


WILEY

Intl. Trans. in Op. Res. 28 (2021) 716–737
DOI: 10.1111/itor.12862INTERNATIONAL
TRANSACTIONS
IN OPERATIONAL
RESEARCH

A simheuristic algorithm for the stochastic permutation flow-shop problem with delivery dates and cumulative payoffs

Pedro A. Villarinho^{a,*} , Javier Panadero^b, Luciana S. Pessoa^a, Angel A. Juan^b
and Fernando L. Cyrino Oliveira^a

^aDepartment of Industrial Engineering, Pontificia Universidade Catolica do Rio de Janeiro, Rua Marquês de São Vicente, 225, Gávea 22453-900, Rio de Janeiro, RJ, Brazil

^bIN3 – Computer Science Department, Universitat Oberta de Catalunya & Euncet Business School, Av. Carl Friedrich Gauss 5, Castelldefels 08860, Spain

E-mail: villarinhopedro@gmail.com [Villarinho]; jpanaderom@uoc.edu [Panadero]; lucianapessoa@puc-rio.br [Pessoa]; ajuanp@uoc.edu [Juan]; cyrino@puc-rio.br [Oliveira]

Received 10 January 2020; received in revised form 24 May 2020; accepted 27 July 2020

Abstract

This paper analyzes the permutation flow-shop problem with delivery dates and cumulative payoffs (whenever these dates are met) under uncertainty conditions. In particular, the paper considers the realistic situation in which processing times are stochastic. The main goal is to find the permutation of jobs that maximizes the expected payoff. In order to achieve this goal, the paper first proposes a biased-randomized heuristic for the deterministic version of the problem. Then, this heuristic is extended into a metaheuristic by encapsulating it into a variable neighborhood descent framework. Finally, the metaheuristic is extended into a simheuristic by incorporating Monte Carlo simulations. According to the computational experiments, the level of uncertainty has a direct impact on the solutions provided by the simheuristic. Moreover, a risk analysis is performed using two well-known metrics: the value-at-risk and conditional value-at-risk.

Keywords: permutation flow-shop problem; stochastic processing times; deliver dates; cumulative payoffs; biased-randomized algorithms; simheuristics

1. Introduction

In the permutation flow-shop scheduling problem with delivery dates and cumulative payoff (PF-SPDP), a finite number of jobs j are processed, following the same order from a release date r_j , by a finite number of machines m . Each time a job finishes before a delivery date, a reward is obtained. Hence, the main goal is to find the permutation of jobs that maximizes the total reward related to the job completion times. Typically, job rewards are assigned following a decreasing stepwise

*Corresponding author.

© 2020 The Authors.

International Transactions in Operational Research © 2020 International Federation of Operational Research Societies
Published by John Wiley & Sons Ltd, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main St, Malden, MA02148, USA.

function, that is, delivery dates define a series of time intervals, and the sooner a job is completed the greater the associated reward. This problem was first studied by Seddik et al. (2013) who modeled a book digitization service in a one-machine environment. Then, Pessoa and Andrade (2018) extended the aforementioned problem into a flow-shop context, and proposed a mathematical formulation and a set of heuristic and metaheuristic methods. As described in the previous references, the problem is motivated by a book-digitization project in a library. The book digitization is a job and this has to be processed in two consecutive stages (machines): digitization and segmentation. Each book requires a given amount of time to be processed in each stage. Due to the operational constraints in the library, not all books are available for digitization at the same time. Hence, a release date is set for each one. This problem is modeled by taking into account two entities: the library, which establishes the delivery dates, and the digitization firm. Each delivery date is a reference point for computing the payment: the sooner each book is processed the higher the corresponding reward. We consider the problem from the point of view of the digitization firm, which wants to maximize its profits.

Our paper takes one step forward and proposes a biased-randomized (BR) algorithm based on the FF heuristic developed by Fernandez-Viagas and Framinan (2015). Our BR-FF algorithm shows a remarkable performance when solving the benchmarks proposed in Pessoa and Andrade (2018). This is coherent with previous results, in which BR algorithms have shown to be competitive approaches for solving different flow-shop problems, including those with nonsmooth objective functions (Ferrer et al., 2016). Since uncertainty is often present in many service and producing systems, the paper also considers a stochastic version of the PFSPDP, named PFSPDP with stochastic times, in which processing times are modeled as random variables. In order to solve this stochastic version, the previous BR-FF algorithm is first integrated into a variable neighborhood descent (VND) metaheuristic framework. Then, the resulting BR-FF-VND metaheuristic is extended into a “simheuristic” algorithm by combining the VND metaheuristic with Monte Carlo simulation. Another significant contribution of this paper relates to the use of two risk metrics: the value-at-risk (VaR) and conditional value-at-risk (CVaR). They are used to complement the simheuristics approach, offering the decision maker an analysis about the worst expected cases concerning the rewards.

The remainder of the paper is distributed as follows. Section 2 contains a more detailed description of the problem under consideration. A literature review on related work is provided in Section 3. Section 4 describes a BR algorithm for solving the deterministic version of the problem, as well as its integration into a metaheuristic framework. Section 5 extends the previous metaheuristic into a simheuristic to cope with the stochastic version of the problem. The settings of the computational experiments are provided in Section 6, while the actual experiments are discussed in Sections 7 (deterministic variant) and 8 (stochastic variant). Finally, Section 9 highlights the main contributions of this work and discusses some open research lines.

2. Problem definition

The deterministic version of the PFSPDP is described next. A finite set of n jobs, $\mathcal{J} = \{1, 2, \dots, n\}$ is processed in a subsequent way by m machines $\mathcal{M} = \{1, 2, \dots, m\}$. For each job $j \in \mathcal{J}$ and machine $l \in \mathcal{M}$, the processing time of job j in machine l is a known constant $p_{jl} > 0$ (note that no

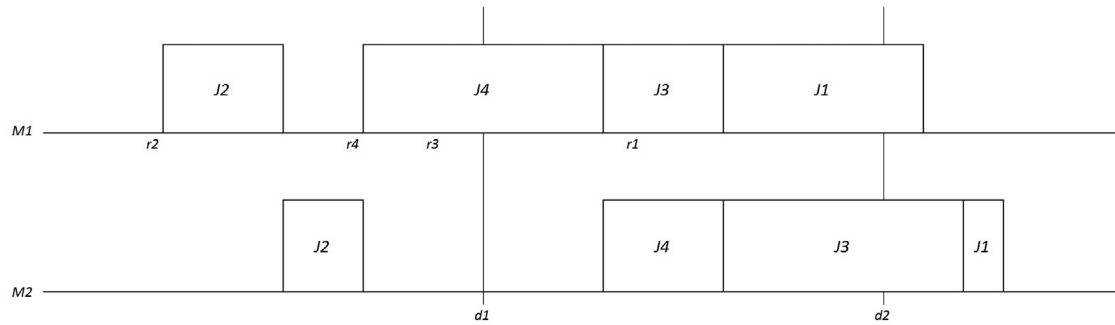


Fig. 1. Example of a flow-shop scheduling with delivery dates.

uncertainty is considered in this deterministic version of the problem). Furthermore, let S (schedule) be a finite set containing the times s_{jl} at which each job j starts being processed at each machine l , that is, $S = \{s_{11}, s_{12}, \dots, s_{1m}, s_{21}, s_{22}, \dots, s_{2m}, \dots, s_{n1}, s_{n2}, \dots, s_{nm}\}$. Given a schedule S , the completion time of a job j in a machine l is given by $c_{jl} = s_{jl} + p_{jl}$. For simplicity, let c_j denote the completion time of job j in the last machine $m \in \mathcal{M}$, that is, $c_j = c_{jm}$.

The problem addressed in this work also considers release dates, $r_j \geq 0$, in which job j becomes available to start being processed by machine 1. Additionally, a set of k delivery dates $\mathcal{D} = \{d_1, d_2, \dots, d_k\}$ is given. Each job j has an associated reward, $\mathcal{F}(c_j)$, which depends on the time it is completed:

$$\mathcal{F}(c_j) = \begin{cases} 0 & \text{if } d_k < c_j \\ 1 & \text{if } d_{k-1} < c_j \leq d_k \\ \vdots & \\ k & \text{if } 0 < c_j \leq d_1. \end{cases} \quad (1)$$

Hence, the goal is to obtain a permutation of jobs that maximizes the aggregated reward obtained, that is, the objective function is $\max \sum_{j=1}^n \mathcal{F}(c_j)$. To illustrate how the reward function works, consider the following simple example (Fig. 1):

- The number of machines $m = 2$, the number of jobs $n = 4$.
- Release dates: $r_1 = 14$, $r_2 = 2$, $r_3 = 9$, and $r_4 = 7$.
- Processing times: $p_{1l} = (5, 1)$, $p_{2l} = (3, 2)$, $p_{3l} = (3, 6)$, and $p_{4l} = (6, 3)$.
- Delivery dates: $d_1 = 10$, $d_2 = 20$.

For this instance, which contains two delivery dates, the reward stepwise function has the following configuration:

$$\mathcal{F}(C_j) = \begin{cases} 0 & \text{if } d_2 < c_j \\ 1 & \text{if } d_1 < c_j \leq d_2 \\ 2 & \text{if } 0 < c_j \leq d_1. \end{cases} \quad (2)$$

As illustrated in Fig. 1, since job 2 has a completion time $c_2 = 7$ (which is lower than delivery date $d_1 = 10$), there is an associated reward of two units. In addition, job 4 has a completion time $c_4 = 16$ (which is lower than delivery date $d_2 = 20$), thus generating a reward of one unit. Also, since jobs 3 and 1 have completion times beyond the delivery date $d_2 = 20$ ($c_3 = 22$ and $c_1 = 23$), no associated rewards are obtained. All in all, the accumulated reward in this case is three units.

