Exact solutions for the two-machine robust flow shop with budgeted uncertainty

Mario Levorato, Rosa Figueiredo, Yuri Frota

Exact solutions for the two-machine robust flow shop with budgeted uncertainty

Mario Levorato[a,b,*], Rosa Figueiredo[b], Yuri Frota[a]

[a] *Instituto de Computação, Universidade Federal Fluminense, Niterói, Brazil*
[b] *Laboratoire Informatique d'Avignon, Avignon Université, Avignon, France*

Abstract

This work proposes an exact solution approach for the two-machine robust flow shop problem, where operation processing times are uncertain and vary in a given interval. Based on the concept of budgeted uncertainty, the objective is to obtain a robust scheduling that minimizes the makespan of the restricted worst-case scenario, where only a subset of job processing times will oscillate to their worst-case values. To our knowledge, this is the first work to obtain optimal solutions to this problem, by extending two classical MILP formulations for the deterministic case and combining them with a Column-and-Constraint Generation framework. For this purpose, a dynamic programming algorithm was also developed, allowing the identification of worst-case scenarios in polynomial time. Experiments on a set of literature instances confirm the efficacy of our approach, including a case study that shows little overhead in the expected solution value of obtained robust solutions.

Keywords: Scheduling, Robust Optimization, Flow Shop, Budget of uncertainty, Column-and-Constraint Generation

1. Introduction

In permutation flow shop scheduling problems, the operations concerning each job are performed in a serial flow (i.e., in a specific production sequence)

*Corresponding author
Email addresses: mlevorato@ic.uff.br (Mario Levorato), rosa.figueiredo@univ-avignon.fr (Rosa Figueiredo), yuri@ic.uff.br (Yuri Frota)

on each machine, and the processing order of the jobs is the same for each subsequent step. Such configuration is commonly used in assembly lines throughout several types of industry, including chemical (Cartwright & Long, 1993), petrochemical (Deal et al., 1994), automobile manufacturing (Yan et al., 2003), electronic (Jin et al., 2002), food and metallurgical (Hall & Sriskandarajah, 1996). In all these applications, maintaining a continuous flow of processing tasks while minimizing idle time and waiting time is highly important, guaranteeing process efficiency and increasing production rates and profits.

The Permutation Flow Shop (PFS) scheduling has been widely investigated considering deterministic processing times, assuming input data are accurate and known in advance. However, in real-world industrial settings, manufacturing systems usually operate in uncertain environments where interruptions (essentially random in nature) restrict the execution of production schedules precisely as they were planned. In particular, variation in processing times and other stochastic events (e.g., machine breakdowns, due-date changes, order cancellations, and raw material shortages) bring increased variability in production systems and may influence the optimization model's quality and feasibility (Sabuncuoglu & Goren, 2009).

A typical solution approach applied to these cases is Stochastic Optimization (Heyman & Sobel, 1982), whose fundamental premise states that the unknown parameters can be described using probabilities. On the other hand, Robust Optimization (RO) (Ben-Tal & Nemirovski, 2002) provides a different approach to optimization problems under uncertain conditions. The unknown parameters are modeled as belonging to an uncertainty set, making it a more suitable modeling approach when the uncertainty interval is known, but not necessarily the probability distribution. The goal for RO is to make a feasible decision, no matter what the constraints turn out to be, which is optimal for the worst-case objective function.

Assuming processing times are uncertain and vary in a given interval, the objective of the present work is to provide exact solutions for the two-machine Robust Permutation Flow Shop (2RPFS). The only information required is the

lower and upper bounds of processing times, obtained from historical data. We are interested in a job permutation that minimizes the worst-case *makespan* for any possible realization of job processing times under the budgeted uncertainty set (Bertsimas & Sim, 2004). Unlike other robust optimization models, which generate only one conservative solution, the budgeted approach allows the adjustment of the level of conservatism of the solution, allowing the incorporation of different attitudes toward risk (e.g., risk-averse, risk-neutral, or risk-seeking). As a result, the decision-maker can select the schedule that achieves the best balance between robustness and optimality.

This text adopts the following structure. Section 2 starts with an introduction to the classical Permutation Flow Shop Problem, whose processing times are certain, followed by Section 3, a literature review on solution methods for the PFS problem with uncertain processing times. The two-machine Robust Permutation Flow Shop Problem is introduced in Section 4, together with two proposed robust counterpart formulations. Our exact solution approach, based on Column-and-Constraint Generation (C&CG), is explained in Section 5. Finally, in Section 6, we discuss the obtained performance, based on extensive computational experiments on existing literature instances.

2. The Deterministic Permutation Flow Shop Problem

This section presents the Permutation Flow Shop Problem (PFSP) to minimize the *makespan*. Following the well-known $\alpha|\beta|\gamma$ notation for scheduling problems, established by Graham et al. (1979), where α represents the machine environment, β stands for job characteristics, and γ symbolizes the objective function, this problem is denoted as $F|prmu|C_{max}$. Since job processing time values are assumed to be known in advance, we will use the term deterministic when referring to this version of the problem.

The problem can be stated as follows. Consider a production planning process consisting of a set $\mathbb{J} = \{J_1, J_2, \ldots, J_n\}$ of n jobs to be executed in a set $\mathbb{M} = \{M_1, M_2, \ldots, M_m\}$ of m machines. In this process, every job J_i is composed of m stages $O_{1,i}, O_{2,i}, \ldots, O_{m,i}$, named operations. Each job operation

$O_{r,i}$ must only be executed on machine M_r. Every operation $O_{r,i}$ has a non-negative processing time $p_{r,i}$ forming the matrix $P \in \mathbb{R}^{+}_{\mathbb{M}\times\mathbb{J}}$. At any time, each machine cannot execute more than one operation. Operation $O_{r,i}$ can only be executed after operation $O_{r-1,i}$ is finished. Preemption is not allowed: once an operation is started, it must be completed without any interruption. The permutation flow shop's particularity is that the sequence in which the jobs are processed (permutation) is the same for all machines. Such sequence is defined by a permutation σ: $\{1,\ldots,n\} \longrightarrow \mathbb{J}$, with $\sigma(j)$ indicating the jth job to be executed. We call Σ the set of all permutations of n jobs, hence $\sigma \in \Sigma$. The completion time of an operation $O_{r,\sigma(j)}$, denoted by $C_{r,\sigma(j)}$, can be defined by the recurrence:

$$C_{r,\sigma(j)} = \begin{cases} p_{r,\sigma(j)} & \text{if } r = 1 \text{ and } j = 1, \\ C_{r,\sigma(j-1)} + p_{r,\sigma(j)} & \text{if } r = 1 \text{ and } j > 1, \\ C_{r-1,\sigma(j)} + p_{r,\sigma(j)} & \text{if } r > 1 \text{ and } j = 1, \\ max(C_{r,\sigma(j-1)},\ C_{r-1,\sigma(j)}) + p_{r,\sigma(j)} & \text{if } r > 1 \text{ and } j > 1. \end{cases}$$

The completion time of a job J_i is defined as its completion time on the last machine: $C_{m,i}$. The *makespan* is the maximum completion time C_{max}, considering all jobs, i.e., $C_{max} = \max_{i\in\{1,\ldots n\}} C_{m,i} = C_{m,\sigma(n)}$. The PFSP objective is to find a sequence of jobs (permutation σ) that minimizes the *makespan*.

As far as the *makespan* objective is concerned, the PFSP was proved strongly NP-hard by Garey et al. (1976) for instances with three or more machines. However, the two-machine version of the problem can be solved in $O(n \log n)$ time with the well-known algorithm proposed by Johnson (1954) in one of the pioneering papers in the scheduling literature. Indeed, for the two-machine case, the optimal *makespan* for the deterministic flow shop problem is the same with or without the permutation constraint (Pinedo, 2016).

Several Mixed-integer linear programming (MILP) models have been developed for the PFSP with an arbitrary number of machines. For a comparative analysis among the best performing PFSP formulations, we refer the reader to Tseng et al. (2004), which presents an empirical study based on a standard set of 60 problem instances.

3. Literature Review

This section provides an overview of the flow shop problem with uncertain processing times, concerning existing Stochastic Optimization methods (Section 3.1) and Robust Optimization approaches (Section 3.2).

3.1. Stochastic Optimization

González-Neira et al. (2017) surveyed 100 papers that study uncertainty in different variations of flow shop scheduling problems, published between 2001 and 2016. According to their analysis, the two most common uncertain parameters are processing times and machine breakdowns. The majority of works are related to Stochastic Optimization, including heuristics (Dodin, 1996; Elmaghraby & Thoney, 1999; Baker & Trietsch, 2011; Framinan & Perez-Gonzalez, 2015), simheuristics (Ferone et al., 2016), probabilistic hybrid heuristics (Laha & Chakraborty, 2007), branch-and-bound (Balasubramanian & Grossmann, 2002), and simulation (Framinan et al., 2018).

It is worth noting that Stochastic Optimization approaches model random parameters using probability distributions, which may be difficult to infer in many cases. Additionally, optimizing the expected value of an objective may not be the best alternative for processes that incorporate only a small number of trials. The benefits of optimum expected value shall only be visible in the long haul, after a large number of tests.

The problem was also investigated from the viewpoint of fuzzy programming (Hong & Chuang, 2005; Mitkowski et al., 2017) and stability analysis (Lai & Sotskov, 1999; Braun et al., 2002).

3.2. Robust Optimization

When applying Robust Optimization techniques, no assumptions are necessary concerning the underlying probability distribution of uncertain data. Also, different approaches towards risk can be incorporated into the problem. As far as robust scheduling problems are concerned, the objective is to optimize a performance measure considering the worst possible scenario, thus developing

a schedule that will perform relatively well under a wide range of possible realizations of processing times. Different optimization criteria may be used to choose a robust solution (Aissi et al., 2009). The first and most straightforward criterion is the *minimax* (also known as the *absolute robust* criterion). In this case, given a minimization problem, the robust decision is made by choosing a solution that minimizes the highest cost over all possible scenarios.

A second possible criterion is minimax regret, aiming to find the least maximum regret over all possible scenarios. Regret can be either defined as the *difference* or the *ratio* between the resulting cost of the candidate decision and the cost of the decision that would have been taken if uncertain input data were known in advance (before the decision time, i.e., before solving the problem). In the first case, where regret is defined as a difference, the so-called *robust deviation* decision is obtained. For the latter case (regret being the ratio of two values), the resulting decision is the *relative robust* decision.

Concerning the uncertain nature of the problem, scenarios represent the set of possible realizations of processing time values. When applying RO, there are two usual ways of describing the set of scenarios. In the discrete case, an explicit scenario list is given, i.e., one processing time matrix P^{λ} for each scenario λ. In the interval case, for each operation $O_{r,i}$ concerning the execution of job i on machine r, a range $[p^L_{r,i}, p^U_{r,i}]$ of lower and upper bounds of processing times is defined. The so-called continuous processing time interval involves, therefore, an infinite number of scenarios. Regardless of the scenario representation approach, interval or discrete, once the robust counterpart formulations have been defined, developing an appropriate solution method remains a challenge.

Table 1 summarizes existing works regarding the makespan-objective Robust Permutation Flow Shop Problem, in terms of optimization criterium (problem type), solution approach (heuristics or exact methods), the number of machines, and how processing time uncertainty was represented (discrete or interval).

