Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

GRASP to minimize total weighted tardiness in a permutation flow shop environment

Lina Paola Molina-Sánchez[a] and Eliana María González-Neira[b*]

[a]*Researcher, Department of Industrial Engineering, School of Engineering, Pontificia Universidad Javeriana, Bogota, Colombia, Cra. 7 No. 40-62 - Edificio José Gabriel Maldonado, Colombia*
[b]*Instructor Professor, Department of Industrial Engineering, School of Engineering, Pontificia Universidad Javeriana, Bogota, Colombia, Cra. 7 No. 40-62 - Edificio José Gabriel Maldonado, Colombia*

C H R O N I C L E

Article history:
Received June 5 2015
Received in Revised Format
June 29 2015
Accepted June 29 2015
Available online
June 30 2015

Keywords:
Permutation Flow Shop (PFS)
Total Weighted Tardiness (TWT)
GRASP
Weighted Modified Due Date (WMDD)
Apparent Tardiness Cost (ATC)
Weighted Earliest Due Date (WEDD)

A B S T R A C T

This paper addresses the scheduling problem in a Permutation Flow Shop (PFS) environment, which is associated with many types of industries such as chemical, petrochemical, automobile manufacturing, metallurgical, textile, etc. Thus, this work intends to solve a PFS scheduling problem in order to minimize the total weighted tardiness, since it is an important sequencing criterion not only for on time delivery jobs but also for customer satisfaction. To solve the problem, GRASP (Greedy Randomized Adaptive Search Procedure) metaheuristic is proposed as a solution, which has shown competitive results compared with other combinatorial problems. In addition, two utility functions called Weighted Modified Due Date (WMDD) and Apparent Tardiness Cost (ATC) are proposed to develop GRASP. These are based on dynamic dispatching rules and also known for solving the problem of total weighted tardiness for single machine scheduling problem. Next, an experimental design was carried out for comparing the GRASP performance with both utility functions and against the WEDD dispatching rule results. The results indicate that GRASP-WMDD could improve the total weighted tardiness in 47.8% compared with WEDD results. Finally, the GRASP-WMDD performance for the PFS total tardiness problem was evaluated, obtaining a relative deviation index of 13.89% and ranking the method over 26 heuristics and metaheuristics.

1. Introduction

Production scheduling seeks optimal sequencing of jobs or tasks for maximum use of limited resources to fulfill the organizations' objectives and policies (Proth, 2007; Pinedo, 2012). This work studies the production schedule in one of the most common environments in industry, Flow Shop (FS) environment. This environment can be found in chemical, metal processing, food, and assembly industries (Li et al., 2015). In the FS, a set of n jobs (J_1,..., J_n) is processed on m machines (M_1,..., M_m). All jobs must go through the machines in the same order, which means all jobs have to be processed first on M_1, M_2

* Corresponding author. Tel: +57-1-3208320 Ext. 5300 Fax: +57-1-3208320 Ext. 5275
E-mail: eliana.gonzalez@javeriana.edu.co (E. M. González-Neira)

doi: 10.5267/j.ijiec.2015.6.004

machines and so on until machine M_m. This results in a set of $(N!)^m$ different candidate solutions. A common constraint in FS environment is that the processing sequence of jobs is the same for all machines. In this case, a candidate solution can be represented as a permutation of the jobs and, therefore, there are n! possible outcomes. This sequence is called Permutation Flow Shop (PFS), which implies that a job is selected to process on each machine according to the rule first in, first out (FIFO). This is often appropriate for real-world applications, given that in-process storage of products is very limited in most situations (Ciavotta et al., 2013). Examples of this situations are factories with conveyors between machines for materials transfer and assembly lines that perform the final assembly of bulky products (Kim, 1995).

In FS literature there are many studies for the PFS and FS problems that minimize the makespan (Ruiz & Stützle, 2008) or the total flowtime (Pan & Ruiz, 2013). Although, there is another important objective that is receiving more attention in literature nowadays due to its applicability to industries, called the on time delivery of jobs (Swaminathan et al., 2007; Vallada et al., 2008). Likewise, late deliveries cause penalties that the company has to assume. Although for many companies it is important to reduce these extra costs. One way to make it happen is not only by delivering all jobs on time but also customer satisfaction (Xiao et al., 2012). This is why in this research the objective is to minimize the total time of late deliveries depending on the importance of customers (Jobs weight). It means to minimize the Total Weighted Tardiness (TWT) in a PFS, which notation corresponds to FS|prmu|∑wjTj.