3. Related work

A simple version of the problem described in Section 2 was initially addressed by Seddik et al. (2013). These authors consider a single machine, being the problem known as the $1|r_j|\sum_{j=1}^n F(c_j)$. Later, Pessoa and Andrade (2018) extended the model to consider a flow-shop environment with m machines, that is, $F|r_j, perm|\sum_{j=1}^n F(c_j)$. This section reviews some related work on the permutation flow-shop scheduling problem (PFSP), both in its deterministic and stochastic versions.

3.1. The deterministic PFSP with different time-related goals

Since the first formulation proposed by Johnson (1954), which aimed at minimizing the makespan in two machines, other formulations were developed on the flow-shop scheduling problem considering heuristic and meta-heuristic methods. For complete reviews on deterministic flow-shop problems employing the makespan minimization criterion, readers are referred to Fernandez-Viagas et al. (2017). Despite the makespan minimization has been the most popular goal in flow-shop problems, other objective functions have been considered as well. Thus, Liu and Reeves (2001) presented the LR constructive heuristic to minimize the total flow time in a PFSP. Similarly, Framinan and Leisten (2003) proposed another constructive heuristic for the same purpose. Also, Framinan et al. (2005) compared several heuristics and proposed two new heuristics, which have proved to be efficient to minimize the total flow time. A biobjective problem considering the minimization of the makespan and the total flow time was tackled by Pasupathy et al. (2006). These authors proposed a genetic algorithm (GA) to construct a Pareto frontier. Framinan and Leisten (2008) proposed a new greedy algorithm based on the variable neighborhood search framework to minimize the total tardiness. Nagano and Moccellini (2008) aimed at minimizing the flow time by proposing a new heuristic and some local search procedures. Also aiming at minimizing the total flow time, an iterated local search (ILS) was proposed by Dong et al. (2009). Laha and Sarin (2009) extended and enhanced the heuristic proposed by Framinan and Leisten (2003) by modifying the selection process of the jobs. Tasgetiren et al. (2011) expanded the discrete differential evolution algorithm proposed by Pan et al. (2008) into a hybrid version, and introduced a discrete artificial bee colony algorithm aiming at minimizing the total flow time. Della Croce et al. (2011) designed a metaheuristic algorithm for minimizing the total completion time in a two-machine flow-shop problem. Pan and Ruiz (2013) made a comprehensive evaluation of 22 heuristics, using as objective function the minimization of the total flow time. Fernandez-Viagas and Framinan (2015) proposed a new heuristic, named FF, which was based on the LR heuristic by Liu and Reeves (2001). Lee and Chung (2013) addressed an m -machine flow-shop problem with learning effects. The goal was to minimize the total tardiness by proposing a branch-and-bound method and two heuristics. Molina-Sánchez and

González-Neira (2016) introduced a greedy randomized adaptive search procedure (GRASP) to tackle the minimization of the total weighted tardiness.

Also to minimize the flow time, Abedinnia et al. (2016) introduced a constructive heuristic based on the one proposed by Laha and Sarin (2009). Yu and Seif (2016) proposed a lower bound based on a GA to minimize the total tardiness and maintenance cost. Rossi et al. (2017) sought to minimize the total flow time by employing a new constructive heuristic based on the FF-NEH heuristic introduced by Fernandez-Viagas and Framinan (2015). With the aim of minimizing the total completion time, Fernandez-Viagas and Framinan (2017) proposed a beam search based constructive heuristic. The minimization of total tardiness was tackled by Fernandez-Viagas et al. (2018). More recently, Rossi and Nagano (2019) tackled a mixed no-idle PFSP with sequence-dependent setup times. With the goal of minimizing the total flow time, they proposed a new set of heuristics based on the beam-search algorithm. Likewise, several matheuristic algorithms were proposed and compared to a GA by Ta et al. (2018), with the purpose of minimizing the total tardiness. Finally, Andrade et al. (2019) proposed a biased random-key GA (BRKGA), with a feature called shaking, to minimize the total completion time.

3.2. The stochastic PFSP

In the real world, unforeseen events are often present in daily-life situations. Service and production systems are not excluded from those situations. On the contrary, they are subject to uncertainties, for example, machine failures, imprecise processing times, employee absences, material unavailability, variations in demand, rush orders and order cancellations, changes in due dates, etc. The literature on flow-shop problems with stochastic components is not as extensive as in the case of the deterministic version. While in the deterministic version of the PFSP one assumes that the processing time of a job j in a machine l , $p_{jl} > 0$, is known in advance and cannot change over time, in the stochastic version it is usual to model this processing time as a random variable, P_{jl} , following a nonnegative probability distribution. By modeling processing times this way, it is possible to incorporate the uncertainty that is present in real-life scenarios. However, the inclusion of random variables in the optimization problem also increases its difficulty. Banerjee (1965) and Makino (1965) were among the pioneers in the study of the PFSP with stochastic processing times. The former aims at minimizing the probability of lateness in a single machine problem, with random processing times following a known probability distribution. In order to do it, the author proposes a decision rule for the single machine problem. The latter author aims at minimizing the expected makespan considering that processing times follow exponential or k -Erlang probability distributions. He also proposes a sequence rule to find a solution in a scenario with two jobs and three machines. Dodin (1996) sought to minimize the expected makespan by representing processing times as random variables following different probability distributions (uniform, normal, and exponential). Kamburowski (1999) drew up an approach aimed at minimizing the makespan in a scenario with two machines and independent processing times. Later, Kamburowski (2000) expanded the previous work to a scenario involving three machines. Gourgand et al. (2000) completed a review about the stochastic flow-shop scheduling problem. Wang et al. (2005) also sought to minimize the makespan while modeling processing times using uniform probability distributions and employing a GA. With the same goal, Kalczyński and Kamburowski (2006) modeled processing times using

Weibull probability distributions. Baker and Trietsch (2011) developed three heuristic procedures for the two-machine permutation flow-shop problem with stochastic processing times. Choi and Wang (2012) used a gamma probability distribution to model processing times. Liefoghe et al. (2012) proposed an indicator-based evolutionary algorithm to deal with a biobjective PFSP with stochastic processing times. Baker and Altheimer (2012) evaluated and compared several heuristic procedures for the stochastic version of the problem. Gonzalez-Neira et al. (2017) presented an overview of PFSP variants with stochastic components. Gholami-Zanjani et al. (2017) introduced robust optimization and fuzzy optimization to model the uncertainty of the input data, and minimized weighted mean completion time in a PFSP considering setup and nondeterministic processing times. Cui et al. (2018) analyzed a biobjective problem including the quality robustness and solution robustness in environments subject to failure uncertainty, which was modeled using Weibull probability distributions. Simheuristics have been recently used to cope with stochastic versions of flow-shop problems (Hatami et al., 2018). Finally, Framinan et al. (2019) explored the use of real-time information to job rescheduling in a PFSP with stochastic processing times.

4. A biased-randomized algorithm (BR-FF) for the PFSPDP

In this section, a BR-FF is proposed for the deterministic version of the PFSP with delivery dates and cumulative payoff. The algorithm is based on the constructive heuristic FF by Fernandez-Viagas and Framinan (2015). This heuristic was also examined by Pessoa and Andrade (2018) who showed their excellent performance when compared to other constructive procedures for the PFSPDP.

4.1. The FF heuristic

The FF heuristic can be considered as an improved version of the LR heuristic proposed by Liu and Reeves (2001). A solution is built by allocating, at each iteration, an unscheduled job in the last position of a partial solution, according to the index function ξ' given in Equation (3):

$$\xi'_{jk} = \frac{(n-k-2)}{a} IT'_{jk} + AT'_{jk}, \quad (3)$$

where AT'_{jk} and IT'_{jk} are defined as follows:

$$IT'_{jk} = \sum_{i=2}^m \frac{m * \max\{c_{(i-1)j} - c_{i[k]}, 0\}}{i - b + k(m - i + b)/(n - 2)} \quad (4)$$

$$AT'_{jk} = c_{mj}. \quad (5)$$

At each iteration, the index function ξ' is computed considering the weighted idle time, IT'_{jk} , between the last job in the partial solution—being represented as k —and each remaining job outside the partial solution—being represented as j . Moreover, the index ξ' approximate the makespan of

the solution, AT'_{jk} . The parameters a and b weight the idle time and makespan in the ξ' function. After ξ' is computed, the jobs are sorted according to the ascending order of its ξ' value. Then, the job with the smallest ξ' value is chosen and scheduled in the partial solution. In case of ties, the job with the smallest value of IT'_{jk} is selected. Pessoa and Andrade (2018) proposed a slight modification of the idle time (IT'_{jk}) for the first machine, in order to consider the job release dates, r_j . In the new expression, $\max\{c_{(i-1)j} - c_{i[k]}, 0\}$ was replaced by $\max\{r_j - c_{i[k]}, 0\}$. Regarding the PFSPDP, the authors show the superior performance of the FF heuristic when compared to other traditional heuristics such as the release dates heuristic (Potts, 1985), the earliest completion time heuristic (Ladhari and Rakrouki, 2009), the classical NEH heuristic (Nawaz et al., 1983), or the iterated earliest completion time heuristic (Pessoa and Andrade, 2018).

