Article

Not peer-reviewed version

Multi-Objective Just-in-Time Permutation Flow Shop: Tools for Analysis of Different Conflict Scenarios

Nícolas Samuel Assis * , Socorro Rangel , Helio Yochihiro Fuchigami *

Posted Date: 20 May 2026

doi: 10.20944/preprints202605.1363.v1

Keywords: scheduling; permutation flow shop; mixed-integer linear programming; multi-objective optimization

Article

Multi-Objective Just-in-Time Permutation Flow Shop: Tools for Analysis of Different Conflict Scenarios

Nícolas Samuel Assis [1,*], Socorro Rangel [2] and Helio Yochihiro Fuchigami [3,*]

[1] SESI-SP, Ribeirão Preto-SP, Brazil
[2] Mathematics Department, São Paulo State University (Unesp), São José do Rio Preto-SP, Brazil
[3] Production Engineering Department, Federal University of São Carlos (UFSCar), São Carlos-SP, Brazil
* Correspondence: nicolas.s.assis@gmail.com (N.S.A.); helio@dep.ufscar.br (H.Y.F.)

Abstract

Permutation flow shop scheduling is an important production planning problem handled in different contexts. Just-in-time measures have been significant in the optimization of real problems and one is specifically addressed here: the total earliness and tardiness of jobs. The most used approach in the literature to mathematically express this measure is to sum them up using unit weights thus obtainning a mono-objective function. In this paper it is shown that this is a simplification of a problem that is inherently multi-objective, highlighting how a more comprehensive approach can better support decision-making. A bi-objective mathematical optimization model and tools capable of analyzing the mono-objective solution within the multi-objective perspective are proposed. A computational study to analyze the benefits and difficulties of the solution using the bi-objective approach is presented. The results show that for large scale instances in which the tardiness factor is small, the conflict between the objectives of minimizing the total earliness and minimizing the total tardiness of jobs increases significantly. Therefore, the multi-objective approach has a greater potential to support decision-makers. Furthermore, we show that the choice of the solution method must be carefully considered, since the Pareto frontier associated with most instances has many non-supported points.

Keywords: scheduling; permutation flow shop; mixed-integer linear programming; multi-objective optimization

1. Introduction

This paper addresses the problem of just-in-time permutation flow shop scheduling (JIT-PFS). Considering there are n jobs and m machines, the solution involves finding a sequence to process the n jobs on all m machines in a unidirectional flow. Each job j has a processing time p_{jk} on each machine k without interruption and an individual due date d_j. The order of execution of the jobs must be the same on all machines, i.e., the solution is a permutation schedule. When a job is completed before its due date, there is early production. Similarly, if a job is completed after its due date, there is tardy production, as illustrated in Figure 1. So the measures of total earliness (E) and total tardiness (T) can be used to compute the just-in-time objective, which implies completing a job exactly at its due date. Given the conflicting nature of the total earliness and total tardiness measures, a bi-objective approach will be taken. Using the standard three-field notation of [1], this problem is denoted as $Fm|prmu|E, T$. More complex variants of this problem, such as hybrid flow shop and distributed permutation flow shop, can be found in the review papers of [2] and [3], respectively.

Figure 1 presents the example of a flow shop problem with $n = 5$ jobs and $m = 3$ machines. The due dates for each job j (d_j) are illustrated with vertical dashed lines. Job J_2 is early, job J_1 completes its execution on time (just-in-time), and the other jobs conclude late. This schedule totals $E = 13$ units of earliness and $T = 199$ units of tardiness. The greater complexity of this problem compared to others lies in the fact that the solution involves both defining the sequence of jobs and determining the timeline of

 doi:10.20944/preprints202605.1363.v1

the schedule, i.e., deciding when each operation will be executed. For example, for the same sequence of jobs, there may be many possible solutions in terms of the start and completion times of each operation.

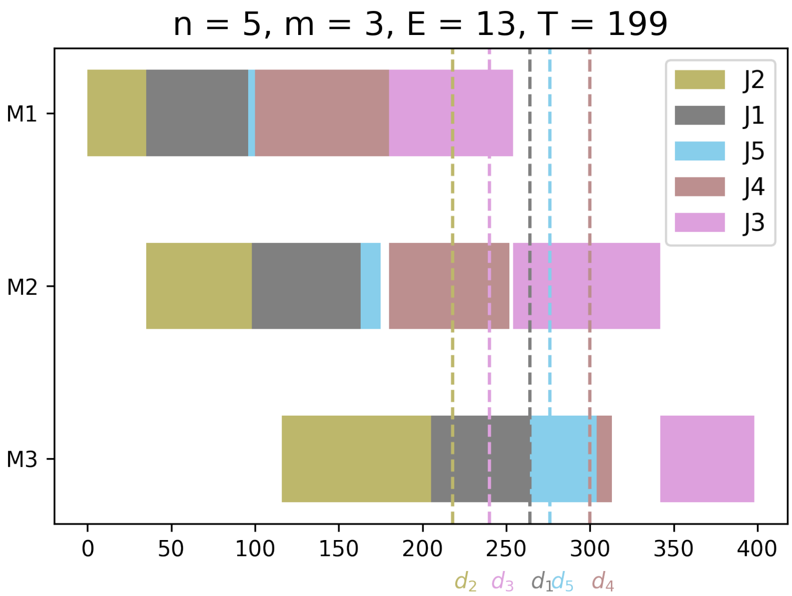


Figure 1. Gantt chart associated to a solution of an instance of the JIT-PFS

In the literature, when the JIT criteria of the optimization is computed using the measures of E and T, it is generally treated mathematically using a mono-objective function obtained by adding up these two measures. Section 2 shows that this is a simplification of a multi-objective solution method called the weighted sum method [4]. There are many approaches to represent this problem using a mathematical optimization model. Ronconi and Birgin [5] compared the solution process of instances of position-based and sequence-based optimization models using commercial solvers. Schaller and Valente [6] transformed the m-machine problem into a single machine and solved it by several heuristics and a branch-and-bound method specially developed for the problem. In general, the JIT-PFS is solved employing heuristics and metaheuristics. Some examples are: Birgin et al. [7] that proposes a constructive heuristic with a beam search; Guevara-Guevara et al. [8], Prata and Fuchigami [9], and Tavana et al. [10] that employ genetic algorithms; and Karacan et al. [11] that uses simulated annealing. Even when the just-in-time objective is proposed more generically, using weights different from one, the values are fixed and just one mono-objective problem is solved. In this strategy, metaheuristics continue to be the most common methodology, including genetic algorithms [12], simulated annealing [13], and others [14]. Table 1 shows relevant articles from the literature to the present time. The first column lists the names of the authors, the second presents the problem solved. The third column informs whether the model is position-based (P) or sequence-based (S). The fourth column indicates the method used: heuristic (H), metaheuristic (MH), mixed-integer programming (MIP), problem transformation (Tr), or branch-and-bound (B&B). Generally, the mathematical optimization models proposed in the literature are only described, with little record of their performance on computational implementation.

