Relationship between common objective functions, idle time and waiting time in permutation flow shop scheduling[1]

Kathrin Maassen[1], Paz Perez-Gonzalez[2], Lisa C. Günther[3]

[1]*University of Duisburg-Essen, Chair of Business Administration and Production Management, Bismarckstr. 90, 47057 Duisburg, Germany*

[2]*University of Seville, School of Engineering, Industrial Management, Camino de los Descubrimientos, s/n, 41092 Sevilla, Spain*

[3]*Fraunhofer Institute for Manufacturing Engineering and Automation IPA, Nobelstrasse 12, 70569 Stuttgart, Germany*

Abstract

Different objective functions have been discussed in static-deterministic flow shop scheduling, being makespan (C_{max}) and total completion time ($\sum C_j$) the most studied objectives. Many scientific papers also dealt with idle time of machines and waiting time of jobs in constraints and objective functions, because both indicators are of high practical relevance, e.g. in steel production or considering expensive machines. The efficiency of a production system can also be evaluated by minimizing idle time and waiting time within the production process (called core idle time, $\sum CIT_i$, and core waiting time, $\sum CWT_j$), because they refer to a high machine utilization and a continuous job flow. Moreover, both objectives can be used to relax the constraints of zero waiting time (no-wait) or zero idle time (no-idle) in shop floors and hence, the alignment between $\sum CIT_i$ and $\sum CWT_j$ and classical objectives like C_{max} and $\sum C_j$ has to be discussed. We start analytically with special cases where $\sum CWT_j$ and $\sum C_j$, as well as $\sum CIT_i$ and C_{max}, are identical objectives. For the general case, an experimental study is executed by means of a set of instances solved by complete enumeration, mixed integer linear programming (MILP) models for single objective and multi-objective (lexicographical) approaches. It can bee seen that the objectives are not identical but at least aligned with varying intensity, e.g. the alignment between $\sum CWT_j$ and $\sum C_j$ is stronger compared to the alignment between $\sum CIT_i$ and C_{max}. The shown properties and results open a new line of problems in permutation flow shop scheduling and shall be further discussed by developing efficient heuristic approaches for increased problem sizes.

[1]This is the Submmited Manuscript of an article published by Elsevier in Computers and Operations Research 121, September 2020 104965 available online: https://doi.org/10.1016/j.cor.2020.104965

Keywords: Permutation Flow Shop, flowshop, total completion time, makespan, waiting time, idle time

1. Introduction

In flow shop scheduling many different objective functions have been considered so far. The most studied objectives are the minimization of makespan (see e.g. the reviews of Framinan et al. (2004), Ruiz and Maroto (2005), Reza Hejazi and Saghafian (2005), Fernandez-Viagas et al. (2017)) and total completion time (see e.g. the surveys of Framinan et al. (2005), Pan and Ruiz (2013)). Makespan (C_{max}) is defined as the completion time of the last job on the last machine. While C_{max} can be solved optimally in a two-machine permutation flow shop by Johnson's algorithm (Johnson (1954)), it was proved to be NP-complete for more than two machines by Garey et al. (1976). Total completion time ($\sum C_j$) is the sum of completion times of all jobs on the last machine and was proved to be NP-complete for two machines by Garey et al. (1976).

Besides C_{max} and $\sum C_j$, there are also other time measures which are important in flow shop scheduling, e.g. waiting time of jobs and idle time of machines are often considered because of their impact on many manufacturing environments and applications. Waiting time has a large influence e.g. in steel-production, where it is not only a waste of time but also a waste of raw material and resources because molten steel can only be processed within a certain range of temperature until the material becomes unusable for production, see e.g. Hall and Sriskandarajah (1996). In this context, the no-wait constraint is often applied which means that jobs are not allowed to wait within the production process. Idle time is an important factor e.g. when expensive machines are part of the considered production layout so that a high utilization is required to guarantee profitability.

In shop layouts, waiting time and idle time can be used to evaluate the efficiency of a production system. E.g. a continuous job flow through the system is an efficiency-oriented objective which means actually that the waiting time of jobs should be minimized because any waiting situation interrupts a continuous flow. This objective is often represented by minimizing $\sum C_j$. However, the minimization of waiting time within the system (called core waiting time, $\sum CWT_j$) is a measure of job flow. The utilization of machines is also one key factor referring to efficient production systems. To achieve highly utilized machines, the minimization of C_{max} is often used

but also the minimization of idle time within the system (called core idle time, $\sum CIT_i$) is a measure of machine utilization. In this context $\sum CWT_j$ and $\sum CIT_i$ are defined for shop floors, and in this work we focus on the permutation flow shop as the easiest shop layout.

In the literature, no-wait and no-idle constraints are considered for problems minimizing C_{max} mainly, see e.g. Allahverdi (2016), Ruiz et al. (2009), being both assumptions strong constraints in scheduling. An interesting approach might be relaxed them by considering $\sum CWT_j$ or $\sum CIT_i$ respectively. As, in the most of the cases, decision makers are interested in the minimization of C_{max} and $\sum C_j$, the alignment among these objectives and the proposed objectives, $\sum CWT_j$ and $\sum CIT_i$, is a first step to relax the no-wait and no-idle constraints. To the best of our knowledge, this alignment has not been considered in the literature, so this work focuses in the analysis of the relationship among the four aforementioned objectives. First, for some special cases, we show that either $\sum CWT_j$ and $\sum C_j$ or $\sum CIT_i$ and C_{max} are identical, e.g. $\sum CWT_j$ and $\sum C_j$ are equivalent if the processing times of all jobs on the first machine are the same. Additionally, the general case is analyzed experimentally and reveals e.g. that the alignment between $\sum CWT_j$ and $\sum C_j$ is stronger compared to the alignment between $\sum CIT_i$ and C_{max}.

Section 2 provides an overview about the different components of idle time and waiting time in a permutation flow shop and explains their formal definitions and practical relevance. Corresponding literature is reviewed in Section 3. The theoretical equivalence of objectives in some special cases are presented in Section 4. Section 5 deals with different methods to analyze experimentally the relationship between objective functions. The paper finishes with conclusions and suggestions for future research lines.

2. Notation and concepts of idle time and waiting time

The relevant notation, Table 1, and the static-deterministic permutation flow shop problem (PFSP) are introduced to provide the basis for explaining the formal and practical relevance of different idle time and waiting time components. A PFSP with following assumptions is considered: n jobs have to be scheduled on m machines which are arranged in series. All jobs and machines are available at time zero. Preemption of jobs is not allowed. The processing times, $p_{i,j}$, of each job j on each machine i are known in advance ($P = \sum_{i=1}^{m} \sum_{j=1}^{n} p_{i,j}$). The sequence of jobs stays the same on all machines and cannot be changed (permutation assumption, $prmu$). Moreover, the

$\alpha|\beta|\gamma$-notation provided by Graham et al. (1979) is used, where α defines the machine layout, β explains the process constraints and γ shows the objective function.

Indexes:	
i	Index of machines, $1 \leq i \leq m$
j	Index of jobs, $1 \leq j \leq n$
$[k]$	Index of job position, $1 \leq k \leq n$
Parameter:	
$p_{i,j}/p_{i,[k]}$	Processing times of job j on machine i / Processing times of job in position k on machine i, $1 \leq i \leq m, 1 \leq k \leq n, 1 \leq j \leq n$
Variables:	
$B_{i,[k]}$	Starting time of job in position k on machine i, $1 \leq i \leq m, 1 \leq k \leq n$
BIT_i	Back idle time of machine i after processing job in position n, $1 \leq i \leq m$
$BWT_{[k]}$	Back waiting time of job in position k after being processed on machine m, $1 \leq k \leq n$
$CIT_{i,[k]}$	Core idle time of machine i before being processed of job in position k, $2 \leq i \leq m, 2 \leq k \leq n$
$CWT_{i,[k]}$	Core waiting time of job in position k before being processed on machine i, $2 \leq i \leq m, 2 \leq k \leq n$
$C_{i,[k]}$	Completion time of job in position k on machine i, $1 \leq i \leq m, 1 \leq k \leq n$
FIT_i	Front idle time of machine i before being processed of job in the first position $1 \leq i \leq m$
$FWT_{[k]}$	Front waiting time of job in position k before being processed on machine 1, $1 \leq k \leq n$
$X_{j,[k]}$	$\begin{cases} 1, \text{ if job } j \text{ is assigned to position } [k] \\ 0, \text{ in all other cases} \end{cases}$ $1 \leq k \leq n, 1 \leq j \leq n$

Table 1: Notation and definitions

2.1. Formal definition of waiting time and idle time

Framinan et al. (2003) and Fernandez-Viagas and Framinan (2014) stated that idle time has been defined in different ways so far. According to this and also referring to waiting time, we first focus on the different components of idle time and waiting time depending on where they could occur in the production process which is the basis to discuss the practical relevance. Regarding waiting times, it can be considered:

- Front Waiting Time of job in position k, $FWT_{[k]}, 2 \leq k \leq n$: the time before the job is processed on the first machine,
- Core Waiting Time of job in position k of machine i, $CWT_{i,[k]}, 2 \leq i \leq m, 2 \leq k \leq n$: the time period between the start of job in position k on machine i and the completion time on the previous one $i-1$. If job j occupies position k then $CWT_j = \sum_{i=2}^{m} CWT_{i,[k]}$.
- Back Waiting Time of job in position k, $BWT_{[k]}, 1 \leq k \leq n-1$: is the time after the job has finished its last operation but before the overall schedule is finished.

In the same way for the idle times, we have:

- Front Idle Time of machine i, $FIT_i, 2 \leq i \leq m$: idle time before a machine processes the first job,
- Core Idle Time of job in position k of machine i, $CIT_{i,[k]}, 2 \leq i \leq m, 2 \leq k \leq n$: the time period between the start of job in position k and the completion time of the previous job on machine i. For a machine i, $CIT_i = \sum_{k=1}^{n} CIT_{i,[k]}$.
- Back Idle Time of machine i, $BIT_i, 1 \leq i \leq m-1$: time after the machine has finished its last operation but before the overall schedule is finished.

In Figure 1 an example is given to visualize the afore mentioned components. The machine-oriented Gantt-chart (left) shows the different forms of idle time, while the components of waiting time are given in the job-oriented Gantt-chart (right). It can be seen that if $B_{i,[k]} = C_{i-1,[k]}$ no core waiting time exists and the job is directly processed on the next machine (see e.g. job 2 in Figure 1). $CIT_{i,[k]} = 0$ if $B_{i,[k]} = C_{i,[k-1]}$ (see e.g job 3 in machine 2 in Figure 1).

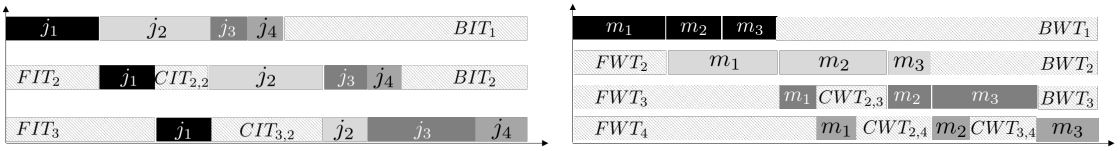

Figure 1: left: Machine-oriented Gantt-chart; right: Job-oriented Gantt-chart

The formal definitions of all components related to waiting time and idle time and their sum-form are described in Table 2, as well as the formal definitions of C_{max} and $\sum C_j$.

