A combinatorial analysis of the permutation and non-permutation flow shop scheduling problems ⋆

Daniel A. Rossit[a,d,*], Óscar C. Vásquez[e], Fernando Tohmé[b,d], Mariano Frutos[a,c], Martín D. Safe[d,f,]

[a] *Engineering Department, Universidad Nacional del Sur, Argentina*
[b] *Economics Department, Universidad Nacional del Sur, Argentina*
[c] *IIESS UNS CONICET, Argentina*
[d] *INMABB UNS CONICET, Argentina*
[e] *Industrial Engineering Department, Universidad de Santiago de Chile, Chile*
[f] *Mathematics Department, Universidad Nacional del Sur, Argentina*

Abstract

In this paper we introduce a novel approach to the combinatorial analysis of flow shop scheduling problems for the case of two jobs, assuming that processing times are unknown. The goal is to determine the dominance properties between permutation flow shop (PFS) and non-permutation flow shop (NPFS) schedules. In order to address this issue we develop a graph-theoretical approach to describe the sets of operations that define the makespan of feasible PFS and NPFS schedules (critical paths). The cardinality of these sets is related to the number of switching machines at which the sequence of the previous operations of the two jobs becomes reversed. This, in turn, allows us to uncover structural and dominance properties between the PFS and NPFS versions of the scheduling problem. We also study the case in which the ratio between the shortest and longest processing times, denoted ρ, is the only information known about those processing times. A combinatorial argument based on ρ leads to the identification of the NPFS schedules that are dominated by PFS ones, restricting the space of feasible solutions to the NPFS problem. We also extend our analysis to the comparison of NPFS schedules (with different number of switching machines). Again, based on the value of ρ, we are able to identify NPFS schedules dominated by other NPFS schedules.

Keywords: non-permutation flow shop scheduling problem, makespan, critical path, unknown processing times, structural and dominance properties

1. Introduction

Scheduling problems have been widely studied, thanks to their great importance in industrial environments [22]. Our goal in this paper is to study such a problem, motivated by real-life

⋆ A preliminary and shorter version of this article has been published in the *Proc. of the Joint EURO/ALIO International Conference 2018 on Applied Combinatorial Optimization* (EURO/ALIO 2018).

* Corresponding author

Email addresses: daniel.rossit@uns.edu.ar (Daniel A. Rossit), oscar.vasquez@usach.cl (Óscar C. Vásquez), ftohme@criba.edu.ar (Fernando Tohmé), mfrutos@uns.edu.ar (Mariano Frutos), msafe@uns.edu.ar (Martín D. Safe)

industrial applications, namely the design of flow shop environments in the case when exact input data about the processing time of jobs operations is lacking. More specifically, we focus on the flow shop scheduling problem with unknown processing times in a two-jobs setting. Formally, the flow shop scheduling problem considered in this paper involves a set $J = \{J_1, J_2\}$ of jobs and a set $M = \{M_1, \ldots, M_m\}$ of m machines. Each job J_j consists of a set $O_j = \{O_{j,1}, \ldots, O_{j,m}\}$ of m operations, where the ith operation is to be carried out by machine M_i. Moreover, operation $O_{j,i+1}$ may start only when $O_{j,i}$ is completed. Operations are performed one at a time on each machine M_i and preemption is not allowed. Furthermore, each operation $O_{j,i}$ has a processing time $p_{j,i} \in \mathbb{N}$ unknown to the scheduler. A solution to the problem consists of a schedule σ of jobs on machines such that the best value of an objective function is obtained. In our case this function is the *makespan*, expressed by $F(\sigma)$, which the optimal schedule minimizes.

The literature identifies two ways in which to address the above scheduling problem. One is by solving its permutation version, which only considers the same sequence of jobs on all the machines. The other way of solving the scheduling problem is by considering its non-permutation version, in which the ordering of jobs can change from a machine to another. The latter version is more general and its space of solutions contains those of the former, i.e. the permutation flow shop (PFS) scheduling problem is a restricted version of the non-permutation flow shop (NPFS) scheduling problem. This means that the optimal PFS schedule may not be optimal for the NPFS version. Furthermore, the optimal NPFS schedule cannot be strictly improved by one for the PFS scheduling problem. For n jobs the downside of this advantage of NPFS schedules is that, instead of considering the $n!$ possible schedules of the PFS scheduling problem, we have to assess the $n!^m$ candidate schedules of NPFS scheduling problem. This feature has spurred the interest in the latter problem [33], [16], [38], [3], [4], [26], [25].

In this paper we extend some preliminary results presented in [27], solving both the PFS and the NPFS versions of the two-jobs and m-machines scheduling problem with unknown values of the processing times. We accomplish this by applying a novel modeling technique that allows to describe the structure of solutions independently of the processing times. We also analyze the particular case in which the only known data about the processing times is the ratio between the maximum and minimum of the processing times.

This paper is organized as follows. Section 2 is a literature review on the particular PFS and NPFS scheduling problems in which the goal is the minimization of the makespan, highlighting the most relevant results in the field. Section 3 presents the definition of scheduling problems, introducing relevant concepts allowing us to characterize their PFS and NPFS versions. Section 4 describes the structural and dominance properties of the PFS and NPFS scheduling problems, which constitute the theoretical results obtained in this research. Finally, the conclusions and prospects for future work are discussed in Section 6.

2. Literature Review

The research on flow shop scheduling was pioneered by Johnson [14], who stated what is currently known as the *classical* flow shop scheduling problem. The literature usually studies versions of this problem in which the processing times or at least their probabilistic distribution are known. The classical versions of the PFS and NPFS scheduling problems of minimization of makespan in two-jobs settings admit optimal solutions in polynomial time. The shortest paths can be found, subject to vertical line segment barriers, examining their graphical representation [2, 1, 32]. In addition, it has been known since the 1960s that there exists an optimal schedule for the NPFS scheduling problem with the same job ordering in the first two and the last machines

[8], as can be shown in a simple exchange argument [11]. In practice, this implies that there exists an optimal solution σ^* for the NPFS scheduling problem such that the job ordering does not change on machines M_1, M_2 and M_m. Thus, when the number of machines m satisfies $m \leq 3$, the NPFS scheduling problem admits a PFS schedule as an optimal solution. For instances with a number of machines $m > 3$, the same schedule may not be optimal simultaneously for both the PFS and NPFS scheduling problems, as shown in Figure 1.

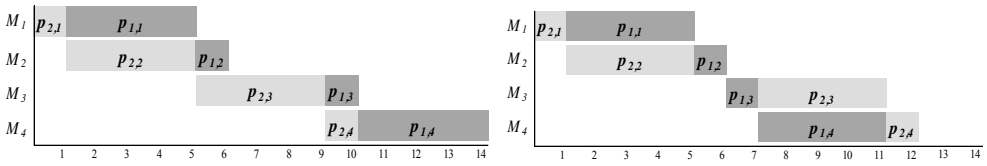

Figure 1: Gantt chart of the optimal schedules of the PFS problem (left) and the NPFS problem (right) for an instance $\mathcal{I}$ with $n = 2$ jobs and $m = 4$ machines. For this setting, the optimal makespan values of PFS and NPFS schedules are 14 and 12, respectively.

The impact of the permutation condition on the flow shop scheduling problem with the goal of minimizing the makespan has been little studied. Potts et al. [23] analyzed the flow shop scheduling problem with zero processing times, and showed that for a family of instances the best permutation schedule is worse than $\sqrt{m}/2$ times the best NPFS schedule. This is shown using the results of Erdős and Szakeres on sequences of distinct integers [9]. Choi et al. [7], studied the processing times defined[1] by $p_{j,i} = p_j/s_i$, where s_i is the processing speed of the machine i, and proved that for $m \geq 4$, PFS schedules are optimal if there exists a slowest machine, i.e., $s_k < s_1 = \ldots = s_{k-1} = s_{k+1} = \ldots = s_m$ for some machine M_k. Nagarajan and Svirdenko [17] study the ratio between permutation and non-permutation scheduling versions in instances with n jobs and m machines, obtaining a tight value of $2\sqrt{\min\{m, n\}}$ based on the results drawn from the analysis of a randomized algorithm. Rebaine [24] studies the structural properties of NPFS scheduling problem in instances with $m = 2$ machines and n jobs with time delays, finding that the ratio is bounded by 2 for arbitrary processing times, and by $\max\{2 - (3/(n + 2)), m\}$ for processing times restricted to be 1. More recently, Panwalkar and Koulamas [19] addressed an ordered flow shop scheduling problem, considering that processing times meet certain conditions such that for any generic machine, denoted k, the processing times can be ordered in a decreasing order, i.e., $p_{i,k} > p_{l,k}$, among other considerations. Furthermore, these authors proved that a PFS schedule is optimal if the processing times are increasing on the successive machines, for every fixed job j (see Lemma 1 therein). A good and updated source of relevant results for the NPFS scheduling problem can be found in [25].

All these works study the PFS and NPFS problems seeking conditions ensuring that PFS schedules are optimal (and therefore, dominant). A common feature of those papers is the imposition of strong conditions on each $p_{j,i}$. This conditioning allows to study ordering rules ensuring optimal solutions. Although there are real cases in which this type of conditioning applies, there are other real cases that can not be modeled in this way. In order to tackle this shortcoming, we approach the problem by lifting all the conditions on the processing times. A recent contribution to the literature already lifts some constraints on the processing times in a 3-machines flow

[1]Here we have chosen to subindexing p by first the job and next by the machine, contrary to the convention in the literature. Since we only assume two jobs, the first subindex can be only either 1 or 2, simplifying the interpretation of $p_{j,i}$.

shop problem in which the displacements of a single robot, taking jobs over from a machine to another, must be scheduled [28]. The authors assume that processing times are independent of the jobs and depend only on the machine in which they are carried out, i.e. $p_{j,i} = p_i$ for each machine i.

Processing times have also been studied in problems with significant learning effects [37], [21],[34]. Other cases in which the processing times are relevant arise when they depend on the discretionary allocation of resources by a scheduler seeking to accelerate the production process [31],[29],[30],[36], [6].

The case analyzed in this paper generalizes all the aforementioned problems, in which the conditions on the processing times are more or less restrictive. We model the processing times as being independent of the parametrization used for each specific problem. This approach allows us to work with graph-theoretical descriptions as well as with the structure of the critical paths of PFS and NPFS schedules. We can conjecture that under those structures it should be possible to find NPFS schedules that are dominated by PFS schedules regardless the values of the processing times. We show, through a combinatorial analysis, that this is the actual case. Even if in this contribution we study only the case of two jobs and m-machines, we provide intuitions on how to generate NPFS schedules for $n > 2$ in which the last job on machine i becomes the first on the $(i + 1)$th. Nevertheless, finding clear analytical results for $n > 2$ jobs goes beyond the scope of this article.

3. Statement of the problem

We focus on flow shop problems with $n = 2$ jobs and m machines, with unknown processing times. We consider schedules corresponding to their PFS and NPFS versions. Given the particular feature of our approach we need to develop new tools to make our solutions independent of the exact values of the processing times. One of these tools is a novel graphical representation of Flow Shop problems. While usually scheduling problems are represented by means of disjunctive graphs [22], the particular structure of flow shop problems (being themselves instances of job shop problems) makes them prone to an alternative representation, recently introduced in [20]. Our own representation is closely related to the latter rerpresentation, based on Gantt charts, but allowing generic processing times and being easier to interpret than disjunctive graphs.