Given the minimax regret makespan criterion, Kouvelis et al. (2000) studied both the discrete and interval cases of processing time uncertainty. Their work proposed branch-and-bound and heuristic procedures to obtain robust solutions,

along with experiments based on a considerable set of randomly generated instances, with no more than 15 jobs, however. They also provided NP-hardness proof for the discrete scenario case of the robust flow shop with $m = 2$. Very recently, the 2-machine interval processing time case has been proven to be NP-hard by Shafransky & Shinkarevich (2020).

Concerning the existing solution approaches involving only discrete-scenario uncertainty, Kasperski et al. (2012) studied the two-machine permutation flow shop problem ($F_2 \mid prmu \mid C_{max}$) with uncertain data, whose deterministic counterpart is known to be polynomially solvable. The work proved that the minimax and minimax regret versions of the problem are strongly NP-hard even for two scenarios. The authors developed a polynomial-time approximation scheme (PTAS) algorithm to be used with the minimax version of the problem when the number of scenarios is constant. They also stated that the minimax regret version is not at all approximable, even for two scenarios.

Regarding the interval scenario case with the minimax makespan criterion, the computational complexity of the robust flow shop with $m = 2$ is still an unsolved open problem (Kasperski et al., 2012). Note that the problem is NP-hard for $m \geq 3$ following the complexity of the deterministic problem. Several solution strategies deal exclusively with processing time uncertainty represented as intervals of known bounds. Averbakh (2006) studied the minimax regret flow shop with m machines but only two jobs. Józefczyk & Siepak (2013) proposed a Scatter Search metaheuristic for the PFSP and other two scheduling problems with interval uncertainty, based on the minimax regret criterion.

Unlike previous works, Ying (2015) adopted a new approach, which searches for a minimax robust schedule given the restricted worst-case scenario, based

Problem Type	Heuristics / Approximation	Exact methods
Minimax Regret	**2 machines**: Greedy (Kouvelis et al., 2000)$^{D, I}$ **3 machines**: Evolutionary (Ćwik & Józefczyk, 2015)^{I} **m machines**: Constructive (Ćwik & Józefczyk, 2018)^{I} Scatter Search (Józefczyk & Siepak, 2013)^{I}	**2 machines**: Branch & Bound (Kouvelis et al., 2000)$^{D, I}$ **2 jobs**: $O(m)$ (Averbakh, 2006)^{I}
Minimax	**2 machines**: PTAS (Kasperski et al., 2012)^{D}	-
Minimax, Budgeted uncertainty	**2 machines**: SA and IG (Ying, 2015)^{I}	-

Tab. 1 Summary of algorithms listed in the literature review regarding the Robust PFSP (*makespan* objective). For each work, we specify how processing time uncertainty was represented: a **D** means discrete processing time matrices; an **I** means processing time intervals.

on the Bertsimas & Sim (2004) budgeted uncertainty set. Simulated annealing (SA) and Iterated Greedy (IG) metaheuristics were applied to solve the problem over a set of 300 randomly generated instances.

In the same year, Ćwik & Józefczyk (2015) proposed an evolutionary algorithm for the minimax regret makespan robust flow shop with three machines, assuming processing times belonged to known intervals. Later, the same authors proposed another solution approach for an arbitrary number of machines (Ćwik & Józefczyk, 2018), where a constructive algorithm based on the Nawaz-Enscore-Ham (NEH) heuristic (Nawaz et al., 1983) has been introduced and experimentally evaluated against two other heuristic algorithms: the authors' evolutionary algorithm and a Middle Interval heuristic.

4. The two-machine Robust Flow Shop Problem

From this moment on, we will concentrate on the two-machine robust flow shop problem. Different optimization criteria may be used to choose a robust solution (Gerodimos et al., 1998; Aissi et al., 2009). Our research focuses on the *minimax* criterion, also known as the *absolute robust* criterion. In this case, considering a minimization problem, the robust decision is made by choosing a solution that minimizes the highest solution value over all possible scenarios, according to a predefined uncertainty set.

Given that the computational complexity remains an open problem for 2RPFS under budgeted uncertainty, in this work, we fill a gap in the literature by providing an exact solution method (see Table 1 from Section 3). Furthermore, it is worth noting that the models and the solution method presented in this section can be generalized to the m-machine variant of the robust PFSP, which is NP-hard for $m \geq 3$, following the complexity of the deterministic problem.

After defining the 2RPFS problem, this section describes the application of the budgeted uncertainty set. Finally, two robust counterpart formulations are proposed, based on well-known Mixed-Integer Linear Programming (MILP) formulations for the deterministic problem.

4.1. Problem statement

Assume the matrix of individual processing times $\mathbf{P} = \{p_{r,i}\}$ as uncertain. A scenario λ is defined as a realization of uncertainty and, for each λ, there is a unique matrix of processing times $\mathbf{P}^{\lambda} = \{p^{\lambda}_{r,i} \in \mathbb{R} : r = 1, 2, i = 1, 2, \ldots, n\}$. We define Λ as the set indexing all possible scenarios.

Let $\varphi(\sigma, \mathbf{P}^{\lambda})$ be the makespan of a sequence $\sigma \in \Sigma$ in scenario λ (i.e., given processing time matrix $\mathbf{P}^{\lambda}$). The objective of the two-machine robust (minimax makespan) flow shop is to find a job permutation $\sigma \in \Sigma$ that *minimizes* the *maximum* possible **makespan** over all possible scenarios $\lambda \in \Lambda$:

$$\textbf{2RPFS:} \quad \min_{\sigma \in \Sigma} \max_{\lambda \in \Lambda} \{\varphi(\sigma, \mathbf{P}^{\lambda})\} \tag{1}$$

For any sequence $\sigma \in \Sigma$, the value

$$\mathcal{Z}(\sigma) := \max_{\lambda \in \Lambda} \{\varphi(\sigma, \mathbf{P}^{\lambda})\} \tag{2}$$

is called the *worst-case makespan* or *robust cost* for σ. The maximizer in (2) is called a worst-case scenario for σ.

4.2. Budgeted uncertainty set for the 2RPFS problem

Ying (2015) compared the three classical Robust-Counterpart Optimization (RCO) models in terms of the number of variables, the number of required constraints, and if the respective formulation is linear or not. The first and simplest model, by Soyster (1973), consists of a linear formulation that presents the smallest number of variables and constraints. However, it is not possible to adjust its degree of solution conservatism. Following a new approach, Ben-Tal & Nemirovski (2000) proposed an RCO model with safety parameters, which allow a trade-off between robustness and performance. However, the resulting formulation is non-linear (conic quadratic) and more challenging to solve than the original problem. Finally, the model developed by Bertsimas & Sim (2004) provided a linear formulation that allows controlling the level of conservatism of the robust solution without resulting in a substantial increase in problem size. With the inclusion of a budget parameter for each constraint, it is possible to adjust the number of coefficients that simultaneously take their largest variations,

thus providing a compromise between robustness and optimality. Therefore, the so-called budgeted uncertainty set, defined by this RCO model, comes as a natural choice to model processing time uncertainty in scheduling problems.

The budgeted uncertainty set for the 2RPFS problem was proposed by Ying (2015). We reproduce their definition below, with some modifications in notation as well as additional comments.

As stated in Section 3, there are two common ways of representing scenario set Λ. Given the interval approach for representing uncertain values, consider two positive processing time matrices $\overline{\mathbf{P}}=\{\overline{p}_{r,i}\}$ and $\widehat{\mathbf{P}}=\{\widehat{p}_{r,i}\}$, that represent the nominal values of and the maximum allowed deviations of $\mathbf{P}$, respectively. Additionally, we introduce two positive integers Γ_1 and Γ_2, which will be called budget parameters, that denote the maximum number of operations whose uncertain processing times can reach their worst-case values in machines M_1 and M_2, respectively. We can define the bounded (processing time) uncertainty sets of operations in M_1 and M_2, denoted as $\mathcal{U}_r$ $(r = 1, 2)$, as follows:

$$\mathcal{U}_r = \left\{ \mathbf{P}_r = \{p_{r,i}\} : p_{r,i} = \overline{p}_{r,i} + \delta_{r,i}\, \widehat{p}_{r,i},\ \delta_{r,i} \in \{0,1\}, i \in \{1,\ldots,n\},\ \sum_{i=1}^{n} \delta_{r,i} \leq \Gamma_r \right\}, \quad (3)$$

where $\mathbf{P}_r$ is the projection of $\mathbf{P}$ in the space defined by machine M_r $(r = 1, 2)$.

We can now define the *budgeted uncertainty set* as:

$$\mathcal{U}_\Gamma = \mathcal{U}_{(\Gamma_1,\Gamma_2)} = \mathcal{U}_1 \times \mathcal{U}_2.$$

Notice that a scenario $\lambda \in \Lambda$ is defined by the matrix $P^\lambda \in \mathcal{U}_\Gamma$. Also, for a given $r \in \mathbb{M}$ and $i \in \mathbb{J}$, let $\delta^\lambda_{r,i}$ be the value defining the deviation of the processing time regarding the execution of job i on machine r in scenario λ, i.e., $p^\lambda_{r,i} = \overline{p}^\lambda_{r,i} + \delta^\lambda_{r,i}\widehat{p}^\lambda_{r,i}$. Therefore, the total number of operations whose processing time can deviate to its maximum value in machine M_r is limited to Γ_r.

The main advantage of applying budgeted uncertainty sets is the ability to model the risk-averseness of the decision-maker by varying Γ_1 and Γ_2. As mentioned by Bertsimas & Sim (2004), the idea is that an event where all uncertain parameters $p_{r,i}$ reach their worst-case values at the same time has a very low probability of happening. In particular, higher values of Γ_1 and Γ_2 lead to larger uncertainty sets and thus more conservative solutions. When

$\Gamma_1 = \Gamma_2 = 0$, the problem is equivalent to the nominal problem, i.e., the PFSP with two machines. If $\Gamma_1 = \Gamma_2 = n$, we obtain the box uncertainty set (Soyster, 1973). For a given value of Γ_1 and Γ_2, there are $\binom{\Gamma_1}{n} \times \binom{\Gamma_2}{n}$ possible worst-case scenarios, given the budgeted uncertainty set $\mathcal{U}_\Gamma$.

Ying (2015) affirmed that obtaining an exact solution for the two-machine Robust PFSP under budgeted uncertainty would be computationally intractable for real-sized problem instances. However, with the method introduced in this research, it turns out that it is possible to obtain exact solutions for most instances from that work in reasonable execution time (see results in Section 6).

4.3. Robust counterparts

We now present the robust counterparts for Wagner (Wagner, 1959) and Wilson (Wilson, 1989) PFSP MILP models. According to the empirical study conducted by Tseng et al. (2004), these two assignment-problem-based models are the best performing ones, based on results obtained on a standard set of 60 problem instances. In both models, the number of constraints and the model matrix size are smaller than the other two competing integer programming models from the literature (Jr & Tseng, 1990; Liao & You, 1992). Experimental data suggests that this factor greatly influences the computational time of the PFS models, apparently more than the number of binary variables.

4.3.1. Robust Counterpart for Wagner PFS Model

Wagner (1959) proposed an all-integer programming model for a three-machine deterministic flow shop, later extended to an m-machine MILP model by Stafford (1988), and commonly named in the literature as *Wagner model*. We now present its robust counterpart for two machines. In this two-stage RO formulation, y and $Z_{i,j}$ are the first-stage variables, while X_j^λ and Y_j^λ are in the second stage.

