Contents lists available at GrowingScience

Journal of Project Management

homepage: www.GrowingScience.com

A two-stage iterated greedy algorithm and a multi-objective constructive heuristic for the mixed no-idle flowshop scheduling problem to minimize makespan subject to total completion time

Marcelo Seido Nagano[a*] and Fernando Luis Rossi[b]

[a]*Universidade de São Paulo, Departamento de Engenharia de Produção, São Carlos, Brazil*
[b]*Federal Institute of São Paulo, Industrial Engeneering Department, São Paulo, Brazil*

C H R O N I C L E

Article history:
Received: February 8, 2023
Received in revised format: April 20, 2023
Accepted: August 31, 2023
Available online:
August 31, 2023

Keywords:
Mixed no-idle
Makespan
Total completion time
Multi-objective

A B S T R A C T

Advanced production systems usually are complex in nature and aim to deal with multiple performance measures simultaneously. Therefore, in most cases, the consideration of a single objective function is not sufficient to properly solve scheduling problems. This paper investigates the multi-objective mixed no-idle flowshop scheduling problem. The addressed optimization case is minimizing makespan subject to an upper bound on total completion time. To solve this problem, we proposed a two-stage iterated greedy and a multi-objective constructive heuristic. Moreover, we developed a new multi-objective improvement procedure focusing on increasing the performance of the developed methods in solving the addressed problem. and a new initialization procedure. We performed several computational tests in order to compare our developed methods with the main algorithms from similar scheduling problems in the literature. It was revealed that the proposed approaches give the best results compared with other state-of-the-art performing methods.

1. Introduction

The permutation flowshop scheduling problem (PFSP) can be described as a set of jobs that has to be processed on a group of machines. The purpose is to generate a sequence of jobs that optimize the objective function. Decision making in modern production systems generally demand the consideration of complex constraints and several performance objectives, which are often conflicting in nature. The particularities of each manufacturing system translate in different constraints which can be associated with properties of the jobs (due dates, release dates) or the machines (no-idle, setup times). Usually, the main objective is to minimize production costs, which may involve the work-in-process inventory, better utilization of machines, due date penalties. Accordingly, the consideration of two or more objectives at the same time is a common occurrence in real case scenarios. Thus, a multi-criteria approach for the flowshop scheduling problem should be considered in order to generate more realistic solutions (T'kindt & Billaut, 2006; Pinedo, 2016).

Recently, the variation known as the no-idle PFSP (NPFSP) has received much attention from the literature (Ruiz, Vallada, & Fernández-Martínez 2009; Fatih Tasgetiren et al., 2013; Shen, Wang, & Wang, 2015; Zhao et al., 2020). In a NPFSP, as the idle times between the machines is not allowed, the jobs must be processed without stop once started, resulting in the start times of the jobs must be delayed further in the scheduling. The no-idle machines are present in complex and expensive operations that require long and costly setup activities to start the processing of jobs. As observed in (Pan & Ruiz 2014), a flowshop consisting only of no-idle machines is highly unlikely to show up in a real industrial environment, being the

* Corresponding author.
E-mail address: drnagano@usp.br (M. S. Nagano)

doi: 10.5267/j.jpm.2023.9.001

consideration of a mixed no-idle flowshop with both no-idle and regular machines a more realistic approach. An example of this situation is present in the metallurgical and ceramic industries, where some steps require continuous processing while others do not (Pan & Ruiz 2014). While mixed flowshops have been studied before, it was addressed with a single objective criterion (Pan & Ruiz 2014; Rossi & Nagano 2020, 2021a, 2021b). This work investigates the multi criteria mixed no-idle permutation flowshop scheduling problem (MNPFSP) with the minimization of the makespan subject to the total flowtime criterion. Both are important optimization criteria, as minimizing the makespan is crucial to increase the resource use efficiency, and the total flowtime is a relevant objective in reducing the work-in-process inventory (Aydilek & Allahverdi 2012; MACCARTHY and LIU 1993). As a result, these objectives have considerable importance in flowshop problems and are critical in practice and have been extensively studied.

This paper addresses for the first time the MNPFSP with the minimization of the makespan subject to total flowtime. We propose an efficient multi-stage heuristic and a two-stage iterated greedy (IG) algorithm for the addressed problem. The developed heuristic has three phases. The first construct a partial sequence using an index-based function. The second uses a variant of the NEH (Nawaz, Enscore, & Ham 1983) to efficiently generate a complete solution. The last and third phase, apply an improving procedure focusing on satisfying the total flowtime upper-bound. The two-stage iterate greedy algorithm has two stages, where the first stage optimizes the makespan criterion and the subsequent phase improves the solution aiming to satisfy the total flowtime objective. Both makespan and total flowtime criteria are important objectives in modern agile production systems, as the first leads to maximum utilization of resources and the last results in a lower work-in-process inventory (Rajendran & Ziegler 1997). Moreover, we performed computational experiments and adapted several state-of-the-art solution methods from the literature. A total of 17 high performance algorithms were compared in a benchmark proposed by (Pan & Ruiz 2014).

The paper is arranged as follows. In Section 3, an extensive literature review is presented on the multi-objective PFSP and the NPFSP. Section 2 presents the addressed problem. We propose our multi-objective constructive heuristic and the two-stage IG algorithm in Section 4. In Section 5, we test our proposed methods and other algorithms from the literature using extensive computational and statistical experimentation. Finally, the conclusion is presented in Section 6.

2. Literature review

As we are studying the MNPFSP with both makespan and total flowtime criteria, we divided the literature review into two parts: multi-objective scheduling problems and NPFSP/MNPFSP. We focused on presenting studies that addressed the makespan and total flowtime criteria. The objective is to identify the main heuristics and metaheuristics proposed for these scheduling problems.

2.1 Multi-objective PFSP

Most of the attention in studying the PFSP has been directed to the single-criterion scheduling. Nonetheless, in real-world situations two or more objectives must be considered simultaneously. Unfortunately, it is common that these criteria conflict with each-other, resulting in the PFSP with a single objective function being insufficient for most practical situations. Recently, multi-objective scheduling problems have attracted attention of researchers and are currently studied with a multitude of different constraints and criteria ((Yenisey & Yagmahan 2014; Singh, Oberoi, and Singh 2021; Sun et al., 2011). The main focus of this review is papers addressing the variants of the PFSP with multi-objective criteria including makespan, total flowtime and total tardiness objective functions.

The flowshop scheduling problem with the minimization of the makespan and total flowtime criteria was addressed under different constraints by many researchers. (Ravindran et al., 2005) presented three heuristic methods for the problem, namely HAMC1, HAMC2 and HAMC3. Their performance was evaluated against another heuristic procedure by (Rajendran 1995). Varadharajan and Rajendran (2005) also considered the multi-objective problem and developed a metaheuristic procedure based on the simulated annealing algorithm, denoted as MOSA. The MOSA method generated two initial sequences using a heuristic procedure and applied three improvement schemes. Pasupathy et al. (2006) proposed a genetic algorithm (GA), called PGA-ALS, which applies a non-dominated sorting, and a metric for crowding distance is used to rank the generated solutions. (Framinan and Leisten (2008) were the first to propose an IG algorithm for the multi-objective problem. The algorithm iterates over a multi-objective heuristic to generate non-dominated solutions. The IG algorithm was evaluated with the HAMC heuristics from (Ravindran et al., 2005). Yagmahan and Yenisey (2008) presented an ant colony optimization (ACO) algorithm and compared it to the methods developed by Nawaz et al. (1983) and Ho and Chang (1991). Dubois-Lacoste et al. (2011) studied five bi-objective flowshop scheduling problems involving makespan, total flowtime and total tardiness criteria. They used an algorithm which applies several local search methods in conjunction to solve the addressed problems. The presented algorithm was compared with the methods of (Varadharajan and Rajendran 2005) and (Pan and Wang 2008a). Minella, Ruiz, and Ciavotta (2011) analysed two multi-objective problems, makespan/flowtime and makespan/total tardiness. They proposed an IG algorithm with a new method for initializing the population and a greedy phase where only non-dominated partial solutions are maintained. The proposed method was compared to the simulated annealing-based metaheuristic form (Varadharajan and Rajendran 2005). Aydilek and Allahverdi (2012) addressed the no-wait PFSP with makespan and subject total flowtime criteria. They developed a heuristic, called HH1, which is based on the simulated annealing algorithm. The HH1 method significantly outperformed the other methods. Aydilek and Allahverdi

(2013) addressed the no-wait PFSP with the minimization of makespan and mean completion time criteria. The authors proposed two heuristics and compared them against the HH1 heuristic from (Aydilek & Allahverdi 2012). Ciavotta et al. (2013) analysed the multi-criteria PFSP with setup times. They presented an algorithm based on a new initialization method and an iterated greedy procedure. Their algorithm outperformed the IG method from (Framinan & Leisten 2008). Shao et al. (2019) proposed a metaheuristic based on the water wave optimization algorithm for the multi-objective blocking PFSP with makespan and total flow time minimization criteria. The authors developed an initialization procedure-based decomposition and a ranking-based operator to direct local exploitation and global exploration. The proposed algorithm was compared NSGA-II (Deb et al., 2002), DDE (Pan, Wang, and Qian 2009), RIPG (Ciavotta, Minella, and Ruiz 2013) and HMOBSA (Lu et al., 2017). Marichelvam et al. (2018) proposed a hybrid crow search algorithm to solve the multi-objective PFSP. The algorithm was compared to traditional metaheuristics as the simulated annealing algorithm, genetic algorithm, artificial bee colony algorithm, among others. (Ye, Li, and Nault 2020) studied the no-wait multi-objective PFSP and proposed heuristic which uses a reconstruction method coupled with a local search based on reinsertion movements. (Marcelo Seido Nagano, Almeida, and Miyata 2021) also addressed no-wait PFSP, and proposed a method based on the NEH heuristic and the iterate greedy algorithm, denoted as G6. The GL method iterates times applying the NEH procedure and the IG algorithm. The algorithm was compared with the HH1 method from (Aydilek and Allahverdi 2012). The results show that the proposed approach offered superior solutions when compared with the HH1 algorithm.