Due to the NP-hardness, in the strong sense of the PFS problem that minimizes tardiness (Du & Leung, 1990), heuristic and metaheuristic methods have been widely used to solve them, considering that exact methods are impractical for medium and large instances (Sayadi et al., 2010; Gupta & Chauhan, 2015). Genetic Algorithm (GA) is the most commonly used metaheuristic to solve the PFS problem for tardiness minimization (Li et al., 2015). Nevertheless, few researches have used Greedy Randomized Adaptive Search procedures (GRASP) to solve this kind of problem although it has shown good results in some scheduling problems. For instance, Caballero-Villalobos and Alvarado-Valencia (2010) and Vega-Mejía and Caballero-Villalobos (2010) used GRASP to minimize TWT in a single machine environment. Similarly Armentano and Araujo (2006) minimized total tardiness considering setup times and Armentano and de França Filho (2007) also used for a parallel machine problem. Moreover, Arroyo and de Souza Pereira (2010) and Shahul Hamid Khan et al. (2007) solved multi-objective PFS problems and Rajkumar et al. (2011) solved a flexible job shop problem. Therefore, this research proposes the use of GRASP metaheuristic to solve the FS|prmu|$\sum w_j T_j$. The proposed GRASP was implemented with two different utility functions that are adaptations of two well-known dispatching rules: Weighted Modified Due Date (WMDD) and Apparent Tardiness Cost (ATC). Later on, the method was tested on 540 instances proposed by Vallada et al. (2008). For each instance the jobs weights (w_j) were generated, due to Vallada et al. (2008) research evaluated only Total Tardiness objective. Moreover, the performance of GRASP was determined through the comparison between the results given by both utility functions in relation with the solutions obtained by sequencing the jobs under Weighted Earliest Due Date (WEDD) dispatching rule. Finally, the GRASP with the utility function that gave the best results for the FS|prmu|$\sum w_j T_j$ was selected to solve the initial problem proposed by Vallada et al. (2008) for assigning a weight $w_j=1$ for all jobs. This allowed ranking the proposed GRASP versus the 40 methods tested by the authors for the FS|prmu|$\sum T_j$ problem.

The remainder of this paper is as follows, in Section 2 the literature review is presented, and in Section 3 a Mixed Integer Programing (MIP) model formulation for the problem is given. Section 4 describes the proposed GRASP procedure. Section 5 shows the computational results and the corresponding analysis, and finally the conclusions and future research are suggested in Section 6.

2. Literature Review

The FS environments goes back to the fifties with the publication of Johnson's rule that solved the problem of two and three machines (Johnson, 1954). Since then, numerous approaches have been proposed for the FS in different areas. Among different methods, we can highlight the algorithms proposed by Palmer (1965), Campbell et al. (1970) and Nawaz et al. (1983).

Table 1
State of the art in FS and PFS (by the authors)