$Z_{i,j} = \begin{cases} 1, \text{ if } \sigma(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma) \\ 0, \text{ otherwise.} \end{cases}$	
X_j^λ	idle time on machine M_2 before the start of operation concerning the job in sequence position j given scenario λ.
Y_j^λ	idle time of the job in sequence position j after it finishes processing on machine M_1 given scenario λ.

$$\min\ y \tag{4}$$

$$\text{st}\ \sum_{i=1}^{n}\left(\overline{p}_{2,i}+\widehat{p}_{2,i}\delta_{2,i}^{\lambda}\right)+\sum_{j=1}^{n}X_j^{\lambda}\leq y, \qquad \lambda\in\Lambda, \tag{5}$$

$$\sum_{i=1}^{n}\left(\overline{p}_{1,i}+\widehat{p}_{1,i}\delta_{1,i}^{\lambda}\right)Z_{i,j+1}+Y_{j+1}^{\lambda}=\sum_{i=1}^{n}\left(\overline{p}_{2,i}+\widehat{p}_{2,i}\delta_{2,i}^{\lambda}\right)Z_{i,j}+X_{j+1}^{\lambda}+Y_j^{\lambda},$$

$$1\leq j\leq n-1, \lambda\in\Lambda, \tag{6}$$

$$\sum_{i=1}^{n}\left(\overline{p}_{1,i}+\widehat{p}_{1,i}\delta_{1,i}^{\lambda}\right)Z_{i,1}=X_1^{\lambda}, \qquad \lambda\in\Lambda, \tag{7}$$

$$\sum_{i=1}^{n}Z_{i,j}=1, \qquad j=1,\ldots,n, \tag{8}$$

$$\sum_{j=1}^{n}Z_{i,j}=1, \qquad i=1,\ldots,n, \tag{9}$$

$$Z_{i,j}\in\{0,1\}, \qquad i,j=1,\ldots,n, \tag{10}$$

$$X_j^{\lambda}\geq 0,\ Y_1^{\lambda}=0,\ Y_j^{\lambda}\geq 0, \qquad j=1,\ldots,n,\lambda\in\Lambda, \tag{11}$$

$$y\geq 0. \tag{12}$$

The objective function (4) and constraint (5) have the goal of finding a robust schedule that minimizes the *makespan* y of the worst-case scenario, among all possible scenarios. Constraints (6) and (7) are the Job-Adjacency and Machine-Linkage (JAML) constraints from the Wagner model, written for each scenario $\lambda\in\Lambda$. We refer the reader to Figure 1 for an illustrative JAML diagram. They ensure that, for each scenario λ: (a) the job in sequence position j cannot begin processing on machine M_2 until it has completed its processing on machine M_1, and (b) the job in sequence position $j+1$ cannot begin its processing on machine M_r until the job in sequence position j has completed its processing on that same machine. Remark that the original Wagner model does not enforce an important aspect: all jobs are processed on machine M_1 without any in-sequence machine idleness, i.e., the idle time before any job is processed on machine M_1 is always zero. As a consequence, the idle time of the first job after processing

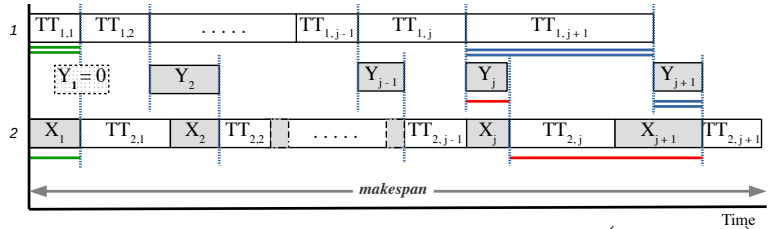

Fig. 1 JAML diagram for Wagner model, where $TT_{r,j}=\sum_{i=1}^{n}\left(\overline{p}_{r,i}+\widehat{p}_{r,i}\delta_{r,i}^{\lambda}\right)Z_{i,j}$.

on M_1 is zero ($Y_1 = 0$).Constraints (8) and (9) are the classical assignment constraints, ensuring, respectively, that each job is assigned to one and only one sequence position; and that each sequence position is filled by one and only one job. Finally, constraints (10)-(12) define the domain of the variables.

4.3.2. Robust Counterpart for Wilson PFS Model

Rather than using equality constraints and idle time variables for controlling the so-called JAML relationships, Wilson (1989) applied sets of inequality constraints based on start time variables defined for each job operation and each machine. This variant of the model uses the following decision variables:

$Z_{i,j} = \begin{cases} 1, \text{ if } \sigma(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma) \\ 0, \text{ otherwise.} \end{cases}$

$B_j^{\lambda} =$ start time of operation concerning job $\sigma(j)$ (in position j) on machine M_2 given scenario λ.

Based on the above definitions, where variables $Z_{i,j}$ and B_j^{λ} are in the first and second stage, respectively, the two-stage robust-counterpart of the Wilson model for 2RPFS can be formulated as follows:

$$\min\ y \tag{13}$$

$$\text{st}\ B_n^{\lambda} + \sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta_{2,i}^{\lambda}\right) Z_{i,n} \leq y, \qquad \lambda \in \Lambda, \tag{14}$$

$$\textstyle\sum_{i=1}^{n} \sum_{\ell=1}^{j} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^{\lambda}\right) Z_{i,\ell} \leq B_j^{\lambda}, \qquad j = 1, \ldots, n, \lambda \in \Lambda, \tag{15}$$

$$B_j^{\lambda} + \textstyle\sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta_{2,i}^{\lambda}\right) Z_{i,j} \leq B_{j+1}^{\lambda}, \qquad j = 1, \ldots, n-1, \lambda \in \Lambda, \tag{16}$$

$$\textstyle\sum_{i=1}^{n} Z_{i,j} = 1, \qquad j = 1, \ldots, n, \tag{17}$$

$$\textstyle\sum_{j=1}^{n} Z_{i,j} = 1, \qquad i = 1, \ldots, n, \tag{18}$$

$$Z_{i,j} \in \{0, 1\}, \qquad i, j = 1, \ldots, n, \tag{19}$$

$$B_j^{\lambda} \geq 0, \qquad j = 1, \ldots, n, \lambda \in \Lambda, \tag{20}$$

$$y \geq 0. \tag{21}$$

The objective function (13) and constraint (14) state that this formulation aims to find a robust schedule that minimizes the *makespan* y of the worst-case scenario, among all possible scenarios. Constraints (15) and (16) guarantee that the robust schedule is feasible and that start time variables are appropriately calculated for each scenario λ. Constraints (17) and (18) are as defined in the

previous formulation. Constraints (19)-(21) define the domain of the variables.

Solving the two models above, for all possible combinations of $\lambda \in \Lambda$, is unrealistic. Therefore, in the next section, we will introduce an algorithm capable of obtaining optimal results for 2RPFS with only a subset of these combinations.

5. Column-and-Constraint Generation applied to 2RPFS problem

This section presents an exact method for solving 2RPFS under budgeted uncertainty. Our approach is based on Column-and-Constraint Generation, a cutting plane procedure for two-stage RO problems which has been recently used to solve different robust scheduling problems (Ruiz Duarte et al., 2020; Silva et al., 2020). The method's name originated from how the decomposition operates: besides new constraints, each cutting plane is also associated with a set of new decision variables for the recourse problem (Zeng & Zhao, 2013).

Given one of the robust counterparts presented in Section 4, the main idea is to relax it into a master problem (MP) where each robust constraint is written only for a finite subset $\mathcal{U}'$ of the uncertainty set $\mathcal{U}_\Gamma$. Then, given a feasible solution to the MP, the solution is checked for feasibility by solving an adversarial separation subproblem (SP). If the SP solution indicates that one or more robust constraints become infeasible, the uncertainty set $\mathcal{U}'$ is expanded by one or more vectors, and the master problem is augmented, according to the column-and-constraint generation procedure.

For the 2RPFS problem, the adversarial separation problem is represented by the worst-case procedure, which, given the sequence σ returned by the MP solution, returns the highest possible *makespan* under the uncertainty set $\mathcal{U}_\Gamma$. Since the uncertainty set $\mathcal{U}_\Gamma$, defined in Section 4, is polyhedral, the number of possible extreme solutions that the procedure can fetch is finite, and the algorithm terminates (Zeng & Zhao, 2013).

5.1. C&CG algorithm

In order to explain the C&CG algorithm, we will consider the 2-stage RO formulations defined in Section 4.3. Given that uncertainty set $\mathcal{U}_{(\Gamma_1,\Gamma_2)}$ is discrete

Algorithm 1: Column-and-constraint generation algorithm

Set $LB = -\infty, UB = +\infty, v = 1$ and $\Theta = \{\lambda_{(0)} : \delta^{(0)}_{r,i} = 0, \forall r = 1, 2, \forall i = 1, \ldots, n\}$;
while UB $-$ LB $> \epsilon$ **do**
  if $model = Wagner$ **then** Solve the MP defined in (4)-(12) with $\Lambda := \Theta$;
  if $model = Wilson$ **then** Solve the MP defined in (13)-(21) with $\Lambda := \Theta$;
  Let $(\mathrm{Z}^*_{(v)},\ y^*,\ \mathcal{R}^*_{model})$ be the MP optimal solution ;
  Update LB $:= \max\left[\mathrm{LB},\ y^*\right]$;
  Call the oracle to solve subproblem (SP) in (22) with $Z := \mathrm{Z}^*_{(v)}$;
  Let $\mathcal{S}^*_{(v)}$ be the SP optimal solution value with associated scenario $\lambda^*_{(v)}$;
  Update UB $:= \min\left[\mathrm{UB},\ \mathcal{S}^*_{(v)}\right]$;
  if UB $-$ LB $> \epsilon$ **then**
    Add recourse decision variables $\mathcal{R}^{(v)}_{model}$ for scenario $\lambda^*_{(v)}$ on MP ;
    if $model = Wagner$ **then** Generate MP constraints (5)-(7)&(11) for $\lambda^*_{(v)}$;
    if $model = Wilson$ **then** Generate MP constraints (14)-(16)&(20) for $\lambda^*_{(v)}$;
    Update $\Theta := \Theta \cup \{\lambda^*_{(v)} : \delta^{(v)} = \delta^*\}$ and set $(v) := (v + 1)$;
Return UB, $\mathrm{Z}^*_{(v)}$

and finite, obtaining a solution for one of these formulations is equivalent to solving a probably large-scale MILP, enumerating all variables and constraints for each scenario λ in the set Λ. This possibility, as we can expect, is unrealistic. Zeng & Zhao (2013) propose an alternative solution approach, generating only a subset of scenarios $\Theta = \{\lambda_1, \ldots, \lambda_v\} \subseteq \Lambda$. With the application of the so-called C&CG procedure, if the problem is formulated in a master-subproblem framework, it can be solved iteratively, with each iteration generating one scenario $\lambda_v \in \Theta$, obtained by solving a subproblem.

With this idea in mind, we define the Master Problem (MP) by choosing an appropriate 2-stage RO formulation, in our case, either Wagner or Wilson robust counterpart models. Let $\mathcal{R}_{Wagner} = \{\mathrm{X}^{(1)}, \ldots, \mathrm{X}^{(v)}, \mathrm{Y}^{(1)}, \ldots, \mathrm{Y}^{(v)}\}$ and $\mathcal{R}_{Wilson} = \{\mathrm{B}^{(1)}, \ldots, \mathrm{B}^{(v)}\}$ be the corresponding recourse decision variables of each model, respectively. The master problem is solved iteratively, with each step generating Wagner constraints (5)-(7) or Wilson constraints (14)-(16), as well as recourse variables linked with one or more scenarios $\lambda_v \in \Lambda$, obtained by solving the associated subproblem.