4.2. Extending the FF heuristic to a BR-FF

We propose a BR extension of the FF heuristic, called BR-FF. Biased-randomization techniques can be used to incorporate an oriented (non-uniform) random behavior into a base heuristic. This random behavior, typically achieved with the use of a skewed probability distribution, allows for encapsulating the modified heuristic into a multistart framework, thus exploring alternative paths during the solution-building process. Still, due to the oriented nature of the randomization process, each of these paths follows the logic behind the heuristic, that is, the most “promising” movements receive a higher probability of being selected during the constructive process. As a result, each time the BR heuristic is executed, chances are that the emerging solution outperforms the one provided by the original heuristic. There is an additional benefit of applying biased-randomization techniques to a constructive heuristic: the resulting procedure is able to quickly generate different alternative solutions of reasonably good quality.

In our case, the biased-randomization effect has been applied to the job selection process. Thus, while in the original FF heuristic the job with the smallest ξ' value is always selected, in our BR-FF a diminishing probability of being selected is assigned to each possible job in the list of jobs sorted from lower to higher ξ' value. This is achieved by employing a geometric probability distribution with a single parameter β ($0 < \beta < 1$). Hence, the random behavior of the selection process depends on the value of β . As this value converges to 1, the selection behavior becomes more greedy (i.e., as in the original heuristic). On the contrary, as this value converges to 0, the selection behavior follows a uniform-random one. Intermediate values of β allow for considering randomization policies between both extremes, which makes the construction process more effective (Quintero-Araujo et al., 2017).

Algorithm 1 illustrates this procedure. It starts by gathering the information of the jobs in the instance. The main loop add one job at a time to the emerging solution S , until all jobs have been incorporated to the permutation. Inside this loop, the index ξ'_{jk} is calculated for each job. Then, the jobs are sorted in ascending order by this index value. The BR selection procedure is employed to select the next job to be added to the solution. Finally, the job selected is removed from the list of unscheduled jobs.

As explained in Section 1, in addition to the use of BR techniques, we have also incorporated a VND metaheuristic framework (Hansen and Mladenović, 2003). Hence, the complete algorithm

Algorithm 1. BR-FF algorithm

```

1  $S \leftarrow \emptyset$ 
2  $S_{best} \leftarrow S$ 
3  $L \leftarrow \{1, 2, \dots, n\}$ 
4 foreach  $j \in L$  do
5   | Compute  $\xi'_{jk}$ 
6 end
7 Sort the jobs in  $L$  using ascending order by  $\xi'_{jk}$ 
8 while  $elapsed < maxTime$  do
9   |  $L_{copy} \leftarrow L$ 
10  while  $L_{copy} \neq \emptyset$  do
11    | Choose, using biased randomization, a job index  $j^* \in L_{copy}$ 
12    |  $S \leftarrow S \cup j^*$ 
13    |  $L_{copy} \leftarrow L_{copy} \setminus j^*$ 
14  end
15  if  $S$  improves  $S_{best}$  then
16    |  $S_{best} \leftarrow S$ 
17  end
18 end
19 return  $S_{best}$ 

```

is denoted as BR-FF-VND. The VND framework adds some local search operators to the BR-FF algorithm. In particular, two classical operators have been added: insertion and interchange. The interchange operator swaps two jobs from the original positions, while the insertion operator attempts to move a job to a new position that improves the solution.

5. Extending the BR-FF-VND algorithm to a simheuristic

Since one of the main goals of this paper is to solve the stochastic version of the PFSPDP, this section explains how Monte Carlo simulation has been integrated into the BR-FF-VND algorithm in order to build a simheuristic. Considered as a special case of simulation optimization, the simheuristic approach is not only able to deal with stochastic versions of most combinatorial optimization problems but it also provides risk and reliability analyses on the stochastic solutions it generates. Thus, simheuristic approaches have been gaining notoriety during the last years, being presented in different fields such as waste collection management (Gruler et al., 2017), arc routing problems (Gonzalez-Martin et al., 2018), inventory routing problems (Gruler et al., 2020), and facility–location problems (Pagès-Bernaus et al., 2019). As a novelty with respect to these previous references on simheuristic algorithms, in this paper we also perform risk analyses based on two well-known metrics: the VaR_α and tCVaR_α . These metrics contribute to provide more complete information to decision makers.

In this stochastic variant of the PFSPDP, the processing times of job j in machine l are modeled as a random variable, P_{jl} , which follows a given probability distribution. Despite our methodology can use any probability distribution that fits the historical data, in our computational experiments we have assumed that P_{jl} follows a log-normal probability distribution with location parameter μ_{jl} and scale parameter σ_{jl} —actually, due to their flexibility, the log-normal and Weibull probability distributions are frequently employed in reliability analysis to model positive failure times. In order to extend the deterministic instances in a natural way, we have assumed that $E[P_{jl}] = p_{jl}$. Likewise, in order to analyze different uncertainty levels, we have assumed that $\text{Var}[P_{jl}] = h \cdot p_{jl}$, where $h \in \{0.1, 0.5, 1.0, 2.0\}$ is a design parameter. Equations (6) and (7) display, for the log-normal probability distribution, the relationship between the location and scale parameters as well as its expected and variance values:

$$\mu_{jl} = \ln E[P_{jl}] - \frac{1}{2} \left(1 + \frac{\text{Var}[P_{jl}]}{E[P_{jl}]^2} \right) \quad (6)$$

$$\sigma_{jl} = \left| \sqrt{\ln \left(1 + \frac{\text{Var}[T_{ij}]}{E[T_{ij}]^2} \right)} \right|. \quad (7)$$

In the deterministic PFSPDP, the reward associated with a given permutation of jobs is generated by the stepwise function provided in Equation (1). However, in the stochastic PFSPDP the reward associated with a given permutation is a random variable that will take a different value each time the solution is tested in a real life or simulated environment. Hence, an estimate of the expected demand associated with a given permutation will be given by the average reward obtained after executing a sufficient number of simulation runs. Algorithm 2 depicts our Sim BR-FF-VND procedure.

The main ideas behind this algorithm are as follows: first, an initial solution for the deterministic PFSPDP is generated by employing the BR-FF algorithm. This is considered as the best solution so far. A fast simulation is performed to measure, in the stochastic environment, the reward provided by the best solution so far. Then, the resulting solution is inserted in the pool of best solutions. While the stopping criterion is not satisfied, a while loop is performed. This loop starts with the generation of a new solution in the deterministic environment. This solution is then enhanced using the VND component (local search operators). The Δ value checks the quality of this new solution in the deterministic environment. If this new solution offers a higher reward than the current base solution, the new solution is considered as a good candidate for solving the deterministic version of the problem, as well as a promising solution for the stochastic version. Note that a correlation between the best deterministic and the best stochastic solutions is assumed in our approach. In order to estimate the quality of the new solution in the stochastic environment, a fast simulation is run. If this new solution offers a higher expected reward than the current base solution (i.e., $\Delta' > 0$), the new solution is considered as a good candidate for solving the stochastic version and the base solution is updated. Moreover, if this new solution shows a higher expected reward than the current best solution, the latter is updated by the former, which is also inserted in the pool of best solutions. Finally, the best solutions found are analyzed in more detail using an intensive simulation (i.e., employing a larger number of simulation runs). This allows an increase in the

Algorithm 2. Sim BR-FF-VND

```

1  initSol  $\leftarrow$  BR-FF
2  bestSol  $\leftarrow$  initSol
3  fastSimulation(bestSol)
4  insert(poolBestSol, bestSol)
5  baseSol  $\leftarrow$  bestSol
6  while elapsed < maxTime do
7    | initNewSol  $\leftarrow$  BR-FF
8    | newSol  $\leftarrow$  initNewSol
9    | while initNewSol.getReward > newSol.getReward do
10   | | newSol  $\leftarrow$  localSearch.Interchange(initNewSol)
11   | | newSol  $\leftarrow$  localSearch.Insertion(newSol)
12   | end
13   |  $\Delta \leftarrow$  newSol.getReward - baseSol.getReward
14   | if  $\Delta > 0$  then
15   | | fastSimulation(newSol)
16   | |  $\Delta' \leftarrow$  newSol.getStochReward - baseSol.getStochReward
17   | | if  $\Delta' > 0$  then
18   | | | baseSol  $\leftarrow$  newSol
19   | | | if newSol.getStochReward > bestSol.getStochReward then
20   | | | | bestSol  $\leftarrow$  newSol
21   | | | | insert(poolBestSol, bestSol)
22   | | | end
23   | | end
24   | end
25 end
26 for solution  $\in$  poolBestSol do
27 | longSimulation(solution)
28 | if solution.getStochReward > bestSol.getStochReward then
29 | | bestSol  $\leftarrow$  solution
30 | end
31 end
32 return bestSol

```

accuracy of the estimated expected rewards associated with each solution and generate a final sort of the best solutions based on this criterion.