It is worth noting that several papers addressed different objective function combinations (makespan, total flowtime, tardiness) under various constraints (no-wait, setup times, blocking). As a result, many methods were developed to solve these extensions of the multi-criteria flowshop scheduling problem: heuristics algorithms (Allahverdi 2004; Jose M. Framinan and Leisten 2006; W. Liu, Jin, and Price 2016), genetic algorithms (Deb et al., 2002; Arroyo and Armentano 2005; Shahsavari Pour, Tavakkoli-Moghaddam, and Asadi 2013; Allahverdi and Aydilek 2013; Allali, Aqil, and Belabid 2022), discrete differential evolution (DDE) algorithms (Pan, Wang, & Qian 2009), simulated annealing (Jarosław, Czesław, and Dominik 2013; Allahverdi, Aydilek, & Aydilek 2018, 2020), iterated greedy algorithms (Ruiz and Allahverdi 2009; Aqil and Allali 2021; Almeida & Nagano 2022), tabu search (Arabameri & Salmasi 2013), among other metaheuristics approaches (Rahimi-Vahed et al., 2008; Rifai, Nguyen, and Dawal 2016; Cho & Jeong 2017; Deng & Wang 2017; Marichelvam, Azhagurajan, & Geetha 2017; Keskin & Engin 2021). Furthermore, energy efficiency based multi-objective problems have also been studied for the PFSP (Jian-Ya Ding, Song, & Wu 2016; Lu et al., 2017; Chen, Wang, & Peng 2019; Fatih Tasgetiren et al., 2019; Wu & Che 2020; Cheng et al., 2021).

2.2 No-idle and mixed no-idle PFSP

In this section, we highlight the main works dealing with the NPFSP with makespan and total flowtime objectives. Early studies on the subject focused on developing exact methods, proposing mathematical models and investigating special properties of the problem (Vachajitpan 1982; Baptiste & Hguny, 1997; Bagga 2003; Kamburowski 2004; Pawel Jan Kalczynski and Kamburowski 2007). Special extensions, such as the three-machine no-idle flowshop were also addressed by the literature (Bagga 2003; Saadani, Guinet, & Moalla 2003; Kamburowski 2004). A heuristic algorithm based on the Travelling Salesman Problem (TSP) was proposed by (Saadani, Guinet, & Moalla 2005). (Pawel Jan Kalczynski and Kamburowski 2005) presented a heuristic based on the (Johnson 1954) rule. In another work, (Pawel Jan Kalczynski and Kamburowski 2007) showed that the no-idle and no-wait flowshops are related to scheduling problems. (Baraz and Mosheiov 2008) presented a two-stage heuristic, which performed better than the algorithm from (Saadani, Guinet, and Moalla 2005). Firstly, the jobs are appended one by one at the final position of the partial sequence being the chosen job, the one that results in the lowest markspan. The second stage, the sequence is improved by a simple procedure based on pairwise job interchange.

Recent developments in the NPFSP algorithms were presented by (Pan and Wang 2008b), where they proposed a hybrid discrete particle swarm algorithm (HDPSO). The HDPSO algorithm outperformed the methods from (Pawel Jan Kalczynski and Kamburowski 2005), (Baraz & Mosheiov 2008) and (Fatih Tasgetiren et al., 2007). The same authors developed a discrete differential evolution algorithm (DDE) in (Pan & Wang 2008a). The method was tested against the algorithms from (Pawel Jan Kalczynski & Kamburowski 2005) and (Baraz & Mosheiov 2008). (Ruiz, Vallada, & Fernández-Martínez 2009) developed an IG algorithm and heuristics procedures. The methods that obtained the best performance were the FRB3 heuristic ((Rad, Ruiz, & Boroojerdian 2009)), as well a modified version of the GH-BM from (Baraz & Mosheiov 2008), called GH-BM2. The IG algorithm surpassed in solution quality the algorithms from (Pan and Wang 2008a), (Pan and Wang 2008b) and (Rad, Ruiz, & Boroojerdian 2009). (Goncharov & Sevastyanov 2009) presented an approximation algorithm which ensures a theoretical performance. Another Discrete Differential Evolution (HDDE) algorithm was presented by (G. Deng and Gu 2012) and compared to algorithms from (Rad, Ruiz, & Boroojerdian 2009), (Pan & Wang, 2008b) and (Pan and Wang 2008a). Fatih Tasgetiren et al. (2013) developed an algorithm based on both the IG procedure and differential evolution operators, which showed superior results when compared to the metaheuristics from (Pan & Wang, 2008a), (Pan & Wang, 2008b) and (Deng & Gu, 2012). (Zhou, Chen, & Zhou 2014) presented an Invasive Weed Optimisation algorithm (IWO), which was compared to other particle swarm optimization algorithms ((Fatih Tasgetiren et al., 2007; Pan and Wang 2008b) and to heuristics algorithms (Pawel Jan Kalczynski & Kamburowski 2005; Baraz & Mosheiov 2008). More recently, (W. Shao, Pi, & Shao 2017) developed a complex memetic algorithm with a hybrid node and edge histogram (MANEH), which was compared to the methods from (Pan & Wang 2008b), (Pan & Wang 2008a), (Ruiz, Vallada, & Fernández-Martínez 2009), (Deng & Gu 2012) and (Fatih Tasgetiren et al., 2013).

Constructive heuristics were also developed for the problem. The LR(x) proposed by (J. Liu and Reeves 2001) is among one the most used methods to solve the PFSP with total flowtime. The LR(x) procedure appends jobs one by one at the end of the sequence until a complete solution is generated. The method generates x sequences by selecting different jobs to be the first appended in the partial sequence. Among the x generated solutions, the method chooses the one that resulted in the best total flowtime. A variant of LR(x) was developed by (Pan and Ruiz 2013), called LR-NEH(x), which combines the LR(x) and NEH methods. More recently, (Fernandez-Viagas and Framinan 2015) developed a variant of the LR(x) procedure, denoted as FF(x). The FF(x) was combined with the LR-NEH(x), resulting in a new extension of the LR-NEH(x) heuristic called FF-NEH(x). (Li, Wang, and Wu 2009) also developed three variants of the LR(x) heuristics, called ICH1, ICH2 and ICH3. They applied new local search procedures based on insertion and permutations movements on the sequence generated by the LR(x) to further increase the solution quality. An improved version of the LR-NEH(x) was proposed by (Pan and Ruiz 2013), where an improvement scheme based on insertion movements is used after the LR-NEH(x) method. Recently, (Fernandez-Viagas, Leisten, and Framinan 2016) presented the FF-ICH1 and FF-PR1 algorithms, which are extensions of the ICH1 and PR1(x) methods by using FF(x) and FF-NEH(x) heuristics instead of the LR(x) and LR-NEH(x) methods by in both algorithms, respectively.

The important variant which considers a MNPFSP addressed in this paper was first considered by (Pan and Ruiz 2014). The authors presented a MILP model and an IG algorithm with referenced local search improvement procedure (RLS). The computational results show that the IG-RLS outperformed several high-performance algorithms from the literature (Pan & Wang 2008b, 2008a; Deng & Gu, 2012; Ruiz, Maroto, & Alcaraz 2006).

From the literature review, it can be noted that, despite the significant number of papers addressing both the NPFSP and the multi-objective PFSP, the consideration of a MNPFSP with a multi-objective criterion has not yet been addressed in the literature. In the next section, we define the addressed problem in more detail.

