Robust permutation flow shop total weighted completion time problem: Solution and application to the oil and gas industry

Mario Levorato, David Sotelo, Rosa Figueiredo, Yuri Frota

Robust permutation flow shop total weighted completion time problem: solution and application to the oil and gas industry

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

[a] *Instituto de Computação, Universidade Federal Fluminense, Niterói, Brazil*
[b] *Laboratoire Informatique d'Avignon, Avignon Université, Avignon, France*
[c] *Operations Research and Data Science Division, Petrobras, Rio de Janeiro, Brazil*

Abstract

This work proposes exact solution approaches for the m-machine robust permutation flow shop problem, where operation processing times are uncertain and vary in a given interval. Following the concept of budgeted uncertainty, the objective is to obtain a robust scheduling that minimizes the total weighted completion time of the restricted worst-case scenario, in which only a subset of operation processing times will deviate to worst-case values. We develop several robust counterpart formulations, which can be used to derive optimal solutions for medium-sized problem instances by using a Column-and-Constraint Generation algorithm. The efficacy of the solution methods is validated through experiments on three sets of randomly-generated instances. Finally, a case study for the maintenance schedule of Brazilian oil platforms is presented.

Keywords: Scheduling, Robust Optimization, Permutation Flow Shop, Total Weighted Completion Time, Offshore platform maintenance

1. Introduction

In the context of the Oil and Gas industry, maintenance scheduling plays a very important role (Ribeiro et al., 2011; Zarei et al., 2014; Fernandez Perez

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

et al., 2018). One major challenge is related to programmed shutdown and maintenance of oil platforms. Due to existing policies, in such schedules, a well-defined set of operations must have their order of execution respected. These tasks have to be performed on all equipment associated with a specific oil well and include, for example, substitution of previously-installed provisional repairs, corrosion removal, replacing damaged paint, and servicing pipes and water re-injection pumps. Moreover, since maintenance tasks have to be executed on every oil-well connected to the platform, they all have to be closed simultaneously. Oil wells will only reopen for production at the end of the process, as soon as their associated maintenance operations finish.

The aforementioned process can be characterized as a *permutation flow shop scheduling* (Pinedo, 2016) in which oil-wells are represented by *jobs*, and maintenance tasks by *machines*. Each oil-well is closed at the start of the schedule, while its associated equipment undergoes a series of maintenance tasks that always follow the same order. The aim is to find a schedule that minimizes the loss of oil production associated with each oil well's flow rate and for how long it remained closed, i.e., the total weighted completion time (TWCT) objective. Such optimization generates huge financial gains, as more oil will be produced, in the order of thousands of dollars.

The TWCT criterion is usually associated with production environments where inventory levels and manufacturing cycle times are of critical concern. With particular interest on the minimization of inventory or holding costs, some production environments aim to minimize the total completion time, assuming all jobs are equal in importance. On specific contexts, however, the importance or value of each job may not be the same. For example, jobs may have different unit costs and holding costs. As is the case for the oil industry, the cost criterion of each job will depend on its completion time and also on its weight.

Solving this scheduling problem considering real-world characteristics is already a challenging task, which demands efficient solution approaches (Ho & Chang, 1991; Rajendran & Ziegler, 1997). Furthermore, job processing times are subject to uncertainty and no probability distribution is known. A viable

alternative, adopted here, consists in applying Robust Optimization to obtain a schedule hedging against worst-case scenarios.

Assuming processing times are uncertain and vary in a given interval, the objective of the present work is to provide efficient solutions for the m-machine Robust Permutation Flow Shop with the *total weighted completion time* objective (RPFS-TWCT). The only information required is the lower and upper bounds of processing times, which can be obtained from historical data. We are interested in a job permutation that minimizes the worst-case cost, for any possible realization of job processing times under the budgeted uncertainty set (Bertsimas & Sim, 2004). Unlike other robust optimization models, which provide only one conservative solution, the budgeted approach allows the adjustment of the solution's level of conservatism according to the decision-maker's risk-aversion.

Concerning uncertain processing times, scheduling problems that minimize the total weighted completion time have been studied from various viewpoints as, for example, single-machine scheduling heuristics (Allahverdi et al., 2014), single-machine branch-and-bound (Pereira, 2016), m-machine heuristics based on probabilistic analysis (Kaminsky & Simchi-Levi, 1998a), as well as stability analysis methods (N. Sotskov. et al., 2011; Lai et al., 2018). To the best of our knowledge, this is the first work to treat the m-machine robust permutation flow shop problem under budgeted uncertainty, which minimizes the *worst-case weighted sum of job completion times*. The solution method is partly based on a previous work (Levorato et al., 2022), developed for the *two*-machine robust permutation flow shop problem, with the *makespan* objective.

This text adopts the following structure. Section 2 introduces the classical deterministic Permutation Flow Shop Problem, minimizing total weighted completion time. The m-machine Robust Permutation Flow Shop Problem is presented in Section 4, together with seven proposed Robust Counterpart (RC) formulations. Our exact solution approach based on Column-and-Constraint Generation (C&CG) is explained in Section 5. An important enhancement to the solution method, which uses a combinatorial branch-and-bound in the master phase of C&CG, is discussed in Section 6. The experimental results are

shown in Section 7, based on extensive computational experiments on three sets of randomly-generated problem instances. Section 8 brings a case study applied to the oil and gas industry, using real data from the operation history of two Brazilian oil platforms. Finally, Section 9 brings the final discussions.

2. The deterministic Permutation Flow Shop Problem

This section presents the Permutation Flow Shop Problem (PFSP) to minimize the *Total Weighted Completion Time* (TWCT), also known as *total weighted flow time* (Pinedo, 2016). For the sake of simplicity, we will refer to this problem as PFSP-TWCT. Following the well-known $\alpha|\beta|\gamma$[1] notation for scheduling problems, established by Graham et al. (1979), this problem is denoted as $F|prmu|\sum w_j C_j$. 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 of jobs $\mathbb{J} = [n]$ to be executed in a set of machines $\mathbb{M} = [m]$[2]. Each job $i \in \mathbb{J}$ has an associated weight w_i and a non-negative processing time $p_{r,i}$ on machine $r \in \mathbb{M}$, forming the matrix $\mathbf{P} \in \mathbb{R}^{+}_{\mathbb{M}\times\mathbb{J}}$. Each job must be processed without preemption on each machine in the same order. At any time, a machine cannot handle more than one job. Also, at any time, a job can only be processed on one machine. We assume intermediate storage between successive machines is unlimited. The permutation flow shop's particularity is that the sequence in which the jobs are to be processed is the same for all machines. Such sequence is defined by a permutation $\sigma : \{1, \ldots, n\} \longrightarrow \mathbb{J}$, with $\sigma(j)$ indicating the jth job to be executed. We call Σ the set of all permutations of n jobs, hence $\sigma \in \Sigma$. Consider an operation $O_{r,\sigma(j)}$, concerning the execution of the jth job on machine r. Its completion time, denoted by $C_{r,\sigma(j)}$,

[1] Where α represents the machine environment, β stands for job characteristics, and γ symbolizes the objective function. In our case: the flow shop problem, single job permutation, minimizing total weighted completion time.

[2] We use the notation $[n] = \{1, \ldots, n\}$.

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 i is defined as its completion time on the last machine $C_{m,i}$. The objective is to find a job sequence that minimizes the total weighed completion times on the final machine, i.e., a permutation σ minimizing $\varphi(\sigma) = \sum_{j \in \{1,\dots n\}} w_{\sigma(j)} C_{m,\sigma(j)}$.

The problem was proved strongly NP-hard by Garey et al. (1976) for instances with two or more machines, when all job weights are equal. It was also studied from the viewpoint of probabilistic analysis (Kaminsky & Simchi-Levi, 1998b), stability approach (Sotskov & Lai, 2012), heuristics (Gelders & Sambandam, 1978; Miyazaki & Nishiyama, 1980; Rajendran & Ziegler, 1997; Wang et al., 2013), combinatorial branch-and-bound (Chung et al., 2002), MIP-based branch-and-bound (Yang & Wang, 2011; Vo & Lenté, 2014) and approximation algorithms (Nagarajan & Sviridenko, 2009). Finally, exact solutions can be obtained with MILP techniques, by adapting the objective function of existing flowshop formulations. For an in-depth description of each MILP model, we refer to the excellent works of (Tseng et al., 2004; Tseng & Stafford, 2008).

3. Literature review of the flow shop under uncertainty

This section provides an overview of the flow shop problem with uncertain processing times, concerning Stochastic and Robust Optimization approaches.

In a survey of 100 papers that study uncertainty in different variations of flow shop scheduling problems, González-Neira et al. (2017) listed the most common uncertain parameters (i.e., processing times and machine breakdowns), along with the approach used to deal with uncertainty. Most 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 should be noted that Stochastic Optimization approaches model random parameters with probability distributions, which may be hard to infer in certain cases. Additionally, improving the expected value of a metric may not be the best choice for processes involving only a small number of trials. In other words, the benefits of optimizing the expected value shall only be visible in the long term, after a large number of observations.

Robust Optimization techniques, on the other hand, require no assumptions on the underlying probability distribution of uncertain data. They also enable the incorporation of different approaches toward risk. The remainder of this section will present a detailed review of existing works that apply RO to flow shop scheduling problems.

3.1. Robust Optimization approaches

When solving robust scheduling problems, the objective is to optimize a performance measure taking into account the worst-case scenario, under a wide range of possible realizations of processing times. Two optimization criteria can be used in robust scheduling (Aissi et al., 2009). The first and simplest one is the *minimax* or *absolute robust* criterion. In a minimization problem, the solution is found by minimizing the highest cost over all possible scenarios.

The other criterion is called minimax regret, and aims 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).

Regarding the uncertain nature of input data, scenarios represent the set of possible realizations of processing times. When applying RO, there are two usual ways of defining the scenario set. 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, a range $[p^{L}_{r,i},\, p^{U}_{r,i}]$ of lower and upper bounds of processing times

is defined for each operation $O_{r,i}$ concerning job i executing on machine r.

Tables 1 and 2 summarize existing works regarding the Robust Permutation Flow Shop Problem with the *makespan* objective and robust scheduling problems with the *TWCT* objective, respectively, in terms of optimization criterion, solution approach (heuristics or exact methods), the number of machines, and how processing time uncertainty was represented (discrete or interval).

Even though several solution methods have been developed for the Robust PFSP with the classical makespan objective (most of them for the 2-machine case, as seen on Tab. 1), the same is not true for the TWCT objective (Tab. 2). To our knowledge, considering robust scheduling with weighted and unweighted versions of the total completion time objective, works about the robust flow shop are nonexistent, and only the single-machine scheduling problem (SMSP) has been addressed. Based on the minimax regret criterion, 2-approximation (Kasperski & Zieliński, 2008), heuristics (Daniels & Kouvelis, 1995) and branch-and-bound (Daniels & Kouvelis, 1995; Pereira, 2016) algorithms were proposed for the single-machine robust scheduling problem, while heuristics (Yang & Yu, 2002; Allahverdi et al., 2014), stability analysis meth-

Criterion	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}	**2 machines**: Column-and-Constraint Generation (Levorato et al., 2022)^{I}

Tab. 1 Summary of algorithms listed in the literature review regarding the Robust PFSP *makespan* objective with processing time uncertainty (table adapted from Levorato et al. (2022)). For each work, we specify how processing time uncertainty was represented: a **D** means discrete scenario set; an **I** means interval scenario set.

Criterion	Heuristics / Approximation	Exact methods
Minimax Regret	**Single machine**: 2-approx (unweighted version) (Kasperski & Zieliński, 2008)^{I} and heuristics (unweighted version) (Daniels & Kouvelis, 1995)^{I}	**Single machine**: Branch & Bound (Pereira, 2016)^{I} and Branch & Bound (unweighted version)(Daniels & Kouvelis, 1995)^{I}
Minimax	**Single machine**: Heuristics (Allahverdi et al., 2014)^{I}, (Yang & Yu, 2002)^{D}	**Single machine**: Stability analysis (N. Sotskov. et al., 2011; Lai et al., 2018)^{I}, Dynamic Programming $O(2^n)$ (Yang & Yu, 2002)^{D}, Branch & Cut(de Farias et al., 2010)^{D}
Minimax, Budgeted uncertainty	**Single machine**: Heuristics (unweighted version) (Lu et al., 2014)^{I}	**Single machine**: MILP(Tadayon & Smith, 2015)^{I}, MILP(unweighted version)(Lu et al., 2014)^{I}

Tab. 2 Summary of algorithms listed in literature review, regarding robust scheduling problems with the *TWCT* objective and processing time uncertainty. For each work, we specify how processing time uncertainty was represented: a **D** means discrete scenario set; an **I** means interval scenario set.

ods (N. Sotskov. et al., 2011; Lai et al., 2018), dynamic programming (Yang & Yu, 2002) and branch-and-cut (de Farias et al., 2010) were developed for the SMSP / minimax optimization criterion. Finally, simple iterative improvement (SII) and simulated annealing (SA) heuristics (Lu et al., 2014), along with MILP models (Lu et al., 2014; Tadayon & Smith, 2015) were developed for the SMSP with minimax budgeted uncertainty.

As mentioned in the introduction, to the best of our knowledge, this is the first research that applies budgeted uncertainty (Bertsimas & Sim, 2004) to the m-machine robust permutation flow shop. Existing works on flow shop and budgeted uncertainty are related to either single or two-machine variants of the problem. In Bougeret et al. (2019), the complexity of the robust single-machine scheduling problem, with the TWCT objective and budgeted uncertainty, was shown to be weakly NP-hard if the budget parameter $\Gamma = 1$ and strongly NP-hard for $\Gamma > 1$. Regarding the two-machine robust flow shop problem with the *makespan* objective, budgeted uncertainty and processing time intervals, Ying (2015) developed two metaheuristic algorithms to solve the problem (SA and IG), but with the *makespan* objective. Finally, in a recent work (Levorato et al., 2022), an exact solution approach was developed for the *makespan* problem, based on Column-and-Constraint Generation.

4. The Robust PFSP to minimize the total weighted completion time

Different optimization criteria can be used to search for a robust solution. This work focuses on the *minimax* or *absolute robust* criterion: the robust decision looks for a solution that minimizes the highest objective value over all possible scenarios, following a predefined uncertainty set.