Table 1. A literature review of PFS with a JIT objective

Authors	Problem	MILP	Method
Akhshabi et al. [13]	$Fm\|d_j fuzzy\|wET$	P	MH
Ronconi and Birgin [5]	$Fm\|block\|ET$	P, S	MIP
Mohammadi [12]	$Fm\|\|wET + C_{max} + N_j$	-	MH
Birgin et al. [7]	$Fm\|prmu, d_j = d\|ET$	S	H
Schaller and Valente [6]	$Fm\|nwt\|ET$	P	Tr, H, B&B
Guevara-Guevara et al. [8]	$Fm\|nwt, s_{ijk}\|ET$	S	MH
Fuchigami and Prata [14]	$Fm\|prmu\|wET + wd$	P, S	MH
Karacan et al. [11]	$Fm\|nwt\|ET$	S	MH
Prata and Fuchigami [9]	$Fm\|block\|ET$	P	MH
Tavana et al. [10]	$Fm\|block\|C_{max}, ET$	P	MH
The current paper	$Fm\|prmu\|E, T$	P	B&B

Research addressing multi-objective problems uses performance indicators (Zitzler et al. [15], Audet et al. [16]) to evaluate and compare Pareto frontiers obtained by different solution methods. After selecting the best-performing Pareto frontier approximation, it is up to the decision-maker to choose which solution will be implemented. Various strategies and resources can be used to support decisions (Greco et al. [17]). One approach not yet found in the literature is embedding a solution point of the mono-objective problem within the Pareto frontier of the multi-objective problem thus providing an extra decision support tool. This approach allows for a direct relationship between mono-objective and multi-objective problems with low computational cost, providing the decision-maker with more information to support their choice.

The goal here is to highlight how the multi-objective optimization approach can be used to support decisions associated to the problem JIT-PFS. For this, a computational study is executed to analyze the conflict between total earliness and total tardiness measures, as well as analyzing the mono-objective problem from a bi-objective perspective. The main contributions are:

1. the proposal of a bi-objective model for the JIT-PFS problem with the goals of minimizing total earliness and minimizing total tardiness;
2. the development of algebraic tools capable of qualitatively analyzing the mono-objective problem within a bi-objective perspective as an additional support for the decision-makers;
3. the application of the bi-objective branch-and-bound method proposed by Parragh and Tricoire [18] to solve the JIT-PFS problem;
4. a computational study of mono-objective and bi-objective approaches evaluating the conflict of the JIT-PFS problem for different scenarios, the potential of the multi-objective approach as an additional support for the decision-makers and the need to use appropriate multi-objective methods for large instances.

To the best of our knowledge, this is the first time that the permutation flow shop scheduling problem is addressed with earliness-tardiness optimization criteria from a multi-objective perspective.

The remainder of the text is organized as follows: Section 2 presents a bi-objective mathematical optimization model for the problem and multi-objective solution methods; in Section 3, the proposed tools for analyzing the problem are described; Section 4 contains a discussion of the computational study; and some final considerations are presented in Section 5.

2. A bi-Objective Mathematical Optimization Model and Solution Strategies

To formulate the permutation flow shop scheduling problem, a position-based model adapted from Wilson [19]'s model is proposed. We employ the assignment variables x_{jh}, which are equal to 1 if job j is processed in position h of the sequence and 0 otherwise. Additionally, the variables C_{hk}, E_h, and T_h are used to calculate the objectives of the problem. All parameters and variables of the mathematical model are presented as follows.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1363.v1

Indexes

- j: index for jobs, with $j = 1 \dots n$;
- k: index for machines, with $k = 1 \dots m$;
- h: index for the position in the sequence where the job is processed, with $h = 1 \dots n$.

Parameters

- p_{jk}: processing time of job j on machine k;
- d_j: due date of job j.

Decision variables

- $x_{jh} = 1$ if job j is to be processed at position h and 0 otherwise;
- C_{hk}: completion time of the job in the position h on machine k;
- E_h: earliness of the job in the position h of the sequence;
- T_h: tardiness of the job in the position h of the sequence.

The proposed bi-objective model for the permutation flow shop scheduling problem is presented in (1)-(9).

$$Min \ \left(\sum_{h=1}^{n} E_h, \sum_{h=1}^{n} T_h\right) \tag{1}$$

$$s.t. \ \sum_{j=1}^{n} x_{jh} = 1, \qquad h = 1, ..., n \tag{2}$$

$$\sum_{h=1}^{n} x_{jh} = 1, \qquad j = 1, ..., n \tag{3}$$

$$C_{11} \geq \sum_{j=1}^{n} p_{j1} x_{j1} \tag{4}$$

$$C_{h,k+1} \geq C_{hk} + \sum_{j=1}^{n} p_{j,k+1} x_{jh}, \qquad h = 1, ..., n, k = 1, ..., m-1 \tag{5}$$

$$C_{h+1,k} \geq C_{hk} + \sum_{j=1}^{n} p_{jk} x_{jh}, \qquad h = 1, ..., n-1, k = 1, ..., m \tag{6}$$

$$\sum_{j=1}^{n} (d_j x_{jh}) - C_{hm} \leq E_h, \qquad h = 1, ..., n \tag{7}$$

$$C_{hm} - \sum_{j=1}^{n} (d_j x_{jh}) \leq T_h, \qquad h = 1, ..., n \tag{8}$$

$$x_{jh} \in \{0, 1\}, C_{hk} \geq 0, E_h \geq 0, T_h \geq 0, \qquad j = 1, ..., n, h = 1, ..., n, k = 1, ..., m. \tag{9}$$

In expression (1), the objectives of total earliness and total tardiness, respectively, are minimized. Constraints (2) and (3) are assignment constraints that ensure only one job is processed in each position, and each job occupies only one position, respectively. Constraint (4) ensures that the completion time of the job in the first position on the first machine has to be equal to or greater than its processing time. Constraints (5) and (6) are necessary to compute the completion times of each job on subsequent machines and the completion times of subsequent jobs on each machine, respectively. Constraints (7) and (8) together with the nonnegativity of E_h and T_h linearize the objectives, respectively, which are defined by the function maximum [20]. In (9), the domains of the variables are defined.

Unlike mono-objective optimization, when considering two or more objective functions (multi-objective optimization), the images of the solution vectors are now multidimensional points and, therefore, other ways of evaluating the optimality of a feasible solution must be established. In this paper, we consider a bi-objective function $f(x) = (f_1(x), f_2(x)), x \in R^n$. Thus, using the notation in Definition 1, it is possible to establish the criterion of optimality according to Pareto [21], as in

 doi:10.20944/preprints202605.1363.v1

Definitions 2 and 3, considering that $\mathcal{X}$ is the set of feasible solutions and $\mathcal{Y}$ is the set of images of the feasible solutions.

Definition 1. *Given* $y^1, y^2 \in \mathbb{R}^2$*, we say that* $y^1 \leq y^2$ *if* $y^1_k \leq y^2_k$ *for* $k = 1, 2$ *and* $y^1 \neq y^2$*.*

Definition 2. *A solution* $x^* \in \mathcal{X}$ *is efficient if there does not exist* $x' \in \mathcal{X}$ *such that* $f(x') \leq f(x^*)$*. The set of all efficient solutions is represented by* $\mathcal{X}_E$*.*

Definition 3. *We say that a point* $f(x^*) \in \mathcal{Y}$ *is non-dominated if* x^* *is an efficient solution. The set of non-dominated points* $\mathcal{Y}_N$ *is called the Pareto frontier.*

By defining the optimality criterion according to Pareto (Definition 3), we can verify if a problem is truly bi-objective through the ideal point (Definition 4). If the ideal point is the image of some feasible solution, then there is no conflict between the objectives and one of the objective functions can be discarded.

Definition 4. *The point* $y^I = (y^I_1, y^I_2)$ *given by*

$$y^I_1 = Min\{E \mid (2) - (9)\} \; e \; y^I_2 = Min\{T \mid (2) - (9)\}$$

is called the ideal point.

The Definition 5 helps in characterizing the solution methods for finding the Pareto frontier of a multi-objective problem.