Definition 1. ***Graph representation***. *Consider a graph $G^\sigma(V, A, P)$, where V is the set of vertices, A is a class of solid and dashed edges and P the class of labels of the edges. $G^\sigma(V, A, P)$ is the* graph representation *of a feasible schedule σ. The set V of vertices is the set of (instantaneous) idle states before or after the execution of jobs. Each solid edge in A represents an operation O_{ji} of a job j on a machine i joining the corresponding idle states before and after the execution of O_{ji}. Each dashed edge in A represents the precedence constraint between operations O_{ji} and $O_{j,i+1}$ joining the idle state after the execution of O_{ji} and the idle state before the execution of $O_{j,i+1}$. Both types of edges connect the vertices in V, indicating changes in machine status, either from idle to busy or from busy to idle. P is the set of processing times weighting the solid edges while the dashed edges are assumed to have null weights.*

A graph representation can be defined both for PFS and NPFS schedules. The type of σ determines which is the case. The main difference between a feasible PFS schedule and a feasible NPFS schedule is that the latter has, on at least one machine, a different sequence than in the previous machines. To make this idea precise we formally introduce the following definition:

Definition 2. ***Switching machine***. *A machine M_k is said to be a switching machine if it reverses the order in which jobs were processed on the previous machine.*

To illustrate Definition 1 and Definition 2 we present Figure 2 where the example of Figure 1 is considered. In Figure 2, the right panels depict the graph representations of the Gantt diagrams from the left panels. The NPFS schedule is at the bottom while the PFS schedule is on the top panels. The graph of the NPFS schedule has, in this case, one switching machine (machine M_3) on which the ordering of jobs becomes reversed. For convenience, we denote S_σ the set of switching machines defined by a particular schedule σ. Notice that machines M_1, M_2 and M_m cannot be switching machines in any optimal schedule due to the results of Conway et al. [8].

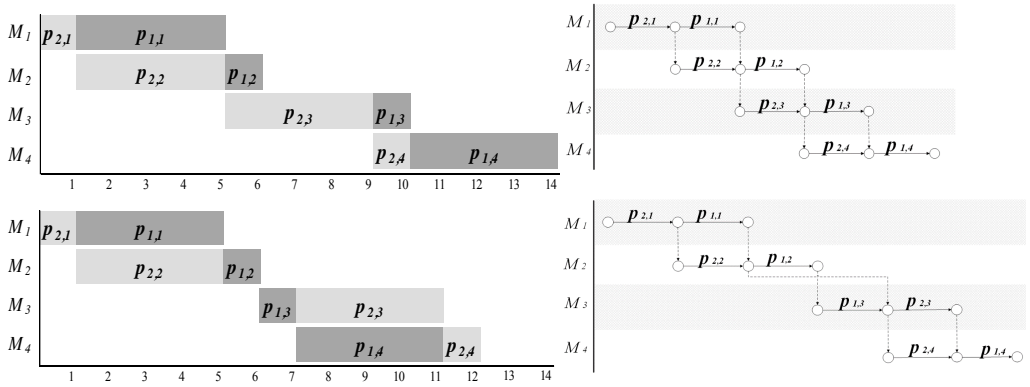

Figure 2: Gantt and graph representations of the optimal schedules of the PFS scheduling problem (top) and the NPFS scheduling problem (bottom) for an instance with $n = 2$ jobs and $m = 4$ machines. The switching machine in the NPFS schedule is M_3.

In general, the graph representation in scheduling problems allows to characterize the makespan (see [15, 13, 5]). This is also the case in our work, where the makespan is characterized by the *flow shop critical path*, as originally defined in [18] in terms of a grid of size $n \cdot m$ for a given schedule π (i.e. all the operations ordered according to π), in which the longest path from start to end yields π's makespan. We start up from this representation in [18], incorporating our own ideas:

Definition 3. ***Flow shop critical path***. *Given a feasible σ, the critical path is constituted by at least one operation on each machine, supporting the makespan of graph $G^\sigma(V, A, P)$.*

In order to illustrate Definition 3 we present Figure 3, in which the example of Figure 1 is revisited. In Figure 3 we highlight the critical path of both the PFS and NPFS schedules.

The solid edges of PFS and NPFS graphs in Figure 3 are labeled by the corresponding processing times, $p_{j,i}$. These labels allow to distinguish which operation belongs to the critical path (critical operations) and which not. Even in the case that processing times are unknown, this labeling allows to distinguish the critical operations from the rest. Thus, each operation has its own different label. A simple inspection shows that there are different possible paths from the first node on the first machine to the last node on the last machine. In the case that processing times are known, the longest of those paths is the critical path and its length in terms of the processing times is the makespan of the schedule. We can see, for instance, in Figure 3 that

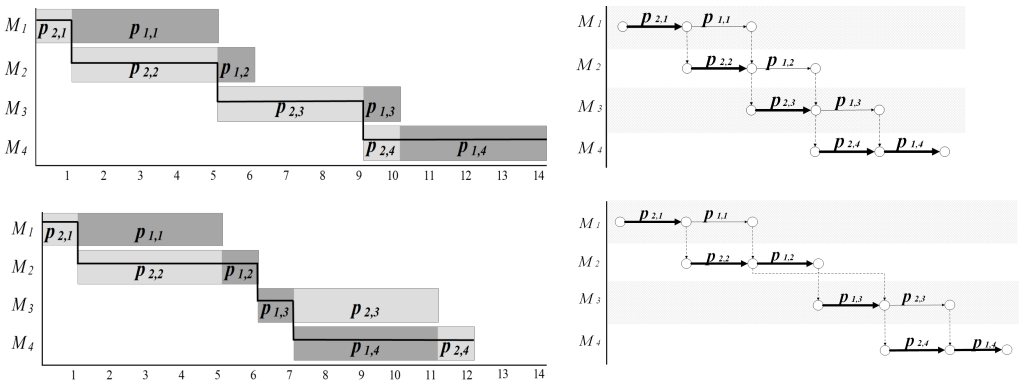

Figure 3: Critical path representations (solid line) of the optimal schedules of the PFS scheduling problem (top) and the NPFS scheduling problem (bottom) for an instance $\mathcal{I}$ with $n = 2$ jobs and $m = 4$ machines.

the critical path in the PFS graph is different from the one in the NPFS graph, supporting also different makespans.

With regards to definition 3, notice that in Figure 3 the *critical paths* include at least one operation on each machine, supporting the makespan of graph $G^{\sigma}(V, A, P)$. Then, by definition, paths that support the same value of makespan but skip operations on some machine cannot obtain. Figure 4 depicts this concept: the dashed line describes an invalid path that has the same makespan as the solid line one (a valid critical path), but skipping operations from M_2.

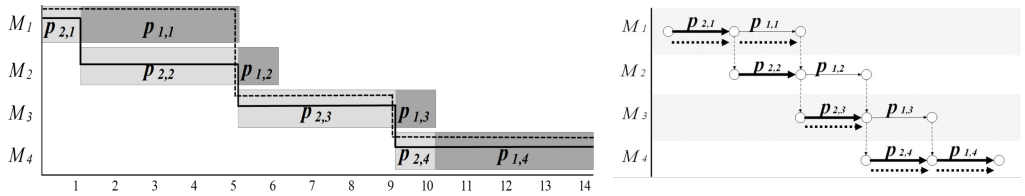

Figure 4: Gantt and graph representations of a critical (solid line) and an invalid critical path (dashed line) of an NPFS schedule (bottom) for an instance $\mathcal{I}$ with $n = 2$ jobs and $m = 4$ machines

The three-field notation introduced in Graham et al. [12] defines what a problem is, allowing to state and compare scheduling problems. However, this notation does not necessarily specify the number of machines and/or jobs. One may obtain different subproblems by setting the number of machings and/or jobs to specific fixed values. Not every such subproblem of $F|prmu|C_{\max}$ is NP-Hard, even if the subproblem obtained by fixing the number of machines to 3 belongs to this complexity class [10]. Then, even when we deal with the same scheduling problem, defined through the same three-field notation, different subproblems may not share the same features. In turn, for each subproblem may exist alternative parameterizations leading to different scheduling problems. Vallada et al. [35] show that there exist distinct parameterizations of flow shop scheduling problems minimizing makespan, and that for the same subproblem (i.e. a given value of n and m) there exist benchmark data making the search harder for iterated greedy algorithms. An important piece of information that help to assess this question is the gap between the lower bound (best relaxed solution) and the best state-of-the-art solution (see Vallada et al. [35]). The

take away for us is that, even if the class of problems ($F|prmu|C_{\max}$), the subproblem and the solution method are fixed, not every case presents the same difficulty for its solution.

Up to now we have presented a new way of representing the feasible schedules σ of the family of problems $F|prmu, n = 2|C_{\max}$ and $F|n = 2|C_{\max}$ by means of graphs $G^{\sigma}(V, A, P)$ with labeled arcs. We also established a relation between graphs and makespan through critical paths. Given a schedule σ the corresponding $G^{\sigma}(V, A, P)$ can be obtained, yielding a critical path supporting the optimal makespan. Since the edges of the graph are labeled, we can distinguish and identify the possible critical paths, characterizing the structure of solutions for the family of problems we are considering.

In what follows we model a parametrization of an instance of a problem, on the basis of its graph representation. In this way we can generate different critical paths for each feasible schedule and obtain all possible makespans for that schedule. For each instance we already know the size of the universe of feasible schedules ($n!$ for PFS scheduling problems and $n!^{m-2}$ for NPFS ones). Then, given an instance, we can generate the universe of makespans up from all the critical paths of each graph allowing the comparison of the structural properties of both the PFS and NPFS schedules. This comparison is based on the presence (or not) of operations in the critical paths. Even if the processing time of an operation is unknown, if it belongs to the critical paths of both PFS and NFPS, it impacts in the same way on their makespans, according to Definition 3.

On the other hand, in the cases in which there are operations that do not belong to both kinds of critical paths, the following magnitude becomes relevant:

Definition 4. ***Ratio*** ρ. *Given* $p_{\max} = \max_{m \in M, j \in J} p_{j,i}$ *and* $p_{\min} = \min_{m \in M, j \in J} p_{j,i}$, *we define* ρ *as the ratio between* $p_{\max}$ *and* $p_{\min}$.

It is interesting to note that even when the actual values of the processing times are unknown, ρ can be determined, at least roughly, in manufacturing settings. For instance, it can be known that the processing time of no operation can take more than 3 times that of the fastest operation.

4. Structural and dominance properties

In this section, we study the structural and dominance properties in both problems $F|n = 2|C_{\max}$ and $F|pmru, n = 2|C_{\max}$, assuming an arbitrary number of machines $m \geq 4$.

Theorem 1. *The length in number of operations of the critical path of* $G^{\sigma}(V, A, P)$ *is* $m + 1 + |S_{\sigma}|$.