In addition to providing accurate estimates of expected rewards, the output provided by the intensive simulation process can also be used to compute two well-known risk metrics: the VaR and CVaR. We will compute these VaR_α and CVaR_α for $\alpha \in \{95\%, 97.5\%, 99\%\}$. Generally associated with a loss distribution, these risk metrics have been approached here in a revenue context. To this end, we have followed the definition provided by Street (2010): CVaR_α represents the conditional expected value of the revenue left-side worst distribution scenario, below a given $1 - \alpha$ quantile,

which is known as VaR_α . The latter is conventionally defined as a maximum loss in a given $1 - \alpha$ quantile (Pflug, 2000). In other words, in a revenue distribution for $\alpha = 95\%$, $\text{VaR}_{95\%}$ is the quantile of the 5% worst benefits. Meanwhile, $\text{CVaR}_{95\%}$ is the average of the 5% worst benefits. Despite both metrics aims at measuring the risk associated with a probability distribution, the CVaR_α developed by Rockafellar and Uryasev (2000) is considered to be a better risk metric than the VaR_α , since the former can be seen as a coherent measure with the four axioms defined by Artzner et al. (1999): monotonicity, translation invariance, positive homogeneity, and subadditivity. Thus, CVaR_α is able to quantify events beyond VaR_α by representing the risks reflected at extreme tails of the probability distribution.

6. Experimental settings

This section describes the instances, computational resources, and design parameters employed in our numerical experiments.

6.1. Benchmark instances

In our computational experiments, we used the 150 instances introduced by Pessoa and Andrade (2018). These instances, which are available at <https://data.mendeley.com/datasets/m2wcd42pvy/1>, were randomly generated and are composed of a set of $n = 100$ jobs that need to be processed by $m = 2$ machines. The deterministic processing times were randomly generated using an integer uniform distribution in the interval $[1, 100]$. The delivery dates were generated using the number k of delivery dates, with $k \in \{1, 2, 3, 5, 7, 10\}$. The first delivery date is set to $d_1 = \gamma \cdot C_{max}/k$, being C_{max} the makespan. The remaining delivery dates are set as $d_k = k \cdot d_1$, with $1 < k \leq K$ and $\gamma \in \{0.1, 0.3, 0.5, 0.7, 1.0\}$.

Following Seddik et al. (2013), release dates were randomly chosen in the interval $r_k = [d_{k-1}, d_{k-1} + r \cdot d_{k-1}]$, with $d_0 = 0$, $k \in \{1, 2, \dots, K\}$, and $r \in \{0.1, 0.3, 0.5, 0.7, 1.0\}$. Instance names can be read as $k < k > a < \gamma > r < r >$. For example, instance $k10a1.0r0.1$ has 10 delivery dates, γ equals to 1.0 and r equals to 0.1

6.2. Computational environment and parameter settings

The proposed algorithms have been implemented using C++ programming language and executed in an Intel Core i7—3960 CPU at 3.30 GHz and 24 GB RAM running on a Windows operating system. We performed a simple experiment in order to identify the best values for the β parameter employed during the biased-randomization process of the BR-FF algorithm. A total of 18 values of β were tested, with a time limit of 60 seconds for each test. Also, as suggested in Pessoa and Andrade (2018), the following values were assigned to the FF heuristic parameters: $a = 4.0$ and $b = 0.0$. For each value of β , the percentage deviation between the proposed BR-FF heuristic and the FF heuristic was computed. We consider $\text{BR-FF}_{\text{reward}}$ as the average of the best solutions provided by a β , and FF_{sol} as the average of the solutions provided by the FF heuristic. Hence, the relative

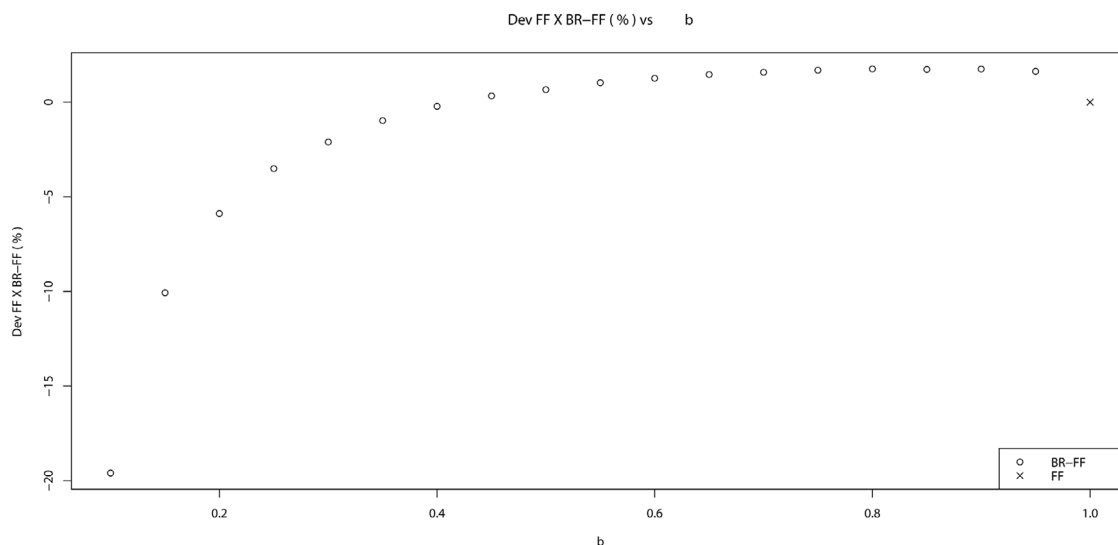


Fig. 2. Deviation of best solutions between the BR-FF and FF heuristics versus the β parameter.

Table 1
A comparison of algorithms for different computing times

Time (minutes)	Algorithm					Deviation (%)			
	FF (1)	FF-GRASP (2)	FF-ILS (3)	BR-FF (4)	BR-FF-VND (5)	(2)–(1)	(3)–(1)	(4)–(1)	(5)–(1)
1	163.71	165.24	166.17	166.35	167.25	0.93	1.50	1.61	2.16
5	163.71	165.66	166.46	166.61	167.63	1.19	1.68	1.77	2.39
10	163.71	165.93	166.60	166.73	167.73	1.36	1.77	1.84	2.46
30	163.71	166.25	166.77	166.93	167.83	1.55	1.87	1.97	2.52

percentage deviation is computed as $(BR-FF_{reward} - FF_{sol}) / FF_{sol}$. Note that a positive value of this percentage means that $BR-FF_{reward}$ is, on average, a better solution than FF_{sol} .

Figure 2 illustrates the deviation between the FF heuristic and our BR-FF algorithm for different values of β . For a β value of 0.45 and higher, the BR-FF is able to outperform the FF heuristic. The best solutions are obtained for $\beta = 0.95$.

7. Testing our BR-FF in the deterministic scenario

According to the experiments carried out by Pessoa and Andrade (2018), the BRKGA and their FF-based ILS are among the best approaches for solving the deterministic PFSPDP. The ILS framework can be easily extended into a full simheuristic algorithm. For this reason, in this section we compare the performance of our BR-FF-VND algorithm against the performance of the FF heuristic (base solving method) and that of the FF-ILS metaheuristic.

All instances were run using each of the solving approaches, allowing up to one minute per instance and approach. As shown in Table 1, the FF-ILS method by Pessoa and Andrade (2018)

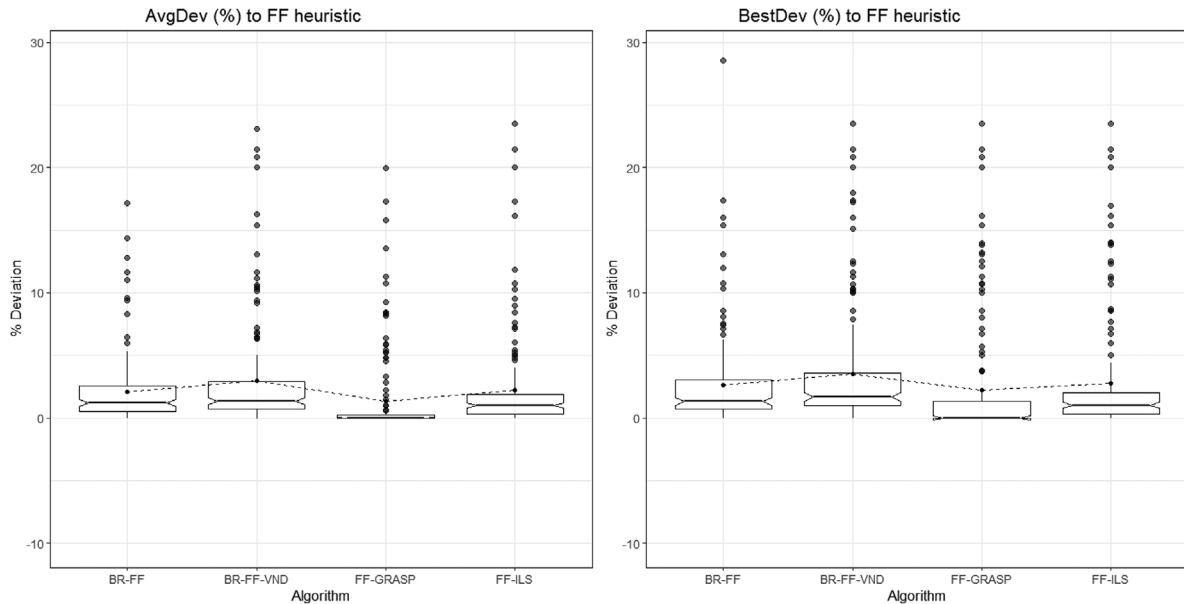


Fig. 3. Box-plot of the percentage deviation $AvgDev$ (%) and $BestDev$ (%).

offered the worst average of best solution values (166.17). Our proposed methods, BR-FF and BR-FF-VND, report averages of 166.35 and 167.25, respectively (i.e., they improve the FF-ILS by 0.11% and 0.65%, respectively). In addition, a GRASP-based algorithm (Resende and Ribeiro, 2016) has also been implemented from the FF constructive method. This FF-GRASP provides results that improve the ones obtained with the FF heuristic. However, these results are somewhat below the ones given by the FF-ILS algorithm. Detailed results regarding all instances are shown in the supplementary material (Villarinho et al., 2020) at <https://data.mendeley.com/datasets/82ykhstgh/4>.