This section starts with a definition of the RPFS-TWCT problem (Section 4.1), followed by the uncertainty set adopted in this work, based on budgeted uncertainty (Section 4.2). Then, seven robust counterpart formulations are proposed (Section 4.3), 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 $\mathbf{P} = \{p_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$ contains uncertain data. A scenario λ is defined as a realization of uncertainty and, for each possible λ, there is a unique matrix of processing times denoted as $\mathbf{P}^\lambda = \{p^\lambda_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$. Let Λ be the set of all possible scenarios λ. Whenever a matrix of processing times $\mathbf{P}^\lambda$ is known, an instance of the deterministic PFS-TWCT is defined.

Let $\varphi(\sigma, \mathbf{P}^\lambda)$ be the *total weighted completion time* of a sequence $\sigma \in \Sigma$ given a scenario $\lambda \in \Lambda$. The objective of the RPFS-TWCT is to find a job permutation $\sigma \in \Sigma$ that *minimizes* the *maximum* possible **total weighted completion time** over all scenarios $\lambda \in \Lambda$:

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

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 total weighted completion time* or *robust cost* for σ. A maximizer in (2) is called a worst-case scenario for σ.

Remark that the RPFS-TWCT problem is NP-hard for $m \geq 2$, following the complexity of the deterministic problem.

4.2. Budgeted uncertainty set for the RPFS-TWCT problem

In Ying (2015), the three classical Robust Counterpart (RC) optimization models (Soyster, 1973; Ben-Tal & Nemirovski, 2000; Bertsimas & Sim, 2004) are compared in terms of the number of variables and required constraints, and if the respective formulation is linear or not. When compared to the other RC models (Soyster, 1973; Ben-Tal & Nemirovski, 2000), the so-called budgeted uncertainty model (Bertsimas & Sim, 2004) fits best for robust scheduling problems, by providing a linear formulation that allows adjusting the level of conservatism of the robust solution, without resulting in a substantial increase in problem size. The inclusion of a budget parameter provides a compromise between robustness and optimality. It is possible to adjust the number of coefficients that

simultaneously take their largest variations, based on application knowledge. For the case of oil-well maintenance, the problem which will be analyzed in the case study section, it is known that the probability of all maintenance tasks simultaneously deviating to their worst-case execution times is low.

Next, the budget uncertainty set for the RPFS-TWCT problem is defined. Consider two positive processing time matrices $\overline{\mathbf{P}}=\{\overline{p}_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$ and $\widehat{\mathbf{P}}=\{\widehat{p}_{r,i}, r \in \mathbb{M}, i \in \mathbb{J}\}$, that represent the nominal value of and the maximum allowed deviation of $\mathbf{P}$, respectively. The processing time of each operation $O_{r,j}$ lies in the interval $[\overline{p}_{r,j}, \overline{p}_{r,j} + \widehat{p}_{r,j}]$. In order to apply budgeted uncertainty, we introduce the budget parameter $\Gamma \in \mathbb{Z}_+ : 0 \leq \Gamma \leq mn$, which denotes the maximum number of operations whose uncertain processing times can reach their worst-case values. The *discrete budgeted uncertainty set* of operation processing times, denoted as $\mathcal{U}^{\Gamma}$, can be defined as follows:

$$\mathcal{U}^{\Gamma} = \left\{ \mathbf{P} = \{p_{r,i}\} \mid p_{r,i} = \overline{p}_{r,i} + \delta_{r,i}\, \widehat{p}_{r,i},\ \delta_{r,i} \in \{0,1\}, \forall r \in \mathbb{M}, \forall i \in \mathbb{J}; \sum_{r=1}^{m} \sum_{i=1}^{n} \delta_{r,i} \leq \Gamma \right\}, \quad (3)$$

Given the uncertainty set $\mathcal{U}^{\Gamma}$, each scenario λ is described by one of the infinite matrices in this set. For a given scenario λ, let $\delta^{\lambda}_{r,i}$ be the value defining the deviation of the processing time regarding the execution of job $i \in \mathbb{J}$ on machine $r \in \mathbb{M}$, i.e., $p^{\lambda}_{r,i} = \overline{p}_{r,i} + \delta^{\lambda}_{r,i}\widehat{p}^{\lambda}_{r,i}$. Therefore, considering all jobs and machines, the total number of operations whose processing time can deviate to its maximum value is limited to Γ. When $\Gamma = 0$, the problem is equivalent to the nominal problem, i.e., the deterministic PFSP-TWCT. If $\Gamma = mn$, we obtain the box uncertainty set (Soyster, 1973). For a given value of Γ, there are $\binom{mn}{\Gamma}$ possible worst-case scenarios, given the budgeted uncertainty set $\mathcal{U}_{\Gamma}$.

4.3. Robust counterparts

A number of MILP models were proposed in the literature for the PFSP. We now present the robust counterparts for the PFSP-TWCT, based on the following seven formulations: Wilson (Wilson, 1989); TBA (Turner & Booth, 1986); Wagner-WST2; TS2 and TS3 (Tseng & Stafford, 2008); Manne (Jr & Tseng, 1990) and Liao-You (Liao & You, 1992). In their original definition, the first five

models rely on assignment constraints in order to find the position occupied by each job in the schedule, while the last two apply disjunctive inequalities with Big-M reformulation to determine if a job appears either before or after another job in the sequence. For more details on the rationale behind each deterministic PFSP model, including illustrative diagrams, we refer the reader to Tseng & Stafford (2008).

It is worth noting that the first five models were further adapted for the TWCT objective. Such adaptation involved additional constraints based on the Big-M method to appropriately calculate the variables which represent the completion time of each job i on the last machine. These variables, which are not present in the original models, had to be defined for the correct calculation of total weighted completion time.

4.3.1. Robust Counterpart for Wilson PFS Model

Wilson (1989) proposed a MILP model for the *makespan*-minimizing flow shop scheduling problem, by applying sets of inequality constraints, based on start time variables, of each job on each machine. In this work, we derived a two-stage robust counterpart of his model, for the *total weighted completion time* objective, with the following decision variables:

$Z_{i,j} =$	$\begin{cases} 1, \text{ if } \sigma(j) = i \text{ (job } i \text{ occupies position } j \text{ in the sequence } \sigma), \\ 0, \text{ otherwise.} \end{cases}$
$B^{\lambda}_{r,j} =$	start time of job $\sigma(j)$ (in position j) on machine M_r given scenario λ.
$F^{\lambda}_{i} =$	completion time of job i on machine M_m in scenario λ.
$y =$	highest (worst-case) total weighted completion time, given all scenarios $\lambda \in \Lambda$.

Based on the above definitions, variables $Z_{i,j}$ are in the first stage, and variables $B^{\lambda}_{r,j}$ and F^{λ}_{i} are in the second stage of this robust counterpart. The two-stage robust counterpart of Wilson model for the RPFS-TWCT can be

formulated as follows:

$$\text{Min } y \tag{4}$$

$$\text{st } \sum_{i=1}^{n} w_i F_i^{\lambda} \leq y, \qquad \lambda \in \Lambda, \tag{5}$$

$$B_{1,1}^{\lambda} = 0, \qquad \lambda \in \Lambda, \tag{6}$$

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

$$B_{r,1}^{\lambda} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^{\lambda}\right) Z_{i,1} = B_{r+1,1}^{\lambda}, \qquad r = 1, \ldots, m-1, \lambda \in \Lambda, \tag{8}$$

$$B_{r,j}^{\lambda} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^{\lambda}\right) Z_{i,j} \leq B_{r+1,j}^{\lambda}, \; r = 1, \ldots, m-1, j = 2, \ldots, n, \lambda \in \Lambda, \tag{9}$$

$$B_{r,j}^{\lambda} + \sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^{\lambda}\right) Z_{i,j} \leq B_{r,j+1}^{\lambda}, \; r = 2, \ldots, m, j = 1, \ldots, n-1, \lambda \in \Lambda, \tag{10}$$

$$F_i^{\lambda} \geq B_{m,j}^{\lambda} + \left(\overline{p}_{m,i} + \widehat{p}_{m,i}\delta_{m,i}^{\lambda}\right) Z_{i,j} - Q(1 - Z_{i,j}), i = 1, ..., n, j = 1, ..., n, \lambda \in \Lambda, \tag{11}$$

$$\sum_{i=1}^{n} Z_{i,j} = 1, \qquad j = 1, \ldots, n, \tag{12}$$

$$\sum_{j=1}^{n} Z_{i,j} = 1, \qquad i = 1, \ldots, n, \tag{13}$$

$$Z_{i,j} \in \{0, 1\}, \qquad i, j = 1, \ldots, n, \tag{14}$$

$$B_{r,j}^{\lambda} \geq 0, \qquad r = 1, \ldots, m, j = 1, \ldots, n, \lambda \in \Lambda, \tag{15}$$

$$F_i^{\lambda} \geq 0, \qquad i = 1, \ldots, n, \lambda \in \Lambda, \tag{16}$$

$$y \geq 0. \tag{17}$$

The objective function (4) and constraint (5) state that this formulation aims to find a robust schedule for the processing of n jobs that minimizes the *weighted sum of completion times* (i.e., total weighted completion time) of the worst-case scenario, among all possible scenarios $\lambda \in \Lambda$. Constraints (6)-(10) guarantee that the robust schedule is feasible and that start time variables are appropriately calculated, for each scenario λ. Constraints (11) are used to determine the completion time of job i on the last machine m, for each scenario λ. In these constraints, assume Q is a large-enough number. The same assumption is made in all other formulations, using a big-M value Q. Constraints (12) and (13) 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 (14)-(17) define the domain of the variables.

4.3.2. Robust Counterpart for TBA PFS Model

Relying on the assignment constraints of Wilson model, Turner & Booth (1986) derived a MILP formulation for the PFSP, here called Turner–Booth alternative (TBA) model. After deriving an equivalent mathematical expression for start time variables, in terms of processing and idle times of each job, the authors applied variable substitution techniques, significantly reducing the number of model constraints. We derived a two-stage robust counterpart of this model, for the TWCT objective, with decision variables $Z_{i,j}$, F_i^{λ} and y, as well as the new ones described below:

$X_{r,j}^{\lambda}$ = idle time on machine M_r before the start of job in sequence position j given scenario λ.

Based on the above definitions, variables $Z_{i,j}$ are in the first stage, and variables $X_{r,j}^{\lambda}$ and F_i^{λ} are in the second stage of this robust counterpart. The two-stage robust counterpart of TBA model for the RPFS-TWCT can be formulated as follows:

$$\text{Min } y \tag{18}$$

$$\text{st } (5), (12), (13), (14), (16), (17),$$

$$X_{1,j}^{\lambda} = 0, \qquad j = 2, \ldots, n, \lambda \in \Lambda, \tag{19}$$

$$\begin{aligned} &\textstyle\sum_{i=1}^{n} \left(\overline{p}_{r-1,i} + \widehat{p}_{r-1,i}\delta_{r-1,i}^{\lambda}\right) Z_{i,1} + \sum_{q=1}^{j-1}\sum_{i=1}^{n} \left(\overline{p}_{r,i} - \overline{p}_{r-1,i}\right) Z_{i,q} \\ &+ \textstyle\sum_{q=1}^{j-1}\sum_{i=1}^{n} \left(\widehat{p}_{r,i}\delta_{r,i}^{\lambda} - \widehat{p}_{r-1,i}\delta_{r-1,i}^{\lambda}\right) Z_{i,q} + \sum_{s=2}^{j} \left(X_{r,s} - X_{r-1,s}\right) \\ &- \textstyle\sum_{i=1}^{n} \left(\overline{p}_{r-1,i} + \widehat{p}_{r-1,i}\delta_{r-1,i}^{\lambda}\right) Z_{i,j} \geq 0, \quad r = 2, \ldots, m, j = 2, \ldots, n, \lambda \in \Lambda, \end{aligned} \tag{20}$$

$$\begin{aligned} F_i^{\lambda} \geq &\textstyle\sum_{r=1}^{m-1}\sum_{\ell=1}^{n} \left(\overline{p}_{r,\ell} + \widehat{p}_{r,\ell}\delta_{r,\ell}^{\lambda}\right) Z_{\ell,1} + \sum_{q=1}^{j}\sum_{\ell=1}^{n} \left(\overline{p}_{m,\ell} + \widehat{p}_{m,\ell}\delta_{m,\ell}^{\lambda}\right) Z_{\ell,q} \\ &+ \textstyle\sum_{s=1}^{j} X_{m,s} - Q(1 - Z_{i,j}), \qquad i = 1, \ldots, n, j = 1, \ldots, n, \lambda \in \Lambda, \end{aligned} \tag{21}$$

$$X_{r,j}^{\lambda} \geq 0, \qquad r = 1, \ldots, m, j = 1, \ldots, n, \lambda \in \Lambda. \tag{22}$$

The objective function (18) and constraints (5), (12) and (13) are as defined in the previous formulation. Constraints (19)-(20) guarantee that the robust schedule is feasible and that idle time variables are appropriately calculated, for each scenario λ. Constraints (21) are big-M constraints used to determine the completion time of job i on the last machine m, for each scenario λ. For an illustrative diagram, we refer the reader to Fig. 2 in (Tseng & Stafford, 2008, p. 1376). Finally, constraints (14), (16), (17) and (22) define the domain of the variables.

4.3.3. Robust Counterpart for WST2 PFS Model

Wagner (1959) proposed an all-integer programming model for a three-machine deterministic flow shop, later extended to a m-machine MILP model by Stafford (1988), and commonly named in the literature as *Wagner model*. In 2002, based on this model and works from other authors, Stafford and Tseng released an improved model called WST and, later on, a second version called WST2 (Tseng & Stafford, 2008), which enforces the initial condition that all jobs are processed on the first machine without any in-sequence machine idleness. In our research, we derived a two-stage robust counterpart of the WST2 model, for the TWCT objective, with decision variables $Z_{i,j}$, $X^{\lambda}_{r,j}$, F^{λ}_{i} and y as described in the previous formulations, and variables $Y^{\lambda}_{r,j}$:

$Y^{\lambda}_{r,j} =$	idle time of job in sequence position j after it finishes processing on machine M_r given scenario λ.