Definition 5. *Let* $x \in \mathcal{X}_E$*, then we say that:*

1. *x is efficient and supported if its image is on the convex part of the Pareto frontier, otherwise it is efficient and non-supported.*
2. *The sets of the images of the supported and non-supported efficient solutions are denoted by* Y_{sN} *and* Y_{nN} *respectively,* $Y_N = Y_{sN} \cup Y_{nN}$*.*

One of the most popular strategies for finding the Pareto frontier is using scalarization methods, which consist of transforming a vector objective function into a scalar objective function. The scalarization method used in this paper is the weighted sum [4], which assigns weights to each objective and adds them up, thus obtaining a mono-objective function. A scalarization of the model (1)-(9) using the weighted sum strategy is presented in (10)-(11).

$$Min \;\; w_1 \sum_{h=1}^{n} E_h + w_2 \sum_{h=1}^{n} T_h \tag{10}$$

$$s.t. \;\; (2) - (9) \tag{11}$$

The weighted sum method is given by solving the problem (10)-(11) iteratively by changing the weights. At first two points, named lexicographic points, are defined using the weighted pairs $(1, 0)$ and $(0, 1)$ in two lexicographic optimizations. In the subsequent iterations, the weights are determined by $w_1 = y^i_2 - y^j_2$ and $w_2 = y^j_1 - y^i_1$ with $y^i_1 < y^j_1$ the pair (i, j) being a combination of the points found two by two. Graphically, choosing these weights makes the level curves of the iteration have exactly the slope of the line determined by the chosen points.

When the problem is solved using only one iteration of the weighted sum method (the mono-objective approach most employed in the literature), it is not possible to individually control the values of E and T obtained, except that the sum of both is the smallest (Theorem 1). This can lead to undesirable solutions for the production system represented by the mathematical model.

 doi:10.20944/preprints202605.1363.v1

Theorem 1. *If* $(x^*_{jh}, C^*_{hk}, E^*_h, T^*_h)$ *is the optimal solution of the model (10)-(11) when* $w_1 = w_2 = 1$ *then there doesn't exist* $(x'_{jh}, C'_{hk}, E'_h, T'_h) \in \mathcal{Y}_{nN}$ *such that* $E' + T' < E^* + T^*$

Proof of Theorem 1. Consider $E = \sum_{h=1}^{n} E_h$ and $T = \sum_{h=1}^{n} T_h$. Suppose that there exists $(x'_{jh}, C'_{hk}, E'_h, T'_h) \in \mathcal{Y}_{nN}$ such that $E' + T' < E^* + T^*$. So we have already got a contradiction, because, in this case, $(x^*_{jh}, C^*_{hk}, E^*_h, T^*_h)$ would not be the optimal solution of (10)-(11). □

Figure 2 shows the Pareto frontier of an instance of the model (1)-(9). Notice that the value of the mono-objective solution, obtained through the model (10)-(11) when $w_1 = 1$ and $w_2 = 1$, belongs to the frontier (Theorem 2). The other points represent the value of other solutions for this problem, which are equally efficient. The weighted sum method only generates some of the points of the Pareto frontier, the ones associated with the supported solutions (Definition 5). To determine the whole Pareto frontier (both supported and non-supported solutions), other multi-objective solution methods are needed, e.g. De Santis et al. [22] and Parragh and Tricoire [18]. Figure 3 highlights the supported and non-supported points of the Pareto frontier.

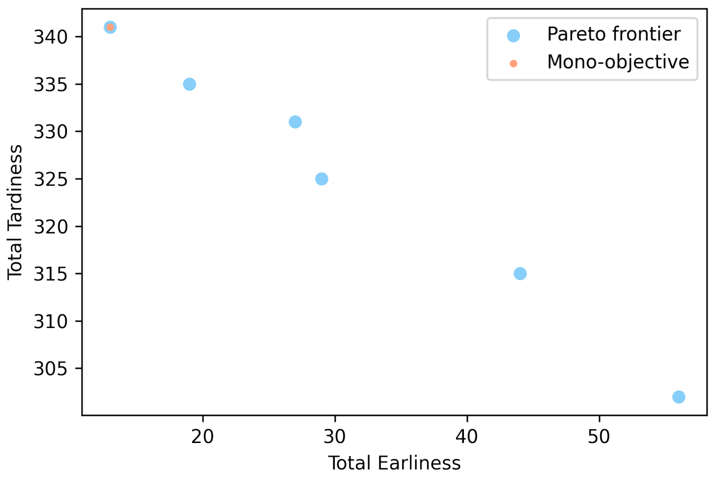


Figure 2. Pareto Frontier with a JIT objective

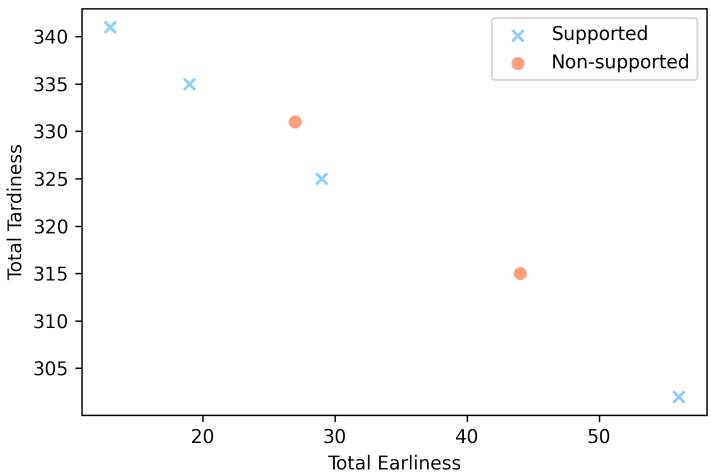


Figure 3. Supported and non-supported points of the Pareto Frontier

Theorem 2. *Let* $\mathcal{Y} \subset \mathbb{R}^p$*, the set of images of the feasible solutions for the problem (1)-(9). If* $(x^*_{jh}, C^*_{hk}, E^*_h, T^*_h)$ *is the optimal solution of (10)-(11) with* $w_1 = w_2 = 1$*, then* $(E^*, T^*) \in \mathcal{Y}_N$.

 doi:10.20944/preprints202605.1363.v1

Proof of Theorem 2. Consider $\sum_{h=1}^{n} E_h = E$ and $\sum_{h=1}^{n} T_h = T$. Suppose that $(E^*, T^*) \notin \mathcal{Y}_N$. Then, there exists $(E', T') \in \mathcal{Y}$ such that $(E', T') \leq (E^*, T^*)$. With this, $E' \leq E^*$, $T' \leq T^*$ and for some of them this inequality is strict. Consequently, we have that $E' + T' < E^* + T^*$ contradicting the fact that $(x^*_{jh}, C^*_{hk}, E^*_h, T^*_h)$ is the optimal solution. □

The second method used in this research is the bi-objective branch-and-bound method (BIOBAB) proposed by Parragh and Tricoire [18]. Like in the mono-objective branch-and-bound method, each node of the branch-and-bound tree is solved by relaxing the domain of the variables, and upon completing the node resolution, if there are one or more infeasible variables for the original problem, one is chosen to be branched. The main difference lies in the resolution of each node, which is done using the weighted sum method, and instead of lower bound and upper bound values, it is necessary to employ the concept of lower and upper bound sets. In general, the solutions found by the weighted sum method form a lower bounding set. When a solution is feasible, it becomes part of the upper bounding set. Although the branch-and-bound method uses the weighted sum to solve each node, it is capable of determining the entire Pareto frontier due to the branching of variables. The major problem with this methodology is the computational effort required to solve each node and the size of the branching tree, which can be large even for small problems.