In order to deal with the scenarios defined by Θ, we assume that an oracle can obtain an optimal solution to the worst-case subproblem for a given value

of the first-stage decision variables $Z_{i,j}$. The subproblem SP is defined as:

$$\text{(SP)} \quad \mathcal{S}(\sigma) = \max_{\lambda_v \in \mathcal{U}_\Gamma} \varphi(\sigma, \mathbf{P}^{\lambda_v}) \tag{22}$$

where job permutation σ is derived using (MP) optimal values of variables $Z_{i,j}$. In our case, the optimal solution for (SP) can be obtained by the worst-case procedure defined in Section 5.2.

The C&CG method is presented in Algorithm 1, where LB denotes the lower bound, UB denotes the upper bound, v is the iteration counter, Θ is the set of worst-case scenarios generated by the method, and $\epsilon \in \mathbb{R}^+$ represents the tolerance of optimality.

5.2. Worst-case evaluation

We now discuss how to determine the worst-case realization under the budgeted uncertainty set $\mathcal{U}_\Gamma$, for a specific sequence of jobs $\sigma = \{\sigma(j), j = 1, \ldots, n\}$. From equation (2), given a protection level $\Gamma = (\Gamma_1, \Gamma_2)$ and a schedule σ, we extend the definition of *worst-case makespan* or *robust cost* $\mathcal{Z}(\sigma, \Gamma)$ as follows:

$$\mathcal{Z}(\sigma, \Gamma) := \max_{\lambda \in \mathcal{U}_\Gamma} \{\varphi(\sigma, \mathbf{P}^{\lambda})\}. \tag{23}$$

We assume that parameters Γ_1 and Γ_2, from the budgeted uncertainty set, are non-negative integers. Based on this assumption, statement (23) reflects a problem with a convex function being maximized over a polytope defined by uncertainty set $\mathcal{U}_\Gamma$. Thus, in order to obtain the worst-case realization of uncertainty, only specific realizations of $\mathcal{U}_\Gamma$ are needed, namely the extreme points of the polytope. For each machine M_r and job J_i, the set of extreme scenarios are characterized either by the values $\overline{p}_{r,i}$ or $\overline{p}_{r,i} + \widehat{p}_{r,i}$ (Kouvelis et al., 2000), i.e., any worst-case realization will use as much budget of uncertainty as possible. Therefore, for the optimal solution of (23), with worst-case scenario λ^*, $\sum_{i=1}^{n} \frac{|p_{r,i}^{\lambda^*} - \overline{p}_{r,i}|}{\widehat{p}_{r,i}} = \Gamma_r, \forall r \in \{1, 2\}$.

We developed a worst-case solution method based on dynamic programming. The complexity of this algorithm is $\mathcal{O}(n^2)$. Given $1 \leq r \leq 2$, $1 \leq k \leq n$, and $0 \leq \gamma \leq n$, let us define a value function $\alpha(r, k, \gamma)$ as the optimal value of the restricted separation problem for machine M_r and job positions $\{1, \ldots, k\}$,

when at most γ jobs are using their maximum processing time on machine M_r. The optimal value of the problem is then defined by $\mathcal{Z}(\sigma, \Gamma) = \alpha(2, n, \Gamma_2)$.

The value-function is defined by the recursion:

$$\alpha(1,k,\gamma) = \max\Big[\overline{p}_{1,\sigma(k)} + \alpha(1,\ k-1,\ \gamma),\ \overline{p}_{1,\sigma(k)} + \widehat{p}_{1,\sigma(k)} + \alpha(1,\ k-1,\ \gamma-1)\Big], \quad for\ 1 \leq k \leq n,\ 0 \leq \gamma \leq \Gamma_1, \tag{24}$$

$$\alpha(2,k,\gamma) = \max\Big[\overline{p}_{2,\sigma(k)} + \max[\alpha(2,\ k-1,\ \gamma),\ \alpha(1,\ k,\ \Gamma_1)]\ , \overline{p}_{2,\sigma(k)} + \widehat{p}_{2,\sigma(k)} + \max[\alpha(2,\ k-1,\ \gamma-1),\ \alpha(1,\ k,\ \Gamma_1)]\Big], \quad for\ 1 \leq k \leq n,\ 0 \leq \gamma \leq \Gamma_2, \tag{25}$$

and the following initialization values:

$$\alpha(r,k,\gamma) = -\infty\ if\ \gamma < 0, \quad \alpha(r,k,\gamma) = 0\ if\ k = 0 \text{ and } \gamma \geq 0,$$
$$\alpha(2,k,0) = \overline{p}_{2,\sigma(k)} + \max[\alpha(2,\ k-1,\ 0),\ \alpha(1,\ k,\ \Gamma_1)],\ for\ 1 \leq k \leq n.$$

If $r = 1$, the first maximizer argument accounts for the case when there is no delay of execution regarding job $\sigma(k)$ on the first machine, while the second expression handles the case where a delay occurs. Similarly, for $r = 2$, we take the maximum of two cases: with or without delay when executing job $\sigma(k)$ on the second machine. However, the job start time has to be computed as the maximum between the previous job's $\sigma(k-1)$ worst-case completion time on the same machine M_2, and the worst-case completion time of the same job $\sigma(k)$ on the previous machine M_1, taking into account its budget of uncertainty Γ_1.

6. Experimental results

We conducted extensive experiments on randomly generated datasets to assess the performance of the proposed solution method, as well as solution robustness, the trade-off between robustness and optimality, and the impact of data uncertainty on the obtained schedules. The analyses employed in this section follow the same lines as recent works on RO under budget uncertainty (Lu et al., 2014; Feizollahi & Feyzollahi, 2015; Silva et al., 2020). Sub-section 6.1 presents the testbed and environment setup, while 6.2 examines the robust method performance regarding execution time and the number of optimal solutions. Finally, based on Monte-Carlo simulation, we close with a case study that analyses the expected behavior of robust, stochastic, and deterministic solutions, to verify a

possible increase in the expected solution cost in the long run.

6.1. Test instances and computational environment

Our experiments were based on a set of random instances generated by Ying (2015). In his work, six groups of instances were created, each one with a different number of jobs $n = \{10, 20, 50, 100, 150, 200\}$. The expected processing time $\overline{p}_{r,i}$ $(r = 1, 2;\ i = 1, \ldots, n)$ is an integer drawn from the uniform distribution [10, 50] and the largest processing time deviation was set as a ratio of the expected processing time (i.e., $\widehat{p}_{r,i} = \alpha \overline{p}_{r,i}$), where $\alpha = \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Ten instances were generated for each combination of n and α for a total of 300 test instances. All test instances are available at `https://github.com/levorato/2RPFS_Cmax_Budget`.

The C&CG algorithm was coded in Julia 1.4.0, and IBM CPLEX 12.9.0 (with default parameters) was used to solve 2RPFS MILP models. All experiments were performed on a workstation with an Intel Xeon® CPU E5640 @2.67GHz with 32 GB RAM, under Ubuntu 18.04 LTS. Time limit was set to 2 hours to solve each instance and the ϵ convergence parameter for C&CG was set to 10^{-8}.

With a particular interest in examining the impact of budget parameters on the performance of the proposed robust scheduling algorithms, when solving each instance, we tested the 2RPFS models by varying Γ_1 and Γ_2 according to five ratios (20%, 40%, 60%, 80%, and 100%) of the number of operations subject to processing time deviation on machines M_1 and M_2, respectively.

6.2. Comparative performance of the algorithms

This section examines the performance and effectiveness of the C&CG algorithm when using either Wagner or Wilson 2RPFS models. The comparison is based on the computational efficiency in terms of CPU time and the percentage of instances solved to optimality (i.e., zero solution gap).

We first present in Tab. 2 overall results, comparing the performance of the algorithms. Wagner-model C&CG is the one that solves the majority of the instances to optimality with the best execution time. The *% Best Performance*

measurement indicates that, from the total number of instances solved to optimality, the Wagner model solved 86% of these instances faster, using less CPU time, followed by Wilson, which solved 14%. Measurements *% Solved 150* and *% Solved 200* indicate that the Wilson-based algorithm could not obtain optimal solutions for most of the 150 and 200-job instances within the time limit. The other presented measurements (*%Solved, Avg % Gap*, and *Median time*) also favor the Wagner model. In this analysis, we present medians to mitigate the effect of instances not solved within the time limit.

Tab. 3 presents, for every instance size, the average performance of the C&CG algorithm with each robust-counterpart model, including average run time values. When using average, the results of all instances (even outliers) are taken into account. Standard deviation is also included as a secondary measure. Additionally, the average number of iterations and the standard deviation are listed. These results evidence that, as instance size grows, the models become harder to solve (especially the Wilson model), as seen on the smaller percentage of instances solved to optimality and increased average execution time.

6.3. Case study on two representative instances

In this subsection, we assess the quality and level of robustness of scheduling solutions for two large problem instances, the first one with small uncertainty ($\alpha = 20\%$) and the second with high uncertainty ($\alpha = 50\%$). The following solution methods were used:

- **Det(P=)**: deterministic PFSP solution with $\mathbf{P} = \{\overline{p}_{r,i}\}, \forall r \in \mathbb{M}, i \in \mathbb{J}$;
- **2RPFS(Γ_1, Γ_2)**: Wagner-based 2RPFS model, solved with the C&CG framework. The Γ parameters are used to control the level of the conservativeness of the robust model, and both vary in the set $\{20\%, 40\%, 60\%, 80\%, 100\%\}$ as a fraction of the number of jobs n. The robust model with $\Gamma_1 = \Gamma_2 = 0$ is equivalent to **Det(P=)**, while the one with $\Gamma_1 = \Gamma_2 = n$ is the deterministic model that is entirely risk-averse and overestimates all parameters. The other values of Γ_1 and Γ_2 model intermediate risk aversions;
- **SimGRASP**: stochastic PFSP simheuristic solution from Ferone et al.

(2016). SimGRASP is a modified GRASP metaheuristic that incorporates Monte Carlo Simulation to solve the PFSP with random processing times. The objective is to find a schedule that minimizes the expected makespan. Given its stochastic nature, we obtained 25 independent runs for each instance file (and respective α parameter). For result comparison, when calculating the robust cost of each (Γ_1, Γ_2) combination, we stored, for each instance, the smallest and largest robust costs found within these 25 simheuristic executions. We call them **SimGRASP-Min(25)** and **SimGRASP-Max(25)**.

We assessed the robustness of each solution method by calculating the *robust cost* at different protection levels $(\Gamma_1\%, \Gamma_2\%)$, using the dynamic programming algorithm defined in Section 5.2. Fig. 2 depicts the *robust cost* $\mathcal{Z}(\sigma)$ of each solution σ under different protection levels $(\Gamma_1\%, \Gamma_2\%)$. For clarity of the graphs, the robust costs for some protection levels were omitted.

Observe that, as the protection level $(\Gamma_1\%, \Gamma_2\%)$ increases, so does the robust cost, i.e., *makespan* of the worst-case scenario defined by the protection

Model	%Best Performance	% Solved	% Solved 10-20	% Solved 50	% Solved 100	% Solved 150	% Solved 200	Avg. % gap	Median time	Median Iterations
Wagner	86%	55%	100%	97%	90%	68%	67%	1.11%	14.05	9.0
Wilson	14%	45%	100%	86%	58%	46%	37%	1.21%	148.62	7.0

Tab. 2 Wagner *vs.* Wilson Robust PFSP C&CG performance comparison, given all instances. % Best Performance is the percentage of instances solved to optimality where the model achieved shorter execution time; % Solved contains the percentage of instances solved to optimality within the time limit; % Solved $< n >$ represents the percentage of solved instances of size n; Avg. % gap is the average percentage gap of solutions from instances not solved to optimality; Median time is the median execution time, in seconds; Median iterations is the median of the number of iterations performed.

