimum makespan service level, as in the work of Beck and Wilson (2007), could be interesting to study. Finally, we intend to explore other service levels such as the tardiness or the weighted sum of the tardiness, by relying on the works in (Mati et al., 2011) and (García-León et al., 2015).

CRediT authorship contribution statement

Mario Flores-Gómez: Investigation, Conceptualization, Methodology, Software, Writing – review & editing. **Valeria Borodin:** Investigation, Conceptualization, Methodology, Supervision, Software, Writing – review & editing, Validation. **Stéphane Dauzère-Pérès:** Investigation, Conceptualization, Methodology, Supervision, Writing – review & editing, Validation, Funding acquisition.

Data availability

Data will be made available on request.

Acknowledgments

This work was partly funded by the French Public Authorities through the Nano 2022 program, which is part of IPCEI (Important Project of Common European Interest).

Appendix A. Nomenclature

$\mathcal{J}$	Set of jobs
$\mathcal{M}$	Set of available machines k
$\mathcal{O}=\cup_{j\in\mathcal{J}}\mathcal{O}_\mathcal{J}$	Set of operations partitioned into a set of jobs $\mathcal{J}$
$\mathcal{M}_i$	Subset of machines capable of executing operation i
$\bar{p}_{ik}$	Deterministic processing time of operation i on machine k

N	Set of nodes. One by operation in set $\mathcal{O} \cup \{0, *\}$
A	Set of conjunctive arcs modeling the routing within jobs
E	Set of disjunctive arcs modeling the potential sequencing of the operations such that $E = \cup_{k \in \mathcal{M}} E^k$
$G = (N, A, E)$	Disjunctive graph presented in Section 4.1
S	Feasible selection of conjunctive arcs from E representing a feasible sequence
$G' = (N, A, S)$	Conjunctive graph presented in Section 4.1
$L(i, j)$	Longest path between operations i and j
r_i	Release date (head) of operation i such that $r_i = L(0, i)$
q_i	Delivery time (tail) of i such that $q_i = L(i, *) - p_{ik'}$ with i assigned to k'
S_{init}	Initial sequence for tabu search
S^*	Final sequence found by a given solution approach
$S^{\mathbb{E}}$	Sequence maximizing $\mathbb{E}(C_{max}(S))$
ξ_{ik}	Random processing time of operation i on machine k
ξ	Multivariate random variable of dimension n
a, b	Shape parameters for the beta probability distribution. $a > 0, b > 0$
$[c_{ik}, d_{ik}]$	Definition interval for the four parameter beta probability law
$f_\beta(x\|a, b, c, d)$	Four parameter beta probability law
μ, σ	Mean and standard deviation of the considered probability distribution
Ω	Set of scenarios ω as described in Section 6.1
$p_{ik}(\omega)$	Realization of p_{ik} in scenario ω
$\bar{p}$	Scenario in which every random processing time is its mean value μ
$C_{max}(S, \omega)$	Makespan of sequence S for processing times from $\omega \in \Omega$
ω_q	Reference scenarios, where random processing times take the value of $c + (d - c) \cdot q$
$\Omega_s \subset \Omega$	Subset of scenarios containing the first s scenarios in Ω
Ω^b_{5000}	bth subset of Ω of 500 scenarios, $b \in \{1, 2, \ldots, 10\}$
$S^*_{\Omega^b_{5000}}$	Sequence found by Algorithm 1 using subset Ω^b_{5000} for intermediate evaluations
T	Predetermined threshold
$\alpha(S, T)$	Makespan service level of sequence S when time is limited to T
$\alpha(S, T, \Omega)$	Estimation of $\alpha(S, T)$ using set of scenarios Ω
$\mathbb{E}(C_{max}(S))$	Expectation on the makespan of sequence S
$\mathbb{E}(S, \Omega)$	Estimation on $\mathbb{E}(C_{max}(S))$ of sequence S using set Ω
$\Delta_{\mathbb{E}}$	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}} [\mathbb{E}(S_{init}, \Omega) - \mathbb{E}(S^{\mathbb{E}}, \Omega)]$
Δ_1	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}} \lvert\alpha(S^E, T, \Omega) - \alpha(S^*, T, \Omega)\rvert$
Estimation error	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}} \lvert\alpha(S^*, T, \Omega_s) - \alpha(S^*, T, \Omega)\rvert$ (Section 7.1)
Improvement gap	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}} \alpha(S^*, T, \Omega) - \alpha(S_{init}, T, \Omega_s)$ (Section 7.1)
Estimation error	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}} \lvert\alpha(S_{init}, T, \Omega) - \alpha(S_{init}, T, \Omega^b_{500})\rvert$ (Section 7.2)
Improvement gap	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}} [\alpha(S^*_{\Omega^b_{500}}, T, \Omega) - \alpha(S_{init}, T, \Omega)]$ (Section 7.2)
$\overline{\mathbb{E}}(S^{\mathbb{E}})$	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}} \mathbb{E}(S^{\mathbb{E}}, \Omega)$
Δ_{CPU}	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}} \lvert\text{CPU}(S^*) - \text{CPU}(S^{\mathbb{E}})\rvert$
$\overline{CPU}$	$\frac{1}{\lvert\mathcal{J}\rvert} \sum_{job \in \mathcal{J}}$ CPU time
$N_{\alpha(S^{\mathbb{E}}, S^*)}$	Number of times per instance that $\alpha(S^E, T, \Omega) > \alpha(S^*, T, \Omega)$
$N_{CPU(S^{\mathbb{E}}, S^*)}$	Number of times per instance that $\text{CPU}(S^{\mathbb{E}}) < \text{CPU}(S^*)$