To improve the performance of the method, Parragh and Tricoire [18] implemented some features such as objective space branching. This improvement uses ideas similar to scalarization methods to prune the objective space. Additionally, it can result in more than one variable being branched at the same node. This is exemplified in Figure 4. The red points represent the lower bounding set, and the green points represent the upper bounding set. When dominated points are filtered by the upper bounding set, two disjoint regions in red are obtained. These are limited search regions where some variables will be branched, not necessarily the same for both regions.

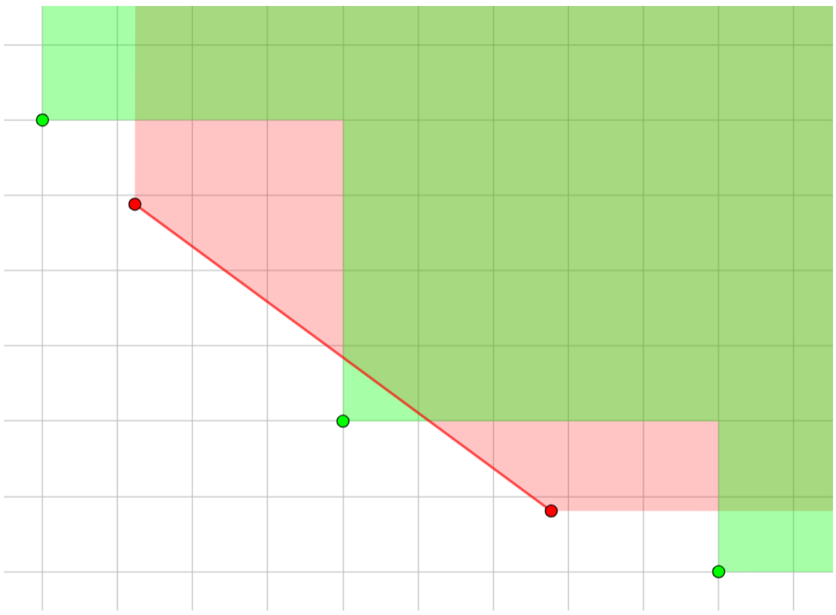


Figure 4. Branching of the objective space

Other improvements have been designed for problems where the variables and their coefficients in the objective function are integers. These improvements exploit the two-dimensional nature of the objective space to speed up node resolution, reduce the region of the objective space and to prune nodes. Since the objectives of the model (1)-(9) consist of continuous variables only the objective space branching and not these latter improvements are used in the computational study.

3. Mathematical Tools for Analysis of Points in the Pareto Frontier

Since, to the best of our knowledge, this research is the first to conduct a study that qualitatively analyzes the mono-objective problem within a bi-objective perspective, mathematical tools needed to be developed. To be clear, these tools are not performance indicators, as they do not verify the characteristics of the Pareto frontier. Rather, they embed the mono-objective optimal value (or the image of any fesible solution) of the problem into the two-dimensional space of the bi-objective problem and check its location on the Pareto frontier. In this way, it is possible to obtain additional insights to support the decison-makers. These tools can be applied to any bi-objective problem that is solved as a mono-objective problem and can be easily extended to a k-objective problem with $k > 2$ using the Euclidean distance for dimension k.

Three of the tools were developed to locate the image of the mono-objective solution on the Pareto frontier. Their purpose is to inform, algebraically, how close this point is to a lexicographic one. To avoid distortions in the calculations, the Pareto frontier is normalized. The first tool (D_I) is based on distance and uses the ideal point (Definition 4) as a reference. The second tool (D_μ) is also based on distance but uses the midpoint between the lexicographic points of the Pareto frontier as the reference point. The third tool (D_Δ) is given by the absolute difference between the objectives. For all these three tools, the higher the returned value, the closer the point is to a lexicographic point, with one being the maximum value and zero the minimum value. We classify a point as balanced when the objectives are not prioritized (between 0 and 0.1), moderate (between 0.1 and 0.5), and unbalanced when one of the objectives is prioritized (between 0.5 and 1).

We will explain how to calculate D_μ, the other two (D_I and D_Δ) are obtained similarly. The way to calculate D_μ is based on the calculation of the epsilon indicator in Zitzler et al. [15] and will use smooth and bump functions. First, the distance from each point to μ is calculated. Then, the maximum (Max) and minimum (Min) of these distances are computed. Finally, a smooth function (12) is used to create the bump function in (13) and from this, D_μ is obtained as shown in (14). D_μ is a smooth bump function that maps the distance between a non-dominated point (or any point in the criteria space) and a reference point to a real number in the range zero and one.

$$f(x) = \begin{cases} 0\,, & if\ x \leq 0 \\ e^{-\frac{1}{x}}\,, & if\ x > 0 \end{cases} \tag{12}$$

$$g(x) = \frac{f(x)}{f(x) + f(1-x)} \tag{13}$$

$$D_\mu(x) = g\left(\frac{x - Min}{Max - Min}\right) = \begin{cases} 0\,, & if\ x \leq Min \\ e^{-\frac{1}{x}}\,, & if\ Min < x < Max \\ 1\,, & if\ x \geq Max \end{cases} \tag{14}$$

Graphically, the D_μ measure can be represented as shown in Figure 5. The green circular arc represents the shortest distances to the reference point μ, illustrated by the black "x". The green arrow indicates the region of balanced points (3 points on this Pareto frontier). The orange circular arc indicates the beginning of the region of moderate points (no points on this frontier), and its arrow shows the extent of this region. Finally, the red circular arc indicates the beginning of the region of unbalanced points (3 points on this Pareto frontier).

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1363.v1

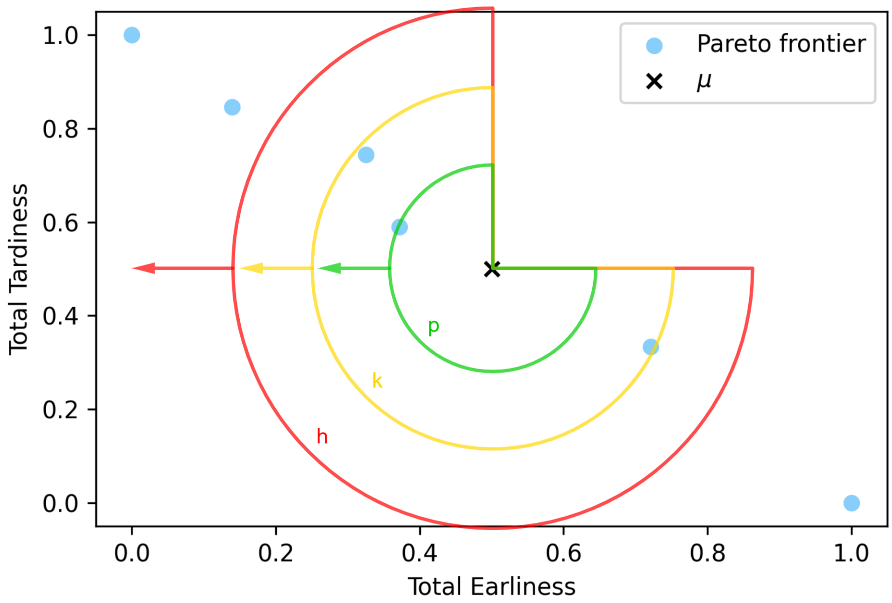


Figure 5. Regions of balance based on the distance having μ as reference

In addition to analyzing the imbalance of the mono-objective solution (D_μ, D_I and D_Δ), we analyze the maximum (SP_{max}), average (SP_μ), and minimum (SP_{min}) percentage increase of the weighted sum of the Pareto frontier points relative to the mono-objective solution. This measure will inform the trade-off between the JIT objective and adapting the solution to an industrial scenario. Figure 6 shows that, for the Pareto frontier shown in Figure 5, the maximum percentage increase does not reach 1.5%, the average increase is 0.61% and there are other solutions that do not show an increase in the value of the objective function. The use of this tool, as shown in Figure 6, is another resource to support the decision-makers, as it allows them to visualize the variation in the objective function while choosing a point on the Pareto frontier or at any point desired.