Proof. We consider an arbitrary instance $\mathcal{I}$. For a feasible schedule σ with $|S_{\sigma}| = 0$, we take the critical path, defined by a set of $m + 1$ horizontal edges in $G^{\sigma}(V, A, P)$: two edges correspond to the first and last operations in the schedule σ. The other $m - 1$ edges are operations defined by each pair of machines M_k and M_{k+1}, $k \in \{1, \ldots, m - 1\}$.

For a feasible schedule σ with $|S_{\sigma}| \geq 1$, we prove the claim by induction. Consider the graph representation $G^{\sigma}(V, A, P)$ with $|S_{\sigma}| = 1$ and separate it into two sub-graphs such that the job of the last operation in one sub-graph is the job of the first operation in the other sub-graph. A switching machine, M_k, defines the split in two sub-graphs. Each sub-graph is free of switching machines, and they include $k - 1$ and $m - k + 1$ machines, respectively. We take then the critical paths from the first and second sub-graphs, of respective lengths $(k - 1) + 1$ and $(m - k + 1) + 1$. Then, the overall critical path of the graph representation $G^{\sigma}(V, A, P)$ is equal to the sum of the critical paths of its subgraphs, $m + 2 = m + 1 + |S_{\sigma}|$ for $|S_{\sigma}| = 1$.

Now consider a feasible schedule σ with $|S_\sigma| = \ell \leq m - 3$ and its graph representation $G^\sigma(V, A, P)$ and assume that the claim is true for any schedule with $\ell - 1$ switching machines. Let us separate it into two sub-graphs such that, on one hand, the job of the last operation in one sub-graph is again the same as the job of the first operation in the other sub-graph. We define the split in such way that the second subgraph does not include a switching machine. We can assume, without loss of generality, that the kth machine, M_k defines the split in two sub-graphs. Since the class of the remaining switching machines has cardinality $\ell - 1$, the critical path in the first sub-graph has $(k-1)+1+\ell-1$ edges while the critical path of the second has $(m-k+1)+1$ edges (recall that it does not contain a switching machine and it includes $m - k + 1$ machines). Thus, the critical path of the graph representation $G^\sigma(V, A, P)$ has $m + \ell + 1$ edges. □

Theorem 1 determines the length of the critical path of a given schedule in terms of the number of operations. That is, it calculates the cardinality of the subset of operations that belong to the critical path. This idea is depicted in Figure 5 which exhibits on its left side the Gantt charts and the critical paths (solid lines) representations of the numerical example presented in Figure 1. On the right side is the representation of the sets of operations that belong to each critical path of the optimal PFS (top) and NPFS schedules (bottom). While the cardinality of the set of operations that belongs to critical path of PFS schedule is 5, the cardinality of the corresponding set in the NPFS schedule is 6, verifying Theorem 1 for a feasible schedule σ with $|S_\sigma| = 1$.

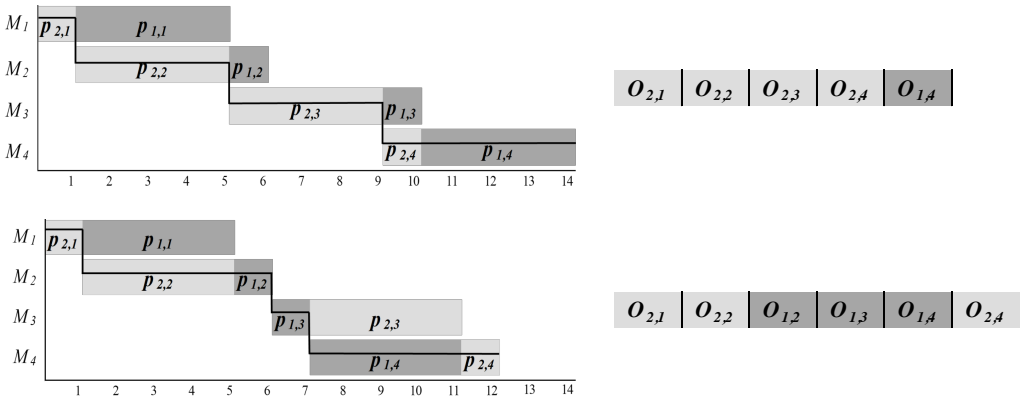

Figure 5: Gantt chart and critical path (solid line) representations (left), and ordered sequence of the operations belonging to the critical paths of the optimal PFS schedule (top) and the optimal NPFS schedule (bottom) for an instance $\mathcal{I}$ with $n = 2$ jobs and $m = 4$ machines.

According to Theorem 1, the NPFS critical path has more or equal operations than the possible PFS critical paths. Let us call a the number of these extra operations. Given that the total number of operations required in an instance $\mathcal{I}$ is constant, it can be inferred that with increasing values of a the NPFS critical path may end up including all the operations in the PFS critical path. Thus, all the operations in the NPFS critical path impact on the makespan of the schedule with a switching machines, according to Definition 3. Then, that critical path has a makespan at least as large as the one of the PFS critical path, since it includes all the operations of the latter and some more. Only if some operations have zero processing times it might happen that the makespans of both critical paths coincide. Otherwise, the makespan of the PFS schedule must improve over the makespan of the NPFS schedule. On the other hand, given the lower computational cost

of solving the PFS version of the scheduling problem, it becomes important to detect either the cases in which the makespans of both scheduling problems versions coincide or those in which the PFS schedule makespan is strictly lower. In order to capture this notion, we introduce two new definitions.

Definition 5. ***Dominance***. *Given* $\sigma_0^* := \arg\min_{\sigma,|S_\sigma|=0} F(\sigma)$ *and an NPFS schedule* π *with* $|S_\pi| > 0$*. The NPFS schedule* π *is dominated by permutation if* $F(\sigma_0^*) \leq F(\pi)$*, where* $F(\sigma_0^*)$ *and* $F(\pi)$ *represent the makespans of* σ_0^* *and* π*, respectively.*

Definition 6. ***Independent set*** $\mathcal{T}_\pi$**.** *Given an NPFS schedule* π *with* $|S_\pi| > 0$*. An independent set* $\mathcal{T}_\pi$ *is the set of operations in the critical path of* $G^{\sigma_0^*}(V, A, P)$ *that are not included in the critical path of* $G^\pi(V, A, P)$*.*

To illustrate Definition 6 we introduce Figure 6, in which an instance $\mathcal{I}$ of $n = 2$ jobs and $m = 5$ machines is considered. Figure 6 illustrates the Gantt charts of two feasible PFS schedules at its top and center panels, while the bottom picture shows an NPFS schedule with $|S_\pi| = 1$, and for each schedule the critical paths (solid lines) are depicted. From the PFS schedules, the optimal one, σ_0^*, corresponds to the center picture, while in the NPFS schedule π the switching machine is M_3. The operations of the independent set $\mathcal{T}_\pi$ are encircled by dashed lines.

As mentioned, it is important to identify those cases in which NPFS schedules are dominated by PFS ones. Thus, we present a relevant new result based on Theorem 1 and on the concept of dominance, which allows to identify cases where the PFS schedule dominates NPFS schedules. This is the gist of the next lemma.

Lemma 1. *Given an instance* $\mathcal{I}$ *and an NPFS schedule* π *with* $|S_\pi| > 0$*, if*

$$\rho \leq 1 + \frac{|S_\pi|}{|\mathcal{T}_\pi|}, \tag{1}$$

then π *is dominated by permutation.*

Proof. We consider an instance $\mathcal{I}$ and schedules σ_0^* and π with $|S_\pi| > 0$. Let $p_1 + \ldots + p_{m+1-|\mathcal{T}_\pi|}$ be the sum of processing times of the operations in the critical path of $G^{\sigma_0^*}(V, A, P)$ that are included in the critical path of $G^\pi(V, A, P)$. From Theorem 1, we have:

$$F(\sigma_0^*) = p_1 + \ldots + p_{m+1-|\mathcal{T}_\pi|} + \sum_{p_{j,i} \in \mathcal{T}_\pi} p_{j,i}$$

The case in which the makespan of the PFS schedule is the longest must obtain if we assume that all the operations of $\mathcal{T}_\pi$ have processing time $p_{\max}$. Then:

$$F(\sigma_0^*) \leq p_1 + \ldots + p_{m+1-|\mathcal{T}_\pi|} + |\mathcal{T}_\pi| p_{\max}. \tag{2}$$

Expression 2 can be formulated in terms of ρ and then:

$$F(\sigma_0^*) \leq p_1 + \ldots + p_{m+1-|\mathcal{T}_\pi|} + \rho \left(\frac{|\mathcal{T}_\pi|}{|\mathcal{T}_\pi| + |S_\pi|} \right) (|\mathcal{T}_\pi| + |S_\pi|)\, p_{\min}. \tag{3}$$

According to our assumptions we have that $\rho \leq 1 + |S_\pi|/|\mathcal{T}_\pi|$. Thus, we obtain the following inequality

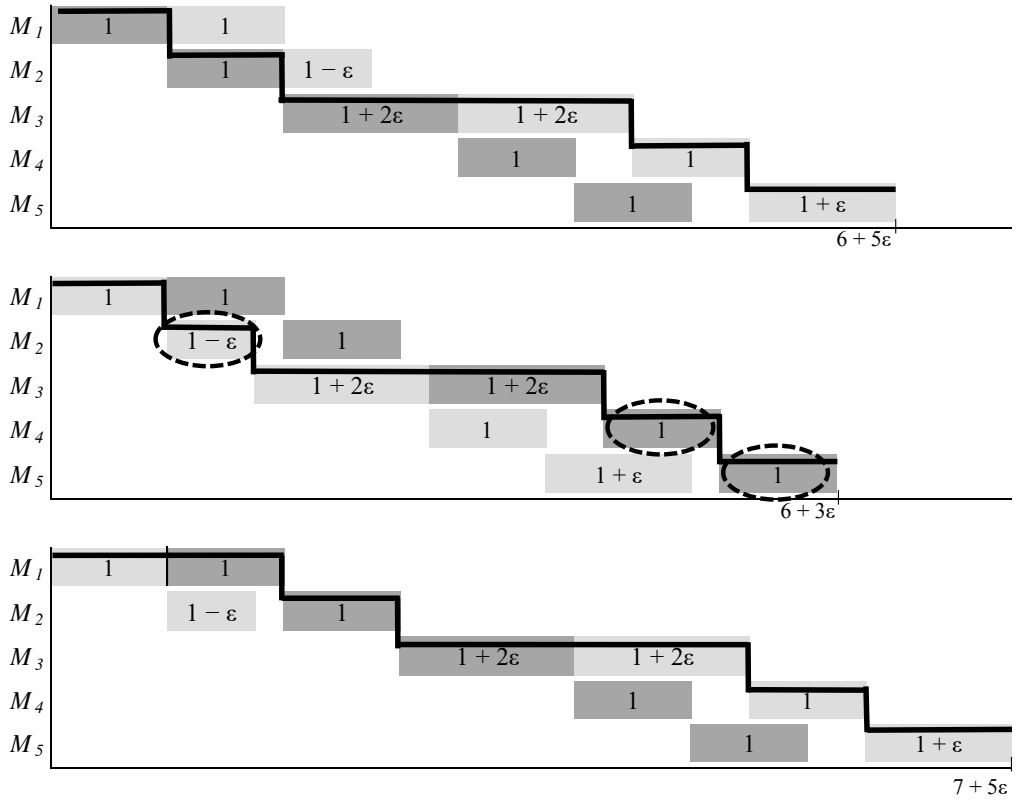

Figure 6: Gantt charts of a PFS schedule σ (top), the optimal PFS schedule σ_0^* (center), and an NPFS schedule π (bottom) with $|S_\pi| = 1$, and their critical paths (solid line) for an instance $\mathcal{I}$ with $n = 2$ jobs and $m = 5$ machines. The switching machine of the NPFS schedule π is M_3. The operations of the independent set $\mathcal{T}_\pi$ are encircled by a dashed line.