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.cor.2023.106237.

References

Adams, J., Balas, E., Zawack, D., 1988. The shifting bottleneck procedure for job shop scheduling. Manage. Sci. 34 (3), 391–401.

Amjad, M.K., Butt, S.I., Kousar, R., Ahmad, R., Agha, M.H., Faping, Z., Anjum, N., Asgher, U., 2018. Recent research trends in genetic algorithm based flexible job shop scheduling problems. Math. Probl. Eng. 2018.

Beck, J.C., Wilson, N., 2007. Proactive algorithms for job shop scheduling with probabilistic durations. J. Artificial Intelligence Res. 28, 183–232.

Birge, J.R., Louveaux, F., 2011. The value of information and the stochastic solution. In: Introduction to Stochastic Programming. Springer, pp. 163–177.

Brandimarte, P., 1993. Routing and scheduling in a flexible job shop by tabu search. Ann. Oper. Res. 41 (3), 157–183.

Brucker, P., Schlie, R., 1990. Job-shop scheduling with multi-purpose machines. Computing 45 (4), 369–375.

Calafiore, G., Campi, M.C., 2005. Uncertain convex programs: randomized solutions and confidence levels. Math. Program. 102 (1), 25–46.

Çaliş, B., Bulkan, S., 2015. A research survey: review of AI solution strategies of job shop scheduling problem. J. Intell. Manuf. 26 (5), 961–973.

Campi, M., Calafiore, G., 2004. Decision making in an uncertain environment: the scenario-based optimization approach. Multiple Particip. Decis. Mak. 99–111.

Çatay, B., Erengüç, Ş.S., Vakharia, A.J., 2003. Tool capacity planning in semiconductor manufacturing. Comput. Oper. Res. 30 (9), 1349–1366.

Chaudhry, I.A., Khan, A.A., 2016. A research survey: review of flexible job shop scheduling techniques. Int. Trans. Oper. Res. 23 (3), 551–591.

Daniels, R.L., Carrillo, J.E., 1997. β-Robust scheduling for single-machine systems with uncertain processing times. IIE Trans. 29 (11), 977–985.

Dauzère-Péres, S., 1994. A Connected Neighborhood Structure for the Multiprocessor Job-Shop Scheduling Problem. Erasmus Universiteit/Rotterdam School of Management, Faculteit Bedrijfskunde.