Reference	Environment	Constraints / Characteristics	Algorithm	Objective function to minimize
Prabhaharan et al. (2005)	FS	NA	GRASP	Makespan
Ruiz et al. (2005)	FS	Sequence dependent setup times (SDST)	GA	Makespan
Liao et al. (2006)	PFS	NA	Tabu search	Due-time, total tardiness and total weighted tardiness
Laha and Chakraborty (2007)	PFS	NA	Various Heuristic	Total flowtime
Ruiz and Stützle (2007)	FS	SDST	Greedy	Makespan, weighted tardiness
Shahul Hamid Khan et al. (2007)	FS	Multi-objective	GRASP	Makespan, maximum tardiness
Swaminathan et al. (2007)	PFS	Uncertain processing times	GA	Total weighted tardiness
Framinan and Leisten (2008)	PFS	Due date	Greedy	Total Tardiness
Chandra et al. (2009)	PFS	Due date	Heuristic	Earliness and Tardiness
Wu and Lee (2009)	PFS	Learning effects	Heuristics Branch and Bound	Total Flow Time
Arroyo and de Souza Pereira (2010)	PFS	Multi-objective	GRASP	Makespan, maximum tardiness, and total flowtime
Sayadi et al. (2010)	PFS	NA	Discrete firefly metaheuristic	Makespan
Anandaraman (2011)	FS	NA	Improved sheep flock heredity algorithm named	Makespan
Araújo and Nagano (2011)	FS	No-wait, SDST	Constructive heuristic	Makespan
Dubois-Lacoste et al. (2011)	FS	Multi-objective	Two- phase local search	Makespan, maximum tardiness, and completion times
Liu et al. (2011)	PFS	NA	GA	Completion time, makespan
Vallada and Ruiz (2011)	PFS	NA	GA and Path relinking	Total Tardiness
Babaei et al. (2012)	FS	SDST Lot sizing	GA Imperialist Competitive Algorithm	Setup, inventory, production and backlogging costs
Bank et al. (2012)	2-machine FS	Deteriorating jobs	Branch and bound	Total Tardiness
Bhongade and Khodke (2012)	FS	Skipping operations	Three heuristics	Makespan
El-Bouri (2012)	FS	Dynamic FS	Cooperative dispatching rules	Mean Tardiness
Khalili and Tavakkoli-Moghaddam (2012)	FS	Skipping jobs	Electromagnetism algorithm	Makespan and Total Weighted Tardiness
Naderi-Beni et al. (2012)	FS	No-wait, SDST, release times	Two-phase fuzzy programming	Bi-objective: Weighted Mean Tardiness and Makespan
Baker (2013)	PFS	NA	Mixed-integer programming approach	Total Tardiness
Ciavotta et al. (2013)	PFS	Multi-objective, SDST	Restarted Iterated Pareto Greedy	Multi-Objective SDST
Lee and Chung (2013)	PFS	Learning effects	Branch and Bound	Total Tardiness
Shahsavari Pour et al. (2013)	FS	NA	GA	Multi-objective: Makespan, Total Waiting Time and Total Tardiness
Schaller and Valente (2013)	PFS	Due date	GA	Total Earliness and Tardiness
Tasgetiren et al. (2013)	PFS	Idle time is not allowed on machines	Discrete artificial bee colony algorithm	Total Tardiness
M'Hallah (2014)	FS	NA	Variable neighborhood search with Mixed integer programming	Earliness and tardiness
Lee et al. (2014)	PFS	Deterioration consideration	Branch & Bound, Particle swarm optimization, Simulated annealing algorithm	Total Tardiness
Gupta and Chauhan (2015)	FS	NA	Heuristic	Makespan

Ruiz and Maroto (2005), Vallada et al. (2008) and Pan and Ruiz (2013) presented extensive reviews for FS and PFS problems with single objective functions, while Sun et al. (2010) and Yenisey and Yagmahan

(2014) applied it for multi-objective FS problems. Although, Ruiz and Maroto (2005) evaluated 25 heuristics and metaheuristics for makespan minimization. They concluded that NEH was the best heuristic and iterated local search and genetic algorithms are the metaheuristics that have the best performances. In addition, Vallada et al. (2008) considered the tardiness minimization. They implemented 40 heuristic and metaheuristic methods presented by different authors in the past and compared them using 540 instances of different sizes. According to the authors, the heuristic methods based on job insertion or job interchanges were those that presented the best results. At the end it was found that simulated annealing outperformed the other ones. Also Pan and Ruiz (2013) compared 22 existing heuristics to minimize the flowtime and proposed five new methods to solve the problem. On the other hand, the reviews of Sun et al. (2010) and Yenisey and Yagmahan (2014), in spite of presenting a multiobjective FS literature survey, mentioned few works dealt with weighted tardiness as one of the objectives assessed and also no review was found in TWT minimization. Table 1 shows the state of art for the reviewed literature that develops the FS and PFS environments during the last decade, since 2005 to nowadays. In this table the problem, its characteristics, solution method and objective function are presented.