$$\rho\left(\frac{|\mathcal{T}_\pi|}{|\mathcal{T}_\pi| + |S_\pi|}\right)(|\mathcal{T}_\pi| + |S_\pi|)\, p_{\min} \leq (|\mathcal{T}_\pi| + |S_\pi|)\, p_{\min}. \tag{4}$$

From expressions 3 and 4, we have

$$F(\sigma_0^*) \leq p_1 + \ldots + p_{m+1-|\mathcal{T}_\pi|} + (|\mathcal{T}_\pi| + |S_\pi|)\, p_{\min} \leq F(\pi),$$

which indicates that the NPFS schedule π is *dominated* by permutation, concluding the proof. □

Lemma 1 justifies the claim that the NPFS schedule is not dominated by the PFS schedule in the numerical example of Figure 1, since the rate between p_{min} and p_{max} is 4, i.e. $\rho = 4$. This, in fact, does not meet the condition of the Lemma 1 that $\rho \leq 1 + |S_\pi|/|\mathcal{T}_\pi|$, being $|S_\pi| = |\mathcal{T}_\pi| = 1$. Nevertheless, Lemma 1 does not provide a necessary condition for the domination of the NPFS schedule, as it can be seen in the case of 2 jobs and 4 machines in which $p_{ji} = 1$, for all j, i, except for $p_{1,3} = p_{2,3} = 2$, for which the NPFS makespan ($C_{max-NPFS} = 8$) is dominated by the PFS one ($C_{max-PFS} = 6$).

A relevant consequence of Lemma 1 is that it implicitly associates the concept of dominance (Definition 5) to the number of switching machines of a feasible schedule σ with $|S_\sigma|$ and ρ (the relation between p_{max} and p_{min}). These are in turn associated to the cardinality of the independent set $|\mathcal{T}_\pi|$. In practice, ρ is a piece of information that is not hard to obtain, at least approximately, while $|S_\sigma|$ is a decision variable. On the contrary $|\mathcal{T}_\pi|$ cannot be determined easily. The next section addresses this problem.

4.1. Comparative combinatorial analysis of PFS and NPFS critical paths

Up to this point we have shown the structure of the critical paths of PFS and NPFS schedules. Simple combinatorial arguments allowed us to analyze the dominance relation between both types of scheduling problem. We intend now to deepen this analysis by generating an upper bound for the cardinality of $\mathcal{T}_\pi$. We will show that $|\mathcal{T}_\pi|$ is at most the difference between $2m$ and the number of operations in $G^\pi(V, A, P)$, i.e. $m + 1 + |S|$. Furthermore, we will show that this maximum is a tight bound.

An upper bound on $|\mathcal{T}_\pi|$ allows to find the minimal $|S_\pi|$ that satisfies the relation established in Lemma 1, which can be obtained by rewriting (1) as follows:

$$|S_\pi| \geqslant (\rho - 1) \cdot |\mathcal{T}_\pi| \tag{5}$$

In order to illustrate how this lower bound can be obtained, we present an example of an instance $\mathcal{I}$ with $n = 2$ jobs and $m = 5$ machines in Figure 7. Figure 7 shows the Gantt charts of the PFS schedule σ_0^* (top panel), an NPFS schedule π (bottom) with $|S_\pi| = 2$, and their critical paths (solid lines). The operations of the independent set $\mathcal{T}_\pi$ are highlighted with dashed encirclements. The switching machines of the NPFS schedule are M_3 and M_4.

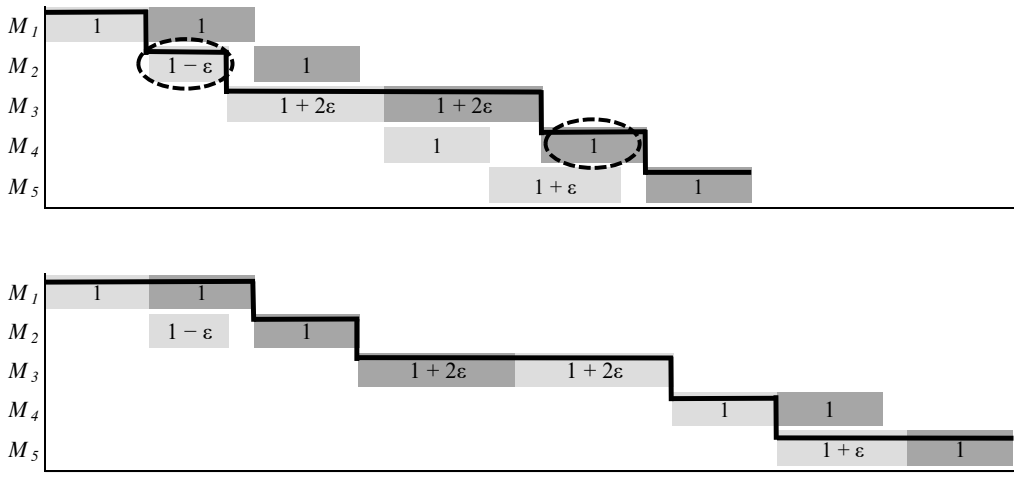

Figure 7: Gantt charts of the PFS schedule σ_0^* (top panel), and an NPFS schedule π (bottom panel) with $|S_\pi| = 2$, and their critical paths (solid lines) for an instance $\mathcal{I}$ with $n = 2$ jobs and $m = 5$ machines. The operations of the independent set $\mathcal{T}_\pi$ are encircled by a dashed line. The switching machines are M_3 and M_4.

We note that the NPFS schedule π is such that $|S_\pi| = 2$ and the universe of possible operations has size $2m = 10$. According to Theorem 1, the critical path of $G^\pi(V, A, P)$ includes $m + 1 +$

Table 1: Distribution of processing times. Here M_k is the first switching machine and $k = m - \ell$.

	M_1	M_2	M_3	...	M_k	M_{k+1}	M_{k+2}	...	M_m
J_1	1	$1-\epsilon$	1	...	$1+2\epsilon$	1	$1+\epsilon$	...	$1+\epsilon$
J_2	1	1	1	...	$1+(m-k)\epsilon$	1	1	...	1

$|S_\pi| = 8$ operations. This means that $G^\sigma(V, A, P)$ is such that the maximum cardinality of the independent set $|\mathcal{T}_\pi|$ is $2m - (m + 1 + |S_\pi|) = 10 - 8 = 2$, indicating that Lemma 1 is tight for the NPFS schedule π of Figure 7.

The main point shown in Figure 7 is that while the critical path of a PFS schedule tends to go through the operations of one of the jobs, the critical path of the NPFS schedule goes through the operations of the other one. In Figure 7, the critical path of the PFS schedule σ_0^* goes through the operations of J_1 up to M_3, and after that, through those of J_2. In contrast, the critical path of the NPFS schedule π (bottom panel) goes through the operations of J_2 up to M_3 and then through those of J_1. This is computed according to the distribution of the ϵ-dependent processing times. Notice that operation $O_{1,2}$ has processing time $p_{1,2} = 1 - \epsilon$, while for $O_{1,3}$ it is $p_{1,3} = 1 + 2\epsilon$. This implies that operation $O_{2,3}$ starts necessarily at the end of $O_{1,3}$. A similar criterion applies downstream: $O_{2,3}$ has $p_{2,3} = 1 + 2\epsilon$, compensating $O_{1,5}$, with processing time $p_{1,5} = 1 + \epsilon$. In this way, the critical path $G^{\sigma_0^*}(V, A, P)$ of the PFS schedule σ_0^* goes in an orderly fashion through J_1 and J_2. On the other hand, in the NPFS schedule π the critical path $G^\pi(V, A, P)$ goes through operations of J_1 and J_2 with processing times that compensate the contribution on the makespan from the operations of σ_0^*. Notice that the processing time $p_{1,5}$ of operation $O_{1,5}$ must be $1 + \epsilon$ in order to ensure that $G^\pi(V, A, P)$ goes through operations of J_1.

The structure in Figure 7 can be extended to m machines. Table 1 shows how to distribute the processing times in order to extend the structure to m machines. It shows that it is enough that $p_{1,2} = 1 - \epsilon$ (of $O_{1,2}$) to ensure that the critical path of the PFS schedule goes through J_1 up to M_k, no matter the number of machines between M_2 and M_k (provided J_1 is the first job in the schedule). Then, at M_k operation $O_{2,k}$ has to be such that $p_{2,k} = 1 + (m - k)\epsilon$ to compensate the processing times of the J_1 operations on machines between M_{k+2} and M_m. The other processing times can be conventionally set at $p_{j,i} = 1$.

In order to formalize the intuition reflected in Table 1, we define a particular instance, which we call *hard instance*, generalizing the structure of the instance from Figure 7.

Definition 7. ***Hard instance*** *An instance $\mathcal{I}$ is a hard instance when the operations of the jobs are defined by three blocks of machines and the number of switching machines, ℓ, is $\ell \leq m - 3$. The blocks are defined as follows:*

initial block: *It includes the first $k - 1$ operations of J_1 and J_2 to be carried out in the first $k - 1$ machines ($m - \ell \leq k \leq m - 1$). Job J_1 is assigned $k - 1$ operations with processing times equal to one, except operation $O_{1,2}$ with a processing time equal to $1 - \epsilon$. The second job J_2 has processing times equal to 1 on all of its $k - 1$ first operations.*

switching machines block: *The operations to be carried out on the ℓ switching machines are included in this block. The first switching machine is kth machine, where $k = m - \ell$, and comes after the machines in the initial block. The kth operation in J_1 has a processing time equal to $1 + 2\epsilon$ while in J_2 the kth operation has processing time $1 + (m - k)\epsilon$. The processing time of each operation on machines in this block is $1 + \epsilon$ for operations of job J_1 (except for machine M_{k+1} that is 1) and 1 for operations of job J_2 .*

final block: *It includes the operations to be carried out on the last machine for both jobs J_1 and J_2. The processing times are $1 + \epsilon$ for J_1 and 1 for J_2.*

A special feature of the hard instance is that among the feasible PFS schedules, the optimal one will always have the same job scheduling, regardless of the value of m. This is shown in the following result:

Lemma 2. *The PFS schedule σ_0^* of a hard instance has job J_1 as the first one in the sequence.*

Proof. We prove this claim by induction. As base case, we consider a hard instance with $m = 4$ machines. Here, the PFS schedule σ_0^* is defined by job J_1 as first job in the sequence, as shown in Figure 8.