3. Problem definition and notations

The MNPFSP with makespan minimization subject to the total flowtime criterion can be defined as $F_m|prmu,\ mixed\ no-idle|C_{max} \setminus \sum C_j$ (Graham et al. 1979)). The definitions and notations are presented as follows:

- The flowshop contains m machines (F_m);
- The order of the jobs is maintained for all machines, resulting in a permutation flowshop ($prmu$);
- The set M' defines the no-idle machines;
- J_j is job j on machine M_i;
- Job j has a processing time of $p_{i,j}$ on machine M_i;
- π is the sequence of jobs;
- Job π_j or] is placed at the jth position of the sequence π;
- $C_{i,j}$ is the completion time for job J_j in M_i and the start time is denoted as $S_{i,j}$;
- C_{max} or makespan denotes the completion time of the last machine for the last job ($C_{m,n}$);
- $\sum C_j$ or total flowtime is the sum of the completion times on the last machine ($\sum_{j=1}^{n} C_{m,j}$)
- The makespan and the total flowtime for a sequence π can be denoted as C_{max} and $TFT(\pi)$, respectively.

4. Proposed algorithms

In this section, we present the two-stage IG algorithm and the heuristic algorithm in more details.

4.1 A multi-objective constructive heuristic

Constructive heuristics are known to generate good solutions with computational efficiency (Ribas, Companys, & Tort-Martorell 2017). As a result, they have been applied successfully for a variety of scheduling problems (Nagano & Moccellin 2002; Framinan & Leisten 2003; Dong, Huang, & Chen 2008; Kalczynski & Kamburowski 2008; Laha & Sarin 2009; Ribas, Companys, & Tort-Martorell 2010; Pan & Ruiz 2013; Fernandez-Viagas & Framinan 2014; Rossi, Nagano, & Sagawa 2017; Marcelo Seido Nagano, Rossi, & Tomazella 2017; Marcelo Seido Nagano, Rossi, & Martarelli 2019). In this work we propose a novel multi-objective constructive heuristic, denoted as MOH. In more detail, the MOH method construct sequences with L jobs ($L \leq n$) using an index based procedure. At each iteration the procedure evaluates all jobs using an index, and the one that results in the lowest value for the index is chosen to be inserted at the last position of the partial sequence. The The rest of the $n - L$ jobs are inserted using an improved variant of the NEH heuristic the rest is and the jobs.

We propose an index which represents the main proprieties of the addressed multi-objective problem. The principle is choosing jobs to be appended that minimizes both makespan and total flowtime objectives. Therefore, we incorporate three measures in our index: (i) idle time between the jobs, (ii) makespan, (iii) total flowtime.

In more detail, a partial sequence with k jobs $\pi = (\pi_1, \dots, \pi_k)$, being J_j the job to be tested at the last position $(k + 1)$ of the sequence π. The index $v_{k,j}$ and is calculated as follows:

$$v_{k,j} = \sum_{2}^{m} m\,ax(S_{j,[k+1]} - C_{j,[k-1]}, 0) + C_{m,[k]}; + \sum_{1}^{k+1} C_{m,[j]}$$

The procedure tests all jobs and chooses the one that results in the lowest value of $v_{k,j}$ to be appended in the sequence. As we are dealing the a multi-objective problem, the principle is that the job which offers a good trade-off between the measures (i), (ii) and (iii) should be selected to be inserted at the last position of the sequence.

In order to construct the rest of the sequence we resort to the construction method presented in (Rossi and Nagano 2021b). The proposed heuristic orders the unscheduled jobs by the non-ascending greater standard deviation (STD) of the processing times of the jobs. Moreover, reinsertions movements are applied in the partial sequence generated at the end of each NEH insertion procedure, where two adjacent jobs are removed, π_k and π_{k+1} ($k = \{1, \dots, n-1\}$), from the sequence, and the same jobs are inserted again in the best possible position of the sequence (lowest value of makespan). The procedure continues by reinserting pairs until the last pair $\pi n - 1$ and π_{n-2} is considered. The principle is to generate a better optimization of the sequences. Similar procedures have been used with great success for similar problems (Laha and Sarin 2009; Rad, Ruiz, and Boroojerdian 2009; Rossi, Nagano, and Neto 2016). In order to increase the computational efficiency, we select a limited set of jobs to the reinserted, where parameter x limits the number of jobs selected to be reinserted.

As we are dealing with the minimization of the makespan and subject to total flowtime objective, it is possible that the solution generated does not result in a total flowtime value lower than the established upper bound. In order to circumvent this problem, we developed a Multi-objective Improvement Procedure (MIP), which focuses on minimizing the total flowtime without compromising the makespan objective. This applies a construct-destruction improvement scheme that iterates until the total flowtime results in a lower value than the upper-bound defined by the scheduler. For the scheduler used to generate the upper-bound for the total flowtime, we choose the NEH heuristics (Nawaz, Enscore, & Ham 1983), as it is a simple method that generates tight upper-bounds for the problem with computational efficiency. Algorithm 1 presents the pseudocode of the MOH$_x$ heuristic and Algorithm 2 details the MIP procedure.

Algorithm 1 Multi-objective Constructive Heuristics MOHx algorithm

```
Calculate the upper-bound for the TFT, UB(TFT)
$\pi = \Phi$
$U = \{J_1, \dots, J_n\}$
For $k = 1$ to $J$ do
    Select the job $J_j \mathcal{E} U$ with the lowest $\delta_{kj}$ (Eq. (1))
    Place it at the end of $\pi$.
    $U = U - J_j$
End for
Order the jobs in $U$ according to the non-ascending order of $STD_j$ generating $\alpha = \{\alpha_1, \dots, \alpha_{n-d}\}$.
For $l = 1$ to $n-d$ do
    Insert job $\alpha_l$ in $\pi$ in the position $b$ that results in the lowest $Cmax$
    For $k = \max(1, b-x)$ to $\min(l, b+x)$, step $k = k + 2$ do
        $\pi' = \pi$
        Remove the jobs $\pi'_k$ and $\pi'_{k+1}$ from $\pi'$.
        Insert the job $\pi'_k$ in the position that results in the lowest Cmax.
        Insert the job $\pi'_{k+1}$ in the position of that results in the lowest Cmax.
        If Cmax ($\pi'$) ¡Cmax ($\pi$) then
            $\pi = \pi'$
        end if
    end for
end for
if $TFT(\pi) \geq UB(TFT)$ then
    Apply the MIP on the $\pi$ solution (Algorithm 2).
end if
Return $\pi$.
```

Algorithm 2 Multi-objective Improvement Procedure (MIP)

```
Define $\pi^s$ as the sequence generated by the scheduler.
Calculate the makespan of $\pi^s$ ($UB(Cmax)$),
$counter = 0$
while TFT($\pi$) > UB (TFT) and $counter < n$ do
    Remove $L$ jobs from $\pi$
    Insert the removed jobs in position of the results in the lowest TFT.
    Calculate the new TFT($\pi$) value.
    $Counter = counter + 1$
end while
Calculate the makespan of the sequence $\pi$.
If $Cmax(\pi) < UB (Cmax)$ then
    $\pi = \pi^s$
end if
Return $\pi$.
```

4.2 The Two-stage IG Algorithm

The IG algorithm iterates through a destruction-reconstruction operator, where a set of jobs is removed from the sequence and then inserted again in the solution. This destruction-reconstruction strategy for solving flowshop scheduling problems has various advantages: as a general-purpose method, it is easy adapted or and has high applicability for a difference set of problems; it is easy to implement, as it has a simple structure; the main parameters are simple and intuitive; and it easily integrates local search methods within its structure. As a result, the IG algorithm has been successfully applied for several flowshop problems (Ruiz & Stützle 2007, 2008; Ding et al., 2015; Tasgetiren et al., 2017). Moreover, IG algorithms have been used in multi-objective problems (Minella, Ruiz, & Ciavotta 2011; Dubois-Lacoste, Lopez-Ibanez, & Stutzle 2011; Ciavotta, Minella, & Ruiz 2013).

In this work, we propose a two-stage IG algorithm, denoted as IG-2S, where the first phase is an improved IG algorithm based on the method proposed by (Pan & Ruiz 2014), and the second phase applies the same improving procedure in case the method results in an infeasible solution, where the resulting total flowtime is not lower than its upper-bound.

As mentioned before, Pan and Ruiz (2014) proposed an IG algorithm, denoted IG-RLS, for the MNPFSP with makespan minimization. A new destruction-reconstruction operator is applied, with a reinsertion procedure and a referenced local search (RLS). In more detail, the IG generates an initial solution using a constructive heuristic, which is subsequently perturbed with a destruction-construction procedure. The destruction phase involves the random removal of jobs from the solution. The removed jobs are then inserted again in sequence, resulting in a reconstruction procedure. A referenced local search is used to improve the solution with a simple simulated annealing phase.