2.2. Practical relevance of waiting time and idle time

After explaining idle time and waiting time formally, it is also essential to discuss their practical importance. One of the main goals in production management is the elimination of all kinds of waste to obtain an efficient system. Among other things, especially waste of time is interesting in scheduling since most of the objectives are time-related. Referring to idle time, we assume that the time before production starts, i.e. FIT_i, and after production has finished, i.e. $\sum BIT_i$, can be used for other necessary but non-value added tasks, like service processes, setups or cleaning procedures. $\sum FIT_i$ only depends on the first scheduled job, see Table 2. $\sum BIT_i$ is influenced by the completion times of the last scheduled job, but a lower bound can be obtained in terms of

Abbr.	Objective		
$\sum FIT_i$	Total front idle time	$\sum_{i=1}^{m} FIT_i$	$\sum_{i=2}^{m} \sum_{i'=1}^{i-1} p_{i',[1]}$
$\sum FWT_j$	Total front waiting time	$\sum_{k=1}^{n} FWT_{[k]}$	$\sum_{j=1}^{n-1} \sum_{k=1}^{j} p_{1,[k]}$
$\sum BIT_i$	Total back idle time	$\sum_{i=1}^{m} BIT_i$	$m \cdot C_{m,[n]} - \sum_{i=1}^{m} C_{i,[n]}$
$\sum BWT_j$	Total back waiting time	$\sum_{k=1}^{n} BWT_{[k]}$	$n \cdot C_{m,[n]} - \sum_{k=1}^{n} C_{m,[k]}$
$\boldsymbol{\sum CIT_i}$	**Total core idle time**	$\boldsymbol{\sum_{i=2}^{m} \sum_{k=2}^{n} CIT_{i,[k]}}$	$\boldsymbol{\sum_{i=2}^{m} \sum_{k=2}^{n} B_{i,[k]} - C_{i,[k-1]}}$ **(1)**
$\boldsymbol{\sum CWT_j}$	**Total core waiting time**	$\boldsymbol{\sum_{i=2}^{m} \sum_{k=2}^{n} CWT_{i,[k]}}$	$\boldsymbol{\sum_{i=2}^{m} \sum_{k=2}^{n} B_{i,[k]} - C_{i-1,[k]}}$ **(2)**
$\boldsymbol{\sum C_j}$	**Total completion time**	$\boldsymbol{\sum_{k=1}^{n} C_{m,[k]}}$ **(3)**	
$\boldsymbol{C_{max}}$	**Makespan**	$\boldsymbol{C_{m,[n]}}$ **(4)**	

Table 2: Formal definitions of objective functions (bold: Objectives further discussed in Sec. 4 - 5)

the processing times of last scheduled job since $\sum BIT_i \geq \sum_{i=1}^{m-1} \sum_{i'=i+1}^{m} p_{i',[n]}$. However, core idle time cannot be estimated in the same manner since its occurrence and length hardly depends on the schedule. Instead, it can be interpreted as an indicator of machine utilization because it indicates that the machine is in a stand-by mode and waiting for work and the minimization leads to an efficient production system (see also Liu et al., 2016).

Considering waiting time, there are also components which could be used efficiently for other tasks. E.g. when a production system consists of multiple sub-systems, front waiting time and back waiting time can be used to transport jobs from one production step to the next one or directly to the customer. So, $\sum FWT_j$ and $\sum BWT_j$ can be used for non-value added tasks in the same way as $\sum FIT_i$ and $\sum BIT_i$. Here, $\sum CWT_j$ can be seen as an indicator of job flow because as soon as jobs have to wait within the process the flow is interrupted, i.e. the highest flow can be achieved when each job is moved through the system without any waiting time.

Related to waiting and idle time, it can be seen that in many practical applications, no-wait schedules or no-idle schedules are required, e.g. in steel industry, food industry or when expensive machines are required. These constraints are mostly combined with the minimization of C_{max} or $\sum C_j$. However, in some practical applications a strict no-wait or no-idle constraint might be

relaxed by $\sum CWT_j$ and $\sum CIT_i$, respectively. E.g. in food industries, cold chains have to be maintained, i.e. waiting time of cooled and frozen food is not allowed and hence, it refers to a no-wait assumption. Nevertheless, in determined conditions the cold chain can be maintained allowing waiting times and so, the minimization of $\sum CWT_j$ instead of the no-wait constraint might be applicable.

2.3. Consequences of semi-active schedules

In this paper we focuses on the permutation flowshop scheduling problem considering semi-active schedules, i.e. the jobs start as early as possible on each machine so that they seem to be left-shifted (see e.g. Pinedo, 2016). Hence, semi-active schedules ensure that the first machine is never idle and the first job never has to wait. Applying the semi-active assumption leads to a reduction of total number of feasible schedules from infinity to $n!$ because each sequence only provides one semi-active schedule for each problem instance. For this reason, scheduling literature often considers only semi-active schedules, which is also assumed here.

Note that $Fm|prmu|\sum CIT_i$ and $Fm|prmu|\sum CWT_j$ are both trivial without this consideration, since if right-shifting of operations is allowed, for each sequence an optimal schedule can be found (as it can be seen in the Fig. 2). Additionally, literature about $Fm|prmu|\sum C_j$ and $Fm|prmu|C_{max}$ considers semi-active schedules which leads to values of $\sum CWT_j$ and $\sum CIT_i$ greater than zero (see Figure 2 a)). Under this consideration, it can be shown that both problems, $Fm|prmu|\sum CIT_i$ and $Fm|prmu|\sum CWT_j$, are NP-complete using a similar reduction to the 3-partition-problem which was used for the proof of NP-completeness of C_{max} and $\sum C_j$ by Garey et al. (1976). Therefore, the semi-active assumption changes the problem of minimizing $\sum CIT_i$ and $\sum CWT_j$ from trivial to NP-hard.

Summing up, idle time and waiting time can be divided in different components depending on their occurrence in the production process. Assuming that front and back components could be used for other tasks, the discussion leads to $\sum CIT_i$ and $\sum CWT_j$ as indicators for efficient production systems and as relaxations of no-wait and no-idle constraints. While $\sum CIT_i$ refers to the utilization of machines, $\sum CWT_j$ evaluates the job flow through the system. Assuming semi-active schedules, the problems considering $\sum CWT_J$ and $\sum CIT_I$ are NP-hard, and to analyse the relationship among these new objectives and the classical objectives is an open issue as it is shown in the literature reviewed in the following section.

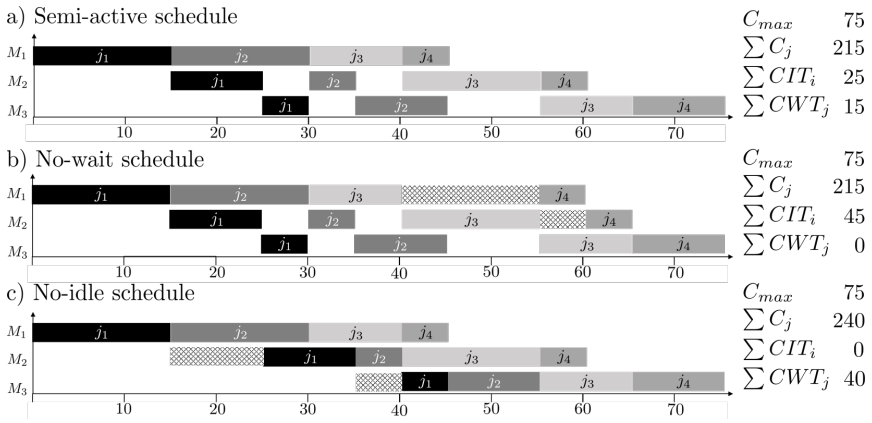

Figure 2: Example of semi-active schedule and non semi-active schedules (under non aditional contraints)

3. Literature review

The discussion about waiting time and idle time is not new because these measures are not only interesting in theoretical scheduling models but are also relevant in many manufacturing environments. These concepts can be found as objective functions (see Table 3), in heuristics methods (see Table 4), and in constraint (see Table 5). In this section, we focus the discussion about waiting time and idle time in objective functions, since this case is directly related to our research work. An extensive review has been carried out including different machine layouts. We do not explain all machine environments and constraints in detail and refer the interested reader to the cited scientific paper. and summarize the reviewed literature.

Merten and Muller (1972) discussed the relationship between completion time and front waiting time by analyzing their mean and variance values in a single machine layout. They showed that minimizing $\overline{\sum C_j}$ or $\overline{FWT}$ leads to the same optimal sequence, while the minimization of completion time variance, $\frac{1}{n}\sum_{j=1}^{n}(C_j - \overline{C_j})^2$, and FWT-variance, $\frac{1}{n}\sum_{k=1}^{n}(FWT_{[k]} - \overline{FWT})^2$, have the same objective value. Eilon and Chowdhury (1977) discussed the objective of minimizing FWT-variance, too, and showed that an optimal schedule has to be V-shaped referring to the processing times of jobs. V-shaped means that the job with the smallest processing time is set in the middle of the schedule, while all jobs in front of this job are sorted in descending order and all jobs after this job in ascending order. Bagchi (1989) dealt with several objectives referring to the variation of front waiting time and completion time, e.g. an optimal algorithm for the total absolute difference of front waiting times, $\sum_{k=1}^{n}\sum_{k'=1}^{n}|FWT_{[k']} - FWT_{[k]}|$, was provided.

Among other things, a bi-objective problem of $\sum FWT_j$ and $\sum_{k=1}^{n}\sum_{k'=1}^{n}|FWT_{[k']} - FWT_{[k]}|$ has been discussed and similarities between completion time and waiting time have been identified. Li et al. (2007) proposed a discussion about the influencing factors when focusing on $\frac{1}{n}\sum_{k=1}^{n}(FWT_{[k]} - \overline{FWT})^2$ in a single machine environment, concluding that the statistical parameters mean and standard deviation of the used processing time distribution makes a strong contribution to FWT-variance. Xu (2011) dealt with the single machine problem and the objective of minimizing the weighted waiting time variance, $\sum_{j=1}^{n} w_j(FWT_j - \overline{FWT})^2$. The author showed that the problem $1||\sum_{j=1}^{n} w_j(FWT_j - \overline{FWT})^2$ is optimally solved by sorting the jobs in a V-shaped manner regarding the measure p_j/w_j, where w_j refers to the weight of job j, if jobs are agreeably weighted, i.e. $p_{j'} > p_j \implies w_{j'} < w_j \quad \forall\, j', j \in \{1, 2, ..., n\}$. Moreover, the dispatching rule 'Shortest Processing Times' can be used for an optimal solution if p_j and w_j of a job are directly proportional. De Matta (2019) discussed the problem of minimizing $\sum CWT_j$ in a two-machine permutation flow shop. The author reduced this problem to a single machine environment and added the requirements of batch-setups and deadlines $\overline{d_j}$ of each job ($1|setup, \overline{d_j}|\sum CWT_j$).