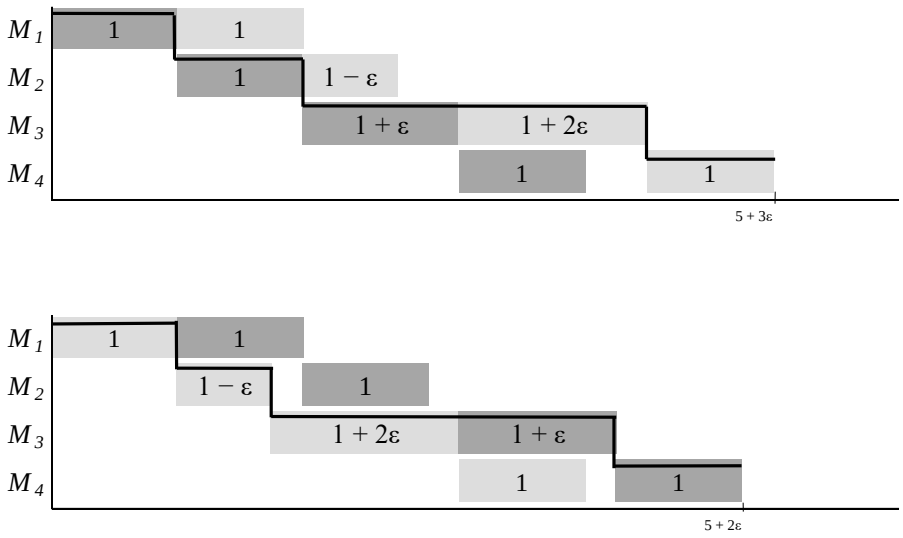

Figure 8: Gantt charts of the two critical paths of the PFS feasible schedule. Their critical paths are depicted with solid lines. The instance $\mathcal{I}$ has $n = 2$ jobs and $m = 4$ machines. In one of the PFS schedules the first job is J_1 (in the bottom panel) while in the other it is J_2 (top panel). The schedule of the bottom panel corresponds to σ_0^*.

For the inductive proof, we increase the number of operations of J_1 and J_2 in the initial and the switching machines blocks of a hard instance. Consider a hard instance with $m = 4 + \mu$ operations defined by the incorporation of μ operations to each job in the initial block with the processing times given in Definition 7. This incorporation of operations increases the makespan of the critical path of σ_0^* by μ units, having as in the base case, J_1 as the first job in the sequence.

Consider again a hard instance with $m = 4+\mu$ operations defined, this time, by the incorporation of μ operations to each job in the switching machines block (with unitary processing times). The kth operation of the job J_2 has a processing time equal to $1 + (m - k) \cdot \epsilon$ and the processing times of the operations on the last machines are defined by $m = 4 + \mu$ (according to the definition of the final block). Then, we compute the makespan of the PFS schedules and obtain an increase of $\mu(1 + 2\epsilon)$ for the PFS schedule in which the first job in the sequence is J_1 and an increase of $\mu(1 + 2\epsilon)$ for the PFS schedule with J_2 as the first job in the sequence. In agreement with the base case, the PFS schedule σ_0^* of the hard instance is defined by job J_1 as first job in the sequence.

This proves that in either case, J_1 is the first job in the sequence. □

The following theorem formalizes the generalization of the evidence depicted in Figure 7 to obtain a tight upper bound for the cardinality of the independent set $\mathcal{T}_\pi$ in an arbitrary instance.

Theorem 2. *$m - 1 - |S_\pi|$ is a tight upper bound on the cardinality of the independent set $\mathcal{T}_\pi$ for an NPFS schedule π with $|S_\pi| > 0$.*

Proof. Consider an NPFS schedule π with $|S_\pi| = \ell \leq m - 3$. Notice that the universe of possible operations has size $2m$. According to Theorem 1, in the critical path in $G^\pi(V, A, P)$, the number of operations included is $m+1+\ell$. This means that $G^\sigma(V, A, P)$ has at most $2m-(m+1+\ell) = m-1-\ell$ possible different operations, and consequently $|\mathcal{T}_\pi| \leq m - 1 - \ell$. We claim that this upper bound is tight for the hard instances that can be defined for this problem. To prove this claim, we need two technical lemmas.

Lemma 3. *Consider a hard instance, an NPFS schedule π with $|S_\pi| > 0$, which starts with the operation of job J_1. The $k-2$ operations on machines M_2 to M_{k-1} in the critical path of the PFS schedule σ_0^* are not in the critical path of the NPFS schedule π.*

Proof. We prove this claim by induction. Consider an NPFS schedule π and a PFS schedule σ_0^*, both starting, by definition, with the same operation. Consider, in particular, an NPFS schedule π with $|S_\pi| = 1$. Then, the number of the operations in the switching machines block is 1 for each job J_1 and J_2, where $S_\pi = \{M_k\}$.

For the base case, we consider $k = 3$ and a hard instance with $m = k + 1 = 4$ machines. In this case, we obtain the largest independent set of the NPFS schedule π, $|\mathcal{T}_\pi| = 2$ as shown in Figure 8.

We now consider $k - 1 = \mu$ and a hard instance with $m = \mu + 2$ machines. Note that μ operations for each job belong to the initial block. Given that the second operation of J_1 has a processing time equal to $1 - \epsilon$ and all remaining operations in the block for jobs J_1 and J_2 have processing time 1, in the PFS schedule σ_0^* the completion times of operation $O_{1,k}$ and $O_{2,k-1}$ are $\mu + 1 + \epsilon$ and $\mu + 1$, respectively. Then, the operations in the critical path of σ_0^* the operations on machines M_2 to M_{k-1} correspond to J_1. For the NPFS schedule π, we have that the first switching machine is M_k and then the operations on machines M_2 to M_{k-1} in the critical path correspond to J_2, concluding the proof.[2] □

Lemma 4. *Consider a hard instance. Only one operation on machines M_{k+1} to M_m in the critical path defined by the PFS schedule σ_0^* is not in the critical path defined for an NPFS schedule π with $|S_\pi| > 0$.*

Proof. We prove this claim again by induction. Consider an NPFS schedule π and a PFS schedule σ_0^*. We know that both schedules start with the same operation. Consider an NPFS schedule π with $|S_\pi| = m - 3$. Then, the number of operations in the initial block is 2 for both J_1 and J_2, where M_k is the first switching machine of the instance, and J_1 is the first job in the sequence.

For the base case $k = 3$, we consider a hard instance with $m = k + 1 = 4$ machines and the largest independent set of an NPFS schedule π is such that $|\mathcal{T}_\pi| = 2$. One operation from the initial block and another operation from the switching machines block are shown in Figure 8.

[2] We can extend the proof by noting that the claim is still true for arbitrary hard instances of an NPFS schedule π with $|S_\pi| > 1$, where M_k is the first switching machine of the instance.

Consider now a hard instance with $k = 4$, $m = k + 1 = 5$ machines and largest independent set of an NPFS schedule π such that $|\mathcal{T}_\pi| = 2$. One operation from the initial block and another from the switching machines block are shown in Figure 7.

For $k = \mu$, we consider a hard instance with $m = \mu + 1$ machines. We have one operation in the independent set from the initial block, since $|S_\pi| = m - 3$ limits the number of machines that can be included in that block. We now analyze the impact of incorporating μ operations in the rest of the structure of the hard instance, i.e., in the switching machines and the final blocks.

Note that operations of both jobs J_1 and J_2 to be performed on the switching machines $M_k, \ldots, M_{m-2}$ are included in the critical path of the NPFS schedule π. By the definition of the hard instance, we have that the operations of jobs J_1 and J_2 to be carried out on the last machine M_m are included in the critical path of the NPFS schedule π (as in Figure 7). Thus, we see that the single operation $O_{2\,m-1}$ defined by the PFS schedule σ_0^* is the only one that is not in the critical path defined for an NPFS schedule π with $|S_\pi| = m - 3$.

Finally, we note that this result holds for an arbitrary hard instance with an NPFS schedule π with $|S_\pi| > 0$ by considering a number of operations in the initial blocks larger than 2 for each of the jobs. This concludes the proof. □

The combination of Lemmas 3 and 4 yields the tightness of the upper bound on $|\mathcal{T}_\pi|$. □

4.2. NPFS feasible space: PFS-bounding

Lemma 1 associates the number of switching machines, ρ (the quotient between p_{max} and p_{min}) and $|\mathcal{T}_\pi|$. We seek now to reformulate that claim in a form more amenable for practice. In order to do that we introduce Corollaries 1 and 2:

Corollary 1. *Given an instance $\mathcal{I}$ with m machines and an NPFS schedule π with $|S_\pi| > 0$, if*

$$\rho \leq \frac{m-1}{m-1-|S_\pi|}. \tag{6}$$

then π is dominated by permutation.

Corollary 2. *Given an instance $\mathcal{I}$ with m machines and an NPFS schedule π with $|S_\pi| > 0$, if $(m-1)\frac{\rho-1}{\rho} \leq |S_\pi|$, then any arbitrary NPFS π' with $|S_\pi| \leq |S_{\pi'}|$ is dominated by permutation.*

Corollaries 1 and 2 follow from replacing $\mathcal{T}_\pi$ by the tight upper bound found in Theorem 2 in expression (1) from Lemma 1.

Corollary 3. *Given an instance $\mathcal{I}$ with m machines, the number of the NPFS schedules dominated is at least*

$$\sum_{i=\left\lceil (m-1)\frac{\rho-1}{\rho} \right\rceil}^{m-2} \binom{m-2}{i}.$$

This result is obtained by considering the number of NPFS schedules dominated by permutation.

We are now in conditions of graphing the *dominance by permutation* zone for NPFS schedules according the number of machines m and the ratio ρ.

Corollary 1 provides a practical summary of most of the results found in this investigation, since ρ as given by equation 6 can be easily estimated in real world contexts. In an industrial

flow shop environment, the number of machines m will be known. With these two parameters (ρ and m) at hand the scheduler can read in Figure 9 the maximal cost-effective $|S|$ discarding larger values of S, reducing the size of the search space for her problem without missing the optimal solution. Suppose the scheduler faces an instance $\mathcal{I}$ of 10 machines ($m = 10$) and 2 jobs, where the operations are such that $\rho = 3$. In Figure 9 we identify the green curve corresponding to $m = 10$, and given $\rho = 3$ we see that for NPFS schedules with a number of the switching machines larger than 6, all possible NPFS solutions will be dominated by PFS ones. Thus, it makes sense to focus on the others.

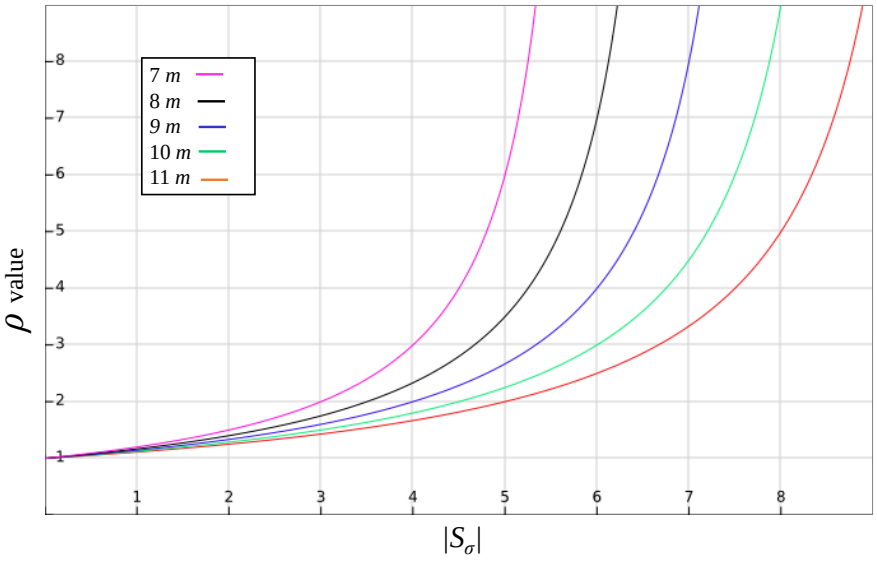

Figure 9: Graphs of ρ as function of the number of switching machines, $|S_\sigma|$ defined by equation 6, parameterized by different values of m.