Variables $Z_{i,j}$ are in the first stage, and variables $X^{\lambda}_{r,j}$, $Y^{\lambda}_{r,j}$ and F^{λ}_{i} are in the second stage of this robust counterpart. The WST2 model for the RPFS-TWCT can be formulated as follows:

$$\text{Min } y \tag{23}$$

$$\text{st } (5), (12), (13), (14), (16), (17), (22),$$

$$X^{\lambda}_{1,1} = 0, \qquad \lambda \in \Lambda, \tag{24}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,j+1} + X^{\lambda}_{r,j+1} + Y^{\lambda}_{r,j+1} = \sum_{i=1}^{n} \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta^{\lambda}_{r+1,i}\right) Z_{i,j} + X^{\lambda}_{r+1,j+1} + Y^{\lambda}_{r,j}, \qquad r = 2, \ldots, m-1, j = 2, \ldots, n-1, \lambda \in \Lambda, \tag{25}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}\right) Z_{i,j+1} + Y^{\lambda}_{1,j+1} = \sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta^{\lambda}_{2,i}\right) Z_{i,j} + X^{\lambda}_{2,j+1} + Y^{\lambda}_{1,j}, \qquad j = 2, \ldots, n-1, \lambda \in \Lambda, \tag{26}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,2} + X^{\lambda}_{r,2} + Y^{\lambda}_{r,2} = \sum_{i=1}^{n} \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta^{\lambda}_{r+1,i}\right) Z_{i,1} + X^{\lambda}_{r+1,2}, \qquad r = 2, \ldots, m-1, \lambda \in \Lambda, \tag{27}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}\right) Z_{i,2} + Y^{\lambda}_{1,2} = \sum_{i=1}^{n} \left(\overline{p}_{2,i} + \widehat{p}_{2,i}\delta^{\lambda}_{2,i}\right) Z_{i,1} + X^{\lambda}_{2,2}, \quad \lambda \in \Lambda, \tag{28}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}\right) Z_{i,1} + X^{\lambda}_{r,1} = X^{\lambda}_{r+1,1}, \qquad r = 1, \ldots, m-1, \lambda \in \Lambda, \tag{29}$$

$$F^{\lambda}_{i} \geq \sum_{p=1}^{j} \sum_{x=1}^{n} \left(\overline{p}_{m,x} + \widehat{p}_{m,x}\delta^{\lambda}_{m,x}\right) Z_{x,p} + \sum_{p=1}^{j} X^{\lambda}_{m,p} - Q(1 - Z_{i,j}), \qquad i = 1, \ldots, n, j = 1, \ldots, n, \lambda \in \Lambda, \tag{30}$$

$$Y^{\lambda}_{r,j} \geq 0, \qquad r = 1, \ldots, m, j = 1, \ldots, n, \lambda \in \Lambda. \tag{31}$$

The objective function (23) and constraints (5), (12) and (13) are as defined in the first formulation. Constraints (24)-(29) guarantee that the robust schedule

is feasible and that idle time variables are appropriately calculated, for each scenario λ. (30) are big-M constraints used to determine the completion time of job i on the last machine m, for each scenario λ. Finally, constraints (14), (16), (17), (22) and (31) define the domain of the variables.

4.3.4. Robust Counterpart for TS2 PFS Model

The TS2 MILP model for the regular flow shop is based on an earlier model with the same name that was developed by Tseng & Stafford Jr (2001) for the sequence-dependent setup times flow shop problem. This model uses the job ending or completion time variables employed in other scheduling models, which eliminates the need for the X and Y variables used in Wagner-WST2 model. Besides adapting the TS2 model to the TWCT objective, we also derived a two-stage robust counterpart with variables $Z_{i,j}$, F_i^λ and y as described in the first formulation, along with variables $E_{r,j}^\lambda$:

$E_{r,j}^\lambda =$	completion time of job in sequence position j after it finishes processing on machine r given scenario λ.

Variables $Z_{i,j}$ are again in the first stage, while variables $E_{r,j}^\lambda$ and F_i^λ are in the second stage of this robust counterpart. The two-stage robust counterpart of TS2 model for the RPFS-TWCT can be formulated as:

$$\text{Min } y \tag{32}$$

$$\text{st } (5), (12), (13), (14), (16), (17),$$

$$E_{r,j+1}^\lambda \geq E_{r,j}^\lambda + \sum_{i=1}^n \left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^\lambda\right) Z_{i,j+1},\ r = 2\ldots, m, j = 1, \ldots, n-1, \lambda \in \Lambda, \tag{33}$$

$$E_{r+1,j}^\lambda \geq E_{r,j}^\lambda + \sum_{i=1}^n \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta_{r+1,i}^\lambda\right) Z_{i,j}, \quad r = 1\ldots, m-1, j = 2, \ldots, n, \lambda \in \Lambda, \tag{34}$$

$$E_{1,j+1}^\lambda = E_{1,j}^\lambda + \sum_{i=1}^n \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^\lambda\right) Z_{i,j+1} \quad j = 1, \ldots, n-1, \lambda \in \Lambda, \tag{35}$$

$$E_{r+1,1}^\lambda = E_{r,1}^\lambda + \sum_{i=1}^n \left(\overline{p}_{r+1,i} + \widehat{p}_{r+1,i}\delta_{r+1,i}^\lambda\right) Z_{i,1}, \quad r = 1, \ldots, m-1, \lambda \in \Lambda, \tag{36}$$

$$E_{1,1}^\lambda = \sum_{i=1}^n \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^\lambda\right) Z_{i,1} \quad \lambda \in \Lambda, \tag{37}$$

$$F_i^\lambda \geq E_{m,j}^\lambda - Q(1 - Z_{i,j}) \quad i = 1, \ldots, n, j = 1, \ldots, n, \lambda \in \Lambda, \tag{38}$$

$$E_{r,j}^\lambda \geq 0, \quad r = 1, \ldots, m, j = 1, \ldots, n, \lambda \in \Lambda. \tag{39}$$

The objective function (32) and constraints (5), (12) and (13) are as defined in the first formulation. Constraints (33)-(37) guarantee that the robust schedule

is feasible and that completion time variables are appropriately calculated, for each scenario λ. Constraints (38) are big-M constraints used to determine the completion time of job i on the last machine m, for each scenario λ. Finally, constraints (14), (16), (17) and (39) define the domain of the variables.

4.3.5. Robust Counterpart for TS3 PFS Model

Using an approach similar to the one applied in the TBA model, Tseng & Stafford (2008) proposed a MILP formulation for the PFSP called TS3. By applying variable substitution on Wilson model, the start time variable, for a given r and j, is replaced by an expression that combines the sum of the processing times of jobs in sequence positions 1 through $j-1$ on machine 1, and the sum of the processing times of the job in position j on machines 1 through $r-1$, incremented of job's idle times (following each of these same machines). To apply Robust Optimization, we derived a two-stage robust counterpart of the TS3 model, adapted to the TWCT objective, with variables $Z_{i,j}$, F_i^λ, $Y_{r,j}^\lambda$ and y as previously defined.

As in previous formulations, variables $Z_{i,j}$ are in the first stage, while variables $Y_{r,j}^\lambda$ and F_i^λ are in stage two. The two-stage robust counterpart of TS3 model can be formulated as follows:

$$\text{Min } y \tag{40}$$

$$\text{st } (5), (12), (13), (14), (16), (17), (31),$$

$$Y_{r,1}^\lambda = 0, \qquad r = 1, \ldots, m-1, \lambda \in \Lambda, \tag{41}$$

$$\sum_{i=1}^{n} \left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^\lambda - \overline{p}_{r,i} - \widehat{p}_{r,i}\delta_{r,i}^\lambda\right) Z_{i,j-1} + \sum_{q=1}^{r-1}\sum_{i=1}^{n} \left(\overline{p}_{q,i} + \widehat{p}_{q,i}\delta_{q,i}^\lambda\right)\left(Z_{i,j} - Z_{i,j-1}\right) + \sum_{q=1}^{r-1}\left(Y_{q,j} - Y_{q,j-1}\right) \geq 0, \qquad r = 2, \ldots, m, j = 2, \ldots, n, \lambda \in \Lambda, \tag{42}$$

$$F_i^\lambda \geq \sum_{q=1}^{j}\sum_{i=1}^{n}\left(\overline{p}_{1,i} + \widehat{p}_{1,i}\delta_{1,i}^\lambda\right) Z_{i,q} + \sum_{r=2}^{m}\sum_{i=1}^{n}\left(\overline{p}_{r,i} + \widehat{p}_{r,i}\delta_{r,i}^\lambda\right) Z_{i,j} + \sum_{r=1}^{m-1} Y_{r,j} - Q(1 - Z_{i,j}), \qquad i = 1, \ldots, n, j = 1, \ldots, n, \lambda \in \Lambda. \tag{43}$$

The objective function (40) and constraints (5), (12) and (13) are as defined in the first formulation. Constraints (41)-(42) guarantee that the robust schedule is feasible and that idle time variables are appropriately calculated, for each scenario λ. Constraints (43) are big-M constraints used to determine the

completion time of job i on the last machine m, for each scenario λ. Finally, constraints (14), (16), (17) and (31) define the domain of the variables.

4.3.6. Robust Counterpart for Manne PFS Model

Manne (Manne, 1960) proposed a dichotomous-constraints integer programming model for the general job shop problem. The model assures that, for two jobs i and k, only one of each pair of completion-time subtraction constraints can hold, i.e., job i either precedes job k somewhere in the processing sequence, or it does not, thus implying that job k precedes job i. Later, Stafford and Tseng (Jr & Tseng, 1990) adapted this model to a permutation flow shop (*makespan* objective). Based on this last model, we developed its robust counterpart, adapted for the TWCT objective, with variables y as described in the first formulation, along with the following decision variables:

$D_{i,k} = \begin{cases} 1, \text{ if job } i \text{ is scheduled any time before job } k \\ 0, \text{ otherwise.} \end{cases}$	
$C^{\lambda}_{r,i}$	completion time of job i on machine r given scenario λ.

In this two-stage RO formulation, $D_{i,k}$ are the first-stage variables, while $C^{\lambda}_{r,i}$ are second stage ones. The robust counterpart for Manne PFS model can be formulated as follows.

$$\text{Min } y \tag{44}$$

$$\text{st } \sum_{i=1}^{n} w_i C^{\lambda}_{m,i} \leq y, \qquad \lambda \in \Lambda, \tag{45}$$

$$C^{\lambda}_{1,i} \geq \overline{p}_{1,i} + \widehat{p}_{1,i}\delta^{\lambda}_{1,i}, \qquad i = 1, \ldots, n, \lambda \in \Lambda, \tag{46}$$

$$C^{\lambda}_{r,i} - C^{\lambda}_{r-1,i} \geq \overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}, \qquad r = 2, \ldots, m, i = 1, \ldots, n, \lambda \in \Lambda, \tag{47}$$

$$C^{\lambda}_{r,i} - C^{\lambda}_{r,k} + QD_{i,k} \geq \overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}, \qquad r = 1, \ldots, m,$$
$$i = 1, \ldots, n-1, k = i+1, \ldots, n, \lambda \in \Lambda, \tag{48}$$

$$C^{\lambda}_{r,i} - C^{\lambda}_{r,k} \leq Q(1 - D_{i,k}) - (\overline{p}_{r,k} + \widehat{p}_{r,k}\delta^{\lambda}_{r,k}), \qquad r = 1, \ldots, m,$$
$$i = 1, \ldots, n-1, k = i+1, \ldots, n, \lambda \in \Lambda, \tag{49}$$

$$C^{\lambda}_{r,i} \geq 0, \qquad r = 1, \ldots, m; i = 1, \ldots, n, \lambda \in \Lambda, \tag{50}$$

$$D_{i,k} \in \{0, 1\}, \qquad i = 1, \ldots, n-1, k = i+1, \ldots, n, \tag{51}$$

$$y \geq 0. \tag{52}$$

The objective function (44) and constraints (45) represent the worst-case total weighted completion time objective, i.e., the minimization of the maximum sum of the weighted completion time of all jobs on the last machine, given all

scenarios $\lambda \in \Lambda$. Constraints (46) insure that the completion time of each job on machine 1 occurs no earlier than the duration of that job's processing time on machine 1. Constraints (47) insure that each job's completion time on machine r is no earlier than the job's completion time on machine $r-1$ plus the job's processing time on machine r (with or without deviation). Constraints (48) and (49) are the paired disjunctive constraints, which insure that job i either precedes job k or follows job k in the sequence, but not both. Finally, constraints (50)-(52) define the domain of the variables.

4.3.7. Robust Counterpart for Liao-You PFS Model

Liao & You (1992) made algebraic combinations of each pair of Manne disjunctive inequality constraints. As a result, they obtained one equality constraint associated to a surplus variable, $q_{r,i,k}$, related to the precedence relationship of jobs i and k on machine r. To ensure feasibility, a second constraint was added to impose an upper bound on these surplus variables. Based on Liao-You model (makespan objective), we developed its robust counterpart, adapted for the TWCT objective, with variables y as described in the first formulation, along with the following additional decision variables:

$S^{\lambda}_{r,i}$	start time of job i on machine r given scenario λ.
$q^{\lambda}_{r,i,k}$	surplus time related to the precedence relationship of jobs i and k on machine r given scenario λ.

In this two-stage RO formulation, $D_{i,k}$ are, as in the previous model, the first-stage variables, while $S^{\lambda}_{r,i}$ and $q^{\lambda}_{r,i,k}$ are on the second stage. The robust

counterpart for Liao-You PFS model can be formulated as follows.

$$\text{Min } y \tag{53}$$

$$\text{st } (51), (52),$$

$$\sum_{i=1}^{n} w_i(S^{\lambda}_{m,i} + \overline{p}_{m,i} + \widehat{p}_{m,i}\delta^{\lambda}_{m,i}) \leq y, \qquad \lambda \in \Lambda, \tag{54}$$

$$S^{\lambda}_{r,i} + \overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i} \leq S^{\lambda}_{r+1,i}, \qquad r = 1, \ldots, m-1, i = 1, \ldots, n, \lambda \in \Lambda, \tag{55}$$

$$S^{\lambda}_{r,i} - S^{\lambda}_{r,k} + QD_{i,k} - (\overline{p}_{r,k} + \widehat{p}_{r,k}\delta^{\lambda}_{r,k}) = q^{\lambda}_{r,i,k}, \qquad r = 1, \ldots, m,$$
$$i = 1, \ldots, n-1, k = i+1, \ldots, n, \lambda \in \Lambda, \tag{56}$$

$$Q - (\overline{p}_{r,i} + \widehat{p}_{r,i}\delta^{\lambda}_{r,i}) - (\overline{p}_{r,k} + \widehat{p}_{r,k}\delta^{\lambda}_{r,k}) \geq q^{\lambda}_{r,i,k}, \qquad r = 1, \ldots, m,$$
$$i = 1, \ldots, n-1, k = i+1, \ldots, n, \lambda \in \Lambda, \tag{57}$$

$$S^{\lambda}_{r,i} \geq 0, \qquad r = 1, \ldots, m; i = 1, \ldots, n, \lambda \in \Lambda, \tag{58}$$

$$q^{\lambda}_{r,i,k} \geq 0, \qquad r = 1, \ldots, m; i = 1, \ldots, n-1, k = i+1, \ldots, n, \lambda \in \Lambda. \tag{59}$$

The objective function (53) and constraints (54) represent the worst-case total weighted completion time objective, i.e., the minimization of the maximum sum of the weighted completion time of all jobs on the last machine, given all scenarios $\lambda \in \Lambda$. Constraints (55) insure that each job's start time on machine $r+1$ is no earlier than the job's start time on machine r plus the job's processing time on machine r (with or without deviation). Constraints (56) and (57) are the paired disjunctive constraints, which insure that job i either precedes job k or follows job k in the sequence, but not both. Finally, constraints (51), (52), (58) and (59) define the domain of the variables.