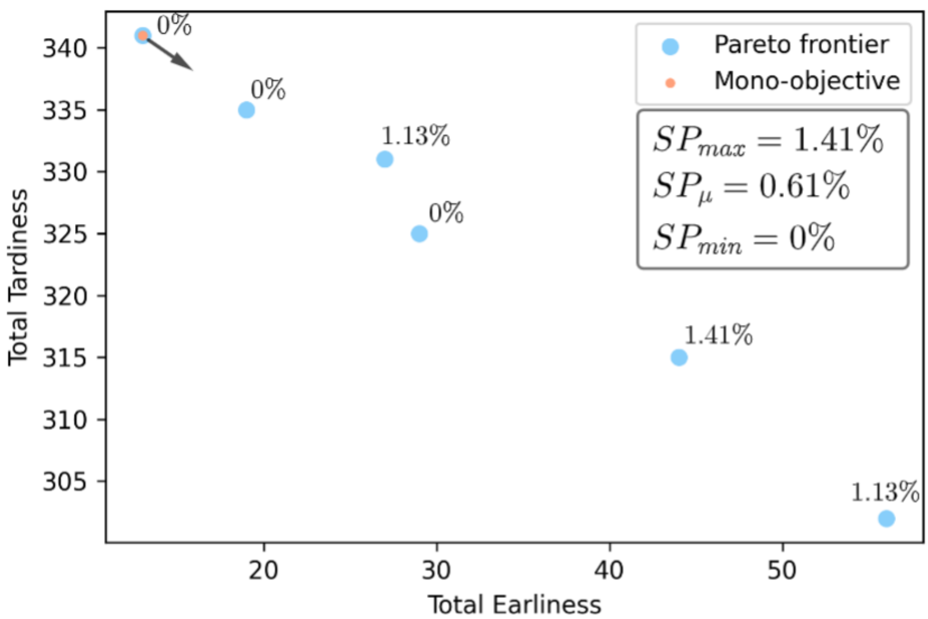


Figure 6. Percentage increase in the weighted sum in relation to the mono-objective solution

The fourth tool proposed is to identify non-supported non-dominated points present on the Pareto frontier. The pseudocode used is shown in Algorithm 1. This algorithm verifies, for each pair of points (A, B) on the Pareto frontier, whether a given point P is above the segment $\overline{AB}$, that is, if it is

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1363.v1

non-supported. Identifying non-supported non-dominated points aims to demonstrate the importance of using appropriate multi-objective methods. If the non-supported points make up the majority of the Pareto frontier and the weighted sum method is used to find the Pareto frontier, the multi-objective approach might not effectively provide decision-makers with a range of choices. This could also lead to the incorrect conclusion that there is little conflict between the objectives. Therefore, it is important to use a method that computes all the non-dominated points, supported and non-supported ones.

Algorithm 1: Pseudocode to identify non-dominated points

Data: $\mathcal{Y}_N$
Result: $\mathcal{Y}_{nN}, \mathcal{Y}_{sN}$
for $(A, B) \in C_2^{|\mathcal{Y}_N|}$ **do**
  for P *entre* A *e* B **do**
    if $y_P > \frac{y_B - y_A}{x_B - x_A}(x_P - x_A) + y_A$ **then**
      P is non-supported
    else
      continue

4. The Computational Study and Discussion of Results

The 160 instances used in this computational study were presented by Ronconi and Birgin [5] and are publicly available. With n as the number of jobs and m as the number of machines, there were four (n, m) pairs: $(5, 3)$, $(10, 3)$, $(10, 5)$, and $(10, 10)$. For the due date generation, four scenarios were also considered, varying in terms of the tardiness factor (TF) and dispersion range of the due dates (DR), that is (TF, DR): $(0.2, 0.6)$, $(0.2, 1.2)$, $(0.4, 0.2)$, and $(0.4, 1.2)$. This resulted in 16 different classes and, for each class, 10 instances were generated, totaling 160 instances. The due dates were generated uniformly in the range $[P(1 - TF - DR/2), P(1 - TF + DR/2)]$, with P being a lower bound for the makespan objective. The processing times were generated uniformly in the range $[1, 99]$. The name of each instance class contains the TF, DR, n and m, for example "0.2_0.6_5_3", as appears in Tables 2, 3 and 4.

The study was conducted on a machine with a 12th generation Core i7 processor at 2.1 GHz, 16 GB of RAM, and a 64-bit Windows 11 Home edition operating system. There were no simultaneous executions and the machine did not perform any other tasks that would limit CPU usage during the study. The methods were implemented using Python 3.8 Foundation [23]. One of the methods used in the study was the BIOBAB proposed by Parragh and Tricoire [18], freely available at *https://github.com/JKU-PLM/BIOBAB*. The parameters used to solve the instances of the proposed model were search depth $(-depth)$ and the non-use of those parameters aimed at improving the solution of problems with integer variables and coefficients ($-nid$, $-nst$, and $-nll$). Details of these parameters can be seen in Parragh and Tricoire [18]. Another method was implementing the proposed model by weighting the objectives with unit weights, reproducing the most commonly used mono-objective methodology in the literature. In this study, we embed the optimal value of the mono-objective problem in the criteria space of the bi-objective problem, but any feasible solution value could be used. Gurobi Optimization, LLC [24] was used by both methods to solve the mathematical optimization subproblems involved. The default configuration was used, except for the search type that was changed to depth-first. The execution time limit was set to 3600 seconds. Out of the 160 instances executed, only 4 were not solved by the BIOBAB due to some inconsistency in the code within the *lowerboundset* module. The analysis of the results will be based on the 156 instances optimally solved within the time limit.

At first, we study the computational effort required to solve the problem in its mono-objective version by the Gurobi and in its bi-objective version by using the BIOBAB. For both strategies the analysis involves the number of nodes (nNod), and execution time (Time) in seconds, and for the BIOBAB we also consider the number of solutions (nSol). These results are shown in Table 2.

doi:10.20944/preprints202605.1363.v1

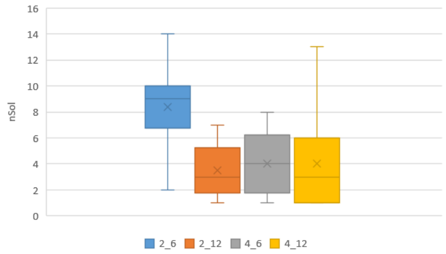
Table 2. Performance of mono-objective and multi-objective approaches (average of 10 instances)

	BIOBAB			GRB_{mono}	
Instance classes	nSol	nNod	Time(s)	nNod	Time(s)
0.2_0.6_5_3	8.40	97.40	0.20	1.00	0.01
0.2_0.6_10_3	40.00	56838.60	172.26	37.80	0,06
0.2_0.6_10_7[1]	16.55	95328.11	723.92	1291.80	0.21
0.2_0.6_10_10[2]	9.75	137406.00	1495.84	5627.70	0,64
0.2_1.2_5_3	3.50	94.40	0.15	1.00	0.01
0.2_1.2_10_3	15.50	132058.20	477.28	426.00	0,10
0.2_1.2_10_7	7.30	115600.40	861.74	644.10	0.21
0.2_1.2_10_10	5.60	171497.60	1418.16	3344.70	0,43
0.4_0.6_5_3	4.00	64.80	0.12	0.01	0.01
0.4_0.6_10_3	20.00	32858.80	109.03	281.20	0.08
0.4_0.6_10_7	7.30	109635.20	731.62	1498.80	0.21
0.4_0.6_10_10	5.30	163233.30	600.42	3390.10	0.39
0.4_1.2_5_3	4.00	73.60	0.12	1.00	0.01
0.4_1.2_10_3[1]	9.44	87927.44	302.16	336.70	0.10
0.4_1.2_10_7	5.60	116169.00	649.27	1463.00	0.24
0.4_1.2_10_10	3.70	122221.60	693.14	3174.10	0.37