These results indicate that NPFS schedules tend to incorporate more operations than PFS ones. It can be inferred that the latter will usually dominate the former under not very demanding conditions on processing times. It is natural to consider that something similar should happen with NPFS schedules with different quantities of switching machines, the ones with lowest number of switching machines dominating those with a larger number of them. This follows from Theorem 1, which indicates that the cardinality of the critical path depends on the number of switching machines. We will present a formal proof of this intuition in next section.

5. Generalizations

In this section we extend and generalize the comparison of NPFS schedules with different quantities of switching machines for flow shop systems with 2 jobs. We follow here the same line of reasoning as for the comparison of PFS and NPFS schedules in Lemma 1, by defining a *generalized dominance* concept according to which a schedule π_2 with $|S_2|$ switching machines dominates a schedule π_1 with $|S_1|$ switching machines, where $|S_1| > |S_2|$.

Definition 8. ***Generalized dominance***. *Given two NPFS schedules π_1 and π_2 with $|S_{\pi_1}|$ and $|S_{\pi_2}|$ switching machines, respectively, and $|S_{\pi_1}| > |S_{\pi_2}|$. π_1 is dominated by π_2 if $F(\pi_2) \leq F(\pi_1)$, where $F(\pi_2)$ and $F(\pi_1)$ represent the makespans of π_1 and π_2, respectively.*

Another extension involves defining a set of independent operations but this time for NPFS schedules with different quantities of switching machines, which we call *generalized independent set* and is analogous to the independent set of Definition 6.

Definition 9. ***Generalized independent set*** $\mathcal{T}_{\pi_1,\pi_2}$. *Given two NPFS schedules π_1 and π_2 with $|S_{\pi_1}|$ and $|S_{\pi_2}|$ switching machines, respectively, and $|S_{\pi_1}| > |S_{\pi_2}|$, the NPFS generalized independent set $\mathcal{T}_{\pi_1,\pi_2}$ is the set of operations in the critical path of $G^{\pi_2}(V, A, P)$ that are not included in the critical path of $G^{\pi_1}(V, A, P)$.*

Following Definition 9, we can formulate an analogous to Lemma 1:

Lemma 5. *Given an instance $\mathcal{I}$ and two NPFS schedules π_1 and π_2 with $|S_{\pi_1}|$ and $|S_{\pi_2}|$ switching machines respectively, and $|S_{\pi_1}| > |S_{\pi_2}|$, if*

$$\rho \leq 1 + \frac{(|S_{\pi_1}| - |S_{\pi_2}|)}{|\mathcal{T}_{\pi_1,\pi_2}|} \tag{7}$$

then π_1 is dominated by π_2.

Proof. Consider an instance $\mathcal{I}$ and schedules π_1 and π_2 with $|S_{\pi_1}|$ and $|S_{\pi_2}|$ switching machines respectively, and $|S_{\pi_1}| > |S_{\pi_2}|$. Let $p_1 + \ldots + p_{m+1-|\mathcal{T}_{\pi_1,\pi_2}|}$ be the sum of processing times of operations in the critical path of $G^{\pi_2}(V, A, P)$ that are included in the critical path of $G^{\pi_1}(V, A, P)$. From Theorem 1, we have:

$$F(\pi_2) = p_1 + \ldots + p_{m+1-|\mathcal{T}_{\pi_1,\pi_2}|} + \sum_{p_{j=2,i} \in \mathcal{T}_{\pi_1,\pi_2}} p_{j,i}$$

If we assume that all the operations from $\mathcal{T}_{\pi_1,\pi_2}$ have processing time $p_{\max}$ we obtain:

$$F(\pi_2) \leq p_1 + \ldots + p_{m+1-|\mathcal{T}_{\pi_1,\pi_2}|} + |\mathcal{T}_{\pi_1,\pi_2}| p_{\max}. \tag{8}$$

Expression 8 can be thus reformulated in terms of ρ:

$$F(\pi_2) \leq p_1 + \ldots + p_{m+1-|\mathcal{T}_{\pi_1,\pi_2}|} + \rho \left(\frac{|\mathcal{T}_{\pi_1,\pi_2}|}{|\mathcal{T}_{\pi_1,\pi_2}| + (|S_{\pi_1}| - |S_{\pi_2}|)} \right) (|\mathcal{T}_{\pi_1,\pi_2}| + (|S_{\pi_1}| - |S_{\pi_2}|))\, p_{\min}. \tag{9}$$

According to our assumptions we have that $\rho \leq 1 + (|S_{\pi_1}| - |S_{\pi_2}|)/|\mathcal{T}_{\pi_1,\pi_2}|$. Thus, we obtain the following inequality

$$\rho \left(\frac{|\mathcal{T}_{\pi_1,\pi_2}|}{|\mathcal{T}_{\pi_1,\pi_2}| + (|S_{\pi_1}| - |S_{\pi_2}|)} \right) (|\mathcal{T}_{\pi_1,\pi_2}| + (|S_{\pi_1}| - |S_{\pi_2}|))\, p_{\min} \leq (|\mathcal{T}_{\pi_1,\pi_2}| + (|S_{\pi_1}| - |S_{\pi_2}|))\, p_{\min}. \tag{10}$$

From expressions 9 and 10, we have

$$F(\pi_2) \leq p_1 + \ldots + p_{m+1-|\mathcal{T}_{\pi_1,\pi_2}|} + (\mathcal{T}_{\pi_1,\pi_2} + (|S_{\pi_1}| - |S_{\pi_2}|))\, p_{\min} \leq F(\pi_1),$$

which indicates that the NPFS schedule π_1 is *dominated* by schedule π_2, concluding the proof. □

Table 2: Distribution of processing times for generalized results when $|S_{\pi_1}|$ and $|S_{\pi_2}|$ are both even or both odd at the same time.

	M_1	M_2	...	M_{k-1}	M_k	M_{k+1}	...	$M_{k+\lvert S_{\pi_2}\rvert-1}$	$M_{k+\lvert S_{\pi_2}\rvert}$	$M_{k+\lvert S_{\pi_2}\rvert+1}$	...	$M_{k+\lvert S_{\pi_1}\rvert-1}$	...	M_m
J_1	1	$1-\epsilon$	1	1	$1+2\epsilon$	1	1	1	1	$1+2\epsilon$	1	1	1	$1+\epsilon$
J_2	1	1	1	1	$1+2\epsilon$	1	1	1	1	$1+2\epsilon$	1	1	1	$1+\epsilon$

Table 3: Distribution of processing times for generalized results when $|S_{\pi_1}|$ and $|S_{\pi_2}|$ are even and odd, respectively, and vice versa.

	M_1	M_2	...	M_{k-1}	M_k	M_{k+1}	...	$M_{k+\lvert S_{\pi_2}\rvert-1}$	$M_{k+\lvert S_{\pi_2}\rvert}$	$M_{k+\lvert S_{\pi_2}\rvert+1}$	...	$M_{k+\lvert S_{\pi_1}\rvert-1}$	...	M_m
J_1	1	$1-\epsilon$	1	1	$1+2\epsilon$	1	1	1	1	1	1	1	1	$1+\epsilon$
J_2	1	1	1	1	$1+2\epsilon$	1	1	1	1	1	1	1	1	$1+\epsilon$

Lemma 5 relates the value of ρ with those of $|S_{\pi_1}|$ amd $|S_{\pi_2}|$. These values are either easy to acquire in practice or are decision variables. But, as in Lemma 1, this result depends on $|\mathcal{T}_{\pi_1,\pi_2}|$ which is not directly measurable in industrial environments. We need thus a new upper bound for this magnitude, defined in terms of other variables.

Since the generalized independent set plays a similar role as the independent set, the new upper bound will be determined analogously. This implies showing that the operations that are not in the critical path of the NPFS schedule π_1 (the one with the largest number of the switching machines) may be in the critical path of π_2 (the one with the lowest number of switching machines). This means that $|\mathcal{T}_{\pi_1,\pi_2}|$ has a tight upper bound $m-(1+|S_{\pi_1}|)$ (which follows from $2m-(m+1+|S_{\pi_1}|)$). To show this we introduce a *generalized instance* in which the bound is tight for every pair $|S_{\pi_1}|$ and $|S_{\pi_2}|$ in which $|S_{\pi_1}|, |S_{\pi_2}| \leq m-3$. The distribution of processing times of the generalized instance is presented in Tables 2 and 3.

Tables 2 and 3 have a similar structure as that for the hard instance in Definition 7 and Table 1. The only difference between Tables 2 and 3 is that the former applies to cases in which $|S_{\pi_1}|$ and $|S_{\pi_2}|$ are either both even or both odd, making schedules π_1 and π_2 start and end in the same order. Table 3 instead, applies to the case in which $|S_{\pi_1}|$ and $|S_{\pi_2}|$ have different parity.

Definition 10. ***Generalized instance***. *An instance $\mathcal{I}$ is a generalized instance when two NPFS schedules π_1 and π_2 with $|S_{\pi_1}|$ and $|S_{\pi_2}|$ switching machines, respectively, satisfy $|S_{\pi_1}| > |S_{\pi_2}|$ ($|S_{\pi_2}| > 0$) and the operations of jobs are defined by three blocks of machines. The blocks are defined as follows:*

Initial block: *This block includes the first $k-1$ operations of J_1 and J_2 to be carried out on the first $k-1$ machines, being the kth machine the first switching machine of π_1 while the first switching machine of π_2 is $k+1$ ($m-|S_{\pi_1}| \leq k \leq m-1$). The processing times of the $k-1$ operations of J_1 are all equal to 1, except for operation $O_{1,2}$, which has processing time $1-\epsilon$. The processing times of the $k-1$ first operations of J_2 are all 1.*

switching machines block: *The operations to be carried out on the S_{π_1} and S_{π_2} switching machines are included in this block (all the switching machines are consecutive). The first switching machine is the kth machine, where $k = m-|S_{\pi_1}|$, and come after the initial block. The operations to be carried out on the $|S_{\pi_2}|$ switching machines are also included, but they start at machine $k+1$. The kth operations of jobs J_1 and J_2 have a processing time equal to $1+2\epsilon$. The other processing times of the operations of both jobs included in this block are equal to 1. The first operation after the last switching machine of S_{π_2}, i.e.,*

$m_{k+|S_{\pi_2}|+1}$. *Two cases are possible, depending on whether* $|S_{\pi_1}|$ *and* $|S_{\pi_2}|$ *are both even or odd simultaneously or not. Tables 2 and 3 show these two cases.*

final block: *It includes the operations to be carried out on the last machine for both* J_1 *and* J_2*. Their processing times are equal to* $1 + \epsilon$*.*

Figure 10 illustrates the generalized instance, presenting the Gantt diagrams for the entries in Table 2. Schedule π_1 is represented in the bottom panel of Figure 10, while schedule π_2 is depicted in the top panel, both with their corresponding critical paths. The Gantt diagram of π_2 identifies the operations in $\mathcal{T}_{\pi_1,\pi_2}$ with dashed circles. We can now determine the upper bound on the cardinality of this last set.