In most cases, papers considering idle time in the objective function refer to the total machine completion time, $\sum_{i=1}^{m} C_i$. Ho and Gupta (1995) discussed two different sets of dominated machines and among other objectives, provided simple and efficient solution methods for $\sum C_i$. Fondrevelle et al. (2008) discussed the minimization of weighted sum of machine completion times with minimal and maximal time lags, $Fm|prmu, wt_{i,j}^{max}, wt_{i,j}^{min}|\sum w_i C_i$. They proved that the problem is NP-hard and provided a branch-and-bound based exact algorithm. In their conclusion, they suggested to discuss the minimization of $\sum CIT_i$. Ruiz-Torres et al. (2011) considered the problem of minimizing C_{max} and $\sum C_i$ with assignment flexibility of tasks, i.e. there are more tasks per job to be processed than machines, but each machine can process each task, $Fm|prmu, flex|\sum C_i$ and $Fm|prmu, flex|C_{max}$. They concluded that there is no solution found which simultaneously minimizes both objectives, i.e. a good solution for C_{max} is also a good sequence for $\sum C_i$ but a good solution for $\sum C_i$ leads to a poor C_{max}. Finally, Liu et al. (2014) considered specifically the objective $\sum CIT_i$ proposing a NEH-based heuristic.

Additionally, some multi-criteria approaches have been considered in the literature regarding idle time objective functions. The sum of $\sum FIT_i$ and $\sum CIT_i$ was considered by Yagmahan and Yenisey (2008) and Sha and Lin (2009), while Liao et al. (2007) only referred to $\sum CIT_i$. Hosseini and Tavakkoli-Moghaddam (2013) proposed two meta-heuristics for the two-machine flow shop

with dynamic arrivals and including learning effects. The multi-objective function consists of the minimization of mean deviation from a common due date and $\sum CIT_i$. In their paper, $\sum CWT_j$ is calculated within the computational experiments but is not considered directly in the objective function. Recently, Liu et al. (2016) modified the well known NEH-heuristic (see Nawaz et al., 1983) for a bi-objective approach of minimizing C_{max} and $\sum CIT_i$.

Summarizing, idle time and waiting time have been discussed in the scheduling literature, but $\sum CIT_i$ and $\sum CWT_j$ have only been rarely discussed as objective functions, and, in particular, the relationship to each other or to C_{max} and $\sum C_j$ has not been considered.

4. Basic observations

As explained in Section 1, $\sum CIT_i$ and $\sum CWT_j$ are efficiency-oriented objectives whose minimization leads to improved utilization and job flow, and they may be used as relaxation of no-wait and no-idle constraints. Therefore, we have observed the interest about to analyse the alignment among $\sum CIT_i$ and $\sum CWT_j$ and the classical objectives C_{max} and $\sum C_j$. Therefore, from an theoretical point of view, the relationship is analysed for some special cases regarding processing times, showing that C_{max} and $\sum CIT_i$, as well as $\sum C_j$ and $\sum CWT_j$ are equivalent. Moreover, we show that the semi-active assumption is fulfilled as long as the no-idle constraint applies to the first machine and the no-wait constraint applies to the first job. This result will be useful for the experimental analysis carried out for the general case in Section 5

4.1. Semi-active schedule assumption

As described in Section 2.3, in the permutation flowshop context, the semi-active schedule verifies that the first machine is never idle and the first job never has to wait within the process. However, these two constraints do not guarantee semi-active schedules in the search space (i.e. the optimal schedule can have some right-shifted operations). Observation 1 proves that in the subspace of all schedules verifying that the first machine is no-idle and the first job is no-wait, the optimal value of $\sum CIT_i$ (and $\sum CWT_j$) is provided by the semi-active schedule.

Observation 1. *Let $\mathcal{S}^\pi$ be the set of all schedules constructed for a given sequence π. Let $\mathcal{S}_c^\pi \subseteq \mathcal{S}^\pi$ be the set of schedules verifying that the first machine is no-idle and the first job is no-wait. Let $S_s \in \mathcal{S}_c^\pi$ be the semi-active schedule for π. Then,*

$\sum CIT_i(S_s) \leq \sum CIT_i(S)\ \forall S \in \mathcal{S}_c^\pi, S \neq S_s$ *and* $\sum CWT_j(S_s) \leq \sum CWT_j(S)\ \forall S \in \mathcal{S}_c^\pi, S \neq S_s$.

α	β	γ	*Results*	*Author*
1	-	$\frac{1}{n}\sum_{k=1}^{n}(FWT_{[k]}-\overline{FWT})^2$	Optimally solved	Eilon and Chowdhury (1977)
1	-	$\sum_{k=1}^{n}\sum_{k'=1}^{n}\lvert FWT_{[k']}-FWT_{[k]}\rvert$	Optimally solved	Bagchi (1989)
1	-	$\sum_{j=1}^{n} w_j(FWT_j)^2$	Decomposition approach	Szwarc and Mukhopadhyay (1995)
1	-	$\frac{1}{n}\sum_{k=1}^{n}(FWT_{[k]}-\overline{FWT})^2$	Comparison to $\frac{1}{n}\sum_{k=1}^{n}(FWT_{[k]}-\overline{FWT})^2$	Zhou and Cai (1996)
1	-	$\frac{1}{n}\sum_{k=1}^{n}(FWT_{[k]}-\overline{FWT})^2$	Two heuristics	Ye et al. (2007)
1	-	$\frac{1}{n}\sum_{k=1}^{n}(FWT_{[k]}-\overline{FWT})^2$	Discussion about influencing factors	Li et al. (2007)
1	-	$\sum_{j=1}^{n} w_j(FWT_j-\overline{FWT})^2$	Optimally solved	Xu (2011)
1	$p_j=f_j\cdot B_j$	$\sum_{k=1}^{n}\sum_{k'=1}^{n}\lvert FWT_{[k']}-FWT_{[k]}\rvert$	Optimally solved	Sun et al. (2011)
1	$setup$, d_j	$\sum CWT_j$	Heuristics algorithms	De Matta (2019)
$F2$	$wt_{2,j}$, $skip$	$\frac{1}{n}\sum_{k=1}^{n}(CWT_{2,[k]}-\overline{CWT})^2$	Approximate solution algorithm	Yu et al. (2017)
Pm	-	$\frac{1}{n}\sum_{k=1}^{n}(FWT_{[k]}-\overline{FWT})^2$	Heuristic algorithms	Xu and Ye (2007)
Jm	-	Total waiting time	Heuristic algorithm	Chu and Portmann (1993)
Fm	idm, ddm	$\sum C_i$	Solution methods	Ho and Gupta (1995)
Fm	$wt_{i,j}^{max}$, $wt_{i,j}^{min}$	$WMCT$	Exact algorithm	Fondrevelle et al. (2008)
Fm	$prmu$, $flex$	$\sum C_i$	Heuristic approach	Ruiz-Torres et al. (2011)
Fm	machine-independent and position-dependent processing times	$\sum C_i$	Solution method	Fiszman and Mosheiov (2018)
Fm		$\sum CIT_i$	Heuristic approach	Liu et al. (2014)
Pm	$p_j=f_j\cdot B_j$	$\sum C_i$	Heuristic approach	Mosheiov (1998)
Pm	$p_j=f_j\cdot B_j$	$\sum C_i$	Lower bound	Jeng and Lin (2007)
Pm	job-dependent and position-dependent deterioration	$\sum C_i$	Polynomial time solution	Mosheiov (2012)
Pm or Rm	deterioration	$\sum C_i$	Polynomial time solutions	Yu et al. (2014)
Rm	deterioration, learning, setup, release dates	$\sum C_i$	Meta-Heuristics	Mir and Rezaeian (2016)
$HF(1,2)$	-	$WMCT$	Approximation algorithm	Cheng et al. (2009)

Table 3: Summary of waiting time and idle time as objective function

γ	Heuristic	Observation	*Author*
$\sum C_j$	Constructive	Definition of CWT and CIT; part of the heuristic approach	Rajendran and Chaudhuri (1992)
$\sum C_j$	Constructive	CIT and CWT part of decision coefficients	Wang et al. (1997)
$\sum C_j$	Constructive	Evaluation of Nodes and Sub-solutions with weighted CIT	Liu and Reeves (2001), Fernandez-Viagas and Framinan (2017)
C_{max}	Constructive	FIT+CIT on the last machine are discussed	Dudek and Teuton Jr (1964)
C_{max}	Constructive	FIT+CIT on the last machine are discussed	Sarin and Lefoka (1993)
C_{max}	Local search algorithm	Idle times as part of the decision process	Liu et al. (2011)
C_{max} or $\sum C_j$	Tie-breaking rule	Idle time consideration	Gupta (1972)
C_{max}	Tie-breaking rule	Idle time on bottleneck is considered	Low et al. (2004)
C_{max}	Tie-breaking rule	$\sum CIT_i$ is part of the rule	Companys et al. (2009)
C_{max}	Tie-breaking rule	$\sum CIT_i$ is part of the rule	Ying and Lin (2013)
C_{max}	Tie-breaking rule	FIT+CIT as part of the rule	Fernandez-Viagas and Framinan (2014)
C_{max}	Tie-breaking rule	FIT+CIT as part of the rule	Liu et al. (2017)

Table 4: Summary of waiting and idle times in algorithms of the PFSP

Description	Graham et al. (1979)-notation	Reference
No-wait	$\alpha\|no\text{-}wait\|\gamma$	detailed survey see Allahverdi (2016)
No-idle	$\alpha\|no\text{-}idle\|\gamma$	surveys of Ruiz et al. (2009) and Goncharov and Sevastyanov (2009)
Limited waiting time	$\alpha \mid wt_{i,j}^{max} \mid \gamma$	Brief overview see Kim and Lee (2019)
Overlapping waiting time	$F3\|prmu, wt_{1-2,j}^{max}, wt_{1-3,j}^{max}\|C_{max}$	Kim and Lee (2019)
Linear dependence of $p_{i,j}$ on waiting times	$F2\|prmu, XX\|XX$	Yang and Chern (1999)
No-wait and no-idle	$F2\|prmu, no\text{-}wait, no\text{-}idle\|C_{max}$	Billaut et al. (2019)
No-wait and no-idle	$F2\|prmu, no\text{-}wait, no\text{-}idle\|\sum C_j$	Della Croce et al. (2018)
Duality relations of no-wait and no-idle	$Fm\|prmu, no\text{-}wait\|C_{max}$ and $Fm\|prmu, no\text{-}idle\|C_{max}$	Kalczynski and Kamburowski (2007)
Relationship between no-wait and no-idle	$Fm\|prmu\|C_{max}$, $Fm\|prmu, no\text{-}wait\|C_{max}$ and $Fm\|prmu, no\text{-}idle\|C_{max}$	Makuchowski (2015)

Table 5: Summary of waiting time and idle time as constraints

Proof. Let S_s be the semi-active schedule for π. Then,

$$\sum_{i=1}^{m} CIT_i(S_s) = \sum_{i=1}^{m} B_{i,[1]} - C_{i,[n]} - \sum_{j=1}^{n} p_{i,j}$$

As $B_{i,[1]}$ is fixed for all schedules $S \in \mathcal{S}_c^{\pi}$, it can be seen that $\sum CIT_i$ depends only on $C_{i,[n]}$.