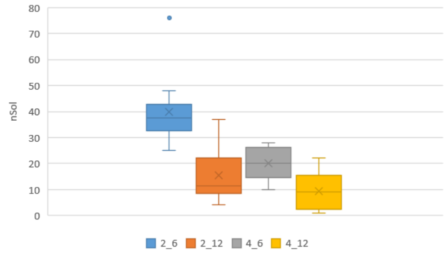
Multi-objective solution methods are used to find different solutions exhibiting the trade-off between the different objectives. Naturally, these methods require greater computational effort since they are effectively solving different problems simultaneously. Consequently, longer execution times were expected, especially when using an exact method like the bi-objective branch-and-bound method. The main interest in this part of the study is the analysis of the conflict between total earliness and total tardiness, which can be measured through the number of solutions on the Pareto frontier. The influence of the parameters DR and TF on the conflict between E and T was analyzed. As can be seen in the results shown in Table 2, as DR increases, the dispersion range of due dates also increases. Additionally, the parameters are generated following a uniform distribution. These two facts contribute to due dates being further apart, reducing the conflict between E and T when assigning the jobs. The TF parameter also contributes to reducing the conflict because, as it increases, the dispersion range of the due dates shifts to the left, meaning all jobs must be completed earlier. This reduces the likelihood of having any job with earliness. If there is no job earliness, the total earliness objective can be ignored and the problem ceases to be bi-objective. In 21 out of the 156 instances (13.46%), the Pareto frontier consists of only one point, indicating the problem is not truly bi-objective. So, when the TF parameter is high, the multi-objective approach is not recommended; it is better to minimize only the total tardiness.

The associated Box plots (Figure 7) confirm the behavior pattern of the number of solutions. As DR and TF increase, there is generally a decrease in the number of solutions. This decrease is more significant for the TF parameter, precisely because it may prevent jobs from being completed early.

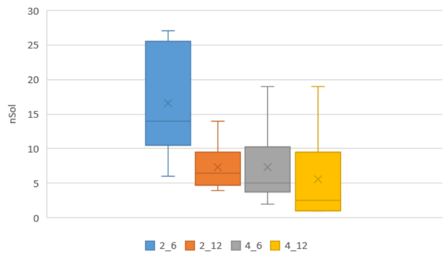
[1] Average of 9 instances
[2] Average of 8 instances



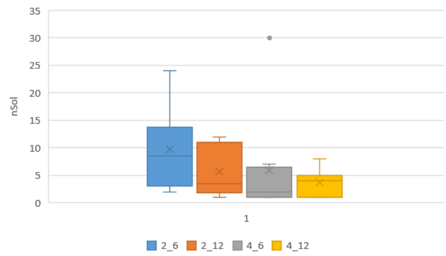
(a) 5 jobs and 3 machines



(b) 10 jobs and 3 machines



(c) 10 jobs and 7 machines



(d) 10 jobs and 10 machines

Figure 7. Number of points on the Pareto frontier by the size of the problem

Table 3 presents the results of the geometric tools and the increase in cost (average of each structure). The first column provides the characteristics of the instances in the format $TF\_DR\_n\_m$. Columns 2 (D_I), 3 (D_μ) and 4 (D_Δ) display the values of the tools for locating the point on the Pareto frontier. Columns 5 (SP_{min}), 6 (SP_μ), and 7 (SP_{max}) show the increases in the value of the objective function when selecting another point on the Pareto frontier. Column 8 ($f_i = f^*$) gives the percentage of points on the Pareto frontier that do not cause any increase in the objective function.

Table 3. Results of geometric analysis tools and the increase in cost (average of 10 instances)

Instance classes	D_I	D_μ	D_Δ	$SP_{min}(\%)$	$SP_\mu(\%)$	$SP_{max}(\%)$	$f_i = f^*(\%)$
0.2_0.6_5_3	0.21	0.43	0.45	7.62	22.75	41.74	40.00
0.2_0.6_10_3	0.00	0.03	0.03	0.53	34.40	107.00	60.00
0.2_0.6_10_7[1]	0.14	0.38	0.28	4.02	18.85	44.50	50.00
0.2_0.6_10_10[2]	0.38	0.56	0.49	4.21	10.57	19.98	60.00
0.2_1.2_5_3	0.70	0.70	0.70	3.20	11.33	18.54	40.00
0.2_1.2_10_3	0.21	0.24	0.27	0.00	24.86	71.88	100.00
0.2_1.2_10_7	0.41	0.43	0.49	0.52	10.29	24.07	60.00
0.2_1.2_10_10	0.70	0.86	0.86	0.88	2.94	5.43	30.00
0.4_0.6_5_3	0.60	0.64	0.63	6.14	13.07	20.94	30.00
0.4_0.6_10_3	0.15	0.38	0.38	0.17	9.41	24.93	80.00
0.4_0.6_10_7	0.50	0.67	0.73	0.85	4.00	9.42	30.00
0.4_0.6_10_10	0.70	0.82	0.83	1.76	3.04	4.23	30.00
0.4_1.2_5_3	0.67	0.73	0.72	2.59	6.60	13.10	20.00
0.4_1.2_10_3[1]	0.38	0.49	0.51	1.02	9.77	18.95	60.00
0.4_1.2_10_7	0.65	0.75	0.71	0.15	1.16	2.95	40.00
0.4_1.2_10_10	0.74	0.84	0.87	0.24	2.01	4.02	40.00

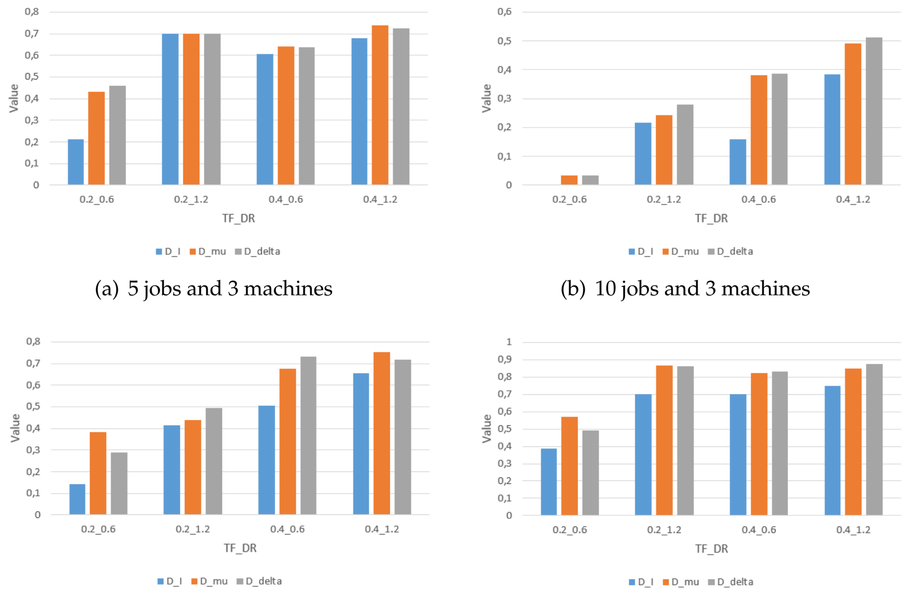
In 14 structures of instances, D_μ, D_I and D_Δ determined the same classification of the mono-objective problem point, in 0.2_0.6_10_10 D_μ classifies the point as unbalanced while D_I and D_Δ classify it as moderate and in 0.4_1.2_10_3 D_Δ classifies the point as unbalanced while D_μ and D_I

[1] Average of 9 instances
[2] Average of 8 instances

classifies it as moderate. Of these 14 structures in which there is agreement between the tools, only one of them was classified as balanced, while 5 (35.71%) of them were classified as moderate and 8 (57.14%) of them were classified as unbalanced. This shows that in most instances the mono-objective point is closer to a lexicographic point, giving more priority to early or tardy production, an important information to a decision-maker since this characteristic might be undesired.