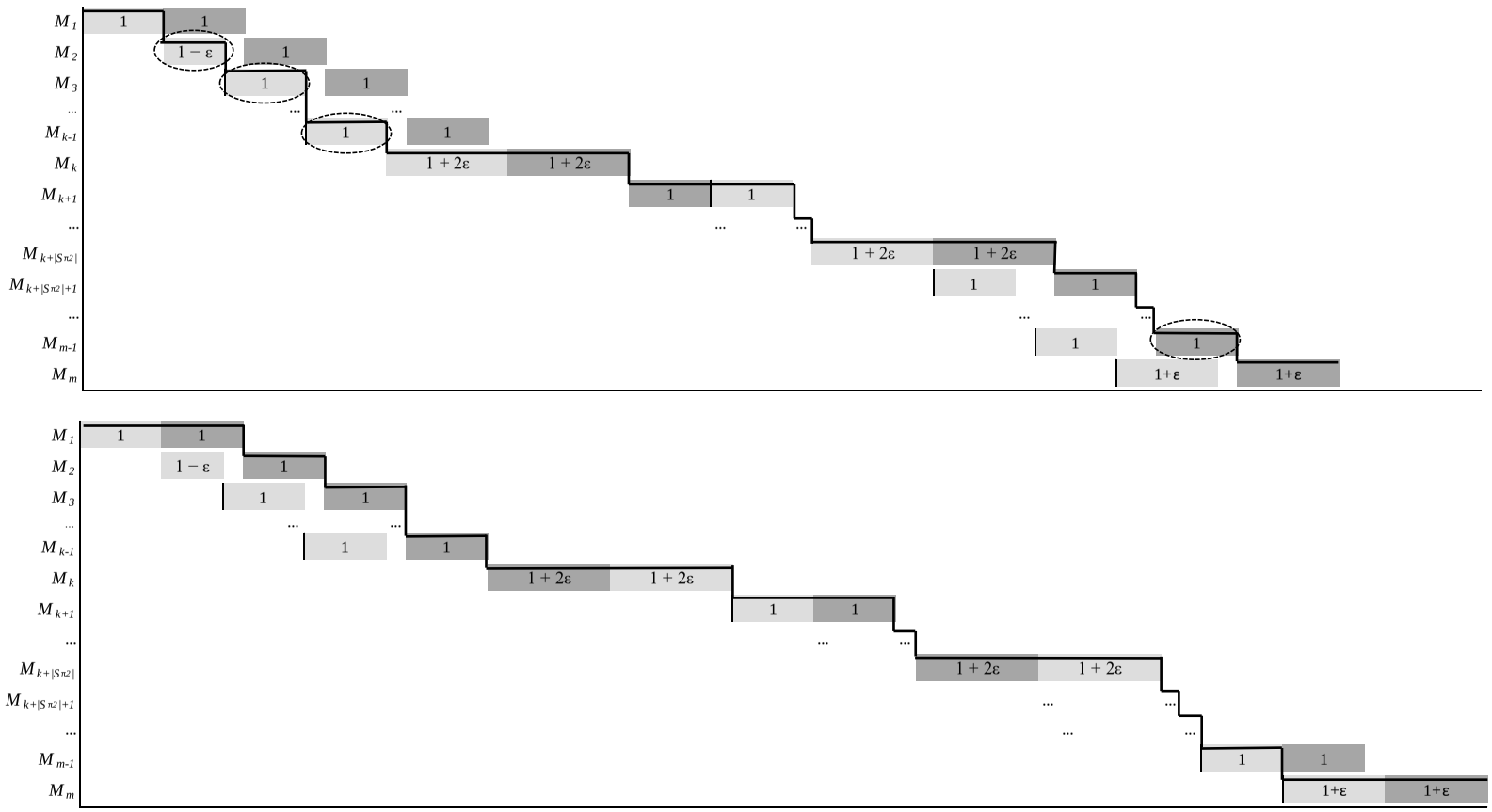

Figure 10: Generalized instance for the case in which $|S_{\pi_1}|$ and $|S_{\pi_2}|$ are even or odd simultaneously. Schedule π_1 is depicted in the bottom panel and schedule π_2 in the top panel.

Theorem 3. *A tight upper bound on the cardinality of the extended independent set $\mathcal{T}_{\pi_1,\pi_2}$ for two NPFS schedules π_1 and π_1 with $|S_{\pi_1}|$ and $|S_{\pi_2}|$ respectively, and $|S_{\pi_1}| > |S_{\pi_2}|$ is $m - 1 - |S_{\pi_1}|$.*

Proof. Consider an NPFS schedule π_1 with $|S_{\pi_1}| \leq m - 3$. Notice that the universe of possible operations has size $2m$. According to Theorem 1, in the critical path in $G^{\pi_1}(V, A, P)$, the number of operations included is $m+1+|S_{\pi_1}|$. That means that $G^{\pi_2}(V, A, P)$ has at most $2m-(m+1+|S_{\pi_1}|) = m - 1 - |S_{\pi_1}|$ possible different operations, and consequently $|\mathcal{T}_{\pi_1,\pi_2}| \leq m - 1 - |S_{\pi_1}|$. We claim that this upper bound is tight for the generalized instances defined for the problem. To prove this claim, we need two technical lemmas.

Lemma 6. *Consider a generalized instance, and two NPFS schedules π_1 and π_2 with $|S_{\pi_1}| > |S_{\pi_2}|$, starting with the operation of job J_1. The $k - 2$ operations on machines M_2 to M_{k-1} in the critical path of π_2 are not in the critical path of π_1.*

(The proof of this Lemma is in the Appendix section).

Lemma 7. *Consider a generalized instance. Only one operation on machines M_{k+1} to M_m in the critical path defined by the NPFS schedule π_2 is not in the critical path defined for an NPFS schedule π_1 such that $|S_{\pi_1}| > |S_{\pi_2}|$.*

(The proof of this Lemma is in the Appendix section).

Thus, combining the results of Lemmas 6 and 7 we obtain the complete set of operations appearing in the formulation of Theorem 3, since Lemma 6 identifies the first $k - 2$ operations and Lemma 7 the $m - (k + |S_{\pi_1}| - 1)$ operations. Thus, adding both we get

$$k - 2 + m - (k + |S_{\pi_1}| - 1),$$

which is

$$m - |S_{\pi_1}| - 1$$

proving that the the bound is tight for the generalized instance. □

5.1. NPFS generalized feasible space

Lemma 5 associates the cardinality of the sets of switching machines $|S_{\pi_1}|$,$|S_{\pi_2}|$, the value of ρ (the relation between p_{max} and p_{min}) and the cardinality of the generalized independent set, $\mathcal{T}_{\pi_1,\pi_2}$. Corollary 4 gives an alternative version:

Corollary 4. *Given a generalized instance $\mathcal{I}$ with m machines and two NPFS schedules π_1 and π_2 with $|S_{\pi_1}| > |S_{\pi_2}|$, if*

$$\rho \leq \frac{m - 1 - |S_{\pi_2}|}{m - 1 - |S_{\pi_1}|}. \tag{11}$$

then π_1 is dominated by π_2.

It is easy to see that Corollary 4 generalizes Corollary 1 and inequality 6. Corollary 1 holds as a particular case, taking into account that $|S_{\pi_2}| = 0$ because the π_2 is a PFS schedule.

Corollary 5. *Given a generalized instance $\mathcal{I}$ with m machines and an NPFS schedule π_1 and π_2 with $|S_{\pi_1}| > |S_{\pi_2}| \geq 0$, if $(m-1)\frac{\rho-1}{\rho} + \frac{|S_{\pi_2}|}{\rho} \leq |S_{\pi_1}|$, then any arbitrary NPFS π' with $|S_{\pi'}| \geq |S_{\pi_1}|$ is dominated by π_2*

These two corollaries obtain by replacing $\mathcal{T}_\pi$ by its tight upper bound in expression 7 of Lemma 5.

Corollary 6. *Given a generalized instance $\mathcal{I}$ with m machines and an NPFS schedule π_1 and π_2 with $|S_{\pi_1}| > |S_{\pi_2}| \geq 0$. The number of NPFS schedules dominated by π_2 is at least*

$$\sum_{i=\left\lceil (m-1)\frac{\rho-1}{\rho} + \frac{|S_{\pi_2}|}{\rho} \right\rceil}^{m-2} \binom{m-2}{i}.$$

This corollary is obtained by considering the number of schedules in NPFS scheduling problem dominated by NPFS schedules of low number of switching machines.

This result, that parallels Corollary 1, is useful in practice, being the only difference that now equation 11 includes both $|S_{\pi_2}|$ and $|S_{\pi_1}|$. Figure 11 depicts the values of ρ for which π_2 dominates π_1 under generalized dominance. Each curve is parameterized by the number of machines. Under the curves, π_2 is the dominant schedule. Finally, in Figure 11 we assume that $|S_{\pi_2}| = 2$ in all cases.

Comparing Figures 9 and 11 we can see that the curves have a very similar behavior, only that in the latter case (in which $|S_{\pi_2}| = 2$) they tend to be displaced to the right. So, revisiting the example analyzed for Figure 9 with $m = 10$ and $\rho \leq 3$, in which we found that PFS schedules dominate NPFS ones when $|S| > 6$, now in the case of two NPFS schedules we can see that the one with less switching machines (i.e. with $|S_{\pi_2}| = 2$) dominates those such $|S_{\pi_1}| > 6$. In fact, for this to be true it is enough that $\rho < 2.5$, approximately. This shows that ρ becomes lower from one figure to the other. This is because the inclusion of switching machines in both schedules reduces the upper bound on $|\mathcal{T}_{\pi_1,\pi_2}|$. In other words, as indicated by equation 11 a larger $|S_{\pi_2}|$ leads to a lower ρ.

6. Conclusions

In this work we have studied the PFS and the NPFS scheduling problems in the case of two jobs, when makespan is the objective function and processing times are not known. We found new structural and dominance properties of these problems. These properties provide a novel perspective on these problems. We found new instances in which a PFS schedule dominates an NPFS one with at least one switching machine. We also described a new bound on the maximum number of feasible schedules of $F||C_{\max}$ problems, when the only information available about the problem is the ratio between $p_{\max}$ and the $p_{\min}$, defined as ρ. This, in turn, allows a drastic reduction of the number of candidate schedules. As a general remark, we can notice that, for the $F||C_{\max}$ problem, the larger the number of switching machines in a schedule, the lower will be the possibility of the schedule being optimal.

Open questions for future work include how to extend this approach when the number of jobs is $n \geq 3$ and how to characterize a tight upper bound on the feasible schedule for particular kinds of instances or for the general problem $F||C_{\max}$ when $|S_\sigma| \geq 2$.