	10		20		50		100		150		200	
	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson	Wagner	Wilson
% Best Performance	61%	39%	81%	19%	95%	5%	95%	5%	93%	7%	95%	5%
% Solved	100%	100%	100%	100%	97%	86%	90%	58%	68%	46%	67%	37%
Avg. % gap					0.40%	0.86%	1.09%	1.68%	1.21%	1.62%	1.09%	0.61%
Avg. time opt. (s)	0	0	2	7	117	310	258	526	338	878	441	2,362
Std. dev. of time opt. (s)	1	1	7	54	616	888	684	1,177	877	1,233	752	1,888
Avg. Iterations	4	4	7	7	17	20	31	21	28	14	30	11
Std. dev. of Iterations	2	2	8	8	24	23	32	15	28	8	44	4

Tab. 3 Wagner *vs.* Wilson Robust PFSP C&CG performance comparison, for each instance size n. % Best Performance is the percentage of instances solved to optimality where the model achieved shorter execution time; % Solved contains the percentage of instances solved to optimality within the time limit; Avg. % gap is the average percentage gap of solutions from instances not solved to optimality; Avg. time opt. and Std. dev. of time opt. are the mean and standard deviation in solution time, respectively, regarding instances solved to optimality; Avg. iterations and Std. dev. of iterations are the mean and standard deviation of the number of iterations performed.

level. In other words, higher values of $\Gamma_1\%$ and $\Gamma_2\%$ are equivalent to a greater quantity of operations with deviated processing times, which directly impacts the *makespan*. In the examples from Fig. 2, the extreme cases occur whenever $\Gamma_2\% = 100$, yielding the highest robust costs.

From the viewpoint of the decision-maker who needs to hedge against worst-case costs, it would be preferable to obtain a solution method that performs well under different protection levels. With this in mind, in the two graphs presented, we identify which scheduling method (and respective solution) presents the best (smallest) robust cost, considering all $(\Gamma_1\%, \Gamma_2\%)$ values. Regarding the first graph (small uncertainty instance), note that both **2RPFS(80,40)** and **2RPFS(60,40)** offer improved protection against worst-case scenarios, regardless of $(\Gamma_1\%, \Gamma_2\%)$ values used for worst-case evaluation. We also highlight the disappointing worst-case performance of both the nominal solution **Det(=)** and the stochastic method. The vast distance between the robust costs of the stochastic method, i.e., **SimGRASP-Min(25)** and **SimGRASP-Max(25)**, reveals a significant exposure to the realization of worst-case scenarios.

In its turn, the "large uncertainty range" instance ($\alpha = 50\%$) presents increased robust cost differences between distinct protection levels. For this instance, the variation of Γ_1 and Γ_2, i.e., the number of operations whose processing times can deviate on each machine, has even more impact on the worst-case makespan. In Fig. 2(b), we can observe that either **2RPFS(40,40)** or **2RPFS(60,60)** offer the best protection against worst-case scenarios, depending on the combination of $(\Gamma_1\%, \Gamma_2\%)$ values. Once again, the solutions **Det(=)** and **SimGRASP-Max(25)** present high robust costs. In particular, for $(\Gamma_1\%, \Gamma_2\%) = (60, 60)$, the solution provided by **2RPFS(60,60)** is 8% cheaper than **Det(=)** and **SimGRASP-Max(25)**.

In summary, the choice of a robust solution depends on the instance and the desired protection level. The examples above illustrate how 2RPFS can provide a pool of robust schedules, depending on the value of (Γ_1, Γ_2). With these options, the decision-makers can choose one of the schedules based on their risk preferences. Also, remark that, if the stochastic method is chosen, depending

on the solution returned by the algorithm, the worst-case performance may be weak, as can be seen on the robust costs achieved by **SimGRASP-Max(25)**. Indeed, neither **SimGRASP** nor the deterministic models have the objective of minimizing the *worst-case makespan*.

As a complementary analysis, we evaluate the expected behavior of obtained problem solutions. The *makespan* distribution of the obtained robust schedules was simulated by subjecting the processing time matrix to random perturbations. In particular, in each Monte Carlo simulation run, the (actual) processing time $\widetilde{p}_{r,i}, \forall r \in \mathbb{M}, i \in \mathbb{J}$, was independently drawn from a predefined probability distribution, yielding a random processing time matrix $\widetilde{P}$. For this purpose, we used lognormal, symmetric triangular, and uniform distributions in $[\overline{p}-\widehat{p}, \overline{p}+\widehat{p}]$ to generate random processing times. We generated $10,000$ processing time matrices $\widetilde{P}$. Then, for each **2RPFS** solution $\sigma^{(\Gamma_1,\Gamma_2)}$, obtained with a specific protection level (Γ_1, Γ_2), we processed the set of all corresponding *makespan* values $\varphi(\sigma^{(\Gamma_1,\Gamma_2)}, \widetilde{P})$ obtained through simulation on $\widetilde{P}$. The same was made

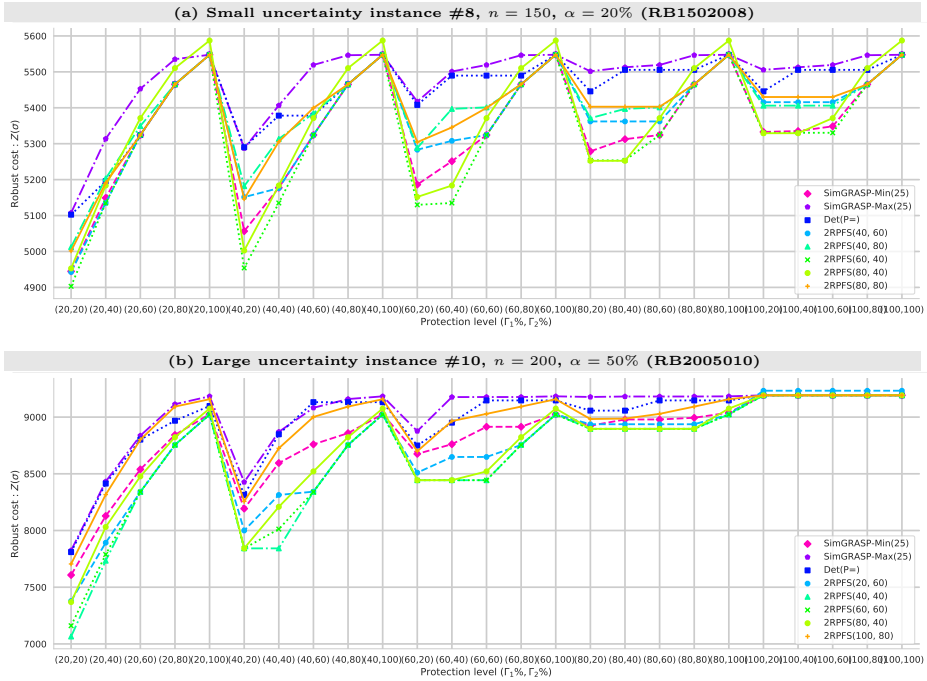

Fig. 2 Robust cost of deterministic, 2RPFS and SimGRASP solutions versus protection level (Γ_1%, Γ_2%). All presented 2RPFS solutions are optimal.

for the solutions returned by **Det(P=)** and **SimGRASP-Min(25)**.

We first focus on simulation results presented in Fig. 3(a). Regarding the small uncertainty instance, the expected makespan performance of **2RPFS(40,60)**, **2RPFS(40,80)**, **2RPFS(60,40)** and **2RPFS(80,80)**, are equivalent to **SimGRASP**. However, depending on the budget parameters, if we return to worst-case evaluation, as seen in Fig. 2(a), the protection against worst-case scenarios varies considerably. The best performing solutions, from smallest to largest robust cost, are: **2RPFS(60,40)**, **2RPFS(80,40)**, **SimGRASP-Min(25)** and **2RPFS(40,60)**. When analyzing the large uncertainty instance in Fig. 3(b), the following robust solutions present expected makespan performance quite similar to **SimGRASP**: **2RPFS(40,40)**, **2RPFS(60,60)** and **2RPFS(80,40)**. However, according to the worst-case evaluation, only the first two provide better protection against worst-case costs.

Finally, Tab. 4 presents some statistics related to the simulation of processing

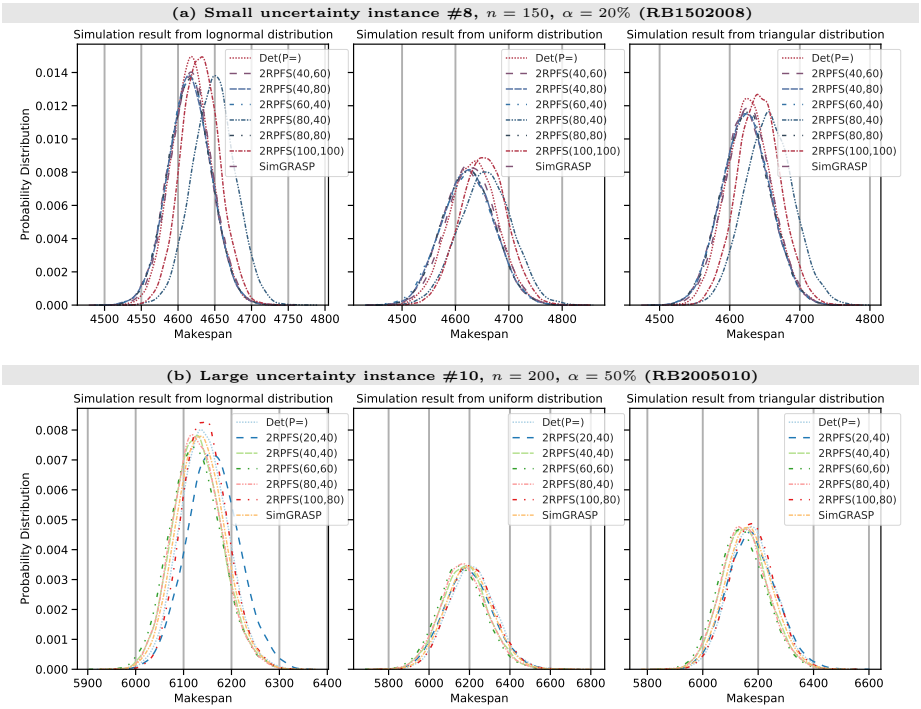

Fig. 3 Probability distributions of *makespan* value for 2RPFS and SimGRASP solutions, according to simulation results from lognormal, triangular, and uniform distributions for uncertain job processing times.

times of the large uncertainty instance. Given $10,000$ processing time matrices $\widetilde{P}$ obtained after simulation runs, let $\varphi(\sigma)$ be the random cost (*makespan*) of scheduling σ, which depends on the realization of P. $E(\varphi(\sigma))$ and $SD(\varphi(\sigma))$ are empirical estimations of expectation and standard deviation of $\varphi(\sigma)$, respectively. Also, $\varphi^{0.95}(\sigma)$ and $\varphi^{0.99}(\sigma)$ are the 0.95 and 0.99 quantiles of $\varphi(\sigma)$, respectively, and $\varphi^{max}(\sigma)$ is the maximum observed $\varphi(\sigma)$ in the simulation.