3. Mixed Integer Programming mathematical model

Here, we present a Mixed Integer Programming (MIP) mathematical model for the FS| prmu |$\Sigma w_j T_j$. In first instance, it is necessary to present the following notations to understand the model:

Sets:

$\mathbf{M}: Machines\ \{1, \dots, m\}$

$\mathbf{J}: Jobs\ \{1, \dots, n\}$

$\mathbf{L}: Positions\ in\ the\ sequence\ \{1, \dots, n\}$

Parameters:

d_j	due date of job $j, \forall j \in \mathbf{J}$
p_{ij}	processing time of job j in machine $i, \forall j \in \mathbf{J}, \forall i \in \mathbf{M}$
w_j	Weight of job $j, \forall j \in \mathbf{J}$
b	Big positive number

Decision variables

$$X_{jl} = \begin{cases} 1, \text{if job } j \text{ is procesed in the } l \text{ position of the sequence}, \forall j \in \mathbf{J}, \forall l \in \mathbf{L} \\ \qquad\qquad 0, \text{otherwise} \end{cases}$$

S_{ij}	Starting time of job j at machine $i, \forall j \in \mathbf{J}, \forall i \in \mathbf{M}$
C_{ij}	Completion time of job j at machine $i, \forall j \in \mathbf{J}, \forall i \in \mathbf{M}$
T_j	Tardiness of job $j, \forall j \in \mathbf{J}$

The objective function is:

$$min \sum_{j \in \mathbf{J}} w_j T_j \tag{1}$$

subject to:

$$\sum_{j \in \mathbf{J}} X_{jl} = 1, \qquad \forall l \in \mathbf{L} \tag{2}$$

$$\sum_{l \in \mathbf{L}} X_{jl} = 1, \qquad \forall j \in \mathbf{J} \tag{3}$$

$$C_{ij} \geq F_{ij} + p_{ij}, \qquad \forall j \in \mathbf{J}, \forall i \in \mathbf{M} \tag{4}$$

$$S_{(i+1)j} \geq C_{ij}, \qquad \forall j \in \mathbf{J}, \forall i, m \in \mathbf{M}, i < Card(\mathbf{M}) \tag{5}$$

$$S_{ij} + \left(1 - X_{j(l+1)}\right) * b \geq C_{ih} - (1 - X_{hl}) * b, \\ \forall h, j \in \mathbf{J}, h \neq j, \forall l \in \mathbf{L}, l < Card(\mathbf{L}), \forall i \in \mathbf{M} \tag{6}$$

$$T_j \geq C_{ij} - d_j, \qquad \forall j \in \mathbf{J}, \forall i \in \mathbf{M} \tag{7}$$

$$T_j \geq 0, \qquad \forall j \in \mathbf{J} \tag{8}$$

$$S_{ij} \geq 0, \qquad \forall j \in \mathbf{J}, \forall i \in \mathbf{M} \tag{9}$$

$$X_{jl} \in \{0,1\}, \qquad \forall j \in \mathbf{J}, \forall l \in \mathbf{L} \tag{10}$$

The objective function (1) is the minimization of the total weighted tardiness. Constraint sets (2) and (3) ensure that every job is sequenced in only one position and every position of the sequence is scheduled. With constraint (4) the completion time of every job in every machine is calculated as the sum of the starting and the processing time. Continually set (5) indicates that the starting time of a job in a specific machine is greater and equal than its completion time on immediately preceding machine. Constraint set (6) controls the starting times of the jobs at the machines. Basically, if a job j is assigned to machine i after job h, its starting time F_{ij} must be greater or equal than the completion time of job h, C_{ih}. Sets (7) and (8) define the tardiness of every job. Set (9) defines the non-negative nature for jobs starting times. Finally, set (10) defines the binary variables.

4. Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP metaheuristic is an iterative process for combinatorial problems that consists of two phases: construction and local search. At first, feasible solution is selected from a set of the best candidates (Restricted Candidate List - RCL). It is constructed in accordance with an established utility function to determine how good the chosen solution is. Then an item is randomly selected from the solution that was built from the RCL. In the second phase, this local search method performs small changes in the solution to find a better value of the objective function (Resende & Ribeiro, 2003). The elements from the RCL given in Eq. (11) can be chosen by including the parameter α [0, 1]. The more the value tends to 0, the greedier is the building list of candidates, which means that the target is not compromised, but still the best solutions can be discarded. If the parameter value approaches 1, the construction is more random and so the value of the utility function will be compromised, but we may find more possible solutions (Resende & Ribeiro, 2003).

$$RCL = \{x \mid L \leq fc(x) \leq L + \alpha(U - L)\}. \tag{11}$$

Here $fc(x)$ is the utility function from the x element; α is a number between 0 and 1; L is the minimum value from the utility function found; U is the maximum utility function value. Algorithm 1 presents the GRASP pseudo-code.