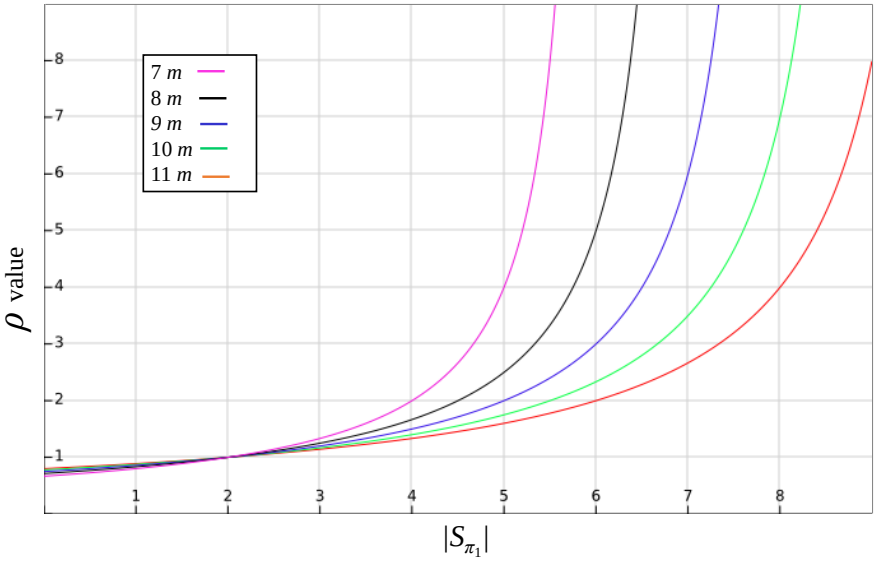

Figure 11: Graphs of ρ as function of $|S_{\pi_1}|$ defined by equation 11, parameterized by m, with $|S_{\pi_2}| = 2$.

Acknowledgements

The authors are grateful for partial support from the following sources: Proyecto DICYT 061817VP, Universidad de Santiago de Chile (Ó. C. Vaśquez),CYTED Ciencia y Tecnología para el Desarrollo [P318RT0165]; Consejo Nacional de Investigaciones Científicas y Técnicas [PIP: 11220150100777]; Universidad Nacional del Sur [PGI: 24/ ZJ35] (D. Rossit, F. Tohmé, Mariano Frutos) ; M.D. Safe acknowledges partial support from ANPCyT Grant PICT-2017-1315 and Universidad Nacional delSur Grants PGI 24/L103 and 24/L115

Appendix A. Proofs of section 5

Proofs of lemmas 6 and 7 respectively.

Proof. Consider two NPFS schedules π_1 and π_2 with $|S_{\pi_1}|$ and $|S_{\pi_2}|$ and $|S_{\pi_1}| > |S_{\pi_2}|$, where both schedules start with the same operation by definition. The first switching machine of π_1 is M_k, while for π_2 is M_{k+1}, as illustrated in Figure 10.

Consider the initial blocks of both Gantt charts of Figure10. Given that the second operation of J_1 has processing time equal to $1-\epsilon$ and all remaining processing times in the block for both jobs J_1 and J_2 are 1, in schedule π_2 the completion times of operations $O_{1,k}$ and $O_{2,k-1}$ are $k+\epsilon$ and k, respectively. Then, the operations in the critical path of π_2 of machines M_2 to M_{k-1} are operations of job J_1. For schedule π_1, we have that the first switching machine is M_k and then, the operations on machines M_2 to M_{k-1} in its critical path are operations of job J_2, concluding the proof of this lemma. □

Proof. Consider two NPFS schedules π_1 and π_2 with $|S_{\pi_1}| > |S_{\pi_2}|$, where both schedules start, by definition, with the same operation. The first switching machine of π_1 is M_k such that machine $M_{k+|S_{\pi_1}|-1}$ coincides with M_{m-1}, while for π_2 it is M_{k+1}, as illustrated in Figure 10.

Since we are considering the case where $|S_{\pi_1}|$ and $|S_{\pi_2}|$ are even or uneven simultaneously (Table 2), both schedules will finish with the same job ordering, as in Figure 10. Note that in this representation the operations of both jobs on machine $M_{k+|S_{\pi_2}|+1}$ have processing time $1+2\epsilon$. This leads to the inclusion of operations of J_2 in the critical path of π_2, as shown in the top panel. However, as the operations of machine $M_{k+|S_{\pi_2}|+1}$ are included in the switching machines block of π_1 (all the switching machines are consecutive), their processing times may not have the same impact on the critical path of π_1. While the processing times of the last operations of both jobs $(1+\epsilon)$ on M_m affect the critical path of π_1 by including operations of J_1 instead of J_2 as happens in the critical path of π_2. Finally, operation $O_{2,m-1}$ is not included in the critical path of π_1. □

References

References

[1] S. Akers. A graphical approach to production scheduling problems. *Operations Research*, 4(2):244–245, 1956.
[2] S. Akers and J. Friedman. A non-numerical approach to production scheduling problems. *Journal of the Operations Research Society of America*, 3(4):429–442, 1955.
[3] A. Benavides and M. Ritt. Two simple and effective heuristics for minimizing the makespan in non-permutation flow shops. *Computers & Operations Research*, 66:160–169, 2016.
[4] A. J. Benavides and M. Ritt. Fast heuristics for minimizing the makespan in non-permutation flow shops. *Computers & Operations Research*, 100:230–243, 2018.
[5] P. Brucker and P. Brucker. *Scheduling algorithms*, volume 3. Springer, 2007.
[6] M. Bultmann, S. Knust, and S. Waldherr. Synchronous flow shop scheduling with pliable jobs. *European Journal of Operational Research*, 270(3):943–956, 2018.
[7] B.-C. Choi, S.-H. Yoon, and S.-J. Chung. Minimizing maximum completion time in a proportionate flow shop with one machine of different speed. *European Journal of Operational Research*, 176(2):964–974, 2007.
[8] R. W. Conway, W. L. Maxwell, and L. W. Miller. *Theory of scheduling*. Courier Corporation, 1967.
[9] P. Erdős and G. Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.
[10] M. Garey, D. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1(2):117–129, 1976.
[11] A. Gharbi, M. Labidi, and M. Louly. The nonpermutation flowshop scheduling problem: Adjustment and bounding procedures. *Journal of Applied Mathematics*, 2014, 2014.
[12] R. Graham, E. Lawler, J. Lenstra, and A. Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

[13] J. James E. Kelley. Critical-path planning and scheduling: Mathematical basis. *Operations Research*, 9(3):296–320, 1961.
[14] S. M. Johnson. Optimal two-and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, 1(1):61–68, 1954.
[15] J. E. Kelley, Jr and M. R. Walker. Critical-path planning and scheduling. In *Papers Presented at the December 1-3, 1959, Eastern Joint IRE-AIEE-ACM Computer Conference*, IRE-AIEE-ACM '59 (Eastern), pages 160–173, New York, NY, USA, 1959. ACM.
[16] C.-J. Liao, L.-M. Liao, and C.-T. Tseng. A performance evaluation of permutation vs. non-permutation schedules in a flowshop. *International Journal of Production Research*, 44(20):4297–4309, 2006.
[17] V. Nagarajan and M. Sviridenko. Tight bounds for permutation flow shop scheduling. *Mathematics of Operations Research*, 34(2):417–427, 2009.
[18] E. Nowicki and C. Smutnicki. A fast tabu search algorithm for the permutation flow-shop problem. *European Journal of Operational Research*, 91(1):160–175, 1996.
[19] S. Panwalkar and C. Koulamas. On the dominance of permutation schedules for some ordered and proportionate flow shop problems. *Computers & Industrial Engineering*, 107:105–108, 2017.
[20] S. Panwalkar and C. Koulamas. The evolution of schematic representations of flow shop scheduling problems. *Journal of Scheduling*, pages 1–13, 2018.
[21] J. Pei, X. Liu, B. Liao, P. M. Pardalos, and M. Kong. Single-machine scheduling with learning effect and resource-dependent processing times in the serial-batching production. *Applied Mathematical Modelling*, 58:245–253, 2018.
[22] M. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer Science & Business Media, 2012.
[23] C. Potts, D. Shmoys, and D. Williamson. Permutation vs. non-permutation flow shop schedules. *Operations Research Letters*, 10(5):281–284, 1991.
[24] D. Rebaine. Flow shop vs. permutation shop with time delays. *Computers & Industrial Engineering*, 48(2):357–362, 2005.
[25] D. Rossit, F. Tohmé, and M. Frutos. The non-permutation flow-shop scheduling problem: a literature review. *Omega*, 77:143–153, 2018.
[26] D. Rossit, F. Tohmé, M. Frutos, J. Bard, and D. Broz. A non-permutation flowshop scheduling problem with lot streaming: a mathematical model. *International Journal of Industrial Engineering Computations*, 7(3):507–516, 2016.
[27] D. A. Rossit, Ó. C. Vásquez, F. Tohmé, M. Frutos, and M. D. Safe. The dominance flow shop scheduling problem. *Electronic Notes in Discrete Mathematics*, 69:21–28, 2018.
[28] D. Shabtay and K. Arviv. Optimal robot scheduling to minimize the makespan in a three-machine flow-shop environment with job-independent processing times. *Applied Mathematical Modelling*, 40(5-6):4231–4247, 2016.
[29] D. Shabtay and G. Steiner. A survey of scheduling with controllable processing times. *Discrete Applied Mathematics*, 155(13):1643–1666, 2007.
[30] D. Shabtay and M. Zofi. Single machine scheduling with controllable processing times and an unavailability period to minimize the makespan. *International Journal of Production Economics*, 198:191–200, 2018.
[31] A. Shioura, N. V. Shakhlevich, and V. A. Strusevich. Preemptive models of scheduling with controllable processing times and of scheduling with imprecise computation: A review of solution approaches. *European Journal of Operational Research*, 2017.
[32] W. Szwarc. Solution of the Akers-Friedman scheduling problem. *Operations Research*, 8(6):782–788, 1960.
[33] M. Tandon, P. Cummings, and M. LeVan. Flowshop sequencing with non-permutation schedules. *Computers & chemical engineering*, 15(8):601–607, 1991.
[34] J. Tilindis and V. Kleiza. Learning curve parameter estimation beyond traditional statistics. *Applied Mathematical Modelling*, 45:768–783, 2017.
[35] E. Vallada, R. Ruiz, and J. M. Framinan. New hard benchmark for flowshop scheduling problems minimising makespan. *European Journal of Operational Research*, 240(3):666–677, 2015.
[36] S. Waldherr and S. Knust. Decomposition algorithms for synchronous flow shop problems with additional resources and setup times. *European Journal of Operational Research*, 259(3):847–863, 2017.
[37] X.-R. Wang and J.-J. Wang. Single-machine scheduling with convex resource dependent processing times and deteriorating jobs. *Applied Mathematical Modelling*, 37(4):2388–2393, 2013.
[38] K.-C. Ying, J. Gupta, S.-W. Lin, and Z.-J. Lee. Permutation and non-permutation schedules for the flowline manufacturing cell with sequence dependent family setups. *International Journal of Production Research*, 48(8):2169–2184, 2010.