Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in DISCRETE MATHEMATICS

Electronic Notes in Discrete Mathematics 69 (2018) 21–28

www.elsevier.com/locate/endm

The Dominance Flow Shop Scheduling Problem [1]

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

[a] *Engineering Department, Universidad Nacional del Sur - CONICET, Bahía Blanca, Argentina*

[b] *Industrial Engineering Department, Universidad de Santiago de Chile, Santiago, Chile*

[c] *Economics Department, Universidad Nacional del Sur - CONICET, Bahía Blanca, Argentina*

[d] *Mathematics Department, Universidad Nacional del Sur, Bahía Blanca, Argentina*

Abstract

We introduce a new line of analysis of Flow Shop scheduling problems, for the case of two jobs and assuming that processing times are unknown. The goal is to determine the domination relations between permutation and non-permutation schedules. We analyze the structural and dominance properties that ensue in this setting, based on the *critical paths* of schedules.

Keywords: Scheduling, Non-permutation Flow Shop, Makespan, Critical Path.

1 Introduction

The research on Flow Shop scheduling was pioneered by Johnson in 1954 [4], who posed what is currently known as the classical Flow Shop problem, which amounts to schedule the processing of a set N of n jobs $j = 1, \ldots, n$ on a set

[1] E-mail addresses: daniel.rossit@uns.edu.ar (D.A. Rossit), oscar.vasquez@usach.cl (Ó.C. Vásquez), ftohme@criba.edu.ar (F. Tohmé), mfrutos@uns.edu.ar (M. Frutos), msafe@uns.edu.ar (M.D. Safe).

https://doi.org/10.1016/j.endm.2018.07.004

M of m machines $i = 1, \dots, m$. All the jobs are assumed to have the same processing sequence, $M_1, \dots, M_m$ and the operations are non-preemptive. Jobs are ready to start at the beginning of the programming and the processing times p_{ij} are positive and known. If jobs are processed in the same order on each machine, the problem is called Permutation Flow Shop (PFS), whereas, if they are not processed in the same order, the problem is deemed Non-Permutation Flow Shop (NPFS). Restricting the problem to the PFS setting implies that the cardinality of the set of feasible solutions is $n!$, while if NPFS schedules are allowed, the cardinality of feasible solutions grows up to $n!^m$. In his original contribution, Johnson detailed probably the most important result on PFS and NPFS problems with the makespan objective, stating that in the optimal solution "the first two machines have the same job ordering, as well as the last two machines". This result was generalized by Conway et al. [3] (Theorems 5–1 and 5–2).

Since then, many contributions to the comparison between the PFS and NPFS approaches have been published. So, for instance, Potts et al. [6] worked on the Flow Shop with missing operations (or zero processing times), and showed that for a family of instances the best permutation schedule is worse than $1/2\sqrt{m}$ times the best NPFS solution. Choi et al. [2], studied the $F|p_{ij} = p_j/s_i|C_{\max}$ instance, where s_i is the processing speed of the machine i. Choi et al. proved that for $m \geq 4$ PFS schedules are optimal if there exists a maximal machine, i.e. $s_k < s_1 = \cdots = s_m$ for some machine M_k. More recently, Panwalkar and Koulamas [5] addressed the ordered Flow Shop problem, this time considering that the processing times meet certain conditions such that for any generic machine, denoted t, the processing times can be ordered in a decreasing order, i.e., $p_{ti} > p_{tl}$, among other considerations. Furthermore, these authors proved that PFS is optimal if the p_{ij}s are increasing on the succesive machines, for every fixed job j (see Lemma 1 therein). For a more extensive review on the NPFS literature visit [7].

All these works study the PFS and NPFS problems seeking conditions ensuring that PFS schedules are optimal (and therefore, dominant). A common feature of these papers is the imposition of strong conditions on p_{ij}. 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 modelled in this way. In order to tackle this shortcoming, we approach the problem lifting all the conditions on the p_{ij}s. Furthermore, we model the processing times as being independent of the parametrization used for each specific problem. This approach allows us to work with graphic descriptions and with the structure of the critical paths of

PFS and NPFS schedules. Using these structures we can find NPFS schedules that are dominated by PFS schedules regardless the value of p_{ij}. We show that this is the case by means of a combinatorial analysis of the problem.