Therefore, right-shifting any job different to the one in the last position $[n]$ implies the same $\sum CIT_i$. Right-shifting the job in the last position leads to an increase of $\sum CIT_i$. Therefore, $\sum CIT_i(S_s) \leq \sum CIT_i(S)\ \forall S \in \mathcal{S}_c^\pi, S \neq S_s$.

Analogously,

$$\sum_{j=1}^{n} CWT_j(S_a) = \sum_{k=1}^{n} C_{m,[k]} - B_{1,[k]} - \sum_{i=1}^{m} p_{i,[k]}$$

Here, $B_{1,[k]}$ is fixed for all schedules $S \in \mathcal{S}_c^\pi$. So, $\sum CWT_j$ depends only on $C_{m,[k]}$. Hence, right-shifting any job on a machine different to the last one leads to the same $\sum CWT_j$, while right-shifting any job in the last machine implies an increase of $\sum CWT_j$. Therefore, $\sum CWT_j(S_s) \leq \sum CWT_j(S)\ \forall S \in \mathcal{S}_c^\pi, S \neq S_s$. □

Observation 1 is illustrated by Figure 3. It can be seen that any right-shifting operation provides worse values at least for one of the objective functions.

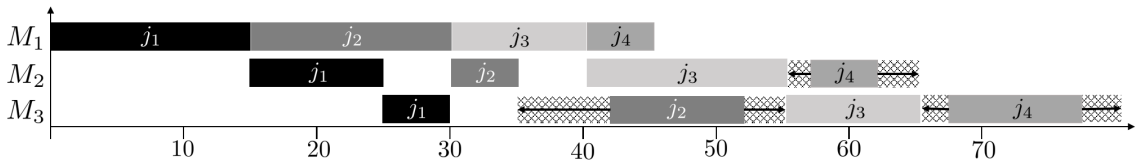

Figure 3: Possible movements maintaining the first machine no-idle and the first job no-wait

4.2. Cases with special processing times

In this section some basic theorems are proved regarding special cases of processing times, where the problems of minimizing $\sum CWT_j$ and $\sum C_j$ or $\sum CIT_i$ and C_{max} are equivalent. Finally, two cases implying $\sum CWT_j = 0$ and $\sum CIT_i = 0$ for all schedules are shown. First, a relationship between $\sum CWT_j$ and $\sum C_j$ is presented (see also Benkel et al., 2015 and Rajendran and Chaudhuri, 1992 for a similar formulation).

Lemma 1. *Total core waiting time can be expressed in terms of total completion time minus total front waiting time and minus the constant $P = \sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij}$, i.e.*

$$\sum_{j=1}^{n} CWT_j = \sum_{j=1}^{n} C_j - \sum_{j=1}^{n} FWT_j - P \tag{5}$$

Proof. The definition of $\sum CWT_j$, given in Eq. 2 (Table 2), is recalled: $\sum_{j=1}^{n} CWT_j = \sum_{i=2}^{m}\sum_{k=2}^{n} B_{i,[k]} - C_{i-1,[k]}$. Taking into account that the job in the first position never has to wait within the process, it is possible to add position $k = 1$ to the previous formula obtaining

$$\sum_{j=1}^{n} CWT_j = \sum_{i=2}^{m}\sum_{k=1}^{n} B_{i,[k]} - C_{i-1,[k]}$$

Disassembling the sum of machines leads to

$$\sum_{j=1}^{n} CWT_j = \sum_{k=1}^{n} B_{2,[k]} - C_{1,[k]} + B_{3,[k]} - C_{2,[k]} + ... + B_{m,[k]} - C_{m-1,[k]}$$

Taking into account that $B_{i,[k]} = C_{i,[k]} - p_{i,[k]} \Rightarrow p_{i,[k]} = C_{i,[k]} - B_{i,[k]}$, $\sum CWT_j$ can be expressed in the following way:

$$\sum_{j=1}^{n} CWT_j = \sum_{k=1}^{n}(B_{m,[k]} - C_{1,[k]} - \sum_{i=2}^{m-1} p_{i,[k]})$$

Additionally, $B_{m,[k]} = C_{m,[k]} - p_{m,[k]}$ and $C_{1,[k]} = B_{1,[k]} + p_{1,[k]}$, i.e.

$$\sum_{j=1}^{n} CWT_j = \sum_{k=1}^{n}(C_{m,[k]} - p_{m,[k]} - B_{1,[k]} - p_{1,[k]} - \sum_{i=2}^{m-1} p_{i,[k]})$$

which can be simplified to

$$\sum_{j=1}^{n} CWT_j = \sum_{k=1}^{n}(C_{m,[k]} - B_{1,[k]} - \sum_{i=1}^{m} p_{i,[k]})$$

where $\sum_{j=1}^{n} C_j = \sum_{k=1}^{n} C_{m,[k]}$, $P = \sum_{k=1}^{n}\sum_{i=1}^{m} p_{i,[k]}$ and $\sum_{j=1}^{n} FWT_j = \sum_{k=1}^{n} B_{1,[k]}$ (see Table 1 and Table 2). □

Using Lemma 1, the following theorem shows that the problems minimizing $\sum C_j$ and $\sum CWT_j$ are equivalent in a special case of the processing times.

Theorem 1. *In an m-machine permutation flow shop where the processing times on machine $i = 1$ are identical for all jobs, i.e. $p_{1,1} = p_{1,2} = ... = p_{1,n}$, the minimization of $\sum CWT_j$ is equivalent to the minimization of $\sum C_j$.*

Proof. This result can be proved with the Gaussian triangular number. If the processing times

are equal on the first machine, p_1, the calculation of $\sum FWT_j$ can be reduced to $\sum_{k=1}^{n} FWT_{[k]} = \frac{n(n+1)}{2} \cdot p_1$ for all sequences, i.e. $\sum FWT_j$ is constant. Then, $\sum CWT_j$ only depends on $\sum C_j$ (Eq. 5). So, the minimization of $\sum CWT_j$ is equivalent to the minimization of $\sum C_j$. □

In the following Lemma it can be seen that $\sum CIT_i$ is related to the completion time of machines (see also Benkel et al., 2015).

Lemma 2. *Total core idle time can be expressed in terms of total machine completion time minus total front idle time and minus the constant* $P = \sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij}$*, i.e.*

$$\sum_{i=1}^{m} CIT_i = \sum_{i=1}^{m} C_i - \sum_{i=1}^{m} FIT_i - P \tag{6}$$

Proof. Considering the definition of $\sum CIT_i$ given in Eq. 1 in Table 2 $\sum_{i=1}^{m} CIT_i = \sum_{i=2}^{m} \sum_{k=2}^{n} B_{i,[k]} - C_{i,[k-1]}$ and keeping the first machine no-idle within the process leads to

$$\sum_{i=1}^{m} CIT_i = \sum_{i=1}^{m} \sum_{[k]=2}^{n} B_{i,[k]} - C_{i,[k-1]}$$

Similar to the previous proof, disassembling the sum of jobs and considering that$p_{i,[k]} = C_{i,[k]} - B_{i,[k]}$ leads to

$$\sum_{i=1}^{m} CIT_i = \sum_{i=1}^{m} B_{i,[2]} - C_{i,[1]} + B_{i,[3]} - C_{i,[2]} + ... + B_{i,[n]} - C_{i,[n-1]} = \sum_{i=1}^{m} (B_{i,[n]} - C_{i,[1]} - \sum_{k=2}^{n-1} p_{i,[k]})$$

Moreover, $B_{i,[n]} = C_{i,[n]} - p_{i,[n]}$ and $C_{i,[1]} = B_{i,[1]} + p_{i,[1]}$, i.e.

$$\sum_{i=1}^{m} CIT_i = \sum_{i=1}^{m} (C_{i,[n]} - p_{i,[n]} - B_{i,[1]} - p_{i,[1]} - \sum_{k=2}^{n-1} p_{i,[k]}) = \sum_{i=1}^{m} (C_{i,[n]} - \sum_{i=1}^{m} B_{i,[1]} - \sum_{k=1}^{n} p_{i,[k]})$$

where $\sum_{i=1}^{m} C_i = \sum_{i=1}^{m} C_{i,[n]}$, $\sum_{i=1}^{m} FIT_i = \sum_{i=1}^{m} B_{i,[1]}$ (see Table 2) and $P = \sum_{k=1}^{n} \sum_{i=1}^{m} p_{i,[k]}$. □

Using Lemma 2, the following theorem proves a two-machines special case, in which minimization of $\sum CIT_i$ is equivalent to minimize makespan.

Theorem 2. *In a two-machine permutation flow shop where the processing times on machine $i = 1$ are identical for all jobs, i.e. $p_{1,1} = p_{1,2} = \cdots = p_{1,n}$, the minimization of $\sum CIT_i$ is equivalent to the minimization of C_{max}.*

Proof. As it is shown in Table 2, $\sum FIT_i$ only depends on the processing time of the first job on the first machine. As long as the processing times are equal on the first machine, $\sum FIT_i$ is also equal for each possible schedule and $\sum CIT_i$ can be reduced to the minimization of $\sum C_i$ (Eq. 6). Because $C_{1,[n]}$ is constant, the minimization only refers to $C_{2,[n]}$ which equals C_{max}. □

In the following theorems, two trivial cases implying $\sum CWT_j = 0$ and $\sum CIT_i = 0$ for all schedules are shown.

Theorem 3. *In an m-machine permutation flow shop where the machines are arranged in a decreasing dominance order, i.e. $min\{p_{i,j}\} \geq max\{p_{i+1,j}\} \quad \forall\, j, \forall i = 1, ..., m-1$, then, $\sum CWT_j$ is zero for every schedule.*

Proof. Under the conditions of this result, Ho and Gupta (1995) shows that:

$$\sum_{j=1}^{n} C_j = \sum_{j=1}^{n}\sum_{k=1}^{j} p_{1,[k]} + \sum_{j=1}^{n}\sum_{i=2}^{m} p_{i,j} \tag{7}$$

It can be observed from Table 2 that $\sum_{j=1}^{n} FWT_j = \sum_{j=1}^{n-1}\sum_{k=1}^{j} p_{1,[k]}$, so

$$\sum_{j=1}^{n} C_j = \sum_{j=1}^{n-1}\sum_{k=1}^{j} p_{1,[k]} + \sum_{k=1}^{n} p_{1,[k]} + \sum_{j=1}^{n}\sum_{i=2}^{m} p_{i,j} = \sum_{j=1}^{n} FWT_j + \sum_{j=1}^{n}\sum_{i=1}^{m} p_{i,j} = \sum_{j=1}^{n} FWT_j + P$$

From Eq. 5 we conclude that in this case $\sum CWT_j = 0$. □

Theorem 4. *In an m-machine permutation flow shop where the machines are arranged in an increasing dominance order, i.e. $max\{p_{i,j}\} \leq min\{p_{i+1,j}\} \quad \forall\, j, \forall i = 1, ..., m-1$, then, $\sum CIT_i$ is zero for every schedule.*

Proof. Similarly to the previous result, and based on the expression of $\sum C_i$ given by Ho and Gupta (1995) under the conditions of this result given in Eq. 8, the result can be derived using the

definition of $\sum FIT_i$ given in Table 2.

$$\sum_{i=1}^{m} C_i = \sum_{i=1}^{m-1} p_{i,[1]} \cdot (m-i) + \sum_{i=1}^{m} \sum_{j=1}^{n} p_{i,j} \tag{8}$$

□

5. Computational analysis

After analyzing C_{max}, $\sum C_j$, $\sum CIT_i$ and $\sum CWT_j$ theoretically on the basis of some special cases, we now focus on the general cases of processing times in order to gain further insights into the alignment of the considered objectives. Different analytical methods are going to be applied in the analysis (Section 5.1). The relationships between the four objectives are analyzed by interpreting the result (Section 5.2).