An additional experiment was carried out to compare the solutions generated by each algorithm with the ones provided by the FF heuristic. For each instance, two indicators were computed: (a) the best percentage deviation, $BestDev$ (%); and (b) the average percentage deviation, $AvgDev$ (%). For each instance, the former measures the percentage gap between the best found solution using a new algorithm—of the 10 runs executed per instance—and the solution provided by the FF heuristic. Similarly, the latter measures the percentage gap between the average solution value—computed from the 10 runs—and the solution value provided by the FF heuristic. Table 2 shows the obtained results for a total of 30 randomly selected instances that were run for one minute each.

Figure 3 depicts a box-plot of both percentage deviations, $AvgDev$ (%) and $BestDev$ (%). The $AvgDev$ (%) box-plot shows that the FF-ILS has a similar expected value (2.24) to the BR-FF (2.09). The same can be observed for the $BestDev$ (%).

Our proposed BR-FF-VND shows the higher deviations of $AvgDev$ (%) and $BestDev$ (%), with an average of 2.94 and 3.53, respectively. Moreover, this algorithm also shows the highest average of the best solutions (167.25) among the methods analyzed. When compared to the FF-ILS, our BR-FF-VND provides higher rewards for 59.33% of the instances.

Table 2
Results for one minute and 30 randomly selected instances

Instance	FF	FF-GRASP		FF-ILS		BR-FF		BR-FF-VND	
	<i>BestSol</i>	<i>BestDev</i>	<i>AvgDev</i>	<i>BestDev</i>	<i>AvgDev</i>	<i>BestDev</i>	<i>AvgDev</i>	<i>BestDev</i>	<i>AvgDev</i>
k1a0.3r0.7	46	0.00	0.00	2.17	2.17	2.17	2.17	2.17	2.17
k1a0.5r0.7	65	0.00	0.00	0.00	0.00	1.54	1.54	0.00	0.00
k1a0.5r1.0	59	0.00	0.00	0.00	0.00	1.69	1.69	1.69	1.02
k1a0.7r0.1	80	0.00	0.00	0.00	0.00	0.00	0.00	1.25	0.37
k1a1.0r1.0	94	0.00	0.00	3.19	3.19	3.19	3.19	4.26	3.30
k2a0.1r0.1	27	3.70	1.48	3.70	1.85	7.41	4.07	7.41	4.44
k2a0.5r1.0	85	0.00	0.00	1.18	1.18	1.18	1.18	1.18	1.18
k2a1.0r0.1	151	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
k2a1.0r0.3	149	0.00	0.00	0.67	0.67	0.67	0.67	0.67	0.67
k2a1.0r0.7	150	0.00	0.00	0.00	0.00	0.67	0.67	0.67	0.13
k3a0.1r0.5	35	8.57	4.57	8.57	5.43	8.57	6.00	8.57	6.29
k3a0.1r0.7	37	0.00	0.00	0.00	0.00	5.41	3.51	2.70	2.16
k3a0.3r0.5	82	0.00	0.00	1.22	1.22	3.66	2.56	3.66	2.56
k3a0.5r1.0	117	0.00	0.00	1.71	1.71	2.56	1.88	2.56	2.14
k3a0.7r0.7	156	0.64	0.06	1.92	1.92	2.56	2.18	2.56	2.50
k5a0.5r0.5	191	0.00	0.00	0.52	0.52	1.05	1.05	1.57	1.20
k5a0.5r1.0	178	0.00	0.00	0.56	0.56	1.12	1.07	1.12	0.96
k5a0.7r0.1	218	0.00	0.00	0.46	0.46	0.92	0.87	0.92	0.73
k5a0.7r0.3	228	0.00	0.00	0.00	0.00	0.44	0.44	0.44	0.44
k5a0.7r0.5	220	0.00	0.00	0.00	0.00	0.45	0.45	0.91	0.73
k7a0.3r0.1	159	3.77	1.01	4.40	2.08	5.66	4.91	6.29	4.91
k7a0.5r0.1	245	0.00	0.00	0.82	0.82	0.82	0.82	1.63	0.90
k7a0.5r0.3	256	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
k7a0.7r0.5	306	0.00	0.00	0.33	0.33	0.33	0.33	0.65	0.36
k7a0.7r1.0	295	0.00	0.00	0.00	0.00	1.36	0.98	1.36	0.81
k7a1.0r1.0	330	0.00	0.00	0.30	0.30	0.00	0.00	0.30	0.30
k10a0.3r0.1	246	1.63	0.44	1.63	1.26	3.66	3.13	3.66	2.89
k10a0.3r0.5	217	1.38	0.34	1.84	1.84	3.23	2.58	3.23	2.53
k10a0.3r0.7	216	0.93	0.13	0.93	0.93	1.39	1.25	1.85	1.20
k10a0.5r0.3	330	0.30	0.03	0.91	0.91	2.12	1.36	1.82	1.45
Average	165.60	0.70	0.27	1.23	0.98	2.13	1.69	2.17	1.61

In summary, according to the experiment results, our BR-FF-VND is able to outperform other state-of-the-art algorithms for the deterministic PFSPDP. Hence, it can be considered as a good candidate to be extended into a simheuristic. With the purpose of analyzing the performance of the proposed methods when more computational time is available, new tests were run. These tests used 5, 10, and 30 minutes, respectively (Table 1). Note that, regardless of the computational time employed, the BR-FF-VND algorithm shows a better average performance than the FF-ILS one.

8. Testing our simheuristic in the stochastic scenario

As explained in Section 6, the aforementioned instances were generalized using the log-normal probability distribution to model-processing times. Then, we solved the stochastic version of the

Table 3

Best and average deviations with respect to BR-FF-VND after 10 runs for 30 randomly selected instances—low variability scenarios

Instance	PFSPDP		PFSPDPST		PFSPDP		PFSPDPST	
	BR-FF-VND		SimBR-FF-VND ($h = 0.1$)		BR-FF-VND		SimBR-FF-VND ($h = 0.5$)	
	<i>BestSol</i>		BestDev (%)	<i>AvgDev</i> (%)	<i>BestSol</i>		BestDev (%)	<i>AvgDev</i> (%)
k1a0.1r0.5	21.00		−0.07	−0.38	21.00		−1.29	−2.71
k1a0.1r0.7	17.00		−0.02	−0.37	17.00		−1.35	−2.33
k1a0.7r0.1	81.00		0.00	−0.69	81.00		−0.41	−1.17
k1a0.7r0.3	81.00		0.05	0.01	81.00		−0.19	−0.31
k1a0.7r0.5	80.00		0.00	0.00	80.00		−0.26	−0.38
k2a0.3r0.1	72.00		−0.15	−1.07	72.00		−1.64	−1.72
k2a0.3r0.3	68.00		−0.02	−0.30	68.00		−0.85	−1.34
k2a1.0r0.7	150.00		0.46	0.41	150.00		0.01	−0.05
k2a1.0r1.0	142.00		−0.15	−0.46	142.00		−0.70	−1.01
k3a0.1r0.1	39.00		0.11	0.01	39.00		−0.56	−0.81
k3a0.1r0.3	33.00		−1.04	−3.99	33.00		−3.29	−6.51
k3a0.3r1.0	87.00		0.30	−0.71	87.00		−0.92	−1.68
k3a0.5r0.1	128.00		−0.03	−0.12	128.00		−0.56	−0.87
k3a0.5r0.3	130.00		−0.36	−0.82	129.00		−0.52	−0.74
k5a0.1r0.3	58.00		0.64	0.33	58.00		−1.56	−1.73
k5a0.5r0.3	186.00		0.06	−0.04	186.00		−0.45	−0.63
k5a0.7r0.7	234.00		−0.08	−0.53	234.00		−0.83	−1.19
k7a0.7r0.7	302.00		−0.07	−0.87	302.00		−0.62	−1.50
k7a0.7r1.0	299.00		−0.03	−0.48	299.00		−0.85	−1.25
k7a1.0r0.1	395.00		−0.04	−0.11	395.00		−0.19	−0.44
k7a1.0r0.3	351.00		0.03	−0.06	351.00		−0.11	−0.21
k7a1.0r0.5	344.00		−0.02	−0.21	344.00		−0.37	−0.49
k7a1.0r0.7	355.00		−0.12	−0.27	355.00		−0.52	−0.74
k7a1.0r1.0	331.00		−0.04	−0.07	331.00		−0.25	−0.37
k10a0.1r0.1	110.00		0.14	−1.78	110.00		−2.14	−3.68
k10a0.1r0.3	109.00		0.07	−0.72	109.00		−2.00	−2.86
k10a0.1r0.5	121.00		0.20	−0.41	121.00		−2.04	−2.73
k10a0.1r0.7	96.00		−0.04	−1.77	96.00		−2.48	−3.31
k10a0.1r1.0	99.00		−0.18	−2.74	99.00		−1.61	−4.28
k10a0.3r0.1	256.00		0.09	−0.85	256.00		−1.19	−2.04
Average	159.17		−0.01	−0.64	159.13		−0.99	−1.64

Note.: PFSPDPST, permutation flow-shop scheduling problem with delivery dates and cumulative payoff

problem by employing the Sim-BR-FF-VND simheuristic introduced in Section 5. Different levels of uncertainty have been considered, that is, $h \in \{0.1, 0.5, 1.0, 2.0\}$. Tables 3 and 4 show detailed results for the considered uncertainty levels (low (0.1 and 0.5) and high (1.0 and 2.0), respectively), where (a) *BestDev* (%) refers to the percentage deviations between the best solutions found by the SimBR-FF-VND (for the stochastic version of the problem) and the best solutions found by the BR-FF-VND (for the deterministic version); and (b) *AvgDev* (%) refers to the percentage deviations between the respective average values. These results are then summarized in Fig. 4 and detailed in the supplementary material (Villarinho et al., 2020) available at <https://data.mendeley.com/datasets/82ykchstgh/4>.