2 Problem Definition

We focus on Flow Shop problems with $n = 2$ jobs and m machines, with unknown processing times. Notice that by specifying m, a problem instance $\mathcal{I}$ is also specified. To the best of our knowledge, this is a novel setting for the comparison between PFS and NPFS schedules. It requires the development of new tools to make our solutions independent of the exact values of the processing times.

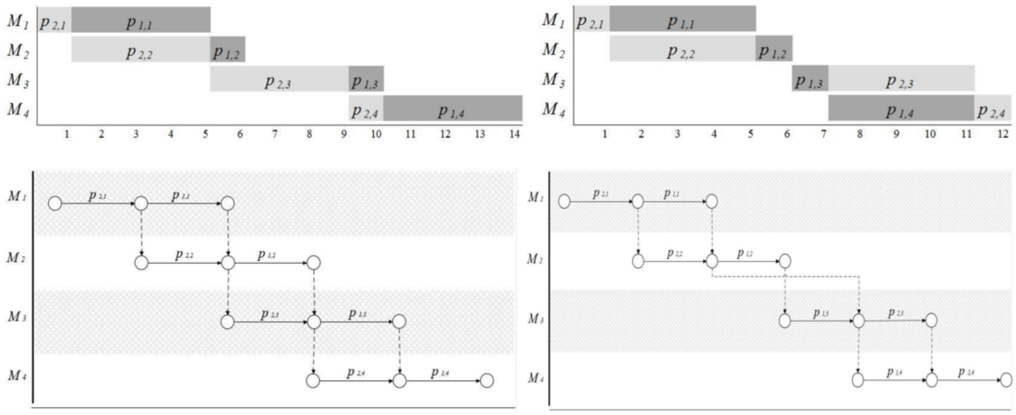

Fig. 1. Gantt and graph representations of the optimal schedules of the PFS problem (top-bottom, left) and the NPFS problem (top-bottom, right) for an instance $\mathcal{I}$ with $n = 2$ jobs and $m = 4$ machines. For this instance, the makespan value of optimal schedules of the PFS and NPFS are 14 and 12, respectively.

To build the intuition for our results, consider Figure 1, in which we present two Gantt diagrams and their corresponding graph-theoretical representations. The former correspond to a numerical example in which the processing times are in fact known, with $n = 2$ jobs and $m = 4$ machines. On the left we present the PFS optimal solution, with a makespan of 14, while on the right side we show the optimal NPFS solution yielding a makespan of 12. The pictures in the bottom panels of Figure 1 show the graphs of the optimal schedules, obtained according to the following definition:

Definition 1 ***Graph representation***. *Consider $G^{\sigma}(V, A, P)$, where V 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*, if solid edges in* A *represent the processing states of the machines while dashed edges represent precedence constraints on jobs. 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 have empty 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 need the following definition:

Definition 2 ***Switching Machine***.
A machine M_k *is said a* switching machine *if it reverses the order in which jobs are processed in the previous machine.*

Note that machines M_1, M_2 and M_m cannot be switching machines in any optimal solution due to the theorems of Conway et al.[3]. Hence, in any optimal solution, there are at most $m - 3$ switching machines.

A feasible schedule σ without switching machines is PFS while if it includes some, it corresponds to the NPFS case. The class of switching machines in a feasible schedule is denoted $S_\sigma \subseteq M$. For convenience, we denote with the capital Greek letter, Σ_a, the set of feasible schedules with exactly a switching machines; i.e., $\sigma \in \Sigma_a$ if and only if $|S_\sigma| = a$. Let $\sigma_0^* := \arg\min_{\sigma \in \Sigma_0} F(\sigma)$ where $F(\sigma)$ denotes the makespan of σ.