In the proposed IG algorithm, we implemented simple and effective methods to improve the performance. In the first stage the original IG algorithm is applied. To improve the IG-RLS algorithm, we propose to use the MOH heuristic to generate the initial solution, as a more robust initialization procedure results in higher quality solutions generated by the IG algorithm (Fernandez-Viagas, Valente, & Framinan, 2018). As we are addressing a multi-objective problem, with makespan subject to the total flowtime, the first stage is dedicated to minimizing the makespan. Therefore, there is a chance that the final solution results in a total flowtime higher than the upper-bound defined by scheduler, as we only minimized the makespan in the first phase. For that motive, we employ in the second stage, the Multi-objective Improvement Procedure (Algorithm 2) focusing on minimizing the total flowtime of the solution. In this way, the IG algorithm optimizes the makespan without compromising the total flowtime upper-bound. All these changes help to solve the PFSP with a multi-criteria objective function.

5. Computational experiments

5.1 Performance evaluation

The algorithms are tested considering two performance measures: (i) solution quality and (ii) efficiency in terms of computational time. The average relative percentage deviation, which is denoted as ARPD(y) measures the solution quality of algorithm y for a set of T problem instances (Eq. (1)). In order to obtain the ARPD value we need to calculate the relative percentage deviation RPD(i,y) for each instance i obtained by heuristic y (Eq. (2)). $C_{max}(i, y)$ is the makespan obtained by the method y for instance i. $C_{max}(i)$ is the best makespan for instance i. RPD measures the distance between the solution obtained by the method and the best solution for the instance.

$$ARPD(y) = \sum_{i=1}^{I} \frac{RPD(i,y)}{T} \tag{1}$$

$$RPD(i,y) = 100 \cdot \frac{C_{max}(i,y) - C_{max}(i)}{C_{max}(i)} \tag{2}$$

The average computational time ACT, which measures the computational efficiency, can be calculate as follows:

$$ACT_i = \sum_{h=1}^{H} \frac{T_{i,h}}{H}$$

All methods were analyzed with the same upper bounds, which were generated by using the NEH method from (Nawaz, Enscore, and Ham 1983). For the metaheuristics, the maximum elapsed CPU time (CPU_{max}) in milliseconds is defined by the following expression:

$$CPU_{max} = \left(n \cdot (m/2.00)\right) \cdot 100$$

The algorithms were implemented in C++ (Intel oneAPI DPC++ 2021 1.2 with O2 flag) and implemented on a PC with an Intel(R) Core(TM) i9-9900k 5.00 GHz CPU and 32 GB RAM.

5.2 Benchmark Instances

We used the well-known set of problems instances proposed by (Pan and Ruiz 2014) for the MNPFSP. This testbed has seven sets of problems instances, each one with a mixed no-idle scenario:

- (1): 100% of no-idle machines;
- (2): The first 50% machines are no-idle machines;
- (3): Inverse of set (2);
- (4): The machines switch between regular and no-idle;
- (5): Randomly 25% of the no-idle machine;
- (6): Randomly 50% of the no-idle machines;
- (7): Randomly 50% of the no-idle machines;.

Each problems instance is generated by the combination of several jobs (50, 100, 150, 200, 250, 300, 350, 400, 450, 500) and a number of machines (10, 20, 30, 40, 50), resulting in a total of 1750 problems instances. The uniform distribution $U[1,99]$ was used to generate the processing times of the jobs.

5.3 Compared methods

Through the extensive literature review presented in Section 2 we were able to identify the best methods available in the literature for related problems. The GL algorithm proposed by (Marcelo Seido Nagano, Almeida, and Miyata 2021) is one of the most recent high-performance methods developed for multi-objective flowshop scheduling. The GL method is based on the IG algorithm and iterates the NEH and a simple IG procedure over L times. We implemented a version that is limited by a maximum elapsed computational time parameter instead of iterating over a specific number of times. We denoted this version as IG-NAM, in this work and compared it with other IG algorithms. From the no-idle PFSP we adapted the best methods recently proposed for the problem. The following list presents the methods selected for comparison:

- Proposed solution methods;
  - MOH(x);
  - IG-2S.
- Heuristics adapted from the literature:
  - GH-BM2 (Ruiz, Vallada, and Fernández-Martínez 2009) (no-idle PFSP);

- FRB3 and FRB4k(Rad, Ruiz, and Boroojerdian 2009) (PFSP with makespan objective);
- FF-PRI1(x) and FF-ICH1(x) (Fernandez-Viagas and Framinan 2015) (PFSP with total flowtime objective);
- GL (L = 6) (Marcelo Seido Nagano, Almeida, and Miyata 2021) (multi-objective PFSP).

- IG algorithms adapted from the literature:
  - IG-RLS (Pan and Ruiz 2014) (NMPFSP);
  - IG-NAM (Marcelo Seido Nagano, Almeida, and Miyata 2021) (multi-objective PFSP).

5.4 Computational results

Computational results of heuristics comparison for the (Pan and Ruiz 2014) benchmark are presented in Table 1. Table 2 shows results grouped by number of machines. The average computational time (ACPU) is shown in Table 3. As noted, MOH$_{70}$ resulted in the best ARPD for almost all subsets of problems, as indicated by the ARPD values. Nevertheless, computational times increase significantly as the number of jobs selected for reinsertion by parameter x increases ($x = 10,30,50,70$). For the proposed methods and those adapted from the literature, the best trade-off between solution quality and average computational time is achieved by FRB4, G6 and FF-PR1(10). The methods FRB3, MOH$_{50}$ and MOH$_{70}$ obtained the best overall ARPD values of 2.62, 2.65 and 2.45, respectively. At the same time, they prove to be very efficient as their overall ACPU values are around 6.69, 7.25 and 7.43 seconds, respectively. Fig. 1 shows the Pareto chart with ARPD vs ACPU for the heuristics, with the Pareto front that shows a set of most efficient solutions methods. These results show that the use of the Multi-Objective Improvement Procedure allows the proposed constructive heuristic MOH$_x$ to generate better solutions to solve a multi-criteria problem when compared to other adapted methods from the literature.

Table 1
APRDs obtained by the heuristics. The best ARPDs are emphasized in bold.

n	GH-BM2	FRB3	FRB4$_5$	FRB4$_{10}$	FRB4$_{15}$	G6	FF-ICH1(5)	FF-ICH1(10	FF-PR1(5)	FF-PR1(10)	MOH$_{10}$	MOH$_{30}$	MOH$_{50}$	MOH$_{70}$
50	7.30	4.89	7.08	6.23	5.59	5.89	6.20	6.20	5.90	5.78	5.56	4.65	4.38	**4.36**
100	6.66	4.22	6.44	5.93	5.46	5.73	5.36	5.36	4.78	4.71	4.84	4.05	3.70	**3.60**
150	5.87	3.26	5.80	5.10	4.63	5.20	4.51	4.51	3.93	3.91	3.99	3.41	3.10	**3.04**
200	5.19	2.91	5.20	4.41	4.18	4.57	3.71	3.71	3.11	3.10	3.50	2.93	2.74	**2.56**
250	4.37	2.24	4.35	3.85	3.49	3.97	3.25	3.25	2.91	2.82	3.10	2.50	2.27	**2.20**
300	4.20	2.17	4.00	3.67	3.34	3.89	2.84	2.84	2.49	2.47	2.73	2.22	2.13	**1.92**
350	3.89	1.83	3.65	3.21	3.04	3.49	2.60	2.60	2.22	2.21	2.44	1.95	1.74	**1.69**
400	3.24	1.63	3.11	2.92	2.61	2.95	2.24	2.24	1.86	1.83	1.98	1.69	1.55	**1.40**
450	3.20	1.60	3.17	2.70	2.47	2.98	2.13	2.13	1.78	1.76	1.92	1.60	1.48	**1.45**
500	3.12	1.50	2.84	2.59	2.42	2.83	1.91	1.91	1.55	1.52	1.86	1.49	1.39	**1.25**
ARPD	4.71	2.62	4.56	4.06	3.72	4.15	3.47	3.47	3.05	3.01	3.19	2.65	2.45	**2.35**

Table 2
APRDs obtained by the heuristics grouped by the number of machines. The best ARPDs are emphasized in bold.

n	GH-BM2	FRB3	FRB4$_5$	FRB4$_{10}$	FRB4$_{15}$	G6	FF-ICH1(5)	FF-ICH1(10	FF-PR1(5)	FF-PR1(10)	MOH$_{10}$	MOH$_{30}$	MOH$_{50}$	MOH$_{70}$
10	2.75	2.51	2.71	2.76	2.65	2.71	1.38	1.38	1.60	1.46	1.48	1.31	1.29	**1.19**
20	3.95	2.59	3.91	3.55	3.26	3.74	2.45	2.45	1.95	1.90	2.39	2.02	1.89	**1.87**
30	4.77	**2.18**	4.66	4.09	3.62	4.22	3.54	3.54	3.09	3.08	3.27	2.68	2.43	2.33
40	5.72	**2.63**	5.45	4.72	4.30	4.75	4.58	4.58	3.97	3.95	4.13	3.41	3.07	2.89
50	6.34	**3.21**	6.08	5.19	4.78	5.33	5.42	5.42	4.67	4.67	4.69	3.83	3.56	3.46
ARPD	4.71	2.62	4.56	4.06	3.72	4.15	3.47	3.47	3.05	3.01	3.19	2.65	2.45	**2.35**