Analysis of the graphs in Figure 8 shows that changing the values of TF and DR tends to position the mono-objective point closer to a lexicographic point on the Pareto frontier. This movement is a direct consequence of the decrease in conflict between the objectives. With fewer points on the Pareto frontier, the likelihood of the mono-objective point being a lexicographic point or being closer to a lexicographic point increases.

When TF and DR are fixed and we analyze the variations in the instance sizes, it is not possible to establish a direct relationship with the values returned by the tools, as there is a drop with 10 jobs and 3 machines and then an increase considering 10 jobs and 7 and 10 machines.



(a) 5 jobs and 3 machines (b) 10 jobs and 3 machines

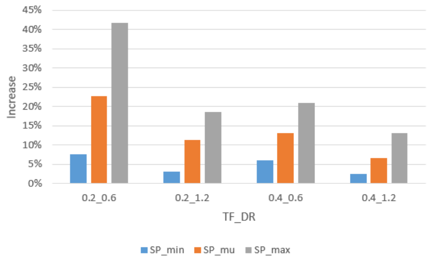
(c) 10 jobs and 7 machines (d) 10 jobs and 10 machines

Figure 8. Values of location tools by the size of the problem

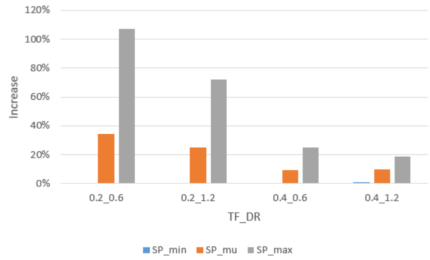
Observing the percentage increases in Figure 9, we notice an inverse movement compared to Figure 8. As the values of TF and DR change, the increases, especially the maximum, decrease. Despite this downward trend in increases, the values are significantly high, with an average of 2.12% for the minimum increase, considering all instances, and a standard deviation of 2.25%. Looking at the standard deviation of this data, we see that the minimum increase can be 0. This is confirmed by analyzing column 8 ($f_i = f^*$) of Table 3, which shows points that do not cause an increase in the objective value, averaging 48.12% of the instances.

The possibility that, in almost 50% of the instances, there are points on the Pareto frontier that do not increase the mono-objective function value and offer a different job sequencing further reinforces the significant potential of the multi-objective approach as support decision for this problem.

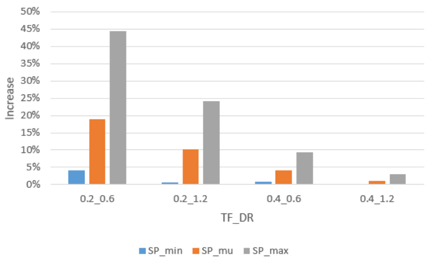
The results shown in Table 4 consider the number of supported and non-supported solutions for 121 (77.56%) out of the 156 instances, the ones that have at least 3 points on the Pareto frontier (average of each structure). The first column provides the characteristics of the instances, columns 2 (Supported) and 3 (Non-supported) shows the number of supported and non-supported points on the Pareto frontier, respectively, column 4 (Non-Sup/Total) shows the percentage of non-supported points on the Pareto frontier and in Column 5 ($|\mathcal{Y}_N| \geq 3$), the number of instances that have at least 3 points on the Pareto frontier are recorded. It is noticeable that in the just-in-time flow shop scheduling problem, non-supported points make up a significant portion of the Pareto frontier. The maximum average reaches 66.71% of non-supported points on the Pareto frontier, with a mean of 43.90% and a standard deviation of 13.84%. This presence becomes more pronounced as the instance size increases. This can be better seen in Figure 10 considering the parameters $(TF, DR) = \{(0.2, 0.6), (0.2, 1.2)\}$, where the conflict between the objectives is higher. When comparing instances with 5 jobs and 3 machines to those with 10 jobs and 10 machines, the increase in non-supported points on the Pareto frontier rises significantly, reaching maximums of 70% and 77.78%, respectively.



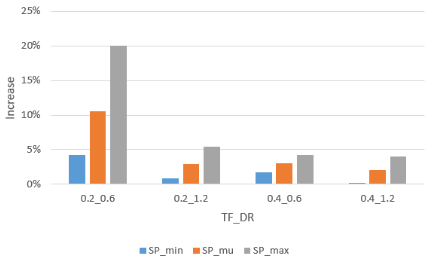
(a) 5 jobs and 3 machines



(b) 10 jobs and 3 machines



(c) 10 jobs and 7 machines



(d) 10 jobs and 10 machines

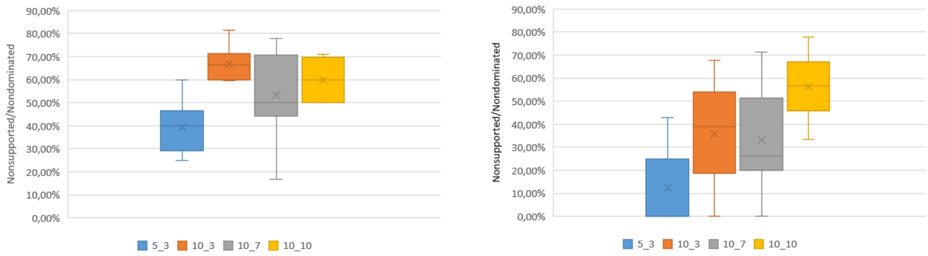
Figure 9. Minimum, medium and maximum increases by the size of the problem

With these results, it is clear that using a multi-objective method that only determines supported points on the Pareto frontier can mask the advantages of the multi-objective approach for the JIT-PFS problem. This issue is exacerbated as the problem size increases, as a large part of the frontier will not be presented, reducing the options of the decision-maker.