5.1. Analytical methods

Based on the observation that one instance can have more than one optimal sequence for a given objective function, and taking into account that, in the set of the optimal solutions for a given objective, the variability of the efficiency of these solutions for a different objective may be high, we propose different analytical methods to compare the objective functions:

- *Complete Enumeration:* This method evaluates all $n!$ schedules. Here, the optimum of an objective can be obtained and, in addition, the number of optimal schedules can be computed. This is the most expensive of the proposed methods from the computational time point of view. Therefore, it is restricted to very small sizes (up to $n = 10$).

- *Classical MILP Model:* In this context, the classical MILP is referred to a single-objective model providing one random optimal sequence among all possible optimums for a given objective function. The disadvantage is that neither the number of optimal schedules nor the relationship to other objective functions is considered, , but computational effort is lower than for the complete enumeration. The MILP formulation of PFSP and C_{max} given by Wilson (1989) (see Wilson, 1989 or Stafford Jr et al., 2005 for the description of the model and its evaluation) is

adapted changing the objective function. The formulations of the objective functions are:

$$C_{max} = B_{m,[n]} + \sum_{j=1}^{n} p_{m,j} \cdot X_{j,[n]} \tag{9}$$

$$\sum_{j=1}^{n} C_j = \sum_{k=1}^{n} \left(B_{m,[k]} + \sum_{j=1}^{n} p_{m,j} \cdot X_{j,[k]} \right) \tag{10}$$

$$\sum_{i=1}^{m} CIT_i = \sum_{i=2}^{m} \left(B_{i,[n]} - B_{i,[1]} - \sum_{k=2}^{n} \sum_{j=1}^{n} p_{i,j} \cdot X_{j,[k-1]} \right) \tag{11}$$

$$\sum_{j=1}^{n} CWT_j = \sum_{k=2}^{n} \left(B_{m,[k]} - B_{1,[k]} - \sum_{i=2}^{m} \sum_{j=1}^{n} p_{i-1,j} \cdot X_{j,[k]} \right) \tag{12}$$

- *Lexicographical Approach:* This multi-objective MILP model is used to identify, among the optimal sequences of a primary objective OF_1, the best one of a secondary objective OF_2 (see e.g. T'kindt and Billaut, 2006). This method consists on the same optimization model than the classical MILP used for a given objective, adding a (hard) constraint where the optimal value for a different objective is imposed. Note that the information provided by the MILP is used, since the optimal value of the secondary objective is needed. This approach provides more information than the classical MILP, but again the computation effort is higher.

The classical MILP as well as the lexicographical MILP models imposes that the first machine and the first jobs are no idle. Observation 1 guarantees that the objective function values for $\sum CWT_j$ and $\sum CIT_i$ are the same that the given by the semi-active schedule.

5.2. Relationship between objectives

The analytical methods previously presented are applied to a set of instances, with $n = [5, 10, 15, 20]$ jobs, $m = [2, 5, 10]$ machines and uniformly distributed processing times ($U[1, 99]$). 30 instances for each problem size have been generated, i.e. in total 360 instances. Each method use different instances due to the computational times needed.

5.2.1. Comparison of results by Complete Enumeration

The complete enumeration is applied to instances with $n = 5$ and $n = 10$ jobs of the set previously described (180 instances in total) to evaluate all solutions of the problem depending on

the objective function, $OF \in \{C_{max}, \sum C_j, \sum CWT_j, \sum CIT_i\}$. Usually in scheduling, results are compared with Relative Percentage Deviation (RPD) which is defined as $RPD = \frac{OF(S)-OF^*}{OF^*}$ where $OF(S)$ is the objective function value of a sequence S and OF^* refers to the optimum. However, an optimal value of $\sum CIT_i$ and $\sum CWT_j$ might be zero, i.e. RPD is not a suitable performance measure for this analysis. Instead, we refer to the Relative Deviation Index (RDI, see e.g. Kim, 1993, Fernandez-Viagas and Framinan (2015) and Perez-Gonzalez et al. (2019)). For the complete enumeration, we define RDI^a_{OF} of a sequence S for a given objective function OF as

$$RDI^a_{OF} = \frac{OF(S) - OF^*}{OF^{max} - OF^*} \quad \forall S \tag{13}$$

where $OF(S)$ is the evaluation of S for objective OF, OF^* is the optimal value and $OF^{max} = \max\{OF(S) \forall S \text{ of instance } I\}$.

To characterize the distributions in detail, the RDI^a-values of each objective are computed and summarized as the frequency of RDI^a within a range of $[0, 1]$ for each problem size. Similar studies have been carried out by Taillard (1990), Perez-Gonzalez and Framinan (2009) and Fernandez-Viagas and Framinan (2015). In Figure 4 represents the empirical distribution for the instances with $m = 2$ (left) and $m = 10$ (right), both cases for $n = 10$. In the figure it can be seen as the empirical distribution for $m = 2$ reveals a problem with more solutions close to the optimal than the case with $m = 10$, for all objective functions except $\sum C_j$. $\sum CIT_i$ is the objective with more solutions close to the optimal, regardless the number of machines.

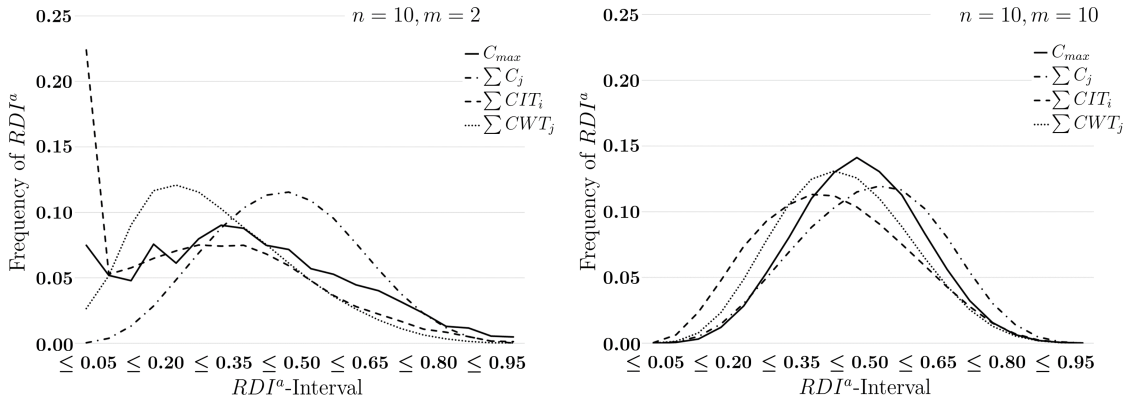

Figure 4: Empirical distribution for each objective function; left: Case $n = 10$ and $m = 2$; right: Case $n = 10$ and $m = 10$

	n = 10, m = 2				n = 10, m = 10			
	mean	STD	skewness	kurtosis	mean	STD	skewness	kurtosis
C_{max}	0,375	0,214	0,364	-0,391	0,481	0,132	0,053	-0,216
$\sum C_j$	0,466	0,158	0,073	-0,386	0,506	0,147	-0,075	-0,334
$\sum CIT_i$	0,283	0,195	0,674	0,205	0,425	0,155	0,224	-0,184
$\sum CWT_j$	0,315	0,154	0,709	0,348	0,448	0,139	0,207	-0,229

Table 6: Statistics indicators for C_{max}, $\sum C_j$, $\sum CIT_i$ and $\sum CWT_j$

To go deeper in the behaviour of the empirical distributions the mean, standard deviation (STD), kurtosis and skewness as shown in Table 6. Considering $m = 2$, all distributions are right-skewed (skewness > 0), significantly for C_{max}, $\sum CIT_i$ and $\sum CWT_j$ but slightly for $\sum C_j$. Moreover, C_{max} and $\sum C_j$ are platykurtic (kurtosis < 0), while $\sum CIT_i$ and $\sum CWT_j$ are the opposite. Figure 4 (left) shows that there are many solutions of $\sum CIT_i$ which are optimal or near-optimal. Because of its strong right-skewness, also many solutions of $\sum CWT_j$ have a RDI^a smaller than 0.5. Considering $m = 10$, the mean-values are all close to 0.5 and the standard deviations are nearly similar for all objectives. While C_{max} and $\sum C_j$ are only slightly skewed, $\sum CIT_i$ and $\sum CIT_i$ are again significantly right-skewed. Moreover, all distributions are platykurtic. Comparing both diagrams, it can be seen that the distribution of $\sum C_j$ is only slightly affected by an increase of machines, while all other objectives provide different distributions depending on the number of machines and are more aligned to a normal distribution when the number of machines increases.

Additionally, Table 7 presents the average of optimal solutions given per size for each case. Note that the problem with $n = 5$ has 120 possible sequences and $n = 10$ has 3,628,800. It can be seen that C_{max}, $\sum CIT_i$ and $\sum CWT_j$ provide many optimal solutions for $m = 2$, $\sum CIT_i$ is the objective providing the highest value in both cases, representing the optimal solutions approximately at 23% and 18% of the overall possible solutions for $n = 5$ and $n = 10$, respectively. This behaviour decreases when the number of machines increases. However, $\sum C_j$ on average only provides approximately one optimal sequence per instance, independently of the problem size.

n	5			10		
m	2	5	10	2	5	10
C^*_{max}	10.1	2.1	1.5	102,195.7	185.6	4.4
$\sum C^*_j$	1.3	1.0	1.0	1.4	1.1	1.0
$\sum CIT^*_i$	27.6	1.8	1.0	657,337.9	64.3	1.2
$\sum CWT^*_j$	9.8	1.1	1.0	17,276.2	1.0	1.0

Table 7: Average number of optimal sequences per instance provided by complete enumeration

Once we know how many optimal sequences are obtained for each objective function, we focus on evaluate them for the rest of the objectives. Therefore, considering only optimal sequences, for each objective $OF \in \{C_{max}, \sum C_j, \sum CWT_j, \sum CIT_i\}$ we compute $RDI^a_{OF}\ \forall S \in \{S^*_{OF'} : OF' \neq OF\}$. The results on average when $n = 10$ and $m = 10$ are shown in Figure 5, evaluating the optimal sequences provided by $\sum CIT_i$ for the rest of the objectives (left) and evaluating the optimal sequences provided by $\sum CWT_j$ for the rest of the objectives (right). The graphs represent the frequency of RDI^a_{OF} in intervals $[0, 0.25]$, $[0.25, 0.5]$, $[0.5, 0.75]$, $[0.75, 1]$ for each pair of objectives. In Figure 5 (left), it can be observed that more than the 75% of optimal sequences of $\sum CIT_i$ are good solutions for C_{max} with an RDI^a lower than 0.25, but $\sum C_j$ and $\sum CWT_j$ do not provide good results in this case. In Figure 5 (right) it can be observed that all optimal solutions (100%) of $\sum CWT_j$ are good solutions for $\sum C_j$ with an RDI^a lower than 0.25. Therefore, it can be concluded that the alignment between $\sum C_j$ and $\sum CWT_j$ is stronger than between C_{max} and $\sum CIT_i$.