Table 3
ACPU of the compared heuristics methods.

n	GH-BM2	FRB3	$FRB4_5$	$FRB4_{10}$	$FRB4_{15}$	G6	FF-ICH1(5)	FF-ICH1(10	FF-PR1(5)	FF-PR1(10)	MOH_{10}	MOH_{30}	MOH_{50}	MOH_{70}
50	0.01	0.02	0.01	0.01	0.01	0.02	0.02	0.02	0.04	0.07	0.01	0.02	0.02	0.02
100	0.05	0.17	0.05	0.07	0.08	0.09	0.09	0.11	0.25	0.46	0.08	0.12	0.16	0.18
150	0.15	0.59	0.17	0.19	0.22	0.25	0.26	0.31	0.77	1.47	0.28	0.37	0.49	0.58
200	0.35	1.41	0.38	0.42	0.48	0.55	0.55	0.65	1.76	3.38	0.77	1.03	0.92	1.45
250	0.68	2.72	0.72	0.79	0.89	1.00	1.00	1.19	3.38	6.59	1.06	1.75	1.76	2.14
300	1.20	4.76	1.23	1.35	1.49	1.66	1.62	1.87	5.88	11.37	3.63	4.04	4.39	4.79
350	1.88	7.63	1.95	2.14	2.32	2.59	2.52	2.83	9.35	18.13	6.67	7.34	7.76	8.09
400	2.80	11.24	2.95	3.17	3.37	3.78	3.64	4.06	14.25	27.46	9.11	10.15	11.12	11.62
450	4.05	16.08	4.22	4.51	4.70	5.28	5.19	5.74	20.32	38.89	16.36	18.22	20.49	21.69
500	5.64	22.32	5.79	6.05	6.42	7.22	7.02	7.68	27.85	53.70	20.45	21.82	25.33	23.71
ACPU	1.68	6.69	1.75	1.87	2.00	2.24	2.19	2.45	8.38	16.15	5.84	6.49	7.25	7.43

We also performed statistical tests to verify whether the difference between the obtained ARPDs is statistically significant. We present a plot of the ARPD values with error bars with 95% confidence intervals. If the error bars overlap, there is no significant statistical difference between the heuristics. Fig. 2 presents that the differences among ARPD between the MOH_{70} the other heuristics is statistically significant.

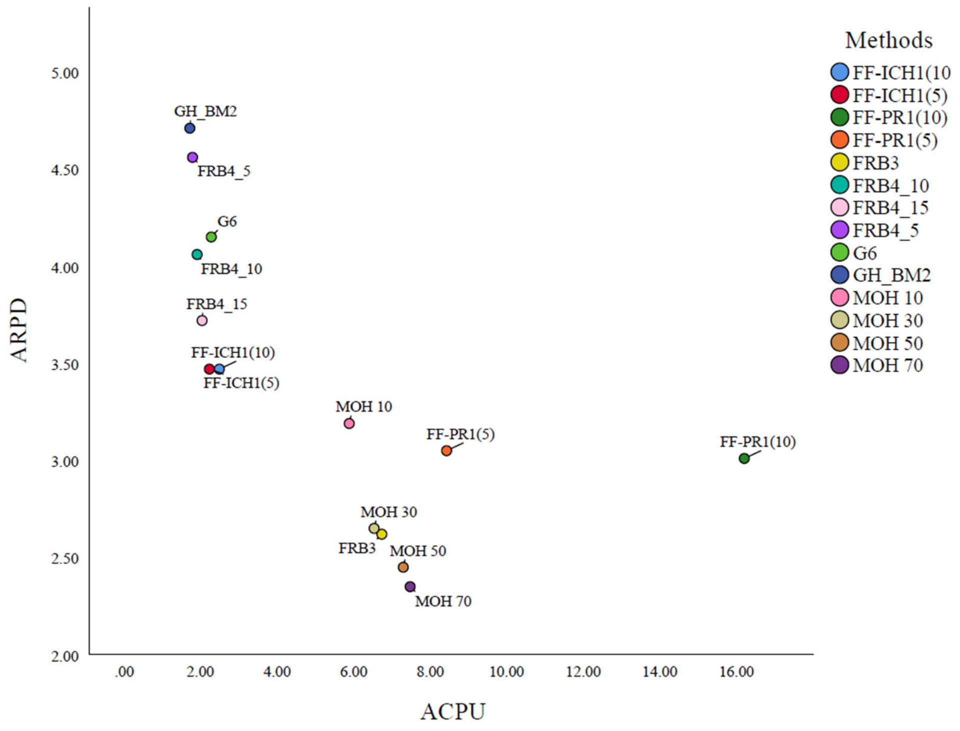

Fig. 1. Pareto chart with ARPD vs ACPU for the heuristics.

Tables 4 and Table 5 show the results for the iterated greedy algorithms IG-NAM, IG-RLS and IG-2S. From the results, the best results are achieved by IG-2S with an ARPD of 0.29, which outperforms the ARPD from IG-RLS and IG-NAM of 0.67 and 0.84, respectively. Also, IG-2S obtains the best ARPD for all subsets of problems, for both cases, when the sets are grouped by number of jobs (Table 4) and number of machines (Table 5). As mentioned, the main difference of the IG-2S is the inclusion of a second stage focusing on addressing the total flowtime objective in case its upper-bound is not satisfied. From the results, the IG-2S algorithm benefits greatly from this strategy, as we can see that the solutions generated by the proposed IG are far superior to those obtained by the other compared IG algorithms. For this comparison, we also performed statistical tests with means to verify if they are statistically different. The results are shown in Fig. 3 and Fig. 4.

Fig. 4 show that the means are statically different for almost every set of problems. These results confirm that the IG-2S is an effective method for solving the multi-objective MNPFSP.

Table 4
APRDs obtained by the metaheuristics. The best ARPDs are emphasized in bold.

n	IG-NAM	IG-RLS	IG-2S
50	0.65	0.33	**0.25**
100	1.02	0.74	**0.28**
150	1.00	0.69	**0.36**
200	1.01	0.83	**0.32**
250	0.90	0.73	**0.28**
300	0.75	0.70	**0.25**
350	0.82	0.69	**0.32**
400	0.65	0.59	**0.24**
450	0.81	0.67	**0.30**
500	0.81	0.73	**0.32**
ARPD	0.84	0.67	**0.29**

Table 5
APRDs obtained by the metaheuristics grouped by the number of machines. The best ARPDs are emphasized in bold.

m	IG-NAM	IG-RLS	IG-2S
10	1.73	1.95	**0.65**
20	0.74	0.85	**0.37**
30	0.43	0.23	**0.15**
40	0.57	0.15	**0.11**
50	0.75	**0.17**	0.18
ARPD	0.84	0.67	**0.29**

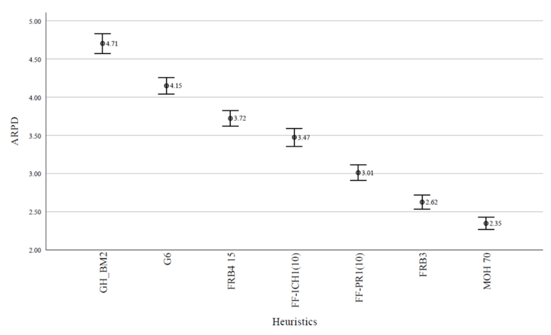

Fig. 2. Means plot for the heuristics (95% confidence intervals).

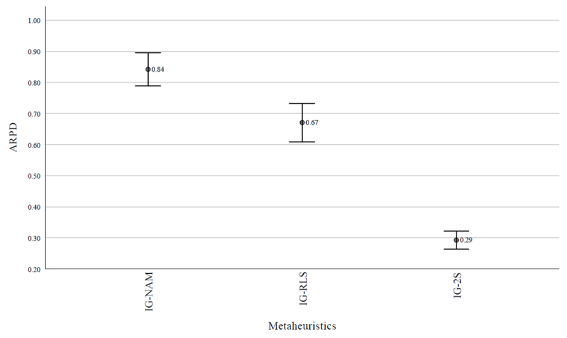

Fig. 3. APRD for the metaheuristics (95% confidence intervals).