RC Model	Binary variables	Continuous variables	Constraints
Wilson TBA WST2 TS2 TS3	$\mathcal{O}(n^2)$	$\mathcal{O}(\lambda mn)$	$\mathcal{O}(\lambda(n^2+mn))$
Manne	$\mathcal{O}(n^2)$	$\mathcal{O}(\lambda mn)$	$\mathcal{O}(\lambda mn^2)$
Liao-You	$\mathcal{O}(n^2)$	$\mathcal{O}(\lambda mn^2)$	$\mathcal{O}(\lambda mn^2)$

Tab. 3 Size complexity of the robust counterpart MILP models.

Tab. 3 presents the size complexity of each robust counterpart MILP model presented in this section. The number of binary variables remains the same of the original deterministic models, as they consist of first-stage variables. Also observe that the number of continuous variables as well as the number of con-

straints grow proportionally to the number of scenarios λ, as expected in robust counterpart formulations. Finally, the number of constraints in the assignment-based models (first five models in Tab. 3) is now quadratic in n, due to the calculation of the weighted completion time of each job, which requires the use of n^2 big-M constraints.

Solving each of the aforementioned models, for all possible scenarios $\lambda \in \Lambda$, is unrealistic. Therefore, in the next section, we will describe an algorithm capable of obtaining optimal results for the RPFS-TWCT problem under budgeted uncertainty, by considering a subset of relevant scenarios. Based on a robust counterpart model and a chosen budget parameter Γ, the solution algorithm will iteratively generate the necessary scenarios (and associated model constraints), so that the Λ set will be expanded at each step. This results in the progressive construction of a robust model which will ultimately solve RPFS-TWCT under budget uncertainty level Γ, and for a specific set of input parameters $\overline{\mathbf{P}}$ and $\widehat{\mathbf{P}}$. It is also worth noting that, by varying the value of Γ when solving each robust counterpart model through this procedure, it is possible to obtain a family of robust solutions with different degrees of conservativeness.

5. Column-and-Constraint Generation applied to the RPFS-TWCT

This section presents an exact method for solving RPFS-TWCT under budgeted uncertainty. Our approach is based on a cutting plane procedure for two-stage RO problems, called Column-and-Constraint Generation (C&CG), recently applied in the efficient solution of robust scheduling problems (Ruiz Duarte et al., 2020; Silva et al., 2020; Levorato et al., 2022). Besides generating new constraints, as usual in this kind of method, each cutting plane of C&CG 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 of the uncertainty set, i.e., for a $\Theta \subseteq \Lambda$. Then, given a

feasible solution to the MP, this solution is checked for feasibility over the whole set Λ, by solving a separation subproblem (SP). If the SP solution indicates that one or more robust constraints become infeasible, the uncertainty set is expanded by adding one or more scenario vectors to Θ. Whenever the master problem is augmented, according to the column-and-constraint generation procedure, the process is repeated.

For the RPFS-TWCT problem, the MP solution represents a permutation σ where $\sigma(j)$ is the order in which job j is executed. The separation problem is then solved by the worst-case procedure, which, given the sequence σ, returns the highest possible *total weighted completion time* under 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 can be fetched by the procedure is finite, and the C&CG algorithm certainly terminates (Zeng & Zhao, 2013).

5.1. C&CG algorithm

We describe the solution method in a general way that can be applied to any two-stage RO formulation from Section 4.3. Following the structure of the C&CG method, we define the Master Problem (MP) by choosing an appropriate 2-stage RO formulation. Considering $\Theta = \{\lambda_1, \ldots, \lambda_v\} \subseteq \Lambda$ a subset of scenarios, let $\mathcal{F}_{model}$ and $\mathcal{R}_{model}$ be the set of corresponding first-stage and recourse decision variables of the RC model, respectively. For instance, $\mathcal{F}_{Wilson} = \{\mathrm{Z_{i,j}}, \forall i, j = 1, \ldots, n\}$ and $\mathcal{R}_{Wilson} = \{\mathrm{B}^{(\lambda)}_{r,j}, \mathrm{F}^{(\lambda)}_{r,j}, \forall \lambda \in \Theta, r = 1, \ldots, m, j = 1, \ldots, n\}$. The master problem (MP) is solved iteratively, with each step generating a subset of problem constraints and associated recourse variables $\mathcal{R}$, regarding one newly-generated scenario $\lambda_v \in \Theta$. The subset of scenarios Θ is iteratively enlarged by solving the associated subproblem at each iteration.

In order to generate the scenarios defined by Θ, we assume that an oracle can obtain an optimal solution to the worst-case subproblem, based on the current MP solution. At iteration v, for a given value of MP first-stage decision variables

$\mathcal{F}$, the subproblem SP is defined as:

$$\text{(SP)} \quad \mathcal{Z}(\sigma) = \max_{\lambda_v \in \Lambda} \varphi(\sigma, \mathbf{P}^{\lambda_v}) \tag{60}$$

where job permutation σ is derived using MP optimal values of first-stage variables $\mathcal{F}$ at iteration v (either $Z_{i,j}^{(v)}$ or $D_{i,j}^{(v)}$, depending on the RC model). The oracle used to find the optimal solution $\lambda_{(v)}^*$ for (SP) is the worst-case MILP described 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, and Θ is the set of worst-case scenarios generated by the method. The procedure starts by considering Θ with a single scenario in which no operation presents processing time oscillation, and stops whenever the tolerance of optimality $\epsilon \in \mathbb{R}^+$ is reached. It returns the optimal solution value of the robust problem, along with the first-stage variables $\mathcal{F}^*$, which represent the optimal permutation σ^*.

5.2. C&CG subproblem: worst-case evaluation based on a MILP model

Solving the SP problem in (60) consists in determining 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 Γ and a schedule σ, we extend the definition of *worst-case total weighted completion time* or *robust cost* to $\mathcal{Z}(\sigma, \Gamma)$ with the equation:

$$\mathcal{Z}(\sigma, \Gamma) := \max_{\mathbf{P}^\lambda \in \mathcal{U}^\Gamma} \{\varphi(\sigma, \mathbf{P}^\lambda)\}. \tag{61}$$

We assume that parameter Γ, from the budgeted uncertainty set, is a non-negative integer. Since any worst-case realization will use as much budget of uncertainty as possible, we can expect that, for the optimal solution of (61), with worst-case scenario $\lambda_{(v)}^*$, $\sum_{r=1}^{m} \sum_{i=1}^{n} \delta_{r,i}^{\lambda^*} = \Gamma$.

The worst-case scenario, and associated robust cost, can be obtained by solving the following proposed SP MILP. As input parameters, besides the processing time matrices $\overline{p}_{r,i}$ and $\widehat{p}_{r,i}$, the SP MILP requires the budget parameter Γ along with the sequence of jobs σ^*, provided by the current Master Problem solution $\mathcal{F}^*$. Since the proposed SP MILP relies on an assignment-based formu-

Algorithm 1: Column-and-constraint generation algorithm

Set $LB = -\infty, UB = +\infty, v = 1, \Theta = \{\lambda_{(0)} : \delta_{r,i}^{(0)} = 0, \forall r = 1, \ldots, m, i = 1, \ldots, n\}$;
while (UB − LB)/LB $> \epsilon$ **do**
  if *model*=Wilson **then** Solve the MP defined in Section 4.3.1 with $\Lambda := \Theta$;
  if *model*=TBA **then** Solve the MP defined in Section 4.3.2 with $\Lambda := \Theta$;
  if *model*=WST2 **then** Solve the MP defined in Section 4.3.3 with $\Lambda := \Theta$;
  if *model*=TS2 **then** Solve the MP defined in Section 4.3.4 with $\Lambda := \Theta$;
  if *model*=TS3 **then** Solve the MP defined in Section 4.3.5 with $\Lambda := \Theta$;
  if *model*=Manne **then** Solve the MP defined in Section 4.3.6 with $\Lambda := \Theta$;
  if *model*=Liao-You **then** Solve the MP defined in Section 4.3.7 with $\Lambda := \Theta$;
  Let $(\mathcal{F}_{(v)}^*,\ y^*,\ \mathcal{R}_{model}^*)$ be the MP optimal solution ;
  Update LB := max $\left[\text{LB},\ y^*\right]$;
  Call the oracle to solve subproblem (SP) in (60) with $\mathcal{F} := \mathcal{F}_{(v)}^*$;
  Let $(\mathcal{Z}_{(v)}^*,\ \lambda_{(v)}^*)$ be the SP optimal solution value and associated worst-case scenario, respectively ;
  Update UB := min $\left[\text{UB},\ \mathcal{Z}_{(v)}^*\right]$;
  if (UB − LB)/LB $> \epsilon$ **then**
    Create recourse decision variables $\mathcal{R}_{(v)}$ for scenario $\lambda_{(v)}^*$ on MP ;
    Update $\mathcal{R}_{model} := \mathcal{R}_{model} \cup \{\mathcal{R}_{(v)}\}$;
    if *model*=Wilson **then** Generate MP constraints (5)-(11),(15)&(16) for $\lambda_{(v)}^*$;
    if *model*=TBA **then** Generate MP constraints (5),(16),(19)-(22) for $\lambda_{(v)}^*$;
    if *model*=WST2 **then** Generate MP constraints (5),(16),(22),(24)-(31) for $\lambda_{(v)}^*$;
    if *model*=TS2 **then** Generate MP constraints (5),(16),(33)-(39) for $\lambda_{(v)}^*$;
    if *model*=TS3 **then** Generate MP constraints (5),(16),(31),(41)-(43) for $\lambda_{(v)}^*$;
    if *model*=Manne **then** Generate MP constraints (45)-(50) for $\lambda_{(v)}^*$;
    if *model*=Liao-You **then** Generate MP constraints (54)-(59) for $\lambda_{(v)}^*$;
    Update $\Theta := \Theta \cup \{\lambda_{(v)}^*\}$ and set $(v) := (v + 1)$;
Return UB, $\mathcal{F}_{(v)}^*$;

lation, an equivalent input matrix of assignment values $z_{i,j}^*$ needs to be derived in the following way:

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

Problem variables	
$\Delta_{r,i} =$	$\begin{cases} 1, \text{ if job } i \text{ will have its processing time deviated on machine } r \\ 0, \text{ otherwise.} \end{cases}$
$C_{r,j} =$	completion time of job $\sigma(j)$ (in position j) on machine r.
$E_{r,j} =$	receives the value corresponding to $\min(C_{r,j-1}, C_{r-1,j})$.
$A_{r,j} =$	contains the value of $\mid C_{r,j-1} - C_{r-1,j} \mid$.
$D_{r,j} =$	(binary) models the disjunction used to calculate $A_{r,j}$ as an absolute value.

The worst-case MILP is stated as follows:

$$\text{Max } \sum_{i=1}^{n} w_i \sum_{j=1}^{n} C_{m,j} z^*_{i,j} \tag{62}$$

$$\text{st } C_{1,1} = \sum_{i=1}^{n} (\overline{p}_{1,i} + \widehat{p}_{1,i}\Delta_{1,i}) z^*_{i,1}, \tag{63}$$

$$C_{1,j} = C_{1,j-1} + \sum_{i=1}^{n} (\overline{p}_{1,i} + \widehat{p}_{1,i}\Delta_{1,i}) z^*_{i,j}, \quad j = 2, \ldots, n, \tag{64}$$

$$C_{r,1} = C_{r-1,1} + \sum_{i=1}^{n} (\overline{p}_{r,i} + \widehat{p}_{r,i}\Delta_{r,i}) z^*_{i,1}, \quad r = 2, \ldots, m, \tag{65}$$

$$E_{r,j} \leq C_{r,j-1}, \quad j = 2, \ldots, n, r = 2, \ldots, m, \tag{66}$$

$$E_{r,j} \leq C_{r-1,j}, \quad j = 2, \ldots, n, r = 2, \ldots, m, \tag{67}$$

$$A_{r,j} \geq C_{r,j-1} - C_{r-1,j}, \quad j = 2, \ldots, n, r = 2, \ldots, m, \tag{68}$$

$$A_{r,j} \geq -(C_{r,j-1} - C_{r-1,j}), \quad j = 2, \ldots, n, r = 2, \ldots, m, \tag{69}$$

$$A_{r,j} \leq C_{r,j-1} - C_{r-1,j} + QD_{r,j}, \quad r = 2, \ldots, m, j = 2, \ldots, n, \tag{70}$$

$$A_{r,j} \leq -(C_{r,j-1} - C_{r-1,j}) + Q(1 - D_{r,j}), \quad r = 2, \ldots, m, j = 2, \ldots, n, \tag{71}$$

$$C_{r,j} \leq \sum_{i=1}^{n} (\overline{p}_{r,i} + \widehat{p}_{r,i}\Delta_{r,i}) z^*_{i,j} + E_{r,j} + A_{r,j}, \quad r = 2, \ldots, m, j = 2, \ldots, n, \tag{72}$$

$$\sum_{r=1}^{m} \sum_{i=1}^{n} \Delta_{r,i} \leq \Gamma \ , \tag{73}$$

$$\Delta_{r,i} \in \{0, 1\}, \qquad r = 1, \ldots, m, i = 1, \ldots, n, \tag{74}$$

$$C_{r,j} \geq 0, \qquad r = 1, \ldots, m, j = 1, \ldots, n, \tag{75}$$

$$A_{r,j} \geq 0, \ E_{r,j} \geq 0, \ D_{r,j} \in \{0, 1\}, \ r = 1, \ldots, m, j = 1, \ldots, n, \tag{76}$$

The objective function (62) states that, given a fixed job permutation $z^*_{i,j}$, this formulation aims to find a worst-case processing time scenario that maximizes the *weighted sum of completion times*, among all possible scenarios defined by $\mathcal{U}^{\Gamma}$. Constraints (63)-(64) are used to determine the completion time of the jobs on the first machine, while constraints (65) define the completion time of the first job on each machine r. For each machine r and job position j, constraints (66) and (67) are used to calculate the minimum value between the completion time of the previous job on the same machine ($C_{r,j-1}$) and the completion time of the same job on the previous machine ($C_{r-1,j}$). Constraints (68)-(69), together with disjunctive constraints (70)-(71) are used to determine the absolute value of the difference between $C_{r,j-1}$ and $C_{r-1,j}$. These absolute values are used to define the completion time $C_{r,j}$. Constraints (72) ensure that the completion time $C_{r,j}$ is bounded by the processing time of job $\sigma^*(j)$ (in position j) on machine r, plus the maximum of $C_{r,j-1}$ and $C_{r-1,j}$, which is equivalent to the minimum of these two variables ($E_{r,j}$) plus the absolute difference between the

same two variables ($A_{r,j}$). Constraints (73) define the budget of uncertainty regarding the maximum allowed processing time deviations $\sum_{r=1}^{m}\sum_{i=1}^{n}\Delta_{r,i}$ given the execution of all jobs i on all machines r. Finally, constraints (74)-(76) define the domain of the variables.