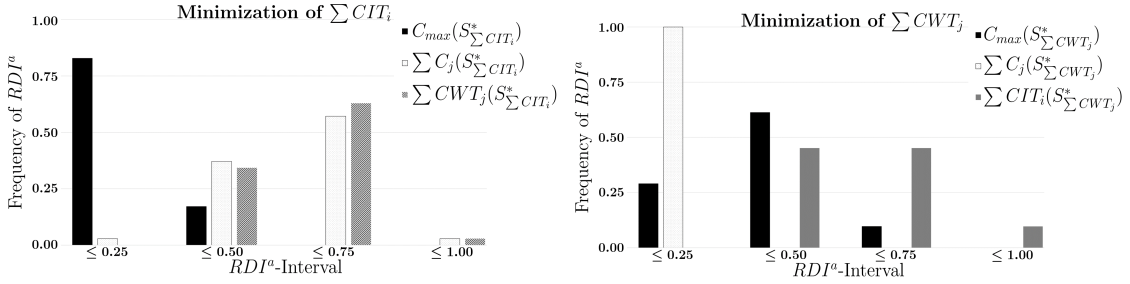

Figure 5: RDI^a for optimal solution problem size $n = 10$, $m = 10$: Left $\sum CIT_i$; Right $\sum CWT_j$

5.2.2. Comparison of results by classical MILP

Due to the computational effort needed to solve the instances by complete enumeration, in this section we use the MILP model in order to obtain, at least, one optimal solution for each of the 360 instances with different sizes. The solver Gurobi (Gurobi Optimization, 2018) with a computational time limit of 900 seconds was used providing optimal solution (or best found in some cases of the biggest sizes) for each instance.

To compare the objectives, we define $RDI^b_{OF_1}$ for each instance I, given a primary objective function OF_1 which is going to be compared to a secondary objective function OF_2. $OF_1(S^*_{OF_2})$ is defined as the value of OF_1 referring to the optimal sequence of OF_2, OF^*_1 is the optimal value for

OF_1 and $OF_1^{worst} = \max_{OF' \neq OF_1}\{OF_1(S^*_{OF'})\}$.

$$RDI^b_{OF_1} = \frac{OF_1(S^*_{OF_2}) - OF_1^*}{OF_1^{worst} - OF_1^*} \quad \forall I \tag{14}$$

The difference between RDI^a and RDI^b is that OF^{max} in RDI^a is the worst value among all the possible sequences for a given instance I, and OF_1^{worst} in RDI^b is the worst value given by OF_1 among the optimal sequences provided for the rest of the objectives for the given instance I.

Table 8 shows the average RDI^b, denoted as $ARDI^b$, and variability of RDI^b in terms of coefficient of variation, denoted as CV (computed by dividing the standard deviation by the average), considering all instances. If CV is lower than 0.75, between 0.75 and 1.33 or greater than 1.33, the variability is low, medium or high, respectively (see Hopp and Spearman, 2008, p. 269). If $ARDI^b$ is zero or close to zero, this is interpreted as a strong alignment between two considered objectives. On the opposite, if $ARDI^b$ is one or close to one, it shows a strong conflict between two objectives.

Considering the results presented in Table 8, $\sum C_j$ and $\sum CWT_j$ are the most aligned objective functions regarding $ARDI^b$. The optimal sequence of $\sum C_j$ provides an $ARDI^b = 0.14$ for $\sum CWT_j$, while the optimal sequence of $\sum CWT_j$ leads to $ARDI^b = 0.28$ for $\sum C_j$, both cases with medium variability. Additionally, it can be seen that the optimal sequence of $\sum CIT_i$ is more aligned to C_{max} (0.54 and low variability) than to $\sum C_j$ or $\sum CWT_j$, and the optimal sequence of C_{max} provides better $\sum CIT_i$ values (0.34 and low variability). Finally, it can be observed that $\sum CWT_j$ and $\sum CIT_i$ are very conflicting objective functions with high average and low variability.

	OF							
	C_{max}		$\sum C_j$		$\sum CIT_i$		$\sum CWT_j$	
	$ARDI^b$	CV	$ARDI^b$	CV	$ARDI^b$	CV	$ARDI^b$	CV
$C_{max}(S^*_{OF})$	-	-	0.58	0.59	0.54	0.69	0.85	0.32
$\sum C_j(S^*_{OF})$	0.40	0.61	-	-	0.96	0.13	0.28	0.99
$\sum CIT_i(S^*_{OF})$	0.34	0.70	0.71	0.37	-	-	0.97	0.15
$\sum CWT_j(S^*_{OF})$	0.55	0.48	0.14	1.01	0.97	0.12	-	-

Table 8: Average and coefficient of variation for RDI^b for each pair of objective functions

As the most promising objective-pairs referring to their alignment are C_{max} with $\sum CIT_i$ and $\sum C_j$ with $\sum CWT_j$, they have been analysed deeply. Figure. 6 shows the 95%-confidence interval

of machine-grouped $ARDI^b$-values referring to these pairs. The results show that the relationship between $\sum C_j$ and $\sum CWT_j$ is stronger than between C_{max} and $\sum CIT_i$. A job-grouped point of view is shown in Figure 7 and the results reveal that there is no influence of the number of jobs on $ARDI^b$ for cases $C_{max}(S^*_{\sum CIT_i})$, $\sum CIT_i(S^*_{C_{max}})$ and $\sum CWT_j(S^*_{\sum C_j})$, while $\sum C_j(S^*_{\sum CWT_j})$ is affected significantly.

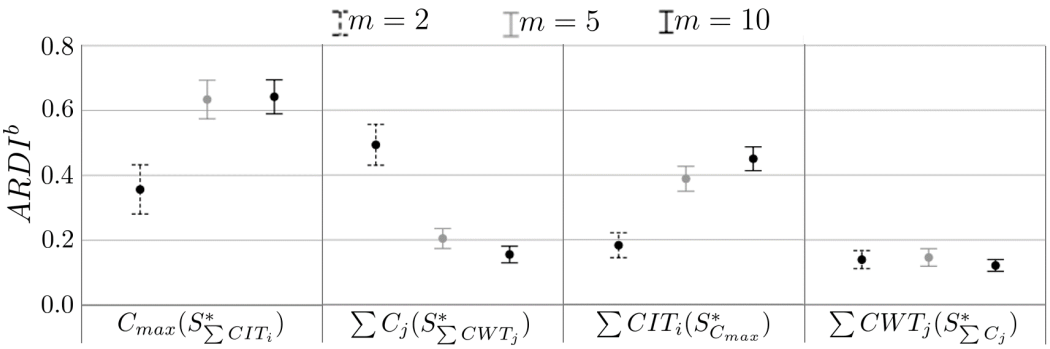

Figure 6: Classic MILP results: $ARDI^b$ grouped by machines

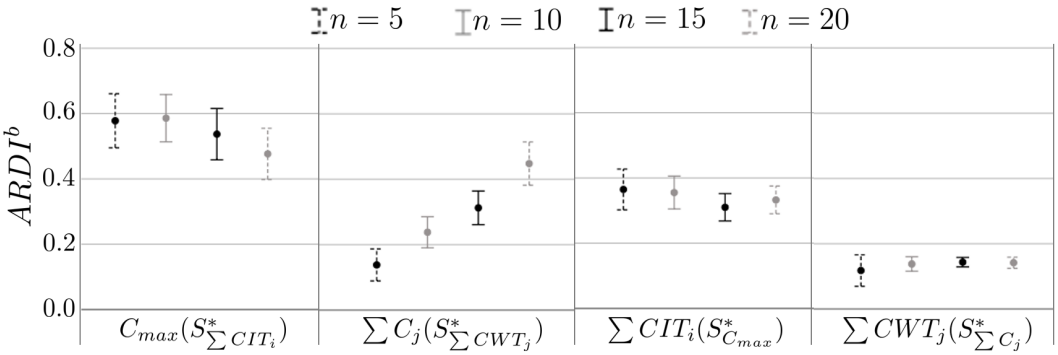

Figure 7: Classic MILP results: $ARDI^b$ grouped by machines

The observation made with complete enumeration, namely that the alignment between $\sum C_j$ and $\sum CWT_j$ is stronger than between C_{max} and $\sum CIT_i$, can also be confirmed for larger problem sizes solved with MILP. However, the classical MILP only provides one random optimal sequence. It might be interesting to analyse a sequence which is optimal for the primary objective and sub-optimal for the secondary objective. For this reason, the lexicographical approach is applied additionally.

5.2.3. Lexicographical Approach

In order to analyse all optimal solutions for a given objective with respect to the rest of the objectives, in this section, the model of Wilson (1989) is extended for a lexicographical approach. This method needs a computational effort higher than the classical MILP, but it is more reasonable than the time needed by complete enumeration, so it is applied to all the instances. In total 12 optimization models (for each pair of objectives) are solved by Gurobi solver, in this case with a stopping criteria of 1800 seconds (due to the difficulty when the additional constraint is added).

A new RDI^b-value, denoted as RDI^b_n, is given for each instance, since the lexicographical approach provides the best sequence for a given objective among the optimal sequences of a different objective. Results are shown in Table 9. Due to the difficulty of the approach, all the instances have not been solved optimally (see the row denoted 'Instances', where the number of instances solved optimally are given). The average RDI^b_n-values, denoted as $ARDI^b_n$, and the improvement with respect to the MILP model (expressed by the difference: $RDI^b - RDI^b_n$) are given for each pair of objectives. To maintain clarity, only improvements > 0.01 are shown and shaded gray if > 0.1. $ARDI^b_n$ close to one implies high goal conflict between the involved objectives, revealing that optimal schedules for the primary objective imply very bad values for the secondary objective. Additionally, small improvements mean poor performance of the lexicographical approach, being in these cases the solution of the MILP similar to the provided by this (computationally expensive) method. From Table 9, it can be seen as example that, in case $m = 2$ and $n = 5$, C_{max} with $\sum CIT^*_i$ is 29.4% better than the MILP by the lexicographical approach. For the majority of problem sizes, $\sum CIT_i$ with C^*_{max} leads to significantly better results compared to C_{max} with $\sum CIT^*_i$. Additionally, $\sum C_j$ and $\sum CWT_j$ are strongly aligned too in the two ways with small $ARDI^b_n$ values for all sizes. When $m > 2$, a goal conflict can be seen for $\sum C_j$ with $\sum CIT^*_i$, and between $\sum CIT_i$ with $\sum CWT_j$ in the two ways.