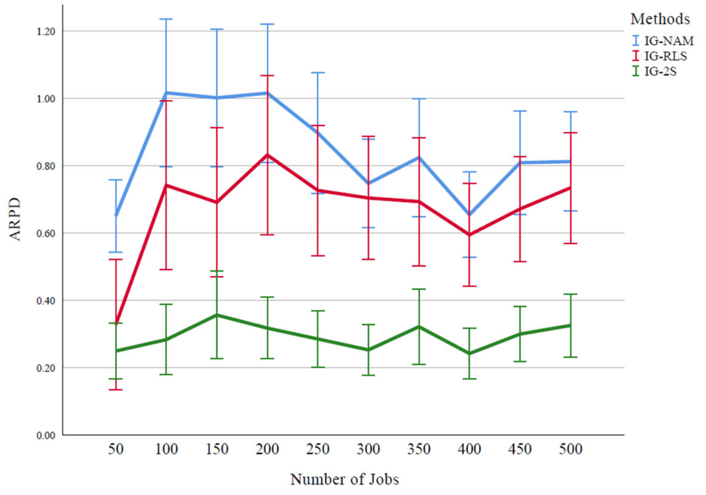

Fig. 4. APRD for the metaheuristics for different number of jobs (95% confidence intervals).

6. Conclusions

We addressed the multi-objective MNPFSP with the objective of minimizing makespan subject to an upper bound on total flowtime. As far as we know, this is the first time that the MNPFSP is studied with a multi-criteria approach. A two-stage iterated greedy (IG-2S) and a new heuristic (MOHx) were proposed to solve the problem. In order to generate a solid computational experimentation for the novel method, we adapted several heuristics and two high performance IG algorithms. Computational results demonstrate the effectiveness of the proposed methods through extensive comparisons. In conclusion, the novel Multi-objective Constructive Heuristic (MOH) and the Two-Stage Iterated Greedy algorithm (IG-2S) can be considered a contribution to the state-of-the-art in solution methods for the multi-objective scheduling problem addressed in this paper. For future research, the proposed algorithms could help solving other similar scheduling problems, as one clear advantage of the proposed algorithms is that they are of easy implementation and adaptation for other multi-objective scheduling problems.

Acknowledgement

This research is partially supported by grant number 312585/2021-7 from the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Brazil.

References

Allahverdi, A. (2004). A new heuristic for m-machine flowshop scheduling problem with bicriteria of makespan and maximum tardiness. *Computers & Operations Research, 31*, 157–80. https://doi.org/10.1016/S0305-0548(02)00143-0.

Allahverdi, A., & Aydilek, H. (2013). Algorithms for no-wait flowshops with total completion time subject to makespan. *The International Journal of Advanced Manufacturing Technology, 68*. https://doi.org/10.1007/s00170-013-4836-x.

Allahverdi, A., Aydilek, H., & Aydilek, A. (2018). No-wait flowshop scheduling problem with two criteria; total tardiness and makespan. *European Journal of Operational Research, 269*(2), 590–601. https://doi.org/https://doi.org/10.1016/j.ejor.2017.11.070.

———. 2020. No-wait flowshop scheduling problem with separate setup times to minimize total tardiness subject to makespan. *Applied Mathematics and Computation, 365*, 124688. https://doi.org/https://doi.org/10.1016/j.amc.2019.124688.

Allali, K., Aqil, S., & Belabid, J. (2022). Distributed no-wait flow shop problem with sequence dependent setup time: Optimization of makespan and maximum tardiness. *Simulation Modelling Practice and Theory*, *116*, 102455. https://doi.org/https://doi.org/10.1016/j.simpat.2021.102455.

Almeida, F. S. D., & Nagano, M. S. (2023). Heuristics to optimize total completion time subject to makespan in no-wait flow shops with sequence-dependent setup times. *Journal of the Operational Research Society*, *74*(1), 362-373. https://doi.org/10.1080/01605682.2022.2039569.

Aqil, S., & Allali, K. (2021). On a bi-criteria flow shop scheduling problem under constraints of blocking and sequence dependent setup time. *Annals of Operations Research, 296*(1), 615–37. https://doi.org/10.1007/s10479-019-03490-x.

Arabameri, S., & Salmasi, N. (2013). Minimization of weighted earliness and tardiness for no-wait sequence-dependent setup times flowshop scheduling problem. *Computers & Industrial Engineering*, *64*(4), 902-916. https://doi.org/https://doi.org/10.1016/j.cie.2012.12.023.

Arroyo, J. E. C., & Armentano, V. A. (2005). Genetic local search for multi-objective flowshop scheduling problems. *European journal of operational research*, *167*(3), 717-738. https://doi.org/https://doi.org/10.1016/j.ejor.2004.07.017.

Aydilek, H., & Allahverdi, A. (2012). Heuristics for no-wait flowshops with makespan subject to mean completion time. *Applied Mathematics and Computation*, *219*(1), 351-359. https://doi.org/https://doi.org/10.1016/j.amc.2012.06.024.

———. (2013). New heuristics for no-wait flowshops with performance measures of makespan and mean completion time. *International Journal of Operations Research, 10*, 29–37.

Bagga, P C. (2003). Minimizing total elapsed time subject to zero total idle time of machines in n X 3 flowshop problem. *Indian Journal of Pure Applied Mathematics, 34*(2), 219–28.

Baptiste, P., & Hguny, L.K. (1997). A branch and bound algorithm for the F/no- idle/Cmax. In *Proceedings of the International Conference on Industrial Engineering and Production Management, IEPM*, *97*, 429–38.

Baraz, D., & Mosheiov, G. (2008). A note on a greedy heuristic for flow-shop makespan minimization with no machine idle-time. *European Journal of Operational Research*, *184*(2), 810-813.

Chen, J. F., Wang, L., & Peng, Z. P. (2019). A collaborative optimization algorithm for energy-efficient multi-objective distributed no-idle flow-shop scheduling. *Swarm and Evolutionary Computation*, *50*, 100557. https://doi.org/https://doi.org/10.1016/j.swevo.2019.100557.

Cheng, C. Y., Lin, S. W., Pourhejazy, P., Ying, K. C., & Lin, Y. Z. (2021). No-Idle Flowshop Scheduling for Energy-Efficient Production: An Improved Optimization Framework. *Mathematics*, *9*(12), 1335. https://doi.org/10.3390/math9121335.

Cho, H. M., & Jeong, I. J. (2017). A two-level method of production planning and scheduling for bi-objective reentrant hybrid flow shops. *Computers & industrial engineering*, *106*, 174-181. https://doi.org/https://doi.org/10.1016/j.cie.2017.02.010.

Ciavotta, M., Minella, G., & Ruiz, R. (2013). Multi-objective sequence dependent setup times permutation flowshop: A new algorithm and a comprehensive study. *European Journal of Operational Research*, *227*(2), 301-313. https://doi.org/https://doi.org/10.1016/j.ejor.2012.12.031.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, *6*(2), 182-197. https://doi.org/10.1109/4235.996017.

Deng, G., & Gu, X. (2012). A hybrid discrete differential evolution algorithm for the no-idle permutation flow shop scheduling problem with makespan criterion. *Computers & Operations Research*, *39*(9), 2152-2160. https://doi.org/http://dx.doi.org/10.1016/jcor201110024.

Deng, J., & Wang, L. (2017). A competitive memetic algorithm for multi-objective distributed permutation flow shop scheduling problem. *Swarm and evolutionary computation*, *32*, 121-131. https://doi.org/https://doi.org/10.1016/j.swevo.2016.06.002.

Ding, J. Y., Song, S., Gupta, J. N., Zhang, R., Chiong, R., & Wu, C. (2015). An improved iterated greedy algorithm with a Tabu-based reconstruction strategy for the no-wait flowshop scheduling problem. *Applied Soft Computing*, *30*, 604-613. https://doi.org/10.1016/j.asoc.2015.02.006.

Ding, J. Y., Song, S., & Wu, C. (2016). Carbon-efficient scheduling of flow shops by multi-objective optimization. *European Journal of Operational Research*, *248*(3), 758-771. https://doi.org/https://doi.org/10.1016/j.ejor.2015.05.019.

Dong, X., Huang, H., & Chen, P. (2008). An improved NEH-based heuristic for the permutation flowshop problem. *Computers & Operations Research*, *35*(12), 3962-3968. https://doi.org/http://dx.doi.org/10.1016/j.cor200705005.

Dubois-Lacoste, J., López-Ibánez, M., & Stützle, T. (2011). A hybrid TP+ PLS algorithm for bi-objective flow-shop scheduling problems. *Computers & Operations Research*, *38*(8), 1219-1236. https://doi.org/10.1016/j.cor.2010.10.008.

Fernandez-Viagas, V., & Framinan, J. M. (2014). On insertion tie-breaking rules in heuristics for the permutation flowshop scheduling problem. *Computers & Operations Research*, *45*, 60-67. https://doi.org/http://dx.doi.org/10.1016/jcor201312012.

———. (2015). A new set of high-performing heuristics to minimise flowtime in permutation flowshops. *Computers and Operations Research, 53*, 68–80. https://doi.org/http://dx.doi.org/10.1016/jcor201408004.

Fernandez-Viagas, V., Leisten, R., & Framinan, J. M. (2016). A computational evaluation of constructive and improvement heuristics for the blocking flow shop to minimise total flowtime. *Expert Systems with Applications*, *61*, 290-301. https://doi.org/https://doi.org/10.1016/j.eswa.2016.05.040.