Table 4

Best and Average deviations with respect to BR-FF-VND after 10 runs for 30 randomly selected instances—high variability scenarios

Instance	PFSPDP	PFSPDPST		PFSPDP	PFSPDPST	
	BR-FF-VND	SimBR-FF-VND ($h = 1.0$)		BR-FF-VND	SimBR-FF-VND ($h = 2.0$)	
	<i>BestSol</i>	<i>BestDev</i> (%)	<i>AvgDev</i> (%)	<i>BestSol</i>	<i>BestDev</i> (%)	<i>AvgDev</i> (%)
k1a0.1r0.5	21.00	-2.60	-3.96	21.00	-4.48	-5.57
k1a0.1r0.7	17.00	-2.94	-3.29	17.00	-4.50	-4.91
k1a0.7r0.1	81.00	-0.83	-1.55	81.00	-1.46	-2.08
k1a0.7r0.3	81.00	-0.52	-0.71	81.00	-1.06	-1.25
k1a0.7r0.5	80.00	-0.62	-0.82	80.00	-1.25	-1.41
k2a0.3r0.1	71.00	-0.62	-0.88	71.00	-1.39	-1.62
k2a0.3r0.3	68.00	-1.60	-2.01	68.00	-2.42	-2.93
k2a1.0r0.7	150.00	-0.21	-0.32	150.00	-0.65	-0.74
k2a1.0r1.0	142.00	-1.18	-1.44	142.00	-1.76	-2.02
k3a0.1r0.1	39.00	-1.53	-1.92	39.00	-3.01	-3.46
k3a0.1r0.3	33.00	-4.86	-8.03	33.00	-7.08	-10.00
k3a0.3r1.0	87.00	-2.01	-2.60	87.00	-3.29	-3.94
k3a0.5r0.1	128.00	-1.04	-1.40	128.00	-1.72	-2.15
k3a0.5r0.3	129.00	-1.13	-1.35	129.00	-1.94	-2.15
k5a0.1r0.3	58.00	-3.65	-4.06	58.00	-6.08	-6.68
k5a0.5r0.3	186.00	-1.00	-1.20	186.00	-1.76	-1.97
k5a0.7r0.7	234.00	-1.45	-1.71	234.00	-2.10	-2.36
k7a0.7r0.7	302.00	-1.11	-2.02	302.00	-1.94	-2.73
k7a0.7r1.0	299.00	-1.55	-1.80	299.00	-2.11	-2.43
k7a1.0r0.1	395.00	-0.49	-0.74	395.00	-0.91	-1.17
k7a1.0r0.3	351.00	-0.27	-0.40	351.00	-0.52	-0.64
k7a1.0r0.5	344.00	-0.64	-0.79	344.00	-1.06	-1.23
k7a1.0r0.7	355.00	-0.77	-1.04	355.00	-1.14	-1.44
k7a1.0r1.0	331.00	-0.53	-0.62	331.00	-0.86	-0.99
k10a0.1r0.1	110.00	-3.71	-5.15	110.00	-5.68	-6.93
k10a0.1r0.3	109.00	-3.54	-4.61	109.00	-5.44	-6.68
k10a0.1r0.5	121.00	-3.52	-4.34	121.00	-5.54	-6.33
k10a0.1r0.7	96.00	-3.81	-4.45	96.00	-5.50	-6.13
k10a0.1r1.0	99.00	-2.67	-5.39	99.00	-4.24	-6.93
k10a0.3r0.1	256.00	-2.17	-2.92	256.00	-3.41	-4.21
Average	159.10	-1.75	-2.38	159.10	-2.81	-3.44

In effect, Fig. 4 shows that the uncertainty level directly impacts the stochastic rewards: for a low level of uncertainty ($h = 0.1$), *BestDev* (%) shows an expected value close to 0 (0.044); on the contrary, as the uncertainty level increases the values *BestDev* (%) diverge from the deterministic ones. A similar effect can be observed for the *AvgDev* (%) values.

Since risk analyses have an important role in the quantification of the worst possible scenarios, we expanded our simheuristic approach to consider a full risk analysis via the quantification of the VaR_α and $CVaR_\alpha$. Thus, for example, in the case of instance k10a0.3r0.1 and uncertainty level $h = 1.0$, Fig. 5 portrays the VaR_α and $CVaR_\alpha$ values associated with three confidence levels, $\alpha \in \{95\%, 97.5\%, 99\%\}$. Note that this instance has an expected reward of 250.44. The VaR_α could be

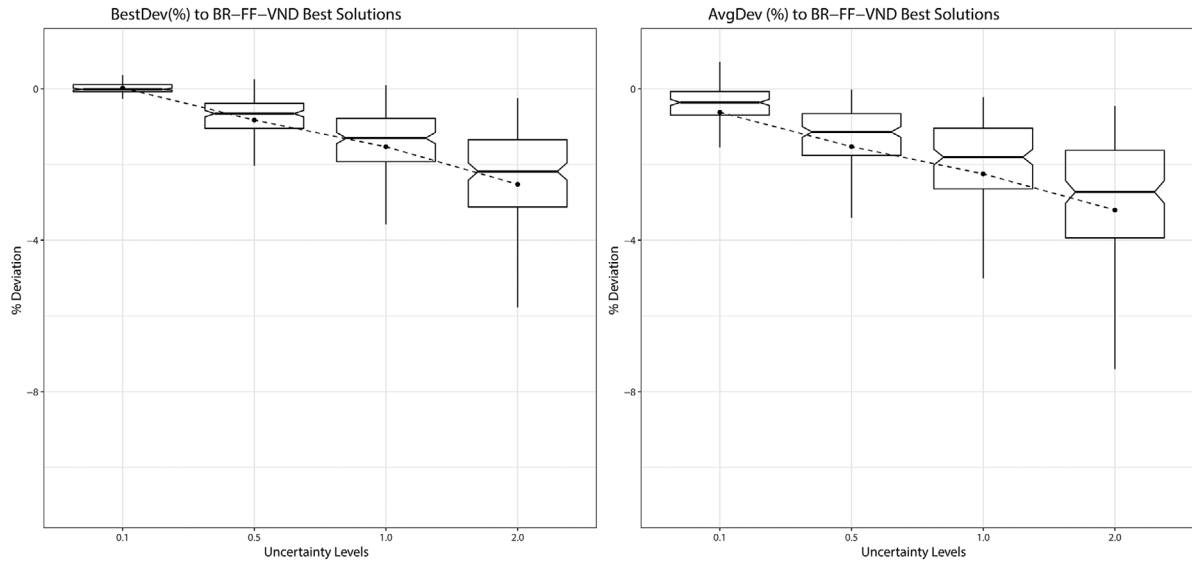


Fig. 4. Percentage deviation of *BestDev* (%) and *AvgDev* (%) to BR-FF-VND in the deterministic environment.

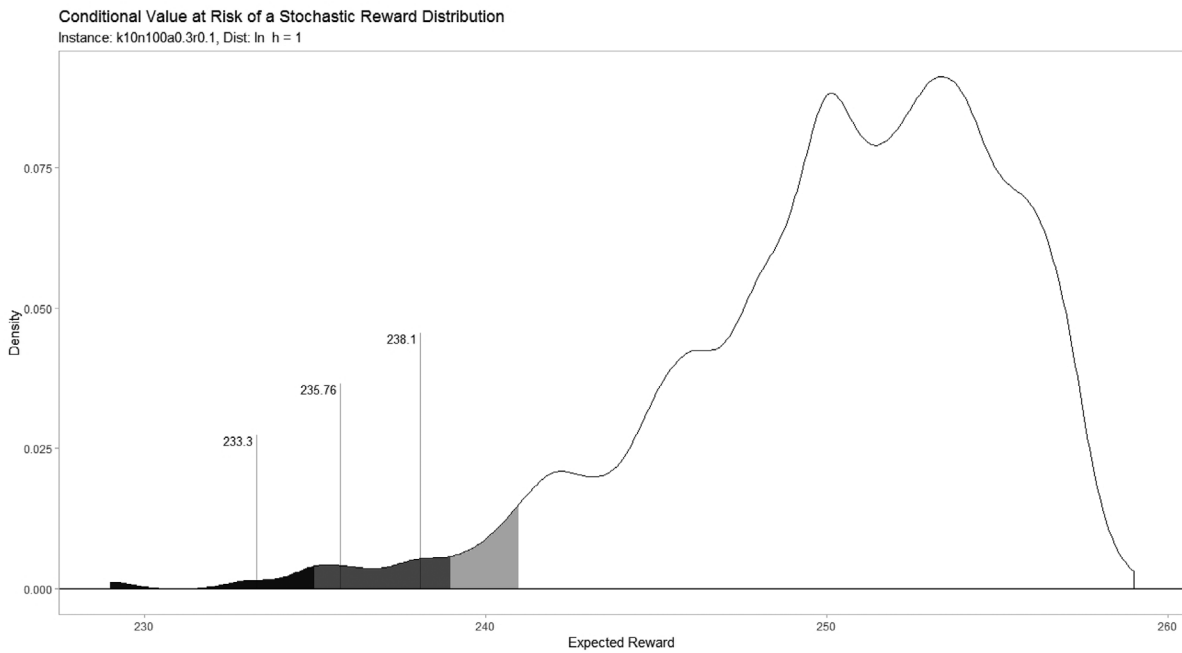


Fig. 5. Conditional VaR for instance k10a0.3r0.1 using different levels of α and $h = 1.0$.