In general, there are two interesting observations: First, we could not gain any significant improvement when $\sum C^*_j$ is considered, i.e. in cases where $\sum C_j$ is considered as objective the lexicographical approach is not suitable to gain better results for the other objectives. Secondly, the improvement is small for all the cases when $m = 10$. These results may be related to the observations done in Section 5.2.1: for problems with $\sum C_j$ as objective, and in general for all objectives when $m = 10$, the number of optimal solutions are small. So, in these cases, it is difficult to find another optimal sequence with better results for a secondary objective function and hence,

the lexicographical approach is not a good option.

Detailed conclusions about this section, as well as about all the paper are explained in the next section.

		Problem Size n/m											
OF/OF^*	$Indicator$	5/2	5/5	5/10	10/2	10/5	10/10	15/2	15/5	15/10	20/2	20/5	20/10
C_{max}	Instances	30	30	30	30	30	30	29	22	11	16	0	0
$\sum C_j^*$	$ARDI_n^b$	0.594	0.493	0.540	0.277	0.628	0.744	0.386	0.652	0.793	0.390	-	-
	Improve.	0.035	0.031	≤ 0.01	0.071	≤ 0.01	≤ 0.01	≤ 0.01	0.020	≤ 0.01	≤ 0.01	-	-
$\sum CIT_i^*$	$ARDI_n^b$	0.104	0.721	0.624	0.030	0.619	0.596	0.011	0.559	0.815	0.011	-	-
	Improve.	0.294	≤ 0.01	≤ 0.01	0.456	≤ 0.01	≤ 0.01	0.448	0.014	≤ 0.01	0.306	-	-
$\sum CWT_j^*$	$ARDI_n^b$	0.581	0.660	0.808	0.587	0.907	0.846	0.597	0.896	0.889	0.560	-	-
	Improve.	0.288	≤ 0.01	≤ 0.01	0.211	≤ 0.01	0.010	0.166	≤ 0.01	≤ 0.01	0.309	-	-
$\sum C_j$	Instances	30	30	30	30	30	30	30	30	28	20	5	3
C_{max}^*	$ARDI_n^b$	0.249	0.259	0.333	0.118	0.307	0.288	0.058	0.256	0.284	0.072	0.160	0.361
	Improve.	0.329	0.011	0.022	0.496	0.086	0.032	0.528	0.270	0.015	0.503	0.263	0.020
$\sum CIT_i^*$	$ARDI_n^b$	0.574	0.987	0.991	0.260	0.913	0.998	0.218	0.729	0.976	0.207	0.664	0.905
	Improve.	0.393	0.013		0.705	0.081		0.767	0.262	0.024	0.756	0.336	0.095
$\sum CWT_j^*$	$ARDI_n^b$	0.111	0.069	0.092	0.189	0.202	0.135	0.202	0.269	0.148	0.178	0.140	0.046
	Improve.	0.111	≤ 0.01	≤ 0.01	0.207	≤ 0.01	≤ 0.01	0.194	≤ 0.01	≤ 0.01	0.382	0.083	0.080
$\sum CIT_i$	Instances	30	30	30	30	30	30	30	16	4	13	0	0
C_{max}^*	$ARDI_n^b$	0.066	0.486	0.414	0.018	0.332	0.472	0.012	0.210	0.505	0.002	-	-
	Improve.	0.109	0.045	≤ 0.01	0.069	0.027	≤ 0.01	0.143	0.107	0.019	0.272	-	-
$\sum C_j^*$	$ARDI_n^b$	0.750	0.859	0.812	0.508	0.693	0.810	0.655	0.758	0.800	0.596	-	-
	Improve.	0.021	≤ 0.01	≤ 0.01	0.055	0.014	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	-	-
$\sum CWT_j^*$	$ARDI_n^b$	0.675	0.959	0.998	0.736	0.984	0.993	0.743	0.991	1.000	0.648	-	-
	Improve.	0.212	≤ 0.01	≤ 0.01	0.207	≤ 0.01	≤ 0.01	0.159	≤ 0.01	≤ 0.01	0.275	-	-
$\sum CWT_j$	Instances	30	30	30	30	30	30	30	30	23	30	10	0
C_{max}^*	$ARDI_n^b$	0.233	0.393	0.425	0.064	0.407	0.388	0.036	0.298	0.386	0.016	0.184	-
	Improve.	0.370	0.021	0.019	0.597	0.117	0.033	0.642	0.296	0.028	0.646	0.293	-
$\sum C_j^*$	$ARDI_n^b$	0.079	0.113	0.078	0.143	0.151	0.116	0.136	0.182	0.122	0.121	0.130	-
	Improve.	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	≤ 0.01	-
$\sum CIT_i^*$	$ARDI_n^b$	0.397	0.980	1.000	0.096	0.867	0.997	0.068	0.620	0.970	0.033	0.427	-
	Improve.	0.560	≤ 0.01	≤ 0.01	0.782	0.133	≤ 0.01	0.854	0.369	0.030	0.889	0.573	-

Table 9: $ARDI_n^b$ and Improvement of all objectives

6. Conclusions

Besides the often discussed objectives of minimizing makespan, C_{max}, and total completion time, $\sum C_j$, there are also other important objectives in static-deterministic scheduling, e.g. idle time of machines and waiting time of jobs whose definitions vary in the literature. We provided a clear structure dividing both measures in front, back and core components. We discussed that

especially the front and back components could be used for other (non-value-added) tasks and focused on the core components, namely $\sum CIT_i$ and $\sum CWT_j$. We explained that $\sum CIT_i$ can be seen as a capacity buffer which is directly related to the core utilization of a permutation flow shop. $\sum CWT_j$ is interpreted as an inventory buffer which can also be seen as an indicator for job flow in a permutation flow shop.

To introduce these new performance measures as objective functions may have interest, for example, to relax no-wait and no-idle constraints. As these constrains use to be considered for C_{max} or $\sum C_j$ minimization, it is relevant to analyse the alignment among the proposed objectives and these classical objectives functions.

Therefore, some basic results show the equivalence between C_{max} and $\sum CIT_i$, and between $\sum C_j$ and $\sum CWT_j$ but only for some special cases related to the processing times. The relationship among the objective functions for the general case (processing times do not follow any rule) has been analyzed. Due to the computational effort needed for the different exact methods, an extensive computational study has been carried out using complete enumeration, MILP models and lexicographical models.

By complete enumeration, for small sizes (up to $n = 10$), the empirical distribution of each objective are plotted, and the number of optimal solutions are provided for each instance. This method reveal the lower number of optimal and close to the optimal solutions for $\sum C_j$ and also for the other objectives when $m = 10$. Additionally, the quality of the optimal solutions for a given objective has been analysed for the rest of the objective functions. Results show that a 75% of the optimal solutions when $\sum CIT_i$ is minimized, are high quality solutions for C_{max}, but not for the others objectives. In a similar way, almost all the optimal solutions when $\sum CWT_j$ is minimized, are high quality solutions for $\sum C_j$, and they are not so bad for C_{max}.

By MILP models, bigger instances are analysed (up to $n = 20$), but only the optimal solution for a given objective can be evaluated for the rest of them. The results show that there is a good/moderate alignment between C_{max} and $\sum CIT_i$ (as shown by Liu et al., 2016) and a strong alignment between $\sum C_j$ and $\sum CWT_j$, while a goal conflict exists between $\sum CIT_i$ and $\sum CWT_j$.

By lexicographical models, higher computational effort is needed than for MILP, but more insights are obtained since, for the optimal value of a given objective, the best sequence for a second objective is provided. From the results it can be conclude that this approach is not worthwhile for $\sum C_j$, but an interesting improvement is obtained when C_{max} and $\sum CIT_i$ are considered, being

better first to minimize C_{max} and consider $\sum CIT_i$ in a second step.

This research is focused on instances generated by a uniform distribution but it might be interesting for future research to deal with the influence of processing times on $\sum CIT_i$ and $\sum CWT_j$. Moreover, the computational study used a set of instances with a maximum of $n = 20$ jobs, so it is interesting to increase the number of jobs and machines and use heuristic approaches to solve the problems and analyze the behaviour of the objective functions. Additionally, several modifications of $\sum CIT_i$ and $\sum CWT_j$ could be discussed, e.g. by weighting them with costs which leads to machine-idle-costs and inventory carrying cost.

Acknowledgment

The authors would like to thank Jose Framinan for his helpful advice and comments. In addition, the authors would like to commemorate the deceased Rainer Leisten whose idea was to discuss idle time and waiting time in detail for the PFSP. Paz Perez-Gonzalez is supported by the Spanish Ministry of Science and Innovation, under grant "PROMISE" with reference DPI2016-80750-P.

References

A. Allahverdi. A survey of scheduling problems with no-wait in process. *European Journal of Operational Research*, 255(3):665–686, 2016.

U. Bagchi. Simultaneous minimization of mean and variation of flow time and waiting time in single machine systems. *Operations Research*, 37(1):118–125, 1989.

K. Benkel, K. Jørnsten, and R. Leisten. Variability aspects in flowshop scheduling systems. In *2015 International Conference on Industrial Engineering and Systems Management (IESM)*, pages 118–127. IEEE, 2015.

J.-C. Billaut, F. Della Croce, F. Salassa, and V. T'kindt. No-idle, no-wait: when shop scheduling meets dominoes, Eulerian paths and Hamiltonian paths. *Journal of Scheduling*, 22(1):59–68, 2019.

T. E. Cheng, B. M. Lin, and Y. Tian. Scheduling of a two-stage differentiation flowshop to minimize weighted sum of machine completion times. *Computers & Operations Research*, 36(11):3031–3040, 2009.

C. Chu and M.-C. Portmann. Job-shop scheduling to minimize total waiting time. *Applied Stochastic Models and Data Analysis*, 9(2):177–185, 1993.

R. Companys, I. Ribas, and M. Mateo. Improvement tools for NEH based heuristics on permutation and blocking flow shop scheduling problems. In *IFIP International Conference on Advances in Production Management Systems*, pages 33–40. Springer, 2009.

R. De Matta. Minimizing the total waiting time of intermediate products in a manufacturing process. *International Transactions in Operational Research*, 26(3):1096–1117, 2019.

F. Della Croce, A. Grosso, and F. Salassa. Minimizing total completion time in the two-machine no-idle no-wait flow shop problem. In *7th International Workshop Matheuristics, Tours*, 2018.

R. A. Dudek and O. F. Teuton Jr. Development of m-stage decision rule for scheduling n jobs through m machines. *Operations Research*, 12(3):471–497, 1964.

S. Eilon and I. Chowdhury. Minimising waiting time variance in the single machine problem. *Management Science*, 23(6):567–575, 1977.