Observe that **2RPFS(60,60)** has the least $E(\varphi(\sigma))$ in lognormal distribution, while **2RPFS(40,20)** presents the smallest expected makespan in symmetric triangular and uniform distributions. When analyzing the largest observed makespan, **2RPFS(60,60)**, **2RPFS(80,80)**, and **2RPFS(80,80)** have the **lowest** $\varphi^{max}(\sigma)$ for lognormal, triangular and uniform distributions, respectively. The best solutions for **Det(P=)** and **SimGRASP** did not provide minimum values for any measure of the simulated distributions. Also, by analyzing the smallest maximum makespan obtained in uniform distribution simulations, the value $\varphi^{max}(\sigma)$ observed for scheduling **2RPFS(80,80)** is 2% cheaper than **SimGRASP**, and, at the same time, its expected makespan is 0.5% less than the stochastic schedule. Based on these observations, the hedge provided by the

Large uncertainty instance #10, $n = 200$, $\alpha = 50\%$ (RB2005010)

	Lognormal Distribution					Symmetric Triangular Distribution					Uniform Distribution				
Method	$E(\varphi(\sigma))$	$SD(\varphi(\sigma))$	$\varphi^{0.95}(\sigma)$	$\varphi^{0.99}(\sigma)$	$\varphi^{max}(\sigma)$	$E(\varphi(\sigma))$	$SD(\varphi(\sigma))$	$\varphi^{0.95}(\sigma)$	$\varphi^{0.99}(\sigma)$	$\varphi^{max}(\sigma)$	$E(\varphi(\sigma))$	$SD(\varphi(\sigma))$	$\varphi^{0.95}(\sigma)$	$\varphi^{0.99}(\sigma)$	$\varphi^{max}(\sigma)$
2RPFS(20,20)	6,125.6	52.4	6,214.5	6,254.6	6,320.1	6,143.6	84.6	6,288.4	6,352.5	6,467.6	6,162.2	116.2	6,360.4	6,446.8	6,627.1
2RPFS(20,40)	6,161.4	54.3	6,252.7	6,291.4	6,348.8	6,177.5	88.7	6,328.5	6,397.0	6,559.7	6,195.0	119.7	6,400.8	6,483.0	6,688.0
2RPFS(20,60)	6,151.5	53.8	6,241.8	6,280.4	6,342.5	6,165.5	87.0	6,312.7	6,379.8	6,499.7	6,182.0	118.7	6,383.1	6,472.4	6,677.8
2RPFS(20,80)	6,164.1	54.5	6,256.3	6,294.4	6,384.2	6,176.8	88.5	6,328.6	6,394.3	6,561.2	6,191.3	119.7	6,394.8	6,491.6	6,703.6
2RPFS(20,100)	6,211.5	54.9	6,302.3	6,340.8	6,444.0	6,228.4	93.9	6,385.6	6,450.2	6,571.4	6,247.2	126.0	6,455.8	6,544.9	6,709.3
2RPFS(40,20)	6,125.6	52.5	6,214.6	6,254.3	6,317.2	6,143.0	84.8	6,288.5	6,353.4	6,476.9	6,160.9	116.5	6,359.3	6,445.3	6,639.6
2RPFS(40,40)	6,129.9	50.7	6,215.7	6,257.6	6,321.3	6,153.9	83.2	6,294.9	6,358.3	6,466.5	6,177.3	114.8	6,371.5	6,462.6	6,667.6
2RPFS(40,60)	6,142.7	53.4	6,232.8	6,273.3	6,347.5	6,157.7	86.6	6,305.5	6,372.1	6,493.2	6,174.6	118.0	6,374.2	6,468.2	6,694.6
2RPFS(40,80)	6,140.7	53.2	6,229.8	6,268.7	6,326.6	6,159.3	86.3	6,307.6	6,372.7	6,508.4	6,178.8	117.1	6,376.1	6,460.3	6,636.4
2RPFS(40,100)	6,234.3	54.4	6,324.3	6,358.5	6,431.3	6,253.3	92.9	6,409.4	6,472.9	6,631.6	6,273.7	128.1	6,485.6	6,573.4	6,740.0
2RPFS(60,20)	6,127.5	51.3	6,214.7	6,254.9	6,336.5	6,147.4	83.8	6,291.9	6,354.5	6,529.1	6,167.7	114.9	6,360.5	6,447.1	6,681.4
2RPFS(60,40)	6,129.1	50.1	6,215.2	6,252.7	6,315.6	6,151.4	82.8	6,291.6	6,357.5	6,491.5	6,172.1	114.0	6,364.7	6,451.6	6,627.3
2RPFS(60,60)	6,125.4	52.5	6,214.4	6,254.2	6,313.5	6,143.5	84.5	6,287.4	6,350.5	6,479.8	6,161.3	115.2	6,353.3	6,446.8	6,643.0
2RPFS(60,80)	6,164.2	54.3	6,255.4	6,295.1	6,362.1	6,176.7	88.8	6,330.5	6,394.2	6,518.9	6,191.9	119.9	6,396.2	6,486.2	6,711.7
2RPFS(60,100)	6,138.8	52.8	6,228.6	6,266.2	6,335.9	6,163.4	85.1	6,307.0	6,372.0	6,506.3	6,189.2	117.6	6,386.3	6,473.2	6,628.6
2RPFS(80,20)	6,269.6	54.6	6,360.7	6,400.4	6,483.5	6,289.8	95.0	6,446.2	6,512.8	6,711.9	6,309.8	129.0	6,524.2	6,613.2	6,884.9
2RPFS(80,40)	6,129.4	50.0	6,214.0	6,253.1	6,318.0	6,151.9	82.9	6,294.5	6,354.0	6,484.0	6,173.1	114.3	6,365.7	6,456.4	6,687.7
2RPFS(80,60)	6,128.7	50.3	6,215.1	6,252.1	6,334.1	6,152.2	82.8	6,291.2	6,355.0	6,478.6	6,175.0	114.3	6,367.7	6,454.8	6,660.6
2RPFS(80,80)	6,126.8	51.7	6,214.3	6,254.8	6,320.3	6,147.4	84.2	6,291.7	6,353.3	6,464.4	6,168.1	114.9	6,361.4	6,451.5	6,623.2
2RPFS(80,100)	6,148.4	53.9	6,238.9	6,276.7	6,345.9	6,167.0	86.7	6,315.8	6,379.3	6,497.7	6,187.8	118.0	6,387.2	6,483.1	6,695.6
2RPFS(100,20)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
2RPFS(100,40)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
2RPFS(100,60)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
2RPFS(100,80)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
2RPFS(100,100)	6,144.5	46.8	6,224.0	6,260.9	6,342.8	6,178.2	81.1	6,312.8	6,372.5	6,531.9	6,208.8	113.6	6,402.6	6,487.4	6,655.5
Det(P =)	6,142.8	49.0	6,223.8	6,264.4	6,325.0	6,178.0	82.4	6,316.9	6,375.9	6,523.4	6,211.0	114.4	6,399.8	6,485.5	6,626.0
SimGRASP	6,136.2	50.6	6,221.8	6,259.3	6,366.0	6,167.4	83.8	6,309.0	6,370.4	6,544.3	6,198.1	115.7	6,391.8	6,477.7	6,783.3

Tab. 4 Simulation summary for 2RPFS, Det(P=), and SimGRASP solution methods from lognormal, triangular, and uniform distributions of processing times. Minimum values for each column are highlighted.

obtained robust solutions does not cause a significant increase in the expected solution cost when compared to stochastic and deterministic solutions.

6.4. *Evaluating price of robustness and hedge value*

Given a protection level Γ, besides robust cost $\mathcal{Z}$, two other measures can be used to evaluate performance: price of robustness η and hedge value H.

$$\eta(\Gamma) = \varphi(\sigma^*_\Gamma, \overline{P}) - \varphi(\bar{\sigma}^*, \overline{P}), \tag{26}$$

$$H(\Gamma) = \mathcal{Z}(\bar{\sigma}^*, \Gamma) - \mathcal{Z}(\sigma^*_\Gamma, \Gamma), \tag{27}$$

where $\varphi(.)$ is the *makespan* function, σ^*_Γ is the optimal solution of **2RPFS**(Γ_1, Γ_2) and $\bar{\sigma}^*$ is the optimal solution of **Det(P=)**.

The first measure, $\eta(\Gamma)$, is defined as the price paid by the decision-maker for employing the robust sequence σ^*_Γ in place of the optimal nominal sequence $\bar{\sigma}^*$ in the scenario of nominal processing times (when $P = \overline{P}$, i.e., no processing time deviations). $H(\Gamma)$ represents the value gained from adopting the robust sequence σ^*_Γ, instead of the optimal nominal sequence $\bar{\sigma}^*$ in the occurrence of the worst-case scenario associated with protection level $\Gamma = (\Gamma_1, \Gamma_2)$. In other words, $\eta(\Gamma)$ can be seen as the *trade-off between robustness and optimality*, and $H(\Gamma)$ represents the *regret of employing sequence* $\bar{\sigma}^*$ *in the worst-case scenario.*

Tab. 5 displays the relative price of robustness $\eta(\Gamma)\% = \frac{\varphi(\sigma^*_\Gamma, \overline{P}) - \varphi(\bar{\sigma}^*, \overline{P})}{\varphi(\bar{\sigma}^*, \overline{P})}$ and hedge value $H(\Gamma)\% = \frac{\mathcal{Z}(\bar{\sigma}^*, \Gamma) - \mathcal{Z}(\sigma^*_\Gamma, \Gamma)}{\mathcal{Z}(\sigma^*_\Gamma, \Gamma)}$ for various protection levels, based on instance #8 with $n = 150$, with different degrees of processing time uncertainty α. Observe that, given a protection level Γ, as α grows, so does the regret $H(\Gamma)\%$ of employing the optimal nominal sequence in the occurrence of the worst-case scenario defined by Γ. The only exception is for extreme values of Γ_1 and Γ_2, where $H(\Gamma)\% = 0$. Regarding the relative price of robustness, for

		$\Gamma_1 = 20$					$\Gamma_1 = 40$					$\Gamma_1 = 60$					$\Gamma_1 = 80$					$\Gamma_1 = 100$				
		Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100	Γ_2=20	Γ_2=40	Γ_2=60	Γ_2=80	Γ_2=100
α = 10%	η %	0.0%	0.0%	0.0%	2.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.7%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
	H %	1.7%	0.2%	0.0%	0.0%	0.0%	3.7%	2.1%	0.2%	0.0%	0.0%	4.4%	3.3%	1.3%	0.0%	0.0%	3.4%	3.4%	1.5%	0.1%	0.0%	2.6%	3.3%	1.5%	0.1%	0.0%
α = 20%	η %	0.0%	0.0%	0.0%	0.3%	5.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.9%	0.0%	0.0%	0.0%	0.4%	0.7%	0.7%	0.0%	0.0%	0.6%	1.5%	1.0%	0.0%	0.0%	0.0%
	H %	4.1%	1.2%	0.0%	0.0%	0.0%	6.8%	4.7%	1.0%	0.0%	0.0%	5.4%	6.9%	3.1%	0.4%	0.0%	3.7%	4.8%	3.4%	0.7%	0.0%	2.2%	3.3%	3.3%	0.7%	0.0%
α = 30%	η %	0.0%	0.0%	0.0%	0.0%	3.7%	0.0%	0.0%	0.0%	0.0%	1.3%	0.0%	0.0%	0.0%	0.0%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%
	H %	6.3%	2.1%	0.0%	0.0%	0.0%	8.3%	7.1%	2.1%	0.0%	0.0%	6.4%	8.6%	4.7%	0.9%	0.0%	4.1%	5.4%	5.1%	1.3%	0.0%	2.2%	3.3%	3.3%	1.3%	0.0%
α = 40%	η %	0.0%	0.0%	0.0%	0.2%	0.0%	0.0%	0.0%	0.0%	0.0%	0.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.1%	0.0%	0.0%	0.0%
	H %	8.4%	3.3%	0.6%	0.0%	0.0%	9.7%	9.3%	3.5%	0.0%	0.0%	7.2%	9.9%	6.1%	1.4%	0.0%	4.4%	5.9%	5.9%	1.8%	0.0%	2.2%	3.3%	3.3%	1.8%	0.0%
α = 50%	η %	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	1.2%	0.1%	0.3%	0.0%	0.0%
	H %	10.4%	4.6%	1.4%	0.0%	0.0%	10.9%	11.3%	4.7%	0.0%	0.0%	7.9%	11.1%	7.4%	1.7%	0.0%	4.7%	6.4%	6.4%	2.3%	0.0%	2.2%	3.3%	3.3%	2.3%	0.0%

Tab. 5 Relative robustness price $\eta(\Gamma)\%$ and hedge value $H(\Gamma)\%$ for instance #8, $n = 150$, for different degrees of uncertainty $\alpha \in \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Best values are highlighted.