We employed two strategies to improve the performance of the SP worst-case MILP model. First, we adopted a problem-specific method when calculating Big-M values, where each Q value varies according to the constraint it belongs to. Second, in order to strengthen the formulation, the following valid inequality was added, improving solution times by a factor of 10:

$$A_{r,j} = C_{r,j-1} + C_{r-1,j} - 2 \times E_{r,j}. \tag{77}$$

The optimal solution of this MILP model, represented by $\Delta^*_{r,i}$ values, consists in a valid worst-case scenario λ^* under budget uncertainty set $\mathcal{U}^\Gamma$. Remark that $\Delta^*_{r,i}$ values are used to define $\delta^\lambda_{r,i}$ values for the scenario added to set Θ, and used in the robust counterparts of the 2-stage RO models.

In the experiments shown in Section 7, the convergence of the C&CG method was also accelerated by generating multiple worst-case scenarios at each iteration, whenever possible.

Our computational experiments have evidenced that the limits of the proposed C&CG solution method lie in the solution of the Master Problem. In particular, we observed a high proportion of time spent when solving Master Problems for instances with 15 jobs and 5 machines. Nonetheless, for the oil and gas maintenance problem at hand, improved solutions are needed for instances of this size. Therefore, in the next section, we will propose an algorithm enhancement to overcome this limitation.

6. Hybrid C&CG Method

Unsurprisingly, the C&CG will suffer of computational limitation as instance size grows, in particular when solving the Master Problem. For this reason, we devise an improved MP solution method, which brings a combinatorial branch-and-bound inside the MILP solver tree structure.

With this new approach, we implemented an alternative Master Problem solution method for assignment-based and dichotomous-based Robust Counterparts. Similarly to the method presented in Algorithm 1, the alternative MP solution method relies on a RC model invoked in an iterative way, based on a list of *C&CG cuts* provided. We denote as *Hybrid C&CG Method* the C&CG solution method that incorporates this new MP solution technique. The main advantage relies on the alternative branching strategy employed, which provides new information used to prune nodes, as well as a powerful combinatorial lower bound.

The implementation of the hybrid C&CG method was based on the CPLEX solver 20.1. Based on its branch callback, we developed a combinatorial branch-and-bound emulation similar to Rubin (2014), which will be described next.

6.1. Branching strategy

Consider the search tree of the classic flow shop combinatorial branch-and-bound (Lageweg et al., 1978), depicted in Fig. 1(a). The root node (at level 0) represents the *null schedule*. A given node N at level s represents a *partial schedule* $\sigma = (\sigma(1), \ldots, \sigma(s))$ of size s, indicating that job $\sigma(j)$ occupies the

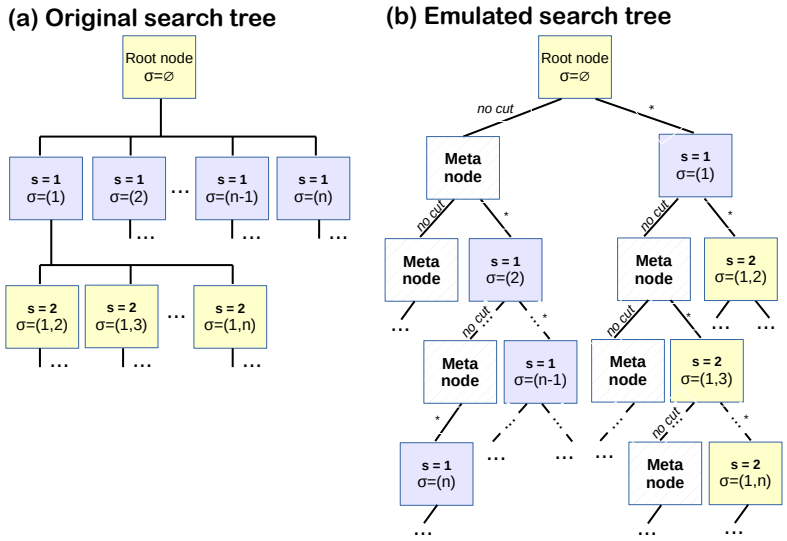

Fig. 1 (a) Search tree of the deterministic flow shop combinatorial branch-and-bound. (b) Diagram illustrating how flow shop multi-way branching was performed in CPLEX.

j-th position on each machine, for $1 \leq j \leq s$, where $1 \leq s \leq n$. Let U be the set of jobs that are not included in the partial schedule σ, i.e., unscheduled jobs. By placing any unscheduled job i in position $(s+1)$, we produce a child node $\sigma i = (\sigma(1), \ldots, \sigma(s), i)$, in level $s+1$.

It is clear that the flow shop search tree requires several branches at each node. However, CPLEX allows the creation of at most two branches at a node. To circumvent this limitation and produce more than two branches, we must emulate multi-way branching by binary branching. To accomplish that, instead of generating the branching tree of Fig. 1(a), we create a branching structure following the diagram in Fig. 1(b). Consider an arbitrary node N from Fig. 1(a). The new branching scheme produces exactly the same offsprings of each original node, but in multiple levels. In this case, one branch is always one of the children to be created (here called a *permutation branch*), while the second branch is a duplicate of the parent node N, which we call a *meta node*.

Whenever a new *permutation branch* is created, an unscheduled job i will be fixed in position j of the partial permutation σ. This new partial permutation has to be reflected, in some way, on the node information manipulated by CPLEX, via a set of *node cuts*. For the flow shop MILP RC models at hand, this means one or more binary variables must have their bounds fixed. Observe that the other branch created, which contains the *meta node*, will receive no additional *node cuts* associated to it, but will receive additional information about the partial sequence generation.

For assignment-based flow shop RC models, variables Z will be fixed:

• Job i occupies sequence position j:

$$Z_{i,j} := 1; \tag{78}$$

• Job i cannot occupy any other position k rather than position j:

$$Z_{i,k} := 0, \forall k \neq j; \tag{79}$$

• No other job $\ell \neq i$ can occupy position j:

$$Z_{\ell,j} := 0, \forall \ell \neq i. \tag{80}$$

For dichotomous-based flow shop RC models, partial order variables D will

be fixed:

• Set all jobs ℓ that come before job i in partial permutation σ:

$$D_{\ell,i} := 1, D_{i,\ell} := 0, \forall \ell \in \sigma, \tag{81}$$

• All jobs ℓ that have not been scheduled yet will necessarily come after job i:

$$D_{i,\ell} := 1, D_{\ell,i} := 0, \forall \ell \in U. \tag{82}$$

6.2. Improved lower bound

When solving the Master Problem, at each node of the B&B tree, in addition to the branching strategy above, an extended combinatorial lower bound can be applied as an additional criterion to prune nodes. Consider the MP is being solved at iteration v of the *hybrid C&CG method*. At this point, a set of $v-1$ *C&CG cuts* (i.e., violated scenarios λ) has already been generated and applied to the MP RC model, as explained in Section 5.1. The list of existing *C&CG cuts* can be then used to calculate the following combinatorial lower bound LB_{MP}:

$$LB_{MP} = \max_{\lambda \in \Lambda} LB_{det}(P^{\lambda}), \tag{83}$$

where LB_{det} is the lower bound of the deterministic PFSP-TWCT, assuming scenario λ and processing time matrix P^{λ}.

To calculate (83), we applied the tight lower bound for the deterministic problem described by Chung et al. (2002). These authors developed a branch and bound algorithm to solve the m-machine permutation flowshop problem, which assumes that a partial permutation is defined at each step. In their work, they considered two possible objectives: the unweighted and weighted total flow-time (i.e., TWCT). Their solution method efficiently handles test problems with $n \leq 15$, thanks to an improved machine-based lower bound, together with a dominance test for pruning nodes.

It is worth noting that, despite the overhead from the combinatorial branch-and-bound emulation, the use of the lower bound LB_{MP} to prune nodes has proved to be essential to the performance gains obtained with this new Master Problem solution method.

7. Experimental results

We conducted extensive experiments to assess the performance of the C&CG solution method as well as the proposed Robust Counterpart formulations.

7.1. Test instances

In the flow shop literature, there is no set of benchmark instances for the *total weighted completion time* objective. In order to verify the effectiveness of the proposed algorithms, our experiments were based on three instance sets[3] obtained by adapting a robust PFSP instance generator described by Ying (2015).

(i) **Two-machine robust PFSP instances with 10 jobs (10x2)**. In his work, Ying (2015) proposed six groups of instances, each one with a different number of jobs $n \in \{10, 20, 50, 100, 150, 200\}$. 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 is set as a fixed ratio of the expected processing time (i.e., $\widehat{p}_{r,i} = \alpha \overline{p}_{r,i}$), with $\alpha \in \{10\%, 20\%, 30\%, 40\%, 50\%\}$. Ten instances were generated for each combination of n and α, for a total of 300 test instances.

(ii) **Robust PFSP instances with 3, 4 and 5 machines**. Following the instance generation algorithm of Ying (2015), we generated random instances with sizes $n \times m \in \{10 \times 3, 10 \times 4, 10 \times 5, 15 \times 5\}$. Ten instances were generated for each combination of $m \times n$ and α, for a total of 200 test instances.

(iii) **Robust PFSP instances with random processing time deviations**. For each instance of the previous two sets with variability level $\alpha = 10\%$, we generated 4 new instances with distinct variability levels $\alpha_{r,i}$ for each operation $O_{r,i}$. First, we define a maximum variability level $\alpha^{max} \in \{30\%, 50\%, 100\%, 200\%\}$. Then, in each generated instance, the variability level $\alpha_{r,i}$ of each operation $O_{r,i}$ is drawn from a uniform distri-

[3] All test instances and results are available at `https://github.com/levorato/RPFS_Budget_TWCT`.

bution in the interval $[0, \alpha^{max})$. Therefore, the maximum processing time deviation of each operation equals $\widehat{p}_{r,i} = \alpha_{r,i}\overline{p}_{r,i}$. The idea behind this new set is to generate instances whose operation processing time deviation follow a completely random behavior, when compared to the previous sets. This way, we will be able to assess the impacts of such behavior on the solution method.

Since all instances above are related to the *makespan* objective, no job weight information is available. Thus, for each instance, job weights (w_j, $\forall j \in \mathbb{J}$) were randomly generated, according to a uniform distribution in the interval $[1, 100]$. These values are based on the job weight distribution from the real-world instances studied in Section 8.

7.2. Computational environment and model observations

The C&CG algorithm was coded in Julia 1.6.0. CPLEX solver 20.1 was used to solve the Master Problems (MP) and Gurobi solver 9.1 was used to solve the subproblems SP, since it obtained improved performance in preliminary experiments. The MILP time limit was set to 7,200 s and the number of threads was set to 16. All experiments were conducted on a workstation with an Intel Xeon® CPU E5640 @2.67GHz with 32 GB RAM, under Ubuntu 18.04 LTS. In the C&CG algorithm, optimality gap tolerance ϵ was set to 10^{-8}.

In the literature (Tseng & Stafford, 2008), empirical tests have shown that the top 3 best performing PFSP MILP models are, in this order: TS3, TBA and Wilson. In this work, we will observe that the performance obtained with the PFSP robust counterparts is rather different to the existing performance of deterministic PFSP MILP models.

7.3. Comparative performance of the Robust Counterpart models

With a particular interest in examining the impact of the budget of uncertainty parameter on scheduling performance, when solving each instance, we tested the RPFS-TWCT RC models by varying Γ according to ten ratios (10,

Model	% Best Performance	% Solved 10x2	% Solved 10x3	% Solved 10x4	% Solved 10x5	% Solved	Avg. % gap	Median iterations	Median time
Manne	45.70	99.56	96.56	91.67	90.89	94.67	0.07	5.00	88.10
Liao-You	34.38	99.44	96.44	91.89	90.11	94.47	0.08	5.00	85.32
Wilson	13.78	99.33	95.00	83.56	80.89	88.38	2.33	5.00	276.03
TS2	6.03	99.89	95.67	85.22	85.67	90.78	3.08	4.00	436.48
TS3	4.95	99.56	92.67	78.11	86.78	89.28	1.65	5.00	334.56
TBA	0.51	99.33	87.78	71.44	69.67	82.06	1.76	5.00	883.80
WST2	0.14	98.00	86.78	69.67	66.44	80.22	2.09	5.00	1,143.74

Tab. 4 Robust PFSP C&CG performance comparison, given all RC models and instances solved. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; % Solved $< n \times m >$ represents the percentage of solved instances of size $n \times m$; 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.

20, 30, 40, 50, 60, 70, 80, 90 and 100%) of operations with uncertain processing times, rounded to their floor values.

Tab. 4 summarizes the obtained results with a performance comparison of the RC models. In this table, we present medians to mitigate the effect of instances not solved within the time limit. Manne C&CG is the one that solves the majority of the instances. It also obtains the lowest average % gap for instances not solved to optimality. The *% Best Performance* measurement indicates that the Manne RC model solved 46% of instances with the best performance, followed by Liao-You, that solved 34%, and Wilson, with 14%. Measurements *% Solved 10 × 4* and *% Solved 10 × 5* reveal that the RC models which rely on job assignment constraints solved less instances to optimality within the time limit. The *%Solved* and *Median time* measurements also favor the dichotomous-based RC models.