V. Fernandez-Viagas and J. M. Framinan. On insertion tie-breaking rules in heuristics for the permutation flowshop scheduling problem. *Computers & Operations Research*, 45:60–67, 2014.

V. Fernandez-Viagas and J. M. Framinan. NEH-based heuristics for the permutation flowshop scheduling problem to minimise total tardiness. *Computers & Operations Research*, 60:27–36, 2015.

V. Fernandez-Viagas and J. M. Framinan. A beam-search-based constructive heuristic for the PFSP to minimise total flowtime. *Computers & Operations Research*, 81:167–177, 2017.

V. Fernandez-Viagas, R. Ruiz, and J. M. Framinan. A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation. *European Journal of Operational Research*, 257(3):707–721, 2017.

S. Fiszman and G. Mosheiov. Minimizing total load on a proportionate flowshop with position-dependent processing times and job-rejection. *Information Processing Letters*, 132:39–43, 2018.

J. Fondrevelle, A. Oulamara, and M.-C. Portmann. Permutation flowshop scheduling problems with time lags to minimize the weighted sum of machine completion times. *International Journal of Production Economics*, 112(1):168–176, 2008.

J. Framinan, R. Leisten, and C. Rajendran. Different initial sequences for the heuristic of Nawaz, Enscore and Ham to minimize makespan, idletime or flowtime in the static permutation flowshop sequencing problem. *International Journal of Production Research*, 41(1):121–148, 2003.

J. M. Framinan, J. N. Gupta, and R. Leisten. A review and classification of heuristics for permutation flow-shop scheduling with makespan objective. *Journal of the Operational Research Society*, 55(12):1243–1255, 2004.

J. M. Framinan, R. Leisten, and R. Ruiz-Usano. Comparison of heuristics for flowtime minimisation in permutation flowshops. *Computers & Operations Research*, 32(5):1237–1254, 2005.

M. R. Garey, D. S. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1(2):117–129, 1976.

Y. Goncharov and S. Sevastyanov. The flow shop problem with no-idle constraints: A review and approximation. *European Journal of Operational Research*, 196(2):450–456, 2009.

R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. R. Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

J. N. Gupta. Heuristic algorithms for multistage flowshop scheduling problem. *AIIE Transactions*, 4(1):11–18, 1972.

L. Gurobi Optimization. Gurobi optimizer reference manual, 2018. URL `http://www.gurobi.com`.

N. G. Hall and C. Sriskandarajah. A survey of machine scheduling problems with blocking and no-wait in process. *Operations research*, 44(3):510–525, 1996.

J. C. Ho and J. N. Gupta. Flowshop scheduling with dominant machines. *Computers & Operations Research*, 22(2):237–246, 1995.

W. J. Hopp and M. Spearman. *Factory Physics*. McGraw-Hill, 2008.

N. Hosseini and R. Tavakkoli-Moghaddam. Two meta-heuristics for solving a new two-machine flowshop scheduling problem with the learning effect and dynamic arrivals. *The International Journal of Advanced Manufacturing Technology*, 65(5-8):771–786, 2013.

A. A.-K. Jeng and B. M. Lin. A note on parallel-machine scheduling with deteriorating jobs. *Journal of the Operational Research Society*, 58(6):824–826, 2007.

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

P. J. Kalczynski and J. Kamburowski. On no-wait and no-idle flow shops with makespan criterion. *European Journal of Operational Research*, 178(3):677–685, 2007.

H.-J. Kim and J.-H. Lee. Three-machine flow shop scheduling with overlapping waiting time constraints. *Computers & Operations Research*, 101:93–102, 2019.

Y.-D. Kim. Heuristics for flowshop scheduling problems minimizing mean tardiness. *Journal of the Operational Research Society*, 44(1):19–28, 1993.

X. Li, N. Ye, X. Xu, and R. Sawhey. Influencing factors of job waiting time variance on a single machine. *European Journal of Industrial Engineering*, 1(1):56–73, 2007.

C.-J. Liao, C.-T. Tseng, and P. Luarn. A discrete version of particle swarm optimization for flowshop scheduling problems. *Computers & Operations Research*, 34(10):3099–3111, 2007.

H. Liu, L. Gao, and Q. Pan. A hybrid particle swarm optimization with estimation of distribution algorithm for solving permutation flowshop scheduling problem. *Expert Systems with Applications*, 38(4):4348–4360, 2011.

J. Liu and C. R. Reeves. Constructive and composite heuristic solutions to the $\mathrm{P}//\sum C_i$ scheduling problem. *European Journal of Operational Research*, 132(2):439–452, 2001.

W. Liu, Y. Jin, and M. Price. A new heuristic to minimize system idle time for flowshop scheduling. In *Poster Presented at the 3rd Annual EPSRC Manufacturing the Future Conference, Glassgow*, 2014.

W. Liu, Y. Jin, and M. Price. A new Nawaz-Enscore-Ham-based heuristic for permutation flowshop problems with bicriteria of makespan and machine idle time. *Engineering Optimization*, 48 (10):1808–1822, 2016.

W. Liu, Y. Jin, and M. Price. A new improved NEH heuristic for permutation flowshop scheduling problems. *International Journal of Production Economics*, 193:21–30, 2017.

C. Low, J.-Y. Yeh, and K.-I. Huang. A robust simulated annealing heuristic for flow shop scheduling problems. *The International Journal of Advanced Manufacturing Technology*, 23(9-10):762–767, 2004.

M. Makuchowski. Permutation, no-wait, no-idle flow shop problems. *Archives of Control Sciences*, 25(2):189–199, 2015.

A. G. Merten and M. E. Muller. Variance minimization in single machine sequencing problems. *Management Science*, 18(9):518–528, 1972.

M. S. S. Mir and J. Rezaeian. A robust hybrid approach based on particle swarm optimization and genetic algorithm to minimize the total machine load on unrelated parallel machines. *Applied Soft Computing*, 41:488–504, 2016.

G. Mosheiov. Multi-machine scheduling with linear deterioration. *INFOR: Information Systems and Operational Research*, 36(4):205–214, 1998.

G. Mosheiov. A note: Multi-machine scheduling with general position-based deterioration to minimize total load. *International Journal of Production Economics*, 135(1):523–525, 2012.

M. Nawaz, E. E. Enscore Jr, and I. Ham. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, 11(1):91–95, 1983.

Q.-K. Pan and R. Ruiz. A comprehensive review and evaluation of permutation flowshop heuristics to minimize flowtime. *Computers & Operations Research*, 40(1):117–128, 2013.

P. Perez-Gonzalez and J. M. Framinan. Scheduling permutation flowshops with initial availability constraint: Analysis of solutions and constructive heuristics. *Computers & Operations Research*, 36(10):2866–2876, oct 2009. ISSN 03050548. doi: 10.1016/j.cor.2008.12.018.

P. Perez-Gonzalez, V. Fernandez-Viagas, M. Z. García, and J. M. Framinan. Constructive heuristics for the unrelated parallel machines scheduling problem with machine eligibility and setup times. *Computers & Industrial Engineering*, 131:131–145, 2019.

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

C. Rajendran and D. Chaudhuri. An efficient heuristic approach to the scheduling of jobs in a flowshop. *European Journal of Operational Research*, 61(3):318–325, 1992.

S. Reza Hejazi and S. Saghafian. Flowshop-scheduling problems with makespan criterion: a review. *International Journal of Production Research*, 43(14):2895–2929, 2005.

R. Ruiz and C. Maroto. A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research*, 165(2):479–494, 2005.

R. Ruiz, E. Vallada, and C. Fernández-Martínez. Scheduling in flowshops with no-idle machines. In *Computational Intelligence in Flow Shop and Job Shop Scheduling*, pages 21–51. Springer, 2009.

A. J. Ruiz-Torres, J. C. Ho, and J. H. Ablanedo-Rosas. Makespan and workstation utilization minimization in a flowshop with operations flexibility. *Omega*, 39(3):273–282, 2011.

S. Sarin and M. Lefoka. Scheduling heuristic for the n-job m-machine flow shop. *Omega*, 21(2): 229–234, 1993.

D. Sha and H. H. Lin. A particle swarm optimization for multi-objective flowshop scheduling. *The International Journal of Advanced Manufacturing Technology*, 45(7-8):749–758, 2009.

E. F. Stafford Jr, F. T. Tseng, and J. N. Gupta. Comparative evaluation of MILP flowshop models. *Journal of the Operational Research Society*, 56(1):88–101, 2005.

L. Sun, L. Sun, and J.-B. Wang. Single-machine scheduling to minimize total absolute differences in waiting times with deteriorating jobs. *Journal of the Operational Research Society*, 62(4):768–775, 2011.

W. Szwarc and S. K. Mukhopadhyay. Minimizing a quadratic cost function of waiting times in single-machine scheduling. *Journal of the Operational Research Society*, 46(6):753–761, 1995.

E. Taillard. Some efficient heuristic methods for the flow shop sequencing problem. *European Journal of Operational Research*, 47(1):65–74, 1990.

V. T'kindt and J.-C. Billaut. *Multicriteria scheduling: theory, models and algorithms*. Springer Science & Business Media, 2006.

C. Wang, C. Chu, and J.-M. Proth. Heuristic approaches for $n/m/F/\sum C_i$ scheduling problems. *European Journal of Operational Research*, 96(3):636–644, 1997.

J. Wilson. Alternative formulations of a flow-shop scheduling problem. *Journal of the Operational Research Society*, 40(4):395–399, 1989.

X. Xu. Minimizing weighted waiting time variance on a single processor. *Computers & Industrial Engineering*, 61(4):1233–1239, 2011.

X. Xu and N. Ye. Minimization of job waiting time variance on identical parallel machines. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(5):917–927, 2007.

B. Yagmahan and M. M. Yenisey. Ant colony optimization for multi-objective flow shop scheduling problem. *Computers & Industrial Engineering*, 54(3):411–420, 2008.

D.-L. Yang and M.-S. Chern. A generalized two-machine flowshop scheduling problem with processing time linearly dependent on job waiting-time. *Computers & Industrial Engineering*, 36(2): 365–378, 1999.

N. Ye, X. Li, T. Farley, and X. Xu. Job scheduling methods for reducing waiting time variance. *Computers & Operations Research*, 34(10):3069–3083, 2007.

K.-C. Ying and S.-W. Lin. A high-performing constructive heuristic for minimizing makespan in permutation flowshops. *Journal of Industrial and Production Engineering*, 30(6):355–362, 2013.

T.-S. Yu, H.-J. Kim, and T.-E. Lee. Minimization of waiting time variation in a generalized two-machine flowshop with waiting time constraints and skipping jobs. *IEEE Transactions on Semiconductor Manufacturing*, 30(2):155–165, 2017.

X. Yu, Y. Zhang, and K. Huang. Multi-machine scheduling with general position-based deterioration to minimize total load revisited. *Information Processing Letters*, 114(8):399–404, 2014.

S. Zhou and X. Cai. Variance minimization–relationship between completion-time variance and waiting-time variance. *The ANZIAM Journal*, 38(1):126–139, 1996.