Fernandez-Viagas, V., Valente, J. M., & Framinan, J. M. (2018). Iterated-greedy-based algorithms with beam search initialization for the permutation flowshop to minimise total tardiness. *Expert Systems with Applications*, *94*, 58-69. https://doi.org/https://doi.org/10.1016/j.eswa.2017.10.050.

Framinan, J. M., & Leisten, R. (2003). An efficient constructive heuristic for flowtime minimisation in permutation flow shops. *Omega*, *31*(4), 311-317. https://doi.org/http://dx.doi.org/10.1016/S0305048303000471.

Framinan, J. M., & Leisten, R. (2006). A heuristic for scheduling a permutation flowshop with makespan objective subject to maximum tardiness. *International Journal of Production Economics*, *99*(1-2), 28-40. https://doi.org/https://doi.org/10.1016/j.ijpe.2004.12.004.

———. (2008). A multi-objective iterated greedy search for flowshop scheduling with makespan and flowtime criteria. *OR Spectrum, 30*(4), 787–804. https://doi.org/10.1007/s00291-007-0098-z.

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

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Kan, A. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. In *Annals of discrete mathematics* (Vol. 5, pp. 287-326). Elsevier. https://doi.org/http://dx.doi.org/10.1016/S016750600870356X.

Ho, J. C., & Chang, Y. L. (1991). A new heuristic for the n-job, M-machine flow-shop problem. *European Journal of Operational Research*, *52*(2), 194-202.

Jarosław, P., Czesław, S., & Dominik, Ż. (2013). Optimizing bicriteria flow shop scheduling problem by simulated annealing algorithm. *Procedia Computer Science*, *18*, 936-945. https://doi.org/https://doi.org/10.1016/j.procs.2013.05.259.

Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, *1*(1), 61-68.

Kalczynski, P. J., & Kamburowski, J. (2005). A heuristic for minimizing the makespan in no-idle permutation flow shops. *Computers & Industrial Engineering*, *49*(1), 146-154. https://doi.org/10.1016/j.cie.2005.05.002.

———. (2007). On no-wait and no-idle flow shops with makespan criterion. *European Journal of Operational Research, 178*(3), 677–85.

Kalczynski, P. J., & Kamburowski, J. (2008). An improved NEH heuristic to minimize makespan in permutation flow shops. *Computers & Operations Research*, *35*(9), 3001-3008. https://doi.org/http://dx.doi.org/10.1016/jcor200701020.

Kamburowski, J. (2004). More on three-machine no-idle flow shops. *Computers & Industrial Engineering*, *46*(3), 461-466.

Keskin, K., & Engin, O. (2021). A hybrid genetic local and global search algorithm for solving no-wait flow shop problem with bi criteria. *SN Applied Sciences*, *3*(6), 628. https://doi.org/10.1007/s42452-021-04615-3.

Laha, D., & Sarin, S. C. (2009). A heuristic to minimize total flow time in permutation flow shop. *Omega*, *37*(3), 734-739. https://doi.org/http://dx.doi.org/10.1016/jomega200805002.

Li, X., Wang, Q., & Wu, C. (2009). Efficient composite heuristics for total flowtime minimization in permutation flow shops. *Omega*, *37*(1), 155-164. https://doi.org/https://doi.org/10.1016/j.omega.2006.11.003.

Liu, J., & Reeves, C. R. (2001). Constructive and composite heuristic solutions to the P//∑ Ci scheduling problem. *European Journal of Operational Research*, *132*(2), 439-452. https://doi.org/http://dx.doi.org/10.1016/S0377221700001375.

Liu, W., Jin, Y., & Price, M. (2016). A new Nawaz–Enscore–Ham-based heuristic for permutation flow-shop problems with bicriteria of makespan and machine idle time. *Engineering Optimization*, *48*(10), 1808-1822. https://doi.org/10.1080/0305215X.2016.1141202.

Lu, C., Gao, L., Li, X., Pan, Q., & Wang, Q. (2017). Energy-efficient permutation flow shop scheduling problem using a hybrid multi-objective backtracking search algorithm. *Journal of cleaner production*, *144*, 228-238. https://doi.org/https://doi.org/10.1016/j.jclepro.2017.01.011.

Maccarthy, B. L., & Liu, J. (1993). Addressing the gap in scheduling research: a review of optimization and heuristic methods in production scheduling. *The International Journal of Production Research*, *31*(1), 59-79. https://doi.org/10.1080/00207549308956713.

Marichelvam, M. K., Azhagurajan, A., & Geetha, M. (2017). A hybrid fruit fly optimisation algorithm to solve the flow shop scheduling problems with multi-objectives. *International Journal of Advanced Intelligence Paradigms*, *9*(2-3), 164-185. https://doi.org/10.1504/IJAIP.2017.082971.

Marichelvam, M. K., & Geetha, M. (2018). A hybrid crow search algorithm to minimise the weighted sum of makespan and total flow time in a flow shop environment. *International Journal of Computer Aided Engineering and Technology*, *10*(6), 636-649. https://doi.org/10.1504/IJCAET.2018.095200.

Minella, G., Ruiz, R., & Ciavotta, M. (2011). Restarted Iterated Pareto Greedy algorithm for multi-objective flowshop scheduling problems. *Computers & Operations Research*, *38*(11), 1521-1533. https://doi.org/https://doi.org/10.1016/j.cor.2011.01.010.

Nagano, M. S., & Moccellin, J. V. (2002). A high quality solution constructive heuristic for flow shop sequencing. *Journal of the Operational Research Society*, *53*, 1374-1379. https://doi.org/10.1057/palgrave.jors.2601466.

Nagano, M. S., de Almeida, F. S., & Miyata, H. H. (2021). An iterated greedy algorithm for the no-wait flowshop scheduling problem to minimize makespan subject to total completion time. *Engineering Optimization*, *53*(8), 1431-1449. https://doi.org/10.1080/0305215X.2020.1797000.

Nagano, M. S., Rossi, F. L., & Martarelli, N. J. (2019). High-performing heuristics to minimize flowtime in no-idle permutation flowshop. *Engineering Optimization*, *51*(2), 185-198. https://doi.org/10.1080/0305215X.2018.1444163.

Nagano, M. S., Rossi, F. L., & Tomazella, C. P. (2017). A new efficient heuristic method for minimizing the total tardiness in a no-idle permutation flow shop. *Production Engineering*, *11*, 523-529. https://doi.org/10.1007/s11740-017-0747-2.

Nawaz, M., Enscore Jr, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, *11*(1), 91-95. https://doi.org/http://dx.doi.org/10.1016/0305048383900889.

Pan, Q. K., & Ruiz, R. (2013). A comprehensive review and evaluation of permutation flowshop heuristics to minimize flowtime. *Computers & Operations Research*, *40*(1), 117-128. https://doi.org/http://dx.doi.org/10.1016/jcor201205018.

———. (2014). An effective iterated greedy algorithm for the mixed no-idle permutation flowshop scheduling problem. *Omega, 44*, 41–50. https://doi.org/http://dx.doi.org/10.1016/jomega201310002.

Pan, Q. K., & Wang, L. (2008a). A novel differential evolution algorithm for no-idle permutation flow-shop scheduling problems. *European Journal of Industrial Engineering*, *2*(3), 279-297. https://doi.org/10.1504/EJIE2008017687.

———. (2008b). No-idle permutation flow shop scheduling based on a hybrid discrete particle swarm optimization algorithm. *The International Journal of Advanced Manufacturing Technology, 39*(7), 796–807. https://doi.org/10.1007/s0017000712520.

Pan, Q. K., Wang, L., & Qian, B. (2009). A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems. *Computers & Operations Research*, *36*(8), 2498-2511. https://doi.org/https://doi.org/10.1016/j.cor.2008.10.008.

Pasupathy, T., Rajendran, C., & Suresh, R. K. (2006). A multi-objective genetic algorithm for scheduling in flow shops to minimize the makespan and total flow time of jobs. *The International Journal of Advanced Manufacturing Technology*, *27*, 804-815. https://doi.org/10.1007/s00170-004-2249-6.

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

Rad, S. F., Ruiz, R., & Boroojerdian, N. (2009). New high performing heuristics for minimizing makespan in permutation flowshops. *Omega*, *37*(2), 331-345. https://doi.org/http://dx.doi.org/10.1016/j.omega200702002.

Rahimi-Vahed, A. R., Javadi, B., Rabbani, M., & Tavakkoli-Moghaddam, R. (2008). A multi-objective scatter search for a bi-criteria no-wait flow shop scheduling problem. *Engineering Optimization*, *40*(4), 331-346. https://doi.org/10.1080/03052150701732509.

Rajendran, C. (1995). Heuristics for scheduling in flowshop with multiple objectives. *European journal of operational research*, *82*(3), 540-555. https://doi.org/https://doi.org/10.1016/0377-2217(93)E0212-G.