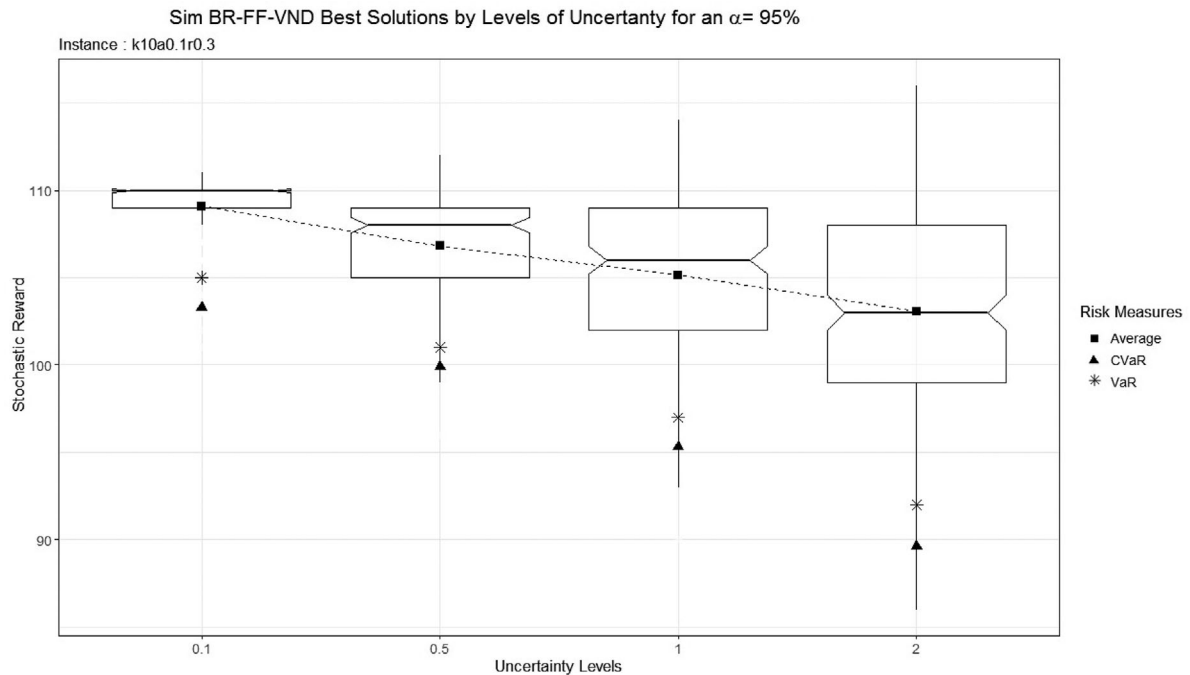


Fig. 6. Behavior of the $VaR_{95\%}$ and $CVaR_{95\%}$ on different levels of uncertainty, for instance k10a0.1r0.3.

interpreted as the worst scenario of a reward for a level of confidence α . In this case, the associated rewards would be 241, 239, and 235, respectively. Likewise, the $CVaR_{\alpha}$ can be interpreted as the average of the worst rewards for a level of confidence α . In this instance, the associated values are 238.10, 235.76, and 233.30, respectively.

Figure 6 depicts the behavior that has been observed across all the instances for the different values of α . Note that, as the level of uncertainty is increased, it has an impact on the processing times variability. This, in turn, influences the stochastic rewards generated, since more jobs will finish later on the last machine. Hence, the values of both risk metrics are reduced as the level of uncertainty increases.

9. Conclusions

This work analyzes a stochastic version of the PFSPDP. This problem is motivated by a real-life case regarding the books digitization in a library. In order to address this challenge, the FF heuristic, proposed by Fernandez-Viagas and Framinan (2015), was first extended into a BR algorithm (BR-FF). Next, a VND metaheuristic framework is incorporated to the BR-FF algorithm. Finally, the BR-FF-VND is extended into a simheuristic algorithm by integrating a Monte Carlo simulation component into it. This allows us to solve both the deterministic version of the PFSPDP and its stochastic counterpart with random processing times.

An extensive set of experiments was carried out to test the quality of the proposed algorithms. The solutions provided by our BR-FF-VND algorithm for the deterministic version of the problem are shown to be competitive when compared with the recently published work by Pessoa and Andrade (2018). Moreover, our simheuristic approach is the first in the literature dealing with the stochastic version of the PFSPDP. The numerical experiments allow us to study how the level of uncertainty affects the solutions obtained for the stochastic version of the problem. In addition, the worst-case scenarios were analyzed using two well-known risk metrics: the VaR and CVaR. As expected, an increase in the level of uncertainty has an impact on the tails of the probability distribution used to model stochastic processing times. This pattern affects the rewards computed by the VaR and CVaR, since both risk metrics consider the behavior of the rewards in the tails of the distribution.

Several research lines can be considered for future work, among them: (a) to consider a biobjective formulation of the problem, where both expected reward and solution risk are simultaneously optimized and the associated Pareto frontier is explored; (b) to incorporate other sources of randomness in the PFSPDP, such as machine breakdowns; and (c) to analyze the performance of our BR algorithm when more complex functions are employed to model rewards by meeting deadlines.

Acknowledgments

This work was supported by the Brazilian Coordination for the Improvement of Higher Level Personnel (CAPES) under grant (number 001); the Brazilian National Council for Scientific and Technological Development (CNPq) under grant (number 307403/2019-0); the Carlos Chagas Filho Research Support Foundation of the State of Rio de Janeiro (FAPERJ) under grants (numbers 202.673/2018, E-26/010.002576/2019 and 211.086/2019) and Spanish Ministry of Science (PID2019-111100RB-C21, RED2018-102642-T).

References

- Abdinnia, H., Glock, C.H., Brill, A., 2016. New simple constructive heuristic algorithms for minimizing total flow-time in the permutation flowshop scheduling problem. *Computers & Operations Research* 74, 165–174.
- Andrade, C.E., Silva, T., Pessoa, L.S., 2019. Minimizing flowtime in a flowshop scheduling problem with a biased random-key genetic algorithm. *Expert Systems with Applications* 128, 67–80.
- Artzner, P., Delbaen, F., Eber, J.M., Heath, D., 1999. Coherent measures of risk. *Mathematical Finance* 9, 3, 203–228.
- Baker, K.R., Althimer, D., 2012. Heuristic solution methods for the stochastic flow shop problem. *European Journal of Operational Research* 216, 1, 172–177.
- Baker, K.R., Trietsch, D., 2011. Three heuristic procedures for the stochastic, two-machine flow shop problem. *Journal of Scheduling* 14, 5, 445–454.
- Banerjee, B., 1965. Single facility sequencing with random execution times. *Operations Research* 13, 3, 358–364.
- Choi, S., Wang, K., 2012. Flexible flow shop scheduling with stochastic processing times: a decomposition-based approach. *Computers & Industrial Engineering* 63, 2, 362–373.
- Cui, W., Lu, Z., Li, C., Han, X., 2018. A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops. *Computers & Industrial Engineering* 115, 342–353.