α	Dist.		Γ1 = 20					Γ1 = 40					Γ1 = 60					Γ1 = 80					Γ1 = 100				
			Γ2=20	Γ2=40	Γ2=60	Γ2=80	Γ2=100	Γ2=20	Γ2=40	Γ2=60	Γ2=80	Γ2=100	Γ2=20	Γ2=40	Γ2=60	Γ2=80	Γ2=100	Γ2=20	Γ2=40	Γ2=60	Γ2=80	Γ2=100	Γ2=20	Γ2=40	Γ2=60	Γ2=80	Γ2=100
α = 10%	logno..	ω	51%	52%	52%	0%	36%	52%	52%	37%	51%	36%	52%	52%	52%	51%	36%	0%	51%	51%	52%	36%	38%	52%	52%	52%	36%
		Δφ	0.0%	0.0%	0.0%	2.2%	0.1%	0.0%	0.0%	0.1%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%	0.7%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%
	trian..	ω	52%	52%	52%	0%	38%	52%	52%	38%	52%	38%	51%	51%	51%	51%	38%	0%	51%	52%	51%	38%	39%	52%	52%	51%	38%
		Δφ	0.0%	0.0%	0.0%	2.2%	0.1%	0.0%	0.0%	0.1%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%	0.7%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%
	unifor..	ω	55%	54%	54%	0%	34%	55%	54%	37%	53%	34%	54%	54%	54%	51%	34%	2%	54%	54%	53%	34%	45%	55%	54%	54%	34%
		Δφ	0.0%	0.0%	0.0%	2.2%	0.1%	0.0%	0.0%	0.1%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%	0.7%	0.0%	0.0%	0.0%	0.1%	0.0%	0.0%	0.0%	0.0%	0.1%
α = 20%	logno..	ω	56%	56%	51%	1%	29%	7%	55%	56%	55%	29%	55%	56%	56%	55%	29%	48%	56%	56%	56%	29%	48%	42%	56%	56%	29%
		Δφ	-0.1%	-0.1%	0.0%	1.7%	0.2%	0.4%	-0.1%	-0.1%	-0.1%	0.2%	-0.1%	-0.1%	-0.1%	-0.1%	0.2%	0.0%	-0.1%	-0.1%	-0.1%	0.2%	0.0%	0.0%	-0.1%	-0.1%	0.2%
	trian..	ω	57%	57%	50%	2%	30%	16%	58%	58%	58%	30%	57%	57%	58%	57%	30%	52%	57%	57%	57%	30%	52%	49%	57%	58%	30%
		Δφ	-0.1%	-0.1%	0.0%	1.6%	0.2%	0.4%	-0.1%	-0.1%	-0.1%	0.2%	-0.1%	-0.1%	-0.1%	-0.1%	0.2%	-0.1%	-0.1%	-0.1%	-0.1%	0.2%	-0.1%	0.0%	-0.1%	-0.1%	0.2%
	unifor..	ω	61%	61%	51%	6%	29%	31%	61%	61%	61%	29%	61%	60%	61%	61%	29%	57%	61%	61%	58%	29%	57%	56%	60%	62%	29%
		Δφ	-0.2%	-0.2%	0.0%	1.5%	0.3%	0.3%	-0.2%	-0.2%	-0.2%	0.3%	-0.2%	-0.2%	-0.2%	-0.2%	0.3%	-0.2%	-0.2%	-0.2%	-0.1%	0.3%	-0.2%	-0.1%	-0.2%	-0.2%	0.3%
α = 30%	logno..	ω	58%	58%	58%	59%	0%	15%	58%	59%	58%	5%	51%	52%	59%	58%	24%	52%	52%	59%	57%	58%	56%	52%	43%	58%	58%
		Δφ	-0.1%	-0.1%	-0.1%	-0.1%	3.6%	0.3%	-0.1%	-0.1%	-0.1%	1.2%	-0.1%	-0.1%	-0.1%	-0.1%	0.4%	-0.1%	-0.1%	-0.1%	-0.1%	-0.1%	-0.1%	-0.1%	0.0%	-0.1%	-0.1%
	trian..	ω	61%	61%	62%	61%	0%	36%	62%	62%	61%	11%	58%	58%	62%	61%	29%	58%	58%	62%	59%	57%	58%	58%	55%	62%	57%
		Δφ	-0.2%	-0.2%	-0.2%	-0.2%	3.5%	0.2%	-0.2%	-0.2%	-0.2%	1.1%	-0.2%	-0.2%	-0.3%	-0.2%	0.4%	-0.2%	-0.2%	-0.2%	-0.2%	-0.1%	-0.2%	-0.2%	-0.1%	-0.2%	-0.1%
	unifor..	ω	65%	63%	62%	62%	1%	47%	64%	64%	63%	17%	62%	63%	65%	63%	31%	62%	63%	65%	59%	56%	61%	62%	61%	63%	56%
		Δφ	-0.5%	-0.3%	-0.3%	-0.3%	3.3%	-0.1%	-0.4%	-0.4%	-0.3%	1.1%	-0.4%	-0.4%	-0.5%	-0.3%	0.4%	-0.4%	-0.4%	-0.5%	-0.2%	-0.1%	-0.4%	-0.4%	-0.4%	-0.4%	-0.1%
α = 40%	logno..	ω	61%	59%	61%	24%	32%	57%	59%	60%	60%	20%	55%	58%	61%	61%	60%	53%	48%	59%	60%	52%	43%	44%	57%	60%	45%
		Δφ	-0.2%	-0.2%	-0.2%	0.4%	0.2%	-0.1%	-0.2%	-0.2%	-0.1%	0.6%	-0.1%	-0.1%	-0.2%	-0.2%	-0.1%	-0.1%	-0.1%	-0.2%	-0.2%	-0.1%	0.0%	0.0%	-0.1%	-0.2%	0.1%
	trian..	ω	64%	62%	64%	32%	35%	63%	63%	62%	62%	27%	62%	63%	64%	64%	62%	61%	60%	64%	63%	55%	50%	56%	63%	65%	47%
		Δφ	-0.4%	-0.3%	-0.4%	0.4%	0.3%	-0.4%	-0.4%	-0.3%	-0.3%	0.6%	-0.4%	-0.4%	-0.4%	-0.3%	-0.3%	-0.3%	-0.3%	-0.4%	-0.4%	-0.1%	0.0%	-0.2%	-0.4%	-0.4%	0.1%
	unifor..	ω	67%	63%	66%	37%	37%	66%	66%	63%	64%	31%	65%	67%	67%	64%	62%	66%	63%	67%	67%	55%	53%	60%	66%	67%	47%
		Δφ	-0.6%	-0.5%	-0.6%	0.4%	0.4%	-0.7%	-0.6%	-0.5%	-0.4%	0.6%	-0.6%	-0.7%	-0.6%	-0.4%	-0.4%	-0.6%	-0.6%	-0.7%	-0.6%	-0.2%	-0.1%	-0.4%	-0.6%	-0.6%	0.1%
α = 50%	logno..	ω	61%	61%	62%	61%	28%	37%	59%	61%	28%	28%	55%	36%	62%	61%	28%	4%	20%	57%	62%	28%	3%	48%	32%	62%	28%
		Δφ	-0.2%	-0.2%	-0.2%	-0.2%	0.2%	0.1%	-0.2%	-0.2%	0.4%	0.2%	-0.1%	0.1%	-0.2%	-0.1%	0.2%	1.2%	0.3%	-0.2%	-0.2%	0.2%	1.1%	-0.1%	0.1%	-0.2%	0.2%
	trian..	ω	66%	66%	66%	60%	35%	55%	65%	63%	36%	35%	62%	56%	65%	59%	35%	23%	47%	64%	65%	35%	20%	60%	54%	65%	35%
		Δφ	-0.5%	-0.5%	-0.5%	-0.2%	0.3%	-0.3%	-0.5%	-0.4%	0.4%	0.3%	-0.4%	-0.3%	-0.5%	-0.2%	0.3%	0.9%	0.0%	-0.5%	-0.5%	0.3%	1.0%	-0.4%	-0.3%	-0.5%	0.3%
	unifor..	ω	67%	68%	67%	57%	36%	60%	66%	63%	40%	36%	62%	62%	67%	58%	36%	34%	55%	67%	67%	36%	30%	64%	60%	67%	36%
		Δφ	-0.8%	-0.8%	-0.7%	-0.3%	0.4%	-0.5%	-0.8%	-0.5%	0.4%	0.4%	-0.5%	-0.6%	-0.7%	-0.3%	0.4%	0.6%	-0.4%	-0.9%	-0.7%	0.4%	0.9%	-0.6%	-0.6%	-0.7%	0.4%

Tab. 6 Simulation results for instance #8, $n = 150$, for different degrees of uncertainty $\alpha \in \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Comparison is based on two measures: (i) $\omega(\Gamma)$ is the % of simulated scenarios (over a total of 10,000) where 2RPFS(Γ) obtained smaller *makespan* cost when compared to Det(P=); (ii) $\Delta\Phi(\Gamma) = Avg_{\lambda \in \mathbb{S}}\left[\frac{\varphi(\sigma^*_\Gamma, P^\lambda) - \varphi(\bar{\sigma}^*, P^\lambda)}{\varphi(\bar{\sigma}^*, P^\lambda)}\right]$ is the average relative cost difference between 2RPFS(Γ) and Det(P=), given all simulated scenarios λ.

several protection levels Γ, the relative robustness price $\eta(\Gamma)\%$ is zero, i.e., in the absence of processing time deviations, most robust schedules present the same makespan as the optimal nominal solution. Among these schedules, the best ones, which maximize hedge value $H(\Gamma)\%$, are **2RPFS(60,20)** for $\alpha = 10\%$, **2RPFS(60,40)** for $\alpha \in \{20\%, 30\%, 40\%\}$, and **2RPFS(40,40)** for $\alpha = 50\%$.

Based on the simulation framework presented in Section 6.3, we close this section with a further analysis of the actual cost overhead of robust solutions in the long run. Two performance measures are calculated for each variability level α, as shown in Tab. 6. The obtained results show that, for different protection levels Γ, several solutions present two important characteristics: (i) high proportion of cheapest solutions ($\omega(\Gamma) > 50\%$), and (ii) smaller expected costs, i.e., negative relative cost difference $\Delta\Phi(\Gamma)$. All in all, 2RPFS provides a pool of robust schedules decision-makers can choose based on their risk preferences.