Table 4. Results of the number of supported and non-supported points on the Pareto frontier (average of $|\mathcal{Y}_N|$ instances)

Instance classes	Supported	Non-supported	Non-Sup/Total	$\lvert\mathcal{Y}_N\rvert \geq 3$
0.2_0.6_5_3	5.55	3.55	39.13%	9
0.2_0.6_10_3	12.60	27.40	66.71%	10
0.2_0.6_10_7	6.55	10.00	53.38%	9
0.2_0.6_10_10	4.50	7.83	59.98%	6
0.2_1.2_5_3	3.71	0.71	12.55%	7
0.2_1.2_10_3	8.20	7.30	35.69%	10
0.2_1.2_10_7	4.60	2.70	33.18%	10
0.2_1.2_10_10	3.33	5.00	56.27%	6
0.4_0.6_5_3	3.50	2.16	35.48%	6
0.4_0.6_10_3	7.40	12.60	58.56%	10
0.4_0.6_10_7	3.77	4.11	45.39%	9
0.4_0.6_10_10	3.80	9.60	57.14%	5
0.4_1.2_5_3	4.40	2.40	28.77%	5
0.4_1.2_10_3	5.00	6.71	48.14%	7
0.4_1.2_10_7	5.00	5.00	39.06%	5
0.4_1.2_10_10	3.14	1.71	32.98%	7

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 May 2026 doi:10.20944/preprints202605.1363.v1



(a) $TF = 0.2$ and $DR = 0.6$

(b) $TF = 0.2$ and $DR = 1.2$

Figure 10. Percentage of non-supported points on the Pareto frontier by scenario

5. Final Considerations

In this research, we studied the just-in-time permutation flow shop scheduling problem considering total earliness and total tardiness performance measures. The literature review showed that this problem has been approached in a simplified manner, not considering its multi-objective nature. A bi-objective model was proposed for the problem and algebraic tools were developed to analyze the mono-objective approach used in the literature from a bi-objective perspective. The computational study allowed the analysis of the conflict between the total earliness and total tardiness objectives considering different scenarios for TF and DR. This analysis demonstrated that this conflict decreases significantly when the TF parameter was increased. This information can be used to approach the just-in-time objective conversely, using only total tardiness, since jobs tend not to be early. In this case, when TF is high, the multi-objective approach is not recommended and using the total tardiness measure is advantageous.

On the other hand, when the TF parameter is small, the conflict between the objectives increases significantly as the size of the instances also increases. This highlights that the multi-objective approach has its greatest potential benefit for larger instances, which are often the focus of the literature. Moreover, for larger instances of the JIT-PFS problem, the choice of the multi-objective method must be carefully considered, as its Pareto frontier is mostly composed of non-supported points. Therefore, not just any multi-objective method will bring real benefit to the decision-maker, only those capable of determining the entire Pareto frontier. Furthermore, the use of exact multi-objective methods capable of determining the entire Pareto frontier is still rarely found in the literature especially when it comes to the JIT-PFS problem. This reveals a significant gap in the literature that can be explored in future research.

Author Contributions: Conceptualization, N.S.A, S.R. and H.Y.F.; methodology, N.S.A, S.R. and H.Y.F.; software, N.S.A.; validation, N.S.A, S.R. and H.Y.F.; formal analysis, N.S.A, S.R. and H.Y.F.; investigation, N.S.A, S.R. and H.Y.F.; resources, S.R and H.Y.F; data curation, N.S.A; writing—original draft preparation, N.S.A; writing—review and editing, N.S.A, S.R. and H.Y.F.; visualization, N.S.A.; supervision, S.R. and H.Y.F.; project administration, S.R. and H.Y.F.; funding acquisition, N.S.A, S.R. and H.Y.F. All authors have read and agreed to the published version of the manuscript.

Funding: The authors are grateful for the financial support of National Council for Scientific and Technological Development (CAPES - grant 2016/01860-1) and Sao Paulo Research Foundation (FAPESP - grants 2022/05803-3, 2013/07375-0).

Data Availability Statement: The instances used in the computational study (Section 4) is a subset of the data presented in Ronconi and Birgin [5] and are publicly available in https://github.com/NicolasSamuelAssis/JIT-PFS.

Conflicts of Interest: The authors declare no conflicts of interest.

References

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

2. Neufeld, J.S.; Schulz, S.; Buscher, U. A systematic review of multi-objective hybrid flow shop scheduling. *European Journal of Operational Research* **2023**, *309*, 1–23.
3. Perez-Gonzalez, P.; Framinan, J.M. A review and classification on distributed permutation flowshop scheduling problems. *European Journal of Operational Research* **2024**, *312*, 1–21.
4. Ehrgott, M.; Gandibleux, X. Bound sets for biobjective combinatorial optimization problems. *Computers & Operations Research* **2007**, *34*, 2674–2694.
5. Ronconi, D.P.; Birgin, E.G. Mixed-integer programming models for flowshop scheduling problems minimizing the total earliness and tardiness. *Just-in-Time systems* **2012**, pp. 91–105.
6. Schaller, J.; Valente, J.M. Minimizing total earliness and tardiness in a nowait flow shop. *International Journal of Production Economics* **2020**, *224*, 107542.
7. Birgin, E.G.; Ferreira, J.E.; Ronconi, D.P. A filtered beam search method for the m-machine permutation flowshop scheduling problem minimizing the earliness and tardiness penalties and the waiting time of the jobs. *Computers & Operations Research* **2020**, *114*, 104824.
8. Guevara-Guevara, A.; Gómez-Fuentes, V.; Posos-Rodríguez, L.; Remolina-Gómez, N.; González-Neira, E. Earliness/tardiness minimization in a no-wait flow shop with sequence-dependent setup times. *Journal of Project Management* **2022**, *7*, 177–190.
9. Prata, B.d.A.; Fuchigami, H.Y. A genetic iterated greedy algorithm for the blocking flowshop to minimize total earliness and tardiness. *Journal of Intelligent Manufacturing* **2023**, pp. 1–14.
10. Tavana, M.; Hajipour, V.; Alaghebandha, M.; Di Caprio, D. A bi-objective hybrid vibration damping optimization model for synchronous flow shop scheduling problems. *Machine Learning with Applications* **2023**, *11*, 100445.
11. Karacan, I.; Senvar, O.; Bulkan, S. A novel parallel simulated annealing methodology to solve the no-wait flow shop scheduling problem with earliness and tardiness objectives. *Processes* **2023**, *11*, 454.
12. Mohammadi, G. Multi-objective flow shop production scheduling via robust genetic algorithms optimization technique. *International Journal of Service Science, Management and Engineering* **2015**, *2*, 1–8.
13. Akhshabi, M.; Akhshabi, M.; Khalatbari, J. Bi-criteria flow shop scheduling with fuzzy simulated annealing algorithm. *African Journal of Business Management* **2012**, *6*, 7478.
14. Fuchigami, H.Y.; Prata, B.d.A. Coronavirus Optimization Algorithms for Minimizing Earliness, Tardiness, and Anticipation of Due Dates in Permutation Flow Shop Scheduling. *Arabian Journal for Science and Engineering* **2023**, pp. 1–33.
15. Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C.M.; Da Fonseca, V.G. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation* **2003**, *7*, 117–132.
16. Audet, C.; Bigeon, J.; Cartier, D.; Le Digabel, S.; Salomon, L. Performance indicators in multiobjective optimization. *European journal of operational research* **2021**, *292*, 397–422.
17. Greco, S.; Figueira, J.; Ehrgott, M. *Multiple criteria decision analysis*; Vol. 37, Springer, 2016.
18. Parragh, S.N.; Tricoire, F. Branch-and-bound for bi-objective integer programming. *INFORMS Journal on Computing* **2019**, *31*, 805–822.
19. Wilson, J.M. Alternative Formulations of a Flow-Shop Scheduling Problem. *The Journal of the Operational Research Society* **1989**, *40*, 395–399.
20. Pinedo, M.L. *Scheduling*; Vol. 29, Springer, 2012.
21. Pareto, V. *Manual d'économie politique (in French)*, 1896.
22. De Santis, M.; Grani, G.; Palagi, L. Branching with hyperplanes in the criterion space: The frontier partitioner algorithm for biobjective integer programming. *European Journal of Operational Research* **2020**, *283*, 57–69.
23. Foundation, P.S. Python 3.8.12 documentation. Disponível em: https://www.python.org, 2021. Acessado: 10 de julho de 2023.
24. Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023.