A second and deeper analysis, grouped by instance size, presents, in Tab. 5, the average performance of each RC 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 its standard deviation are listed. As we could expect, these results show that, as instance size grows, the RC models become harder to solve, as seen on the smaller percentage of solved instances and increased average execution time, mainly the *Avg. MP time*. In fact, our results show that this is especially true for the assignment-based RC models. As

		Dichotomous-based		Assignment-based				
Instance size	Measure	Manne	Liao-You	Wilson	TS2	TS3	TBA	WST2
10x2	% Best Performance	36.83	17.43	31.10	12.90	0.78	1.23	0.11
	% Solved	99.56	99.44	99.33	99.89	99.56	99.33	98.00
	Avg. % gap	0.00	0.00	0.07	0.00	5.78	0.52	0.78
	Std. dev. of % gap	0.00	0.00	0.18	0.00	11.39	0.68	0.85
	Avg. iterations	7.69	8.64	5.60	3.81	4.57	3.93	3.98
	Std. dev. of iterations	56.32	66.12	19.00	3.53	16.31	4.71	2.47
	Avg. MP time	112.94	104.42	137.05	316.55	210.11	375.57	666.10
	Avg. SP time	0.85	0.95	0.38	0.39	0.55	0.51	0.37
	Avg. time	113.79	105.37	137.43	316.95	210.67	376.07	666.47
	Std. dev. of time	532.32	546.95	618.42	802.44	604.03	944.73	1,354.99
10x3	% Best Performance	49.65	32.26	6.78	6.85	8.15	0.00	0.00
	% Solved	96.56	96.44	95.00	95.67	92.67	87.78	86.78
	Avg. % gap	0.00	0.04	4.88	1.20	1.20	2.26	1.23
	Std. dev. of % gap	0.00	0.21	9.66	1.63	1.71	8.64	1.68
	Avg. iterations	18.87	17.70	6.21	6.20	6.17	5.62	5.71
	Std. dev. of iterations	76.10	67.38	7.72	7.62	5.21	3.36	4.18
	Avg. MP time	416.65	477.63	1,144.08	1,379.19	1,276.20	2,151.87	2,302.40
	Avg. SP time	7.35	9.19	8.97	8.13	4.36	5.53	3.47
	Avg. time	424.00	486.82	1,153.05	1,387.32	1,280.56	2,157.40	2,305.87
	Std. dev. of time	1,317.86	1,386.44	1,888.22	2,005.33	2,104.25	2,440.00	2,559.65
10x4	% Best Performance	30.26	60.27	10.11	2.09	4.13	0.31	0.48
	% Solved	91.67	91.89	83.56	85.22	78.11	71.44	69.67
	Avg. % gap	0.12	0.11	1.65	1.98	2.04	2.20	2.25
	Std. dev. of % gap	0.33	0.35	2.20	2.34	2.51	2.79	2.89
	Avg. iterations	23.26	24.42	10.11	7.75	7.73	6.57	6.25
	Std. dev. of iterations	69.66	77.97	18.05	6.80	6.47	3.78	3.48
	Avg. MP time	970.49	930.11	2,076.57	2,234.34	2,524.83	3,394.95	3,544.79
	Avg. SP time	52.43	46.19	48.41	43.88	53.68	34.56	28.33
	Avg. time	1,022.92	976.31	2,124.98	2,278.23	2,578.52	3,429.51	3,573.12
	Std. dev. of time	2,047.19	2,018.60	2,643.78	2,568.41	2,828.57	2,819.48	2,881.46
10x5	% Best Performance	66.95	29.10	4.40	1.04	7.09	0.32	0.00
	% Solved	90.89	90.11	80.89	85.67	86.78	69.67	66.44
	Avg. % gap	0.04	0.08	1.15	1.10	1.11	1.16	2.37
	Std. dev. of % gap	0.16	0.26	1.56	1.65	1.40	1.64	5.82
	Avg. iterations	35.26	32.64	8.23	8.03	6.72	6.79	5.93
	Std. dev. of iterations	109.28	100.17	12.70	12.17	4.90	6.61	4.75
	Avg. MP time	807.29	882.73	2,204.63	1,971.19	1,850.99	3,242.81	3,535.38
	Avg. SP time	196.24	229.35	208.22	280.31	206.78	177.76	298.72
	Avg. time	1,003.53	1,112.08	2,412.85	2,251.50	2,057.77	3,420.57	3,834.10
	Std. dev. of time	2,086.74	2,166.21	2,726.14	2,532.00	2,523.15	2,916.78	2,878.93

Tab. 5 Robust PFSP C&CG performance comparison, for each instance size $n \times m$ and RC model. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; Avg. % Gap and Std. dev. of % Gap are the mean and standard deviation of the percentage gap of solutions from instances not solved to optimality; Avg. iterations and Std. dev. of iterations are the mean and standard deviation of the number of iterations performed; Avg. time MP(SP) is the average time to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

instance size grows, the harder to solve these models at each C&CG iteration, with more time spent at each iteration and less iterations performed on average.

A complementary investigation, based on the α and α^{max} parameters, is portrayed in Tab. 6. In this context, we explore solution statistics regarding the four best performing models, when solving 10×5 instances. It is possible to note the decrease of model performance as the α and α^{max} values grow. This can be observed in the *% Solved*, *Avg. % gap* and *Avg. time* rows, from columns $\alpha = 10\%$ until $\alpha = 50\%$, and from columns $\alpha^{max} = 30\%$ until $\alpha^{max} = 200\%$.

Model	Variable	α=10%	α=20%	α=30%	α=40%	α=50%	α^{max}=30%	α^{max}=50%	α^{max}=100%	α^{max}=200%
Manne	% Best Performance	55.10	75.00	82.14	75.00	38.89	60.61	80.21	65.98	67.01
	% Solved	98.00	96.00	85.00	73.00	73.00	99.00	96.00	100.00	98.00
	Avg. % gap	0.00	0.00	0.00	0.03	0.06	0.00	0.00	0.00	0.57
	Std. dev. of % gap	0.00	0.00	0.00	0.11	0.20	0.00	0.00	0.00	0.08
	Avg. iterations	10.02	20.02	57.25	95.02	82.81	6.71	24.29	8.82	12.36
	Avg. MP time	198.80	343.21	1,151.25	1,996.76	1,997.19	150.97	425.11	365.75	636.58
	Avg. SP time	13.34	103.91	215.03	457.65	575.86	13.88	81.86	89.68	214.94
	Avg. time	212.14	447.12	1,366.27	2,454.41	2,573.06	164.85	506.97	455.43	851.52
	Std. dev. of time	1,007.64	1,401.65	2,531.63	3,052.05	2,960.42	719.07	1,419.05	861.10	1,350.03
Liao-You	% Best Performance	40.82	22.92	17.86	23.61	58.33	25.25	16.67	27.84	32.99
	% Solved	98.00	96.00	84.00	72.00	72.00	99.00	96.00	97.00	97.00
	Avg. % gap	0.00	0.00	0.00	0.06	0.10	0.00	0.00	0.10	0.85
	Std. dev. of % gap	0.00	0.00	0.01	0.19	0.27	0.00	0.00	0.12	0.66
	Avg. iterations	10.47	17.30	49.77	86.26	80.75	5.20	23.08	8.77	12.19
	Avg. MP time	230.66	395.28	1,314.84	2,080.92	2,109.72	178.98	470.81	466.47	696.90
	Avg. SP time	19.05	120.31	270.29	537.54	640.74	13.57	98.86	110.64	253.15
	Avg. time	249.71	515.59	1,585.13	2,618.46	2,750.46	192.55	569.67	577.11	950.05
	Std. dev. of time	1,021.04	1,419.79	2,593.14	3,060.68	3,027.40	723.47	1,426.31	1,258.07	1,532.42
TS2	% Best Performance	0.00	2.06	0.00	0.00	0.00	3.03	1.10	2.33	0.00
	% Solved	100.00	97.00	83.00	75.00	70.00	99.00	91.00	86.00	70.00
	Avg. % gap	0.00	0.00	0.22	0.55	0.89	0.00	0.19	1.63	2.46
	Std. dev. of % gap	0.00	0.00	0.33	0.63	1.14	0.00	0.29	1.41	2.50
	Avg. iterations	3.76	5.99	8.20	11.61	11.20	4.76	10.37	7.47	8.94
	Avg. MP time	1,017.22	1,584.34	2,063.25	2,798.91	3,133.31	669.08	1,549.04	1,800.18	3,125.41
	Avg. SP time	197.62	354.42	496.12	465.72	542.74	15.98	73.83	126.85	249.46
	Avg. time	1,214.85	1,938.76	2,559.38	3,264.63	3,676.05	685.06	1,622.87	1,927.04	3,374.87
	Std. dev. of time	1,404.58	2,030.54	2,545.68	2,696.03	2,697.30	1,182.24	2,337.84	2,468.36	3,084.71
Wilson	% Best Performance	3.09	1.25	1.32	5.71	5.88	9.09	6.59	4.71	0.00
	% Solved	97.00	80.00	76.00	70.00	68.00	99.00	91.00	85.00	62.00
	Avg. % gap	0.07	0.34	0.68	0.79	1.17	0.00	0.23	1.93	2.18
	Std. dev. of % gap	0.12	0.34	0.55	0.76	1.10	0.00	0.37	1.77	2.46
	Avg. iterations	3.90	7.98	8.42	9.12	13.13	4.70	8.53	7.26	11.00
	Avg. MP time	1,165.72	2,266.70	2,619.72	3,019.32	3,257.35	747.88	1,508.29	1,941.32	3,315.41
	Avg. SP time	133.02	203.90	321.46	349.65	352.45	22.32	68.82	132.49	289.86
	Avg. time	1,298.74	2,470.60	2,941.17	3,368.97	3,609.79	770.20	1,577.11	2,073.81	3,605.27
	Std. dev. of time	1,944.91	2,777.29	2,764.50	2,903.50	2,836.88	1,265.93	2,319.08	2,529.15	3,098.39

Tab. 6 Robust PFSP C&CG performance comparison for instance size 10×5, grouped by α and α^{max} values, and RC model. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit; Avg. % Gap is the average percentage gap of solutions from instances not solved to optimality; Avg. iterations is the average number of iterations performed; Avg. time MP(SP) is the average time to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

Instance size	Variable	TS2	Wilson	Liao-You	Manne	Liao-You-Hybrid	Manne-Hybrid	Wilson-Hybrid	TS3-Hybrid
15x5	% Best Performance	0.00	0.00	0.00	0.00	5.59	26.95	36.61	36.34
	% Solved	5.71	6.90	12.98	14.15	31.78	41.67	47.33	48.22
	Avg. % gap	20.13	7.46	6.16	4.62	4.04	3.59	5.29	5.00
	Std. dev. of % gap	14.53	11.75	10.15	8.42	6.19	5.53	7.39	7.20
	Avg. iterations	2.14	3.05	3.00	3.31	4.76	5.91	5.83	5.82
	Std. dev. of iterations	0.73	0.87	1.23	1.49	1.85	2.48	3.63	3.20
	Avg. MP time	11,073.67	13,277.03	11,681.69	11,218.73	10,795.78	9,344.76	8,893.31	9,090.81
	Avg. SP time	3,050.96	358.64	1,834.73	2,101.60	640.71	988.21	920.33	780.12
	Avg. time	14,124.64	13,635.67	13,516.42	13,320.34	11,436.49	10,332.97	9,813.64	9,870.92
	Std. dev. of time	1,222.25	2,849.15	2,907.75	3,302.32	5,220.83	5,755.23	5,902.10	5,902.38

Tab. 7 Robust PFSP C&CG performance comparison, for instance size 15×5, RC models TS2, Wilson, Liao-You and Manne, along with Hybrid C&CG models Liao-You-Hybrid, Manne-Hybrid, Wilson-Hybrid, and TS3-Hybrid. % Best Performance is the percentage of instances where the model achieved shorter execution time (ties included); % Solved contains the percentage of instances solved within the time limit of 14400s; Avg. % Gap and Std. dev. of % Gap are the mean and standard deviation of the percentage gap of solutions from instances not solved to optimality; Avg. iterations and Std. dev. of iterations are the mean and standard deviation of the number of iterations performed; Avg. time MP(SP) is the average time to solve the Master(Sub) Problem; Avg. time and Std. dev. of time are the mean and standard deviation in solution time (in seconds), respectively.

According to our experiments, the Liao-You and Manne robust counterparts are the ones that perform best when solving the RPFS-TWCT problem. Regarding the *% Best Performance* measure, these two RC models dominate

every other model, regardless of the $\alpha(\alpha^{max})$ value. Together with the *% Solved* measure, this indicates that the dichotomous-based RC models are the best choice when solving 10×5 instances. The possible reason is related to how the objective function is calculated in each robust counterpart model. In all assignment-based RC models (namely Wilson, WST2, TBA, TS2 and TS3), when calculating the total weighted completion time, we multiply the weight of job i by its corresponding completion time. However, in these models, there are no variables representing the completion time of job index i. Instead, they represent the completion time of job in *position* k. To properly calculate the objective function, the adopted solution involves the creation of an auxiliary variable F_i, representing the completion time of job i (on the last machine M_m). Then, in order to calculate each value of F_i, it is necessary to apply several Big-M constraints, which make the MILP model relaxation weaker.

On the other hand, the Liao-You and Manne robust counterparts, which are based on disjunctive constraints, directly offers these variables which represent the completion time of job index i, so the only Big-M constraints in the model are the ones already present in the original model.

Finally, when solving the 15×5 instances, the largest ones in the test-bed, we perceived a drastic performance reduction of the algorithm. For this reason, besides extending the time limit parameter to $14,400$ seconds, we chose to solve these instances with the four best performing RC models so far: Manne, Liao-You, TS2 and Wilson. As shown in Tab. 7, the percentage of solved instances drops from 98% to 14% when applying the best-performing Manne RC model. Also, when analyzing the % gap of instances not solved to optimality, the average % gap is considerably higher in all models, as well as its variance. We can see that, for these instances, the C&CG algorithm was not able to perform more than 3 iterations on average.

7.4. Hybrid C&CG method performance

We will now discuss the performance of the hybrid algorithm enhancement, designed to overcome the performance limitation observed when solving the

C&CG master problems. In our experiments, the hybrid method was first used to solve both 10×4 and 10×5 instances, using all seven RC models presented in section 4.3 as a basis. The obtained results were then used to select the four best-performing hybrid models: the two dichotomy models (Liao-You-Hybrid and Manne-Hybrid) and the Wilson-Hybrid and TS3-Hybrid assignment models. Finally, these hybrid RC models were used to solve the larger 15×5 instances.