Dauzère-Pérès, S., Castagliola, P., Lahlou, C., 2005. Niveau de service en ordonnancement. In: Billaut, J., Moukrim, A., Sanlaville, E. (Eds.), Flexibility and Robustness in Scheduling. Hermes Science Publications, pp. 97–113, (Chapter 5).

Dauzère-Pérès, S., Castagliola, P., Lahlou, C., 2013. Service level in scheduling. In: Billaut, J., Moukrim, A., Sanlaville, E. (Eds.), Flexibility and Robustness in Scheduling. Wiley, pp. 99–121, (Chapter 5).

Dauzère-Pérès, S., Paulli, J., 1994. Solving the General Multiprocessor Job-Shop Scheduling Problem. Management Report Series (182).

Dauzère-Pérès, S., Paulli, J., 1997. An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search. Ann. Oper. Res. 70, 281–306.

Flores Gómez, M., Borodin, V., Dauzère-Pérès, S., 2021. A Monte Carlo based method to maximize the service level on the makespan in the stochastic flexible job-shop scheduling problem. In: 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE). IEEE, pp. 2072–2077.

Gao, J., Sun, L., Gen, M., 2008. A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. Comput. Oper. Res. 35 (9), 2892–2907.

García-León, A.A., Dauzère-Pérès, S., Mati, Y., 2015. Minimizing regular criteria in the flexible job-shop scheduling problem. In: Proceedings of the 7th Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA). pp. 443–456.

García-León, A.A., Dauzère-Pérès, S., Mati, Y., 2019. An efficient Pareto approach for solving the multi-objective flexible job-shop scheduling problem with regular criteria. Comput. Oper. Res. 108, 187–200.

Garey, M.R., Johnson, D.S., Sethi, R., 1976. The complexity of flowshop and jobshop scheduling. Math. Oper. Res. 1 (2), 117–129.

Gendrau, M., Potvin, J.-Y., 2010. Handbook of Metaheuristics. Springer, pp. 41–57, (Chapter 2).

Ghaleb, M., Zolfagharinia, H., Taghipour, S., 2020. Real-time production scheduling in the industry-4.0 context: Addressing uncertainties in job arrivals and machine breakdowns. Comput. Oper. Res. 123, 105031.

Giles, M.B., 2015. Multilevel monte carlo methods. Acta Numer. 24, 259–328.

Golenko-Ginzburg, D., Kesler, S., Landsman, Z., 1995. Industrial job-shop scheduling with random operations and different priorities. Int. J. Prod. Econ. 40 (2–3), 185–195.

Hammersley, J.M., Handscomb, D.C., 1964. General principles of the monte carlo method. In: Monte Carlo Methods. Springer, pp. 50–75.