- Della Croce, F., Grosso, A., Salassa, F., 2011. A matheuristic approach for the total completion time two-machines permutation flow shop problem. Proceedings of the European Conference on Evolutionary Computation in Combinatorial Optimization. Springer, Berlin, pp. 38–47.
- Dodin, B., 1996. Determining the optimal sequences and the distributional properties of their completion times in stochastic flow shops. *Computers & Operations Research* 23, 9, 829–843.
- Dong, X., Huang, H., Chen, P., 2009. An iterated local search algorithm for the permutation flowshop problem with total flowtime criterion. *Computers & Operations Research* 36, 5, 1664–1669.
- Fernandez-Viagas, V., Framinan, J.M., 2015. A new set of high-performing heuristics to minimise flowtime in permutation flowshops. *Computers & Operations Research* 53, 68–80.
- Fernandez-Viagas, V., Framinan, J.M., 2017. A beam-search-based constructive heuristic for the PFSP to minimise total flowtime. *Computers & Operations Research* 81, 167–177.
- Fernandez-Viagas, V., Ruiz, R., Framinan, J.M., 2017. A new vision of approximate methods for the permutation flowshop to minimise makespan: state-of-the-art and computational evaluation. *European Journal of Operational Research* 257, 3, 707–721.
- Fernandez-Viagas, V., Valente, J.M., Framinan, J.M., 2018. Iterated-greedy-based algorithms with beam search initialization for the permutation flowshop to minimise total tardiness. *Expert Systems with Applications* 94, 58–69.
- Ferrer, A., Guimarans, D., Ramalhinho, H., Juan, A.A., 2016. A BRILS metaheuristic for non-smooth flow-shop problems with failure-risk costs. *Expert Systems with Applications* 44, 177–186.
- Framinan, J.M., Fernandez-Viagas, V., Perez-Gonzalez, P., 2019. Using real-time information to reschedule jobs in a flowshop with variable processing times. *Computers & Industrial Engineering* 129, 113–125.
- Framinan, J.M., Leisten, R., 2003. An efficient constructive heuristic for flowtime minimisation in permutation flow shops. *Omega* 31, 4, 311–317.
- Framinan, J.M., Leisten, R., 2008. Total tardiness minimization in permutation flow shops: a simple approach based on a variable greedy algorithm. *International Journal of Production Research* 46, 22, 6479–6498.
- Framinan, J.M., Leisten, R., Ruiz-Usano, R., 2005. Comparison of heuristics for flowtime minimisation in permutation flowshops. *Computers & Operations Research* 32, 5, 1237–1254.
- Gholami-Zanjani, S.M., Hakimifar, M., Nazemi, N., Jolai, F., 2017. Robust and fuzzy optimisation models for a flow shop scheduling problem with sequence dependent setup times: a real case study on a PCB assembly company. *International Journal of Computer Integrated Manufacturing* 30, 6, 552–563.
- Gonzalez-Martin, S., Juan, A.A., Riera, D., Elizondo, M.G., Ramos, J.J., 2018. A simheuristic algorithm for solving the arc routing problem with stochastic demands. *Journal of Simulation* 12, 1, 53–66.
- Gonzalez-Neira, E., Montoya-Torres, J., Barrera, D., 2017. Flow-shop scheduling problem under uncertainties: review and trends. *International Journal of Industrial Engineering Computations* 8, 4, 399–426.
- Gourgand, M., Grangeon, N., Norre, S., 2000. A review of the static stochastic flow-shop scheduling problem. *Journal of Decision Systems* 9, 2, 1–31.
- Gruler, A., Fikar, C., Juan, A.A., Hirsch, P., Contreras-Bolton, C., 2017. Supporting multi-depot and stochastic waste collection management in clustered urban areas via simulation–optimization. *Journal of Simulation* 11, 1, 11–19.
- Gruler, A., Panadero, J., de Armas, J., Moreno, J.A., Juan, A.A., 2020. A variable neighborhood search simheuristic for the multiperiod inventory routing problem with stochastic demands. *International Transactions in Operational Research* 27, 1, 314–335.
- Hansen, P., Mladenović, N., 2003. Variable neighborhood search. In Martí, R., Pardalos, P., Resende, M. (eds) *Handbook of Metaheuristics*. Springer, Cham, pp. 145–184.
- Hatami, S., Calvet, L., Fernandez-Viagas, V., Framinan, J.M., Juan, A.A., 2018. A simheuristic algorithm to set up starting times in the stochastic parallel flowshop problem. *Simulation Modelling Practice and Theory* 86, 55–71.
- Johnson, S.M., 1954. Optimal two- and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly* 1, 1, 61–68.
- Kalczynski, P.J., Kamburowski, J., 2006. A heuristic for minimizing the expected makespan in two-machine flow shops with consistent coefficients of variation. *European Journal of Operational Research* 169, 3, 742–750.
- Kamburowski, J., 1999. Stochastically minimizing the makespan in two-machine flow shops without blocking. *European Journal of Operational Research* 112, 2, 304–309.

- Kamburowski, J., 2000. On three-machine flow shops with random job processing times. *European Journal of Operational Research* 125, 2, 440–448.
- Ladhari, T., Rakrouki, M.A., 2009. Heuristics and lower bounds for minimizing the total completion time in a two-machine flowshop. *International Journal of Production Economics* 122, 2, 678–691.
- Laha, D., Sarin, S.C., 2009. A heuristic to minimize total flow time in permutation flow shop. *Omega* 37, 3, 734–739.
- Lee, W.C., Chung, Y.H., 2013. Permutation flowshop scheduling to minimize the total tardiness with learning effects. *International Journal of Production Economics* 141, 1, 327–334.
- Liefooghe, A., Basseur, M., Humeau, J., Jourdan, L., Talbi, E.G., 2012. On optimizing a bi-objective flowshop scheduling problem in an uncertain environment. *Computers & Mathematics with Applications* 64, 12, 3747–3762.
- Liu, J., Reeves, C.R., 2001. Constructive and composite heuristic solutions to the $p//\Sigma c_i$ scheduling problem. *European Journal of Operational Research* 132, 2, 439–452.
- Makino, T., 1965. On a scheduling problem. *Operations Research Society of Japan* 8, 32–44.
- Molina-Sánchez, L., González-Neira, E., 2016. GRASP to minimize total weighted tardiness in a permutation flow shop environment. *International Journal of Industrial Engineering Computations* 7, 1, 161–176.
- Nagano, M.S., Moccellini, J., 2008. Reducing mean flow time in permutation flow shop. *Journal of the Operational Research Society* 59, 7, 939–945.
- Nawaz, M., Enscore Jr, E.E., Ham, I., 1983. A heuristic algorithm for the m -machine, n -job flow-shop sequencing problem. *Omega* 11, 1, 91–95.
- Pagès-Bernaus, A., Ramalhinho, H., Juan, A.A., Calvet, L., 2019. Designing e-commerce supply chains: a stochastic facility–location approach. *International Transactions in Operational Research* 26, 2, 507–528.
- Pan, Q.K., Ruiz, R., 2013. A comprehensive review and evaluation of permutation flowshop heuristics to minimize flow-time. *Computers & Operations Research* 40, 1, 117–128.
- Pan, Q.K., Tasgetiren, M.F., Liang, Y.C., 2008. A discrete differential evolution algorithm for the permutation flowshop scheduling problem. *Computers & Industrial Engineering* 55, 4, 795–816.
- Pasupathy, T., Rajendran, C., Suresh, R., 2006. A multi-objective genetic algorithm for scheduling in flow shops to minimize the makespan and total flow time of jobs. *The International Journal of Advanced Manufacturing Technology* 27, 7–8, 804–815.
- Pessoa, L.S., Andrade, C.E., 2018. Heuristics for a flowshop scheduling problem with stepwise job objective function. *European Journal of Operational Research* 266, 3, 950–962.
- Pflug, G.C., 2000. Some remarks on the value-at-risk and the conditional value-at-risk. In Uryasev, S.P. (ed.) *Probabilistic Constrained Optimization*. Springer, Boston, MA, pp. 272–281.
- Potts, C., 1985. Analysis of heuristics for two-machine flow-shop sequencing subject to release dates. *Mathematics of Operations Research* 10, 4, 576–584.
- Quintero-Araujo, C.L., Caballero-Villalobos, J.P., Juan, A.A., Montoya-Torres, J.R., 2017. A biased-randomized meta-heuristic for the capacitated location routing problem. *International Transactions in Operational Research* 24, 5, 1079–1098.
- Resende, M.G., Ribeiro, C.C., 2016. *Optimization by GRASP*. Springer, Berlin.
- Rockafellar, R.T., Uryasev, S., 2000. Optimization of conditional value-at-risk. *Journal of Risk* 2, 21–42.
- Rossi, F.L., Nagano, M.S., 2019. Heuristics for the mixed no-idle flowshop with sequence-dependent setup times and total flowtime criterion. *Expert Systems with Applications* 125, 40–54.
- Rossi, F.L., Nagano, M.S., Sagawa, J.K., 2017. An effective constructive heuristic for permutation flow shop scheduling problem with total flow time criterion. *The International Journal of Advanced Manufacturing Technology* 90, 1–4, 93–107.
- Seddik, Y., Gonzales, C., Kedad-Sidhoum, S., 2013. Single machine scheduling with delivery dates and cumulative pay-offs. *Journal of Scheduling* 16, 3, 313–329.
- Street, A., 2010. On the conditional value-at-risk probability-dependent utility function. *Theory and Decision* 68, 1–2, 49–68.
- Ta, Q.C., Billaut, J.C., Bouquard, J.L., 2018. Matheuristic algorithms for minimizing total tardiness in the m -machine flow-shop scheduling problem. *Journal of Intelligent Manufacturing* 29, 3, 617–628.
- Tasgetiren, M.F., Pan, Q.K., Suganthan, P.N., Chen, A.H., 2011. A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flow shops. *Information Sciences* 181, 16, 3459–3475.

- Villarinho, P.A., Panadero, J., Pessoa, L.S., Juan, A.A., Cyrino Oliveira, F.L., 2020. Supplementary material with individualized results for all test instances in “a simheuristic algorithm for the stochastic permutation flow-shop problem with delivery dates and cumulative payoffs.” *Mendeley Data*, V4, <https://doi.org/10.17632/82ykchstgh.4>.
- Wang, L., Zhang, L., Zheng, D.Z., 2005. A class of hypothesis-test-based genetic algorithms for flow shop scheduling with stochastic processing time. *The International Journal of Advanced Manufacturing Technology* 25, 11–12, 1157–1163.
- Yu, A.J., Seif, J., 2016. Minimizing tardiness and maintenance costs in flow shop scheduling by a lower-bound-based GA. *Computers & Industrial Engineering* 97, 26–40.