7. Concluding remarks

This work proposed the first exact solution method for the two-machine robust flow shop problem, based on budgeted uncertainty (Bertsimas & Sim,

2004). As main contributions, we developed a worst-case determination procedure for the problem, using polynomial-time dynamic programming. Together with new robust-counterpart formulations, we employed Column-and-Constraint Generation techniques. Extensive experimental results demonstrated that the proposed algorithm was effective in obtaining optimal robust schedules for small and medium-sized problems (e.g., instances with $n \leq 100$). However, it requires more computational power and thus CPU time for larger instances ($n \geq 150$) and as processing time variability level α increases.

Based on a case study with two representative instances, we have also assessed the trade-off between solution quality and cost, comparing robust solutions to deterministic and stochastic ones. The adoption of the budget of uncertainty avoids the over-conservativeness of conventional robust scheduling approaches and, at the same time, provides a pool of robust schedules, many of which perform well under different levels of realization of uncertainty. Also, according to simulations based on three probability distributions, such robust schedules presented only a small overhead in the expected solution cost.

Subsequent research will be devoted to solving the m-machine version of the same problem using metaheuristics. In addition, the computational complexity of the two-machine problem under budgeted uncertainty remains an important topic to be studied. Another direction for future research involves extending the solution method to similar scheduling problems with different objective functions, such as total tardiness, total flow time, or their weighted combinations.

Acknowledgments

We thank the anonymous reviewers, as well as David Sotelo (from PETROBRAS) and Michael Poss (from LIRMM), for their feedback and discussions on the topics discussed in this manuscript. This research was conducted during a visiting period at Avignon Université through a Doctoral Exchange Program sponsored by CAPES Foundation, Ministry of Education, Brazil (Process No.: 88881.187708/2018-01).

References

Aissi, H. et al. (2009). Min-max and min-max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*,

197, 427–438. doi:10.1016/j.ejor.2008.09.012.

Averbakh, I. (2006). The minmax regret permutation flow-shop problem with two jobs. *European Journal of Operational Research*, *169*, 761–766. doi:10.1016/j.ejor.2004.07.073.

Baker, K. R., & Trietsch, D. (2011). Three heuristic procedures for the stochastic, two-machine flow shop problem. *Journal of Scheduling*, *14*, 445–454. doi:10.1007/s10951-010-0219-4.

Balasubramanian, J., & Grossmann, I. E. (2002). A novel branch and bound algorithm for scheduling flowshop plants with uncertain processing times. *Computers and Chemical Engineering*, *26*, 41–57. doi:10.1016/S0098-1354(01)00735-9.

Ben-Tal, A., & Nemirovski, A. (2000). Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical programming*, *88*, 411–424.

Ben-Tal, A., & Nemirovski, A. (2002). Robust optimization–methodology and applications. *Mathematical Programming*, *92*, 453–480.

Bertsimas, D., & Sim, M. (2004). The Price of Robustness. *Operations Research*, *52*, 35–53. doi:10.1287/opre.1030.0065.

Braun, O., Lai, T. C., Schmidt, G., & Sotskov, Y. N. (2002). Stability of Johnson's schedule with respect to limited machine availability. *International Journal of Production Research*, *40*, 4381–4400. doi:10.1080/00207540210159527.

Cartwright, H. M., & Long, R. A. (1993). Simultaneous optimization of chemical flowshop sequencing and topology using genetic algorithms. *Industrial & engineering chemistry research*, *32*, 2706–2713.

Ćwik, M., & Józefczyk, J. (2015). Evolutionary Algorithm for Minmax Regret Flow-Shop Problem. *Management and Production Engineering Review*, *6*, 3–9. doi:10.1515/mper-2015-0021.

Ćwik, M., & Józefczyk, J. (2018). Heuristic algorithms for the minmax regret flow-shop problem with interval processing times. *Central European Journal of Operations Research*, *26*, 215–238. doi:10.1007/s10100-017-0485-8.

Deal, D., Yang, T., & Hallquist, S. (1994). Job scheduling in petrochemical production: two-stage processing with finite intermediate storage. *Computers & chemical engineering*, *18*, 333–344.

Dodin, B. (1996). Determining the optimal sequences and the distributional properties of their completion times in stochastic flow shops. *Computers and Operations Research*, *23*, 829–843. doi:10.1016/0305-0548(95)00083-6.

Elmaghraby, S. E., & Thoney, K. A. (1999). The two-machine stochastic flowshop problem with arbitrary processing time distributions. *IIE Transactions*, *31*, 467–477. doi:10.1080/07408179908969849.

Feizollahi, M. J., & Feyzollahi, H. (2015). Robust quadratic assignment problem with budgeted uncertain flows. *Operations Research Perspectives*, *2*, 114–123. doi:10.1016/j.orp.2015.06.001.

Ferone, D., Festa, P., Gruler, A., & Juan, A. A. (2016). Combining simulation with a GRASP metaheuristic for solving the permutation flow-shop problem with stochastic processing times. In *2016 Winter Simulation Conference (WSC)* (pp. 2205–2215). IEEE. doi:10.1109/WSC.2016.7822262.

Framinan, J. M., & Perez-Gonzalez, P. (2015). On heuristic solutions for the

stochastic flowshop scheduling problem. *European Journal of Operational Research*, *246*, 413–420. doi:10.1016/j.ejor.2015.05.006.

Framinan, J. M. et al. (2018). The value of real-time data in stochastic flowshop scheduling: A simulation study for makespan. In *Proceedings - Winter Simulation Conference* (pp. 3299–3310). doi:10.1109/WSC.2017.8248047.

Garey, M. R. et al. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, *1*, 117–129.

Gerodimos, A., Kouvelis, P., & Yu, G. (1998). Robust Discrete Optimization and Its Applications. *The Journal of the Operational Research Society*, *49*, 1303. doi:10.2307/3010157.

González-Neira et al. (2017). Flow-shop scheduling problem under uncertainties: Review and trends. *International Journal of Industrial Engineering Computations*, *8*, 399–426. doi:10.5267/j.ijiec.2017.2.001.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Rinnooy Kan, A. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. In *Annals of discrete mathematics* (pp. 287–326). Elsevier volume 5.

Hall, N. G., & Sriskandarajah, C. (1996). A survey of machine scheduling problems with blocking and no-wait in process. *Operations research*, *44*, 510–525.

Heyman, D. P., & Sobel, M. J. (1982). *Stochastic models in operations research: stochastic optimization* volume 2. Courier Corporation.

Hong, T.-P., & Chuang, T.-N. (2005). Fuzzy Gupta Scheduling for Flow Shops with More than two Machines. *International Journal of Computers and Applications*, *27*, 169–177. doi:10.1080/1206212X.2005.11441765.

Jin, Z., Ohno, K., Ito, T., & Elmaghraby, S. (2002). Scheduling hybrid flowshops in printed circuit board assembly lines. *Production and Operations Management*, *11*, 216–230.

Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval Research Logistics (NRL)*, *1*, 61–68.

Józefczyk, J., & Siepak, M. (2013). Scatter search based algorithms for min-max regret task scheduling problems with interval uncertainty. *Control and Cybernetics*, *42*, 667–698.

Jr, E. F., & Tseng, F. T. (1990). On the Srikar-Ghosh MILP model for the N x M SDST flowshop problem. *International Journal of Production Research*, *28*, 1817–1830. doi:10.1080/00207549008942836.

Kasperski, A., Kurpisz, A., & Zieliński, P. (2012). Approximating a two-machine flow shop scheduling under discrete scenario uncertainty. *European Journal of Operational Research*, *217*, 36–43.

Kouvelis, P. et al. (2000). Robust scheduling of a two-machine flow shop with uncertain processing times. *Iie Transactions*, *32*, 421–432.

Laha, D., & Chakraborty, U. K. (2007). An efficient stochastic hybrid heuristic for flowshop scheduling. *Engineering Applications of Artificial Intelligence*, *20*, 851–856. doi:10.1016/j.engappai.2006.10.003.

Lai, T. C., & Sotskov, Y. N. (1999). Sequencing with uncertain numerical data for makespan minimisation. *Journal of the Operational Research Society*, *50*, 230–243. doi:10.1057/palgrave.jors.2600690.

Liao, C.-J., & You, C.-T. (1992). An improved formulation for the job-shop scheduling problem. *Journal of the Operational Research Society*, *43*, 1047–1054.

Lu, C.-C., Ying, K.-C., & Lin, S.-W. (2014). Robust single machine scheduling for minimizing total flow time in the presence of uncertain processing times. *Computers Industrial Engineering*, *74*, 102 – 110. doi:https://doi.org/10.1016/j.cie.2014.04.013.

Mitkowski, W., Kacprzyk, J., Oprzędkiewicz, K., & Skruch, P. (2017). Blocks for the flow shop scheduling problem with uncertain parameters. *Advances in Intelligent Systems and Computing*, *577*, 703–711. doi:10.1007/978-3-319-60699-6.

Nawaz, M., Enscore, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, *11*, 91–95. doi:10.1016/0305-0483(83)90088-9.

Pinedo, M. L. (2016). *Scheduling: theory, algorithms, and systems*. Springer.

Ruiz Duarte, J. L., Fan, N., & Jin, T. (2020). Multi-process production scheduling with variable renewable integration and demand response. *European J. of Operational Research*, *281*, 186–200. doi:10.1016/j.ejor.2019.08.017.

Sabuncuoglu, I., & Goren, S. (2009). Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research. *International Journal of Computer Integrated Manufacturing*, *22*, 138–157. doi:10.1080/09511920802209033.

Shafransky, Y., & Shinkarevich, V. (2020). On the complexity of constructing a minmax regret solution for the two-machine flow shop problem under the interval uncertainty. *Journal of Scheduling*, *23*, 745–749.

Silva, M., Poss, M., & Maculan, N. (2020). Solution algorithms for minimizing the total tardiness with budgeted processing time uncertainty. *European Journal of Operational Research*, *283*, 70–82. doi:10.1016/j.ejor.2019.10.037.

Soyster, A. L. (1973). Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations research*, *21*, 1154–1157.

Stafford, E. F. (1988). On the development of a mixed-integer linear programming model for the flowshop sequencing problem. *Journal of the Operational Research Society*, *39*, 1163–1174.

Tseng, F. T., Stafford Jr, E. F., & Gupta, J. N. (2004). An empirical analysis of integer programming formulations for the permutation flowshop. *Omega*, *32*, 285–293.

Wagner, H. M. (1959). An integer linear-programming model for machine scheduling. *Naval Research Logistics Quarterly*, *6*, 131–140.

Wilson, J. (1989). Alternative formulations of a flow-shop scheduling problem. *Journal of the Operational Research Society*, *40*, 395–399.

Yan, H.-S., Xia, Q.-F., Zhu, M.-R., Liu, X.-L., & Guo, Z.-M. (2003). Integrated production planning and scheduling on automobile assembly lines. *Iie Transactions*, *35*, 711–725.

Ying, K. C. (2015). Scheduling the two-machine flowshop to hedge against processing time uncertainty. *Journal of the Operational Research Society*, *66*, 1413–1425. doi:10.1057/jors.2014.100.

Zeng, B., & Zhao, L. (2013). Solving two-stage robust optimization problems using a column-and- constraint generation method. *Operations Research Letters*, *41*, 457–461. doi:10.1016/j.orl.2013.05.003.