Homem-de-Mello, T., Bayraksan, G., 2014. Monte Carlo sampling-based methods for stochastic optimization. Surv. Oper. Res. Manag. Sci. 19 (1), 56–85.
Horng, S.-C., Lin, S.-S., Yang, F.-Y., 2012. Evolutionary algorithm for stochastic job shop scheduling with random processing time. Expert Syst. Appl. 39 (3), 3603–3610.
Hurink, J., Jurisch, B., Thole, M., 1994. Tabu search for the job-shop scheduling problem with multi-purpose machines. Oper.-Res.-Spektrum 15 (4), 205–215.
Jacobs, J., Etman, L., Van Campen, E., Rooda, J., 2003. Characterization of operational time variability using effective process times. IEEE Trans. Semicond. Manuf. 16 (3), 511–520.
Jamrus, T., Chien, C., Gen, M., Sethanan, K., 2018. Hybrid particle swarm optimization combined with genetic operators for flexible job-shop scheduling under uncertain processing time for semiconductor manufacturing. IEEE Trans. Semicond. Manuf. 31 (1), 32–41.
Kacem, I., Hammadi, S., Borne, P., 2002. Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems. IEEE Trans. Syst. Man Cybern. C (Appl. Rev.) 32 (1), 1–13.
Karabuk, S., 2001. Coordinating Capacity Decisions for the Supply Chain in High-Tech Industry. Lehigh University.
Knopp, S., Dauzère-Pérès, S., Yugma, C., 2017. A batch-oblivious approach for Complex Job-Shop scheduling problems. European J. Oper. Res. 263 (1), 50–61.
Luedtke, J., Ahmed, S., 2008. A sample approximation approach for optimization with probabilistic constraints. SIAM J. Optim. 19 (2), 674–699.
Mahdavi, I., Shirazi, B., Solimanpur, M., 2010. Development of a simulation-based decision support system for controlling stochastic flexible job shop manufacturing systems. Simul. Model. Pract. Theory 18 (6), 768–786.
Marshall, A.W., Olkin, I., 2007. Gamma and beta functions. In: Life Distributions. Springer, pp. 717–727.
Mati, Y., Dauzère-Pérès, S., Lahlou, C., 2011. A general approach for optimizing regular criteria in the job-shop scheduling problem. European J. Oper. Res. 212 (1), 33–42.
Mokhtari, H., Dadgar, M., 2015. Scheduling optimization of a stochastic flexible job-shop system with time-varying machine failure rate. Comput. Oper. Res. 61, 31–45.
Mönch, L., Fowler, J.W., Dauzère-Pérès, S., Mason, S.J., Rose, O., 2011. A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. J. Sched. 14 (6), 583–599.
Mönch, L., Schabacker, R., Pabst, D., Fowler, J.W., 2007. Genetic algorithm-based subproblem solution procedures for a modified shifting bottleneck heuristic for complex job shops. European J. Oper. Res. 177 (3), 2100–2118.
Paulli, J., 1995. A hierarchical approach for the FMS scheduling problem. European J. Oper. Res. 86 (1), 32–42.
Pezzella, F., Morganti, G., Ciaschetti, G., 2008. A genetic algorithm for the flexible job-shop scheduling problem. Comput. Oper. Res. 35 (10), 3202–3212.
Rossi, A., Dini, G., 2007. Flexible job-shop scheduling with routing flexibility and separable setup times using ant colony optimisation method. Robot. Comput.-Integr. Manuf. 23 (5), 503–516.
Roy, B., Sussmann, B., 1964. Les problemes d'ordonnancement avec contraintes disjonctives. Note Ds 9.
Shen, L., Dauzère-Pérès, S., Neufeld, J.S., 2018. Solving the flexible job shop scheduling problem with sequence-dependent setup times. European J. Oper. Res. 265 (2), 503–516.
Silver, E.A., Pyke, D.F., Peterson, R., et al., 1998. Inventory Management and Production Planning and Scheduling, Vol. 3. Wiley New York.
Szántai, T., 2009. Approximation of multivariate probability integrals. Encyclopedia Optim. 1, 53–59.
Tamssaouet, K., Dauzère-Pérès, S., Knopp, S., Bitar, A., Yugma, C., 2022. Multiobjective optimization for complex flexible job-shop scheduling problems. European J. Oper. Res. 296 (1), 87–100.
Wu, C.W., Brown, K.N., Beck, J.C., 2009. Scheduling with uncertain durations: Modeling β-robust scheduling with constraints. Comput. Oper. Res. 36 (8), 2348–2356.
Xing, L.-N., Chen, Y.-W., Yang, K.-W., 2009. Multi-objective flexible job shop schedule: design and evaluation by simulation modeling. Appl. Soft Comput. 9 (1), 362–376.
Yazdani, M., Amiri, M., Zandieh, M., 2010. Flexible job-shop scheduling with parallel variable neighborhood search algorithm. Expert Syst. Appl. 37 (1), 678–687.
Zhang, R., Song, S., Wu, C., 2012. A two-stage hybrid particle swarm optimization algorithm for the stochastic job shop scheduling problem. Knowl.-Based Syst. 27, 393–406.