Rajendran, C., & Ziegler, H. (1997). An efficient heuristic for scheduling in a flowshop to minimize total weighted flowtime of jobs. *European Journal of Operational Research*, *103*(1), 129-138. https://doi.org/https://doi.org/10.1016/S0377-2217(96)00273-1.

Ravindran, D., Selvakumar, S. J., Sivaraman, R., & Haq, A. N. (2005). Flow shop scheduling with multiple objective of minimizing makespan and total flow time. *The international journal of advanced manufacturing technology*, *25*, 1007-1012. https://doi.org/10.1007/s00170-003-1926-1.

Ribas, I., Companys, R., & Tort-Martorell, X. (2010). Comparing three-step heuristics for the permutation flow shop problem. *Computers & Operations Research*, *37*(12), 2062-2070. https://doi.org/http://dx.doi.org/10.1016/jcor201002006.

———. (2017). Efficient heuristics for the parallel blocking flow shop scheduling problem. *Expert Systems with Applications, 74*(Supplement C): 41–54. https://doi.org/https://doi.org/10.1016/j.eswa.2017.01.006.

Rifai, A. P., Nguyen, H. T., & Dawal, S. Z. M. (2016). Multi-objective adaptive large neighborhood search for distributed reentrant permutation flow shop scheduling. *Applied Soft Computing*, *40*, 42-57. https://doi.org/https://doi.org/10.1016/j.asoc.2015.11.034.

Rossi, F. L., & Nagano, M. S. (2020). Heuristics and metaheuristics for the mixed no-idle flowshop with sequence-dependent setup times and total tardiness minimisation. *Swarm and Evolutionary Computation*, *55*, 100689. https://doi.org/https://doi.org/10.1016/j.swevo.2020.100689.

———. (2021a). Heuristics and iterated greedy algorithms for the distributed mixed no-idle flowshop with sequence-dependent setup times. *Computers & Industrial Engineering, 157*, 107337. https://doi.org/https://doi.org/10.1016/j.cie.2021.107337.

———. (2021b). Heuristics for the mixed no-idle flowshop with sequence-dependent setup times. *Journal of the Operational Research Society, 72*(2), 417–43. https://doi.org/10.1080/01605682.2019.1671149.

Rossi, F. L., Nagano, M. S., & Neto, R. F. T. (2016). Evaluation of high performance constructive heuristics for the flow shop with makespan minimization. *The International Journal of Advanced Manufacturing Technology*, *87*, 125-136. https://doi.org/10.1007/s00170-016-8484-9.

Rossi, F. L., Nagano, M. S., & Sagawa, J. K. (2017). An effective constructive heuristic for permutation flow shop scheduling problem with total flow time criterion. *The International Journal of Advanced Manufacturing Technology*, *90*, 93-107. https://doi.org/10.1007/s00170-016-9347-0.

Ruiz, R., & Allahverdi, A. (2009). New heuristics for no-wait flow shops with a linear combination of makespan and maximum lateness. *International Journal of Production Research*, *47*(20), 5717-5738. https://doi.org/10.1080/00207540802070942.

Ruiz, R., Maroto, C., & Alcaraz, J. (2006). Two new robust genetic algorithms for the flowshop scheduling problem. *Omega*, *34*(5), 461-476.

Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European journal of operational research*, *177*(3), 2033-2049. https://doi.org/http://dx.doi.org/10.1016/jejor200512009.

———. (2008). An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives. *European Journal of Operational Research, 187*(3), 1143–59.

Ruiz, R., Vallada, E., & Fernández-Martínez, C. (2009). Scheduling in flowshops with no-idle machines. *Computational intelligence in flow shop and job shop scheduling*, 21-51. https://doi.org/10.1007/97836420283662.

Saadani, N. E. H., Guinet, A., & Moalla, M. (2003). Three stage no-idle flow-shops. *Computers & industrial engineering*, *44*(3), 425-434.

———. (2005). A travelling salesman approach to solve the F/no-idle/Cmax problem. *European Journal of Operational Research, 161*(1), 11–20. https://doi.org/http://dx.doi.org/10.1016/jejor200308030.

Pour, N., Tavakkoli-Moghaddam, R., & Asadi, H. (2013). 5. Optimizing a multi-objectives flow shop scheduling problem by a novel genetic algorithm. *International Journal of Industrial Engineering Computations*, *4*(3), 345-354. https://doi.org/10.5267/j.ijiec.2013.03.008.

Shao, W., Pi, D., & Shao, Z. (2017). Memetic algorithm with node and edge histogram for no-idle flow shop scheduling problem to minimize the makespan criterion. *Applied Soft Computing*, *54*, 164-182. https://doi.org/https://doi.org/10.1016/j.asoc.2017.01.017.

Shao, Z., Pi, D., & Shao, W. (2019). A novel multi-objective discrete water wave optimization for solving multi-objective blocking flow-shop scheduling problem. *Knowledge-Based Systems*, *165*, 110-131. https://doi.org/https://doi.org/10.1016/j.knosys.2018.11.021.

Shen, J. N., Wang, L., & Wang, S. Y. (2015). A bi-population EDA for solving the no-idle permutation flow-shop scheduling problem with the total tardiness criterion. *Knowledge-Based Systems*, *74*, 167-175.

Singh, H., Oberoi, J. S., & Singh, D. (2021). Multi-objective permutation and non-permutation flow shop scheduling problems with no-wait: a systematic literature review. *RAIRO-Operations Research*, *55*(1), 27-50. https://doi.org/10.1051/ro/2020055.

Sun, Y., Zhang, C., Gao, L., & Wang, X. (2011). Multi-objective optimization algorithms for flow shop scheduling problem: a review and prospects. *The International Journal of Advanced Manufacturing Technology*, *55*, 723-739.

T'kindt, V., & Billaut, J. C. (2006). *Multicriteria scheduling: theory, models and algorithms*. Springer Science & Business Media. https://doi.org/10.1007/b106275.

Tasgetiren, M. F., Liang, Y. C., Sevkli, M., & Gencyilmaz, G. (2007). A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. *European journal of operational research*, *177*(3), 1930-1947.

Tasgetiren, M. F., Pan, Q. K., Suganthan, P. N., & Buyukdagli, O. (2013). A variable iterated greedy algorithm with differential evolution for the no-idle permutation flowshop scheduling problem. *Computers & Operations Research*, *40*(7), 1729-1743. https://doi.org/http://dx.doi.org/10.1016/jcor201301005.

Tasgetiren, M. F., Kizilay, D., Pan, Q. K., & Suganthan, P. N. (2017). Iterated greedy algorithms for the blocking flowshop scheduling problem with makespan criterion. *Computers & Operations Research*, *77*, 111-126. https://doi.org/10.1016/j.cor.2016.07.002.

Tasgetiren, M. F., Öztop, H., Gao, L., Pan, Q. K., & Li, X. (2019). A variable iterated local search algorithm for energy-efficient no-idle flowshop scheduling problem. *Procedia Manufacturing*, *39*, 1185-1193. https://doi.org/https://doi.org/10.1016/j.promfg.2020.01.351.

Vachajitpan, P. (1982). Job sequencing with continuous machine operation. *Computers & Industrial Engineering*, *6*(3), 255-259.

Varadharajan, T. K., & Rajendran, C. (2005). A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs. *European Journal of Operational Research*, *167*(3), 772-795. https://doi.org/https://doi.org/10.1016/j.ejor.2004.07.020.

Wu, X., & Che, A. (2020). Energy-efficient no-wait permutation flow shop scheduling by adaptive multi-objective variable neighborhood search. *Omega*, *94*, 102117. https://doi.org/https://doi.org/10.1016/j.omega.2019.102117.

Yagmahan, B., & Yenisey, M. M. (2008). Ant colony optimization for multi-objective flow shop scheduling problem. *Computers & Industrial Engineering*, *54*(3), 411-420. https://doi.org/https://doi.org/10.1016/j.cie.2007.08.003.

Ye, H., Li, W., & Nault, B. R. (2020). Trade-off balancing between maximum and total completion times for no-wait flow shop production. *International Journal of Production Research*, *58*(11), 3235-3251. https://doi.org/10.1080/00207543.2019.1630777.

Yenisey, M. M., & Yagmahan, B. (2014). Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. *Omega*, *45*, 119-135. https://doi.org/https://doi.org/10.1016/j.omega.2013.07.004.

Zhao, F., Zhang, L., Zhang, Y., Ma, W., Zhang, C., & Song, H. (2020). A hybrid discrete water wave optimization algorithm for the no-idle flowshop scheduling problem with total tardiness criterion. *Expert Systems with Applications*, *146*, 113166. https://doi.org/https://doi.org/10.1016/j.eswa.2019.113166.

Zhou, Y., Chen, H., & Zhou, G. (2014). Invasive weed optimization algorithm for optimization no-idle flow shop scheduling problem. *Neurocomputing*, *137*, 285-292. https://doi.org/http://dx.doi.org/10.1016/jneucom201305063v