The additional results obtained with the new hybrid solution method are depicted in the last four columns of Tab. 7. The TS3-Hybrid model reached the best performance, for solving the highest proportion of instances to optimality (48%), and also for obtaining the lowest average gap (5.00%) of instances not solved to optimality, when compared to the other RC models. Moreover, the performance of Wilson-Hybrid and TS3-Hybrid models are practically equivalent (Wilson-Hybrid achieved 47% of instances solved to optimality and 5.29% of average gap). In general, all four hybrid solution methods achieved drastic performance improvements when compared to the initial solution method results, where the best-performing conventional C&CG algorithm, Manne, had obtained only 14% of instances solved to optimality, along with an average % gap (standard deviation of % gap) of 4.62 (8.42), respectively.

One possible answer to the obtained results may be found in how the hybridization of the solution method works. The partial permutation σ is built iteratively, using the same job fixation order of the combinatorial branch-and-bound method, i.e., fixing one job at a time, from left to right. Given this solution representation, whenever a new job k is fixed in the partial permutation, new cuts have to be added to the existing MILP model of the corresponding node in the B&B tree, in order to make the solutions from combinatorial B&B and MILP compatible. Experimental data shows that, in the case of Wilson-Hybrid and TS3-Hybrid methods, the cuts added to each node, which are based on the job assignment binary variables $Z_{i,j}$, turn out to be stronger than the cuts added to the Liao-You-Hybrid and Manne-Hybrid RC models, which are based on the job precedence variable $D_{i,k}$.

8. Case study on two real instances

In this section, we assess the quality and level of robustness of scheduling solutions for two real problem instances, the first one representing a platform with 9 *oil-wells* and 4 *maintenance tasks* (9×4) and the second one a different platform with size 15×5. The processing time matrices $\overline{p}$ and $\widehat{p}$ were obtained from the available operation history. The following solution methods were used:

- **Det**: deterministic PFSP solution (Wilson, 1989) with $\mathbf{P}$=$\{\overline{p}_{r,i}\}, r \in \mathbb{M}, i \in \mathbb{J}$.
- **RPFS(Γ)**: TS3-Hybrid RPFS solution method, described in Section 6. The Γ parameter is used to control the level of the conservativeness of the robust model. Rounded to the floor value of the fraction of the number of operations $m \times n$, it varies from 5% to 100%, with 5% intervals. The robust counterpart model with $\Gamma = 0\%$ is equivalent to **Det**, while the one with $\Gamma = 100\%$ is the deterministic model that is entirely risk-averse and overestimates all parameters. The other values of Γ model intermediate risk aversions.
- **SimGRASP**: stochastic PFSP simheuristic method from Ferone et al. (2016), properly modified to find the schedule that minimizes the *expected total weighted completion time*. SimGRASP is a modified GRASP metaheuristic that incorporates Monte Carlo Simulation to solve the PFSP with random processing times. Given its stochastic nature, we obtained 25 independent runs for each instance file. Then, for result comparison purposes, for each independent run, we calculated the robust cost $\mathcal{Z}$ at each Γ protection level. Finally, 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 obtained solution σ by calculating the *robust cost* at different protection levels Γ, using the worst-case MILP model defined in Section 5.2. Fig. 2 depicts the *robust cost* $\mathcal{Z}(\sigma, \Gamma)$ of each solution σ under different protection levels Γ. For clarity of the graphs, the robust costs for some protection levels were omitted.

Observe that, as the protection level Γ increases, so does the robust cost, i.e.,

the total weighted completion time (TWCT) of the worst-case scenario defined by the protection level. In other words, higher values of Γ are equivalent to a greater quantity of operations with deviated processing times, which directly impacts the *solution cost* $\mathcal{Z}(\sigma, \Gamma)$. In the case studies from Fig. 2, the extreme cases occur whenever $\Gamma \geq 60\%$, yielding the highest robust costs.

From the viewpoint of the decision-maker at the oil company who needs to hedge against worst-case maintenance 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 Γ values. Regarding the first graph (9×4 instance), note that **RPFS(10)** offers improved protection against worst-case scenarios for $10\% \leq \Gamma \leq 20\%$, while **RPFS(25)** is the best-performing robust solution considering $25\% \leq \Gamma \leq 35\%$. Finally, **RPFS(50)** is indicated for higher protection levels $\Gamma \geq 45\%$. 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, which is represented by the highlighted area in the graph.

In its turn, the larger instance (15×5) presents a distinct behavior in robust cost differences between distinct protection levels. In Fig. 2(b), we can observe that **RPFS(50)** presents the best overall protection against worst-case scenarios, considering $\Gamma \geq 20\%$. Once again, the solutions **Det** and **SimGRASP-Max(25)** present high robust costs. In particular, for $\Gamma = 30\%$, the robust cost provided by **RPFS(35)** is 2% cheaper than **Det** and 3% cheaper than **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 RPFS can provide a pool of robust schedules, depending on the value of Γ. With these options, the decision-makers can choose one of the schedules based on their risk preferences. Also, remark that, if the stochastic heuristic 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 TWCT*.

8.1. Analysis based on Monte-Carlo simulation

As a complementary analysis, we evaluate the expected behavior of obtained problem solutions. The $TWCT$ 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

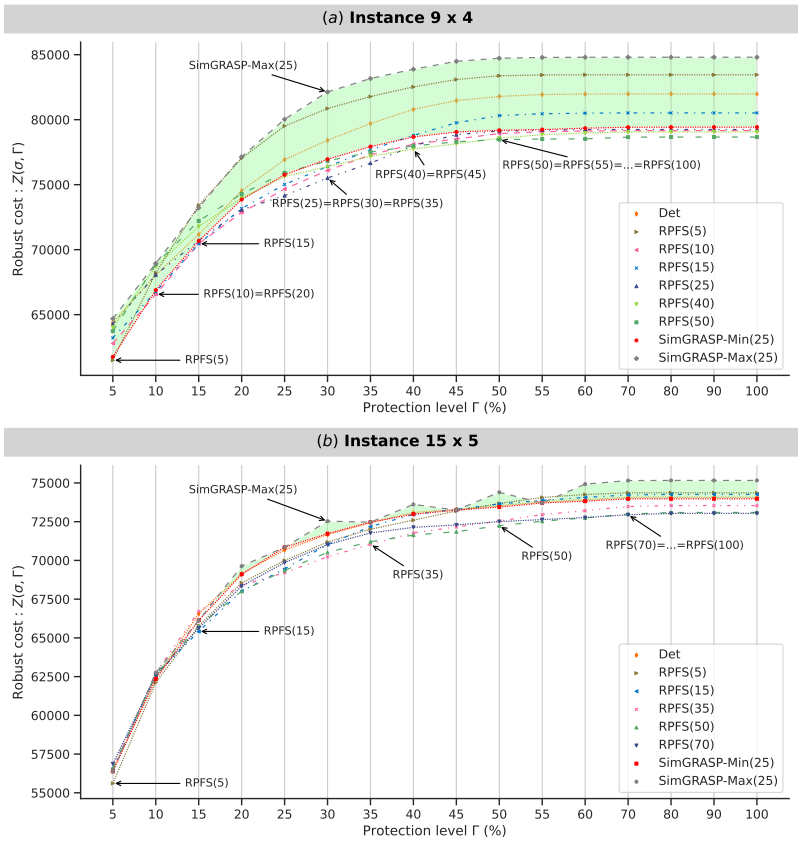

Fig. 2 Robust cost of deterministic, RPFS and SimGRASP solutions versus protection level Γ%. All presented RPFS solutions are optimal.

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 **RPFS(Γ)** solution σ^{Γ}, obtained with a specific protection level Γ, we processed the set of all corresponding $TWCT$ values $\varphi(\sigma^{\Gamma}, \widetilde{P})$ obtained through simulation on $\widetilde{P}$. The same was made for the solutions returned by **Det** and **SimGRASP-Min(25)**.

We first focus on simulation results presented in Fig. 3(a). Regarding the 9×4 instance, we can observe that, in the long run, the TWCT performance of **RPFS(5)**, **RPFS(10)** and **RPFS(25)** are equivalent to **SimGRASP**, a method specialized at optimizing the expected TWCT. On the other hand, as stated in the worst-case analysis of Fig. 2(a), not all schedules are sufficiently immune against worst-case scenarios. For instance, if the decision-maker assumes an intermediate protection level of $\Gamma \leq 35\%$, the two most appropriate schedules, with smallest robust costs, are **RPFS(25)** and **RPFS(10)**.

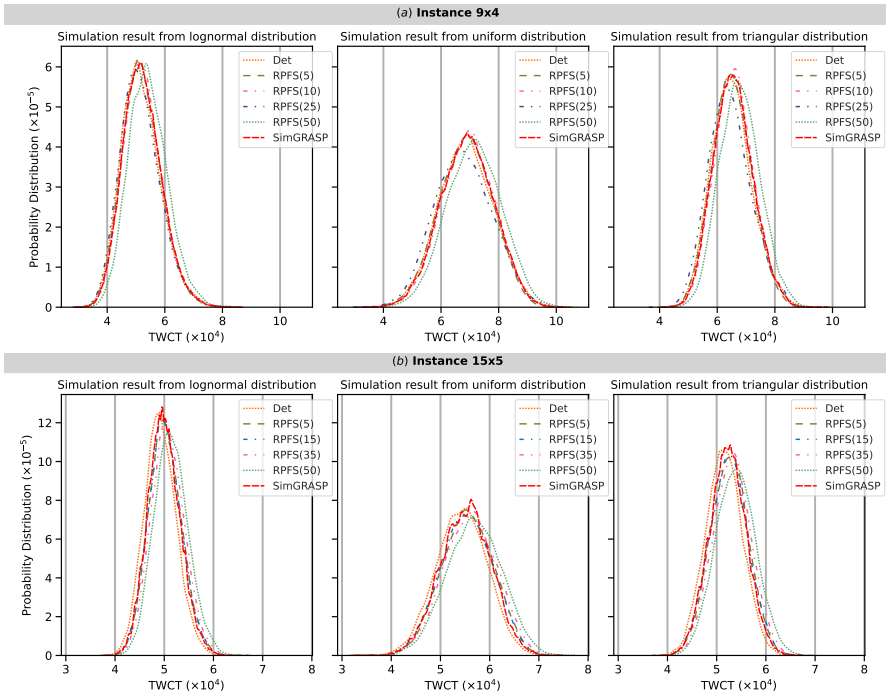

Fig. 3 Probability distributions of $TWCT$ value for RPFS(Γ), Det and SimGRASP solutions, according to simulation results from lognormal, triangular, and uniform distributions for uncertain processing times.

Distribution	Measure	Method: RPFS(5)	RPFS(10, 20)	RPFS(15)	RPFS(20)	RPFS(25,30,35)	RPFS(40, 45)	RPFS(50,...,100)	Det	SimGRASP
lognormal	E($\varphi(\sigma)$)	52,028	52,340	51,647	52,340	51,548	54,375	54,221	51,774	52,254
	SD($\varphi(\sigma)$)	6,619	6,606	6,820	6,606	6,941	6,749	6,772	6,841	6,725
	VaR($\varphi(\sigma)$)	63,528	63,812	63,712	63,812	63,939	66,126	65,959	63,751	63,972
	CVaR($\varphi(\sigma)$)	66,839	67,319	67,344	67,319	67,843	69,442	69,447	67,677	67,532
	Max($\varphi(\sigma)$)	79,857	82,729	83,287	82,729	82,151	83,973	82,780	84,934	86,193
triangular	E($\varphi(\sigma)$)	65,110	65,854	64,481	65,854	64,435	68,049	67,890	65,121	65,599
	SD($\varphi(\sigma)$)	6,739	6,687	7,176	6,687	7,392	6,893	6,946	7,027	6,865
	VaR($\varphi(\sigma)$)	76,531	77,215	76,973	77,215	77,362	79,567	79,620	77,399	77,372
	CVaR($\varphi(\sigma)$)	79,360	80,181	80,060	80,181	80,801	82,532	82,649	80,683	80,383
	Max($\varphi(\sigma)$)	90,901	94,283	93,286	94,283	93,362	96,123	96,399	96,200	97,623
uniform	E($\varphi(\sigma)$)	68,566	69,399	67,838	69,399	67,730	71,538	71,426	68,739	69,143
	SD($\varphi(\sigma)$)	9,153	9,066	9,706	9,066	9,878	9,331	9,414	9,432	9,280
	VaR($\varphi(\sigma)$)	83,425	84,246	83,831	84,246	84,266	86,717	86,839	84,490	84,393
	CVaR($\varphi(\sigma)$)	86,487	87,533	87,398	87,533	88,013	90,096	90,223	88,086	87,699
	Max($\varphi(\sigma)$)	98,076	100,090	100,462	100,090	102,587	102,107	102,600	101,782	102,039

Tab. 8 Simulation summary for instance 9×4 with RPFS(Γ), Det and SimGRASP solutions after $10,000$ simulation runs under lognormal, triangular, and uniform distributions of operation processing times. Minimum and maximum values, for each row, are highlighted. Similar robust solutions for different Γ values are grouped in the same column (e.g., RPFS(10, 20)).

When analyzing the 15×5 instance in Fig. 3(b), the following robust solutions present expected TWCT performance quite similar to **SimGRASP**: **RPFS(5)**, **RPFS(15)** and **RPFS(35)**. Taking the worst-case evaluation into account and considering a protection level $15\% \leq \Gamma \leq 40\%$, these three robust solutions also provide better protection against worst-case costs, when compared to the stochastic solution method.

Finally, Tab. 8 presents some statistics related to the simulation of processing times of the 9×4 instance. In this analysis, whenever the same robust solution has been obtained for more than one Γ parameter value, their (equivalent) statistics were reported in the same column. Given $10,000$ processing time matrices $\widetilde{P}$ obtained after simulation runs, let $\varphi(\sigma)$ be the random cost ($TWCT$) 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, $VaR(\varphi(\sigma))$ and $CVaR(\varphi(\sigma))$ are the value-at-risk and conditional value-at-risk of $\varphi(\sigma)$, respectively, both at 95% confidence level. In other words, $VaR(\varphi(\sigma))$ is equivalent to the 0.95 quantile of $\varphi(\sigma)$, while $CVaR(\varphi(\sigma))$ represents the average of the largest 5% values of $\varphi(\sigma)$. Finally, $Max(\varphi(\sigma))$ is the maximum observed $\varphi(\sigma)$ in the simulation.

Observe that **RPFS(25,30,35)** has the least $E(\varphi(\sigma))$ in all distributions. When analyzing the largest observed TWCT, **RPFS(5)**, has the **lowest** $Max(\varphi(\sigma))$ for lognormal, triangular and uniform distributions. The best solu-

tions for **Det** and **SimGRASP** did not provide minimum values for any measure of the simulated distributions. Indeed, **SimGRASP** presented the worst values for the largest observed TWCT in lognormal and triangular distributions.