In Figure 1 the bottom panels depict the graph representations of the Gantt diagrams in the top panels. The NPFS graph corresponds to a schedule in Σ_1, i.e., having a single switching machine (machine M_3).

In general, the graph representation of scheduling problems allows to characterize the makespan (see [1]). In our work, we characterize the makespan defining a particular critical path, which may not be unique.

Definition 3 ***Flow Shop Critical Path***. *Given a feasible* σ*, a path in* $G^\sigma(V, A, P)$ *is said critical if it includes at least one operation on each machine, and supports the makespan of the graph.*

The solid edges of PFS and NPFS graphs in Figure 1 are labeled by operations. These labels allow to describe the critical paths in terms of operations. Even if the processing times are unknown, we do not assume that they are all equal. Thus, each operation has its own different label. It is easy to see 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 the 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. Thus, for instance, we can see in Figure 1 that the critical paths of the PFS graph are different from those of the NPFS graph.

3 Main Results

In this section, we consider a given instance $\mathcal{I}$ and 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 3.1 *The length 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 a 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 $G^\sigma(A, V, P)$ with $|S_\sigma| = 1$ and separate it into two sub-graphs such that the job of last operation performed in one sub-graph is the same job of the first operation in the other sub-graph. A switching machine, M_k, defines this split into 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, and the length is $m + 2$ which verifies $m + 1 + |S_\sigma|$ 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 performed in one sub-graph is the same job of the first operation in the other sub-graph. On the other hand, 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 k-th machine, M_k, defines the split into 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 $(m-k+1)+1$ (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. □

According to Theorem 3.1, each possible NPFS critical path has more operations than each possible PFS critical path. Let us call a the number of those 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 a NPFS critical path may end up including all the operations in a PFS critical path. Then, 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 longest 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 makespan of both critical paths coincide. Otherwise, the PFS makespan would be better than the makespan of the NPFS schedule. We capture this notion as follows:

Definition 4 ***Dominance.*** *A feasible NPFS schedule σ with $|S_\sigma| > 0$ is* dominated by permutation *if $F(\sigma) \geq F(\sigma_0^*)$, where $F(\sigma)$ and $F(\sigma_0^*)$ represent the makespans of σ and σ_0^* respectively.*

Proposition 1 *Given an instance $\mathcal{I}$, if the number of machines m is odd, each critical path of $G^{\pi}(V, A, P)$ with $|S_\pi| = m-3$ includes all the operations from some critical path of $G^{\sigma_0^*}(V, A, P)$.*

Proof. Consider a schedule π with $|S_\pi| = m-3$. Since we assume two jobs, the universe of possible operations has size $2m$. According to Theorem 3.1, in the critical path in $G^{\pi}(V, A, P)$, the number of operations included is $m+1+m-3 = 2m-2$. That means that $G^{\pi}(V, A, P)$ has all the possible operations except two of them. On the other hand, consider the schedule σ_0^*. Again by Theorem 3.1, the critical path in $G^{\sigma_0^*}(V, A, P)$ has $m+1$ operations (since the set of its switching machines is empty). If we show that the two operations that are not included in the critical path of $G^{\pi}(V, A, P)$ are not included in the critical path $G^{\sigma_0^*}(V, A, P)$, it follows that the former includes all the operations of the latter plus some more. Then, the makespan of schedule σ_0^* is shorter than that of π.

To see this point let us assume, without loss of generality, that the first operation of schedule π and that of schedule σ_0^* are the same, and then, since the number of switching machines, $m-3$, is even (and thus the last sub-graph in the graph representation features the same order as in the first one), the last operation must also the same for both of them. In addition, we have the same job ordering in the first two machines and in the last two machines, according to [3]. This implies, since the first machine carries out the same operation in both graphs, that the critical path of $G^{\pi}(V, A, P)$ has the same two first operations as $G^{\sigma_0^*}(V, A, P)$ plus one more, but not all the four possibilities on the two machines. Then, the operation that is not included in the critical path of $G^{\pi}(V, A, P)$ (which by the previous argument includes three of the four operations) is also not included in the critical path of $G^{\sigma_0^*}(V, A, P)$. A similar analysis can be carried out for the last two machines. Then, we can assert that the critical path of $G^{\pi}(V, A, P)$ includes the same operations included in the critical path of $G^{\sigma_0^*}(V, A, P)$, while the operations discarded from the critical path of $G^{\pi}(V, A, P)$ are also discarded from the critical path of $G^{\sigma_0^*}(V, A, P)$.□