Also, by analyzing the smallest maximum TWCT obtained in triangular distribution simulations, the value $Max(\varphi(\sigma))$ observed for scheduling **RPFS(25,30,35)** is 4.3% cheaper than **SimGRASP**, and, at the same time, its expected TWCT is 1.8% less than the stochastic schedule. Following these observations, the decision-maker of the oil company can evaluate the hedge provided by the obtained robust solutions, and choose a specific solution (and associated protection level) that does not cause a significant increase in the expected solution cost, when compared to stochastic and deterministic solutions.

8.2. Evaluating hedge value and price of robustness

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

$$H(\Gamma) = \mathcal{Z}(\bar{\sigma}^*, \Gamma) - \mathcal{Z}(\sigma^*_\Gamma, \Gamma), \tag{84}$$

$$\eta(\Gamma) = \varphi(\sigma^*_\Gamma, \overline{P}) - \varphi(\bar{\sigma}^*, \overline{P}), \tag{85}$$

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

The first measure, $H(\Gamma)$, 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 Γ. A visual interpretation of $H(\Gamma)$ can be made in Fig. 2, by analysing to the robust cost difference between the solutions from **Det** ($\bar{\sigma}^*$) and **RPFS(Γ)** (σ^*_Γ) methods, at each protection level Γ. The second 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). In other words, $H(\Gamma)$ can be seen as the *regret of employing sequence $\bar{\sigma}^*$ in the worst-case scenario*, and $\eta(\Gamma)$ represents the *trade-off between robustness and optimality.*

	Instance Name / Measure			
	9 x 4		15 x 5	
Γ %	η %	H %	η %	H %
0	0.00%	0.00%	0.00%	0.00%
5	2.11%	4.69%	1.81%	1.62%
10	2.97%	2.24%	1.69%	0.82%
15	1.95%	1.02%	2.72%	1.75%
20	2.97%	2.30%	4.86%	1.72%
25	2.28%	3.73%	4.50%	2.33%
30	2.28%	3.85%	5.70%	1.90%
35	2.28%	3.97%	3.47%	1.95%
40	7.47%	3.92%	4.74%	1.98%
45	7.47%	4.26%	4.97%	1.87%
50	6.87%	4.23%	4.58%	1.77%
55	6.87%	4.36%	5.35%	1.63%
60	6.87%	4.39%	4.91%	1.64%
70	6.87%	4.22%	4.86%	1.55%
80	6.87%	4.22%	4.86%	1.43%
90	6.87%	4.22%	4.86%	1.42%
100	6.87%	4.22%	4.86%	1.42%

Tab. 9 Relative robustness price $\eta(\Gamma)\%$ and hedge value $H(\Gamma)\%$ for instances 9×4 and 15×5.

	Distribution / Measure					
	lognormal		triangular		uniform	
Γ %	ω	$\Delta\eta$	ω	$\Delta\eta$	ω	$\Delta\eta$
5	44.6%	0.5%	49.7%	0.0%	52.6%	-0.3%
10	34.5%	1.1%	31.3%	1.1%	36.0%	1.0%
15	53.9%	-0.2%	68.5%	-1.0%	67.4%	-1.3%
20	34.5%	1.1%	31.3%	1.1%	36.0%	1.0%
25	52.2%	-0.4%	62.0%	-1.1%	63.6%	-1.5%
30	52.1%	-0.4%	62.0%	-1.1%	63.6%	-1.5%
35	52.2%	-0.4%	62.0%	-1.1%	63.6%	-1.5%
40	13.5%	5.0%	10.8%	4.5%	17.1%	4.1%
45	13.5%	5.0%	10.9%	4.5%	17.1%	4.1%
50	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
55	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
60	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
70	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
80	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
90	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%
100	12.1%	4.7%	9.0%	4.3%	15.8%	3.9%

Tab. 10 Simulation results for instance 9×4, for different protection levels $\Gamma \in \{5\%, 15\%, \ldots, 100\%\}$. Comparison is based on two measures: (i) $\omega(\Gamma)$ is the % of simulated scenarios (over a total of 10,000) where RPFS(Γ) obtained smaller $TWCT$ cost when compared to Det(P=); (ii) $\Delta\eta(\Gamma) = Avg_{\lambda\in\mathbb{S}}\left[\frac{\varphi(\sigma^*_\Gamma,P^\lambda)-\varphi(\bar{\sigma}^*,P^\lambda)}{\varphi(\bar{\sigma}^*,P^\lambda)}\right]$ is the average relative cost difference between RPFS(Γ) and Det(P=), given all simulated scenarios λ.

Tab. 9 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 the two instances from this case study. Among the schedules obtained when solving RPFS with different Γ levels, the best ones, which maximize hedge value $H(\Gamma)\%$, are **RPFS(5)** for instance 9×4, and **RPFS(25)** for instance 15×5.

Based on the simulation framework presented in Section 8.1, 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 instance 9×4, as shown in Tab. 10. The obtained results show that, for the protection levels Γ used in the study, robust solutions **RPFS(15)**, **RPFS(25)**, **RPFS(30)**, and **RPFS(35)** present two important characteristics: (i) high proportion of cheapest solutions ($\omega(\Gamma) > 50\%$), and (ii) smaller expected costs, i.e., negative relative cost difference $\Delta\eta(\Gamma)$. All in all, RPFS provides a pool of robust schedules decision-makers can choose based on their risk preferences.

9. Concluding remarks

This work provided an exact solution method for the m-machine robust permutation flow shop problem to minimize the *total weighted completion time*. The models developed in this work are based on budgeted uncertainty, a powerful tool for handling robustness and the trade-off between cost and risk, avoiding the over-conservativeness of the conventional robust scheduling approaches.

As main contributions, besides proposing a set of benchmark instances for the problem, we developed seven robust counterpart formulations, coupled with an exact solution method based on Column-and-Constraint Generation (C&CG). Additionally, we implemented a hybrid C&CG method which relies on two strategies to enhance the processing of larger problem instances. First, a branching strategy used in the combinatorial branch-and-bound for scheduling problems. Second, a new lower bound for the robust problem, based on an existing bound used in the deterministic case. Computational experiments suggest that the improved algorithm can handle test problems with $n \leq 15$, reaching the same instance-size limit of the best-performing deterministic solution methods. Despite the longer average processing time required to solve the larger 15×5 instances, good-quality solutions were obtained by the hybrid C&CG based on the TS3 formulation, with 48% of the instances solved to optimality. For the remaining solutions, average gaps of 5% were obtained.

We have also assessed the cost of the solutions returned by the robust model

and compared them to deterministic and stochastic solutions. According to simulations based on three probability distributions, our solution method was capable of protecting against worst-case scenarios, with just a small overhead in the expected solution cost.

Experimental results indicate the feasibility of applying this robust solution method to real-world problem instances, such as the ones from the oil and gas industry, whose current solutions are obtained through methods that disregard either uncertainty or the impact of worst-case scenarios. Based on their risk preferences, decision-makers can then choose an appropriate schedule from a pool of robust solutions, with different levels of exposure to uncertainty.

Future research may attempt to develop efficient heuristics to the problem, which largely depends on the solution of the worst-case problem. Given that the robust problem is NP-hard, this would be particularly important, and our exact approach could therefore play a useful role in evaluating the performance of any heuristic.

Acknowledgments

Research of Mario Levorato 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). Research of Yuri Frota was sponsored by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) grant No. 301254.

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.

Allahverdi, A., Aydilek, H., & Aydilek, A. (2014). Single machine scheduling problem with interval processing times to minimize mean weighted completion time. *Computers and Operations Research*, *51*, 200–207. URL: http://dx.doi.org/10.1016/j.cor.2014.06.003. doi:10.1016/j.cor.2014.06.003.

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.

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

Bougeret, M., Pessoa, A. A., & Poss, M. (2019). Robust scheduling with budgeted uncertainty. *Discrete Applied Mathematics*, *261*, 93–107. URL: https://doi.org/10.1016/j.dam.2018.07.001. doi:10.1016/j.dam.2018.07.001.

Chung, C.-S., Flynn, J., & Kirca, O. (2002). A branch and bound algorithm to minimize the total flow time for m-machine permutation flowshop problems. *International Journal of Production Economics*, *79*, 185–196.

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.

Daniels, R. L., & Kouvelis, P. (1995). Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management science*, *41*, 363–376.

de Farias, I. R., Zhao, H., & Zhao, M. (2010). A family of inequalities valid for the robust single machine scheduling polyhedron. *Computers Operations Research*, *37*, 1610–1614. doi:https://doi.org/10.1016/j.cor.2009.12.001.

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.

Fernandez Perez, M. A., Oliveira, F., & Hamacher, S. (2018). Optimizing workover rig fleet sizing and scheduling using deterministic and stochastic programming models. *Industrial & engineering chemistry research*, *57*, 7544–7554.

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.

Gelders, L. F., & Sambandam, N. (1978). Four simple heuristics for scheduling a flow-shop. *The International Journal of Production Research*, *16*, 221–231.

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

Ho, J. C., & Chang, Y.-L. (1991). A new heuristic for the n-job, m-machine flow-shop problem. *European Journal of Operational Research*, *52*, 194–202.

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.

Kaminsky, P., & Simchi-Levi, D. (1998a). Probabilistic Analysis and Practical Algorithms for the Flow Shop Weighted Completion Time Problem. *Operations Research*, *46*, 872–882. URL: http://pubsonline.informs.org/doi/abs/10.1287/opre.46.6.872. doi:10.1287/opre.46.6.872.

Kaminsky, P., & Simchi-Levi, D. (1998b). Probabilistic analysis and practical algorithms for the flow shop weighted completion time problem. *Operations Research*, *46*, 872–882. doi:10.1287/opre.46.6.872.

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.

Kasperski, A., & Zieliński, P. (2008). A 2-approximation algorithm for interval data minmax regret sequencing problems with the total flow time criterion. *Operations Research Letters*, *36*, 343–344.

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

Lageweg, B., Lenstra, J. K., & Rinnooy Kan, A. (1978). A general bounding scheme for the permutation flow-shop problem. *Operations Research*, *26*, 53–67.

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., Egorova, N. G., & Werner, F. (2018). The optimality box in uncertain data for minimising the sum of the weighted job completion times. *International Journal of Production Research*, *56*, 6336–6362. doi:10.1080/00207543.2017.1398426.

Levorato, M., Figueiredo, R., & Frota, Y. (2022). Exact solutions for the two-machine robust flow shop with budgeted uncertainty. *European Journal of Operational Research*, *300*, 46–57. doi:https://doi.org/10.1016/j.ejor.2021.10.021.

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 and Industrial Engineering*, *74*, 102–110. URL: http://dx.doi.org/10.1016/j.cie.2014.04.013. doi:10.1016/j.cie.2014.04.013.

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

Miyazaki, S., & Nishiyama, N. (1980). Analysis for minimizing weighted mean flow-time in flow-shop scheduling. *Journal of the Operations Research Society of Japan*, *23*, 118–133.

N. Sotskov., Y., G. Egorova., N., Lai., T., & Werner., F. (2011). The stability box in interval data for minimizing the sum of weighted completion times, . (pp. 14–23). doi:10.5220/0003574400140023.

Nagarajan, V., & Sviridenko, M. (2009). Tight bounds for permutation flow shop scheduling. *Mathematics of Operations Research*, *34*, 417–427.

Pereira, J. (2016). The robust (minmax regret) single machine scheduling with interval processing times and total weighted completion time objective. *Computers and Operations Research*, *66*, 141–152. doi:10.1016/j.cor.2015.08.010.

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

Rajendran, C., & Ziegler, H. (1997). An efficient heuristic for scheduling in a flowshop to minimize total weighted flowtime of jobs. *European Journal of Operational Research*, *103*, 129–138.

Ribeiro, G. M., Mauri, G. R., & Lorena, L. A. N. (2011). A simple and robust simulated annealing algorithm for scheduling workover rigs on onshore oil fields. *Computers & Industrial Engineering*, *60*, 519–526.

Rubin, P. A. (2014). *OR in an OB World*. URL: https://orinanobworld.blogspot.com/2014/10/multiple-children-again.html (accessed February 1, 2022).

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.

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.

Sotskov, Y., & Lai, T.-C. (2012). Minimizing total weighted flow time under uncertainty using dominance and a stability box. *Computers Operations Research*, *39*, 1271 – 1289. doi:https://doi.org/10.1016/j.cor.2011.02.001. Special Issue on Scheduling in Manufacturing Systems.

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.

Tadayon, B., & Smith, J. C. (2015). Algorithms and complexity analysis for robust single-machine scheduling problems. *Journal of Scheduling*, *18*, 575–592.

Tseng, F. T., & Stafford, E. F. (2008). New MILP models for the permutation flowshop problem. *Journal of the Operational Research Society*, *59*, 1373–1386. doi:10.1057/palgrave.jors.2602455.

Tseng, F. T., & Stafford Jr, E. F. (2001). Two milp models for the n× m sdst flowshop sequencing problem. *International Journal of Production Research*, *39*, 1777–1809.

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.

Turner, S., & Booth, D. (1986). A new integer programming model for the n job m machine flow shop problem. In S. MJ (Ed.), *Proceedings of the Midwest Decision Science Institute* (p. 229). Lincoln: Nebraska, IL.

Vo, N.-V., & Lenté, C. (2014). From maxplus algebra to general lower bounds for the total weighted completion time in flowshop scheduling problems. *Lecture Notes in Management Science*, *6*, 128–137.

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

Wang, X.-Y., Zhou, Z., Zhang, X., Ji, P., & Wang, J.-B. (2013). Several flow shop scheduling problems with truncated position-based learning effect. *Computers Operations Research*, *40*, 2906 – 2929. doi:https://doi.org/10.1016/j.cor.2013.07.001.

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

Yang, J., & Yu, G. (2002). On the robust single machine scheduling problem. *Journal of Combinatorial Optimization*, *6*, 17–33.

Yang, S. H., & Wang, J. B. (2011). Minimizing total weighted completion time in a two-machine flow shop scheduling under simple linear deterioration. *Applied Mathematics and Computation*, *217*, 4819–4826. doi:10.1016/j.amc.2010.11.037.

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.

Zarei, F., Muradov, K., Davies, D. et al. (2014). Optimal well work-over scheduling: application of intelligent well control optimisation technology to conventional wells. In *SPE Intelligent Energy Conference & Exhibition*. Society of Petroleum Engineers.

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.