From Proposition 1, we immediately have the following:

Proposition 2 *Given an instance $\mathcal{I}$, if the number of machines m is odd, a feasible schedule σ with $|S| = m - 3$ is dominated by permutation.*

Proposition 3 *Given an instance $\mathcal{I}$, if the number of machines m is even, each critical path of $G^{\pi}(V, A, P)$ with $|S_\pi| = m - 3$ has all the operations, except at most one operation, of those in some critical path of $G^{\sigma_0^*}(V, A, P)$.*

Proof. The proof is similar to that of Proposition 1. Consider a feasible schedule π with $|S_\pi| = m - 3$, where the total number of operations is $2m$. According to Theorem 3.1, the critical path of $G^{\pi}(V, A, P)$ has $2m - 2$ operations. Without loss of generality, we assume that the first operation in schedule π and in schedule σ_0 is the same, and then, by Theorem 3.1, they have the same job ordering in the first two machines . Since m is even and thus $|S_\pi| = m-1$ is odd, schedules π and σ_0^* do not have the same job orderings on the last two machines. On these last machines the critical path of $G^{\pi}(V, A, P)$ takes $2m - 4$ operations from a total of $2m - 3$ operations. Therefore, the critical path of $G^{\pi}(V, A, P)$ has at most one operation that differs from those in the critical path of $G^{\sigma_0^*}(V, A, P)$. On the rest of the machines, the critical path of $G^{\pi}(V, A, P)$ includes all the possible operations. Thus, the critical path of $G^{\pi}(V, A, P)$ has at most one operation that differs from those in the critical path of $G^{\sigma_0^*}(V, A, P)$. □

Consider p_{max} and p_{min} as the maximum and minimum of p_{ij}, respectively.

Proposition 4 *Given an instance $\mathcal{I}$, if the number of machines m is even and $p_{max} \leq (m - 3)p_{min}$, a feasible schedule π, with $|S_\pi| = m - 3$, is not optimal.*

Proof. Recall that π has $m-3$ more operations than any PFS critical path, according to Theorem 3.1. Thus, from Proposition 3 a PFS critical path has at most one operation not included in any NPFS critical path. The worst case is when the missing operation from the PFS critical path is p_{max} and the $m-3$ extra operations from the NPFS are p_{min}. Then, if the p_{max} is less or equal than $m-3$ times p_{min}, the makespan of the NPFS critical path is larger or equal to the makespan of the PFS critical path. In any other case PFS yields a better makespan than NPFS. □

4 Final Remaks

In this work we analyzed a variant of the Flow Shop problem in a two-jobs setting, independently of the processing times on machines. We identified NPFS solutions that are dominated by PFS schedules. This assessment is based on the consideration of structural and dominance properties of the critical paths of PFS and NPFS schedules. Extending this analysis to three or more jobs is matter of further work.

References

[1] P. Brucker. *Scheduling algorithms.* Springer, Berlin, fifth edition, 2007.

[2] 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.

[3] R. W. Conway, W. L. Maxwell, and L. W. Miller. *Theory of scheduling.* Addison-Wesley, Reading, Mass., 1967.

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

[5] 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.

[6] C. Potts, D. Shmoys, and D. Williamson. Permutation vs. non-permutation flow shop schedules. *Operations Research Letters*, 10(5):281–284, 1991.

[7] D. Rossit, F. Tohmé, and M. Frutos. The non-permutation flow-shop scheduling problem: a literature review. *Omega*, 77:143–153, 2018.