Algorithm 1
GRASP Pseudocode

```
Input Data:
M: Machines {i=1,2,3,…,m}
J: Jobs {j=1,2,3,…,n}
L: Sequences positions.
α: Value [0,1] for the selection of candidates to be sequenced.
$w_j$: Weight of job j.
$d_j$: Due date of job j.
$P_{ij}$: Processing time of job j on machine i.
$S_{ij}$: Start time of job j on machine i.
$C_{ij}$: Completion time of job j on machine i.
eje: Number of executions for instance.
seconds: Algorithm execution time.

Variables:

f(): Objective function.
SQ: Solution.
BS: Best solution.
cont: Counter

START PROCEDURE
        FOR k=1 until eje, DO
                Read de data from the problem (M,J, α, $d_j$, $w_j$)
                cont=1
                WHILE seconds ≤ (n*m*90)/1000
                        3. SQ  Construction phase.
                        4. SQ   Local search phase.
                        5. IF f(SQ)< f(BS) THEN
                                BS=SQ
                           END IF
                        6. cont=cont+1
                END WHILE
                Return BS
        END FOR
END PROCEDURE
```

4.1 GRASP Construction Phase

In this phase, a feasible solution is iteratively constructed with an element at a time. The GRASP probabilistic component is characterized by randomly choosing one of the top best candidates (Restricted Candidate List - RCL for its acronym), but not necessarily the best one. GRASP uses the utility function for the RCL construction, this function is calculated from all not selected elements yet, and arises to choose a sequence that minimizes the objective function. This technique allows the choice of different solutions to be obtained at each GRASP iteration, but does not necessarily compromise the power of the adaptive greedy component method. For this paper two different utility functions are used based on dispatching rules. Apparent Tardiness Cost (ATC) was developed by Vepsalainen and Morton (1987) and Weighted Modified Due Date (WMDD) was given by Kanet and Li (2004) for the proposed total weighted tardiness problem, but for a single machine. ATC rule for job *j* at the time *t*, is given by Eq. (12) as follows,

$$ATC_j = \frac{w_j}{P_j} exp\left(-max\{\,0, d_j - P_j - t\}/K\bar{P}\right), \tag{12}$$

where $\bar{P}$ is the average of the total processing times of the total Jobs *j* that are under construction in that specific moment (waiting), *t* is the time where the previous job is expected to finish (the first job to be sequenced has a t = 0) and *K* is a scale parameter. The following method is used for selecting the *K* value Eq. (13):

$$K = \begin{cases} 4.5 + R, & R < 0.5 \\ 6 - 2R, & R \geq 0.5 \end{cases} \tag{13}$$

The *R* value is the Due Date Range, which in this case, the values are already inside the instances used from Vallada et al. (2008). The WMDD dispatching rule is given by Eq. (14):

$$WMDD_j = \frac{1}{w_j} max\{P_j, (d_j - t)\} \tag{14}$$

This rule can be interpreted as a combination of Weighted Shortest Processing Time (WSPT) and Weighted Remaining Allowance (WRA) and WSPT rule minimizes the weighted tardiness when all jobs are behind (Smith, 1956) and on the other hand, the WRA rule gives priority to work with larger weighted tardiness and few gaps. Both dispatching rules presented have P_j, d_j, w_j and t in common, where P_j are the processing times, d_j are the due dates or the deadlines for jobs submission, w_j is the penalty cost or the jobs weight and t is the expected time from the previous job. This time t is dynamic, because as the jobs are sequencing, the time is changing every time it recalculates the utility function for the next job to be sequenced. As in this case the environment type presented is composed of various machines, then the P_j calculation is $P_j = C_j - S_j$, remembering that C_j is the time when finishes the job j in the last machine and S_j the time that starts job j in the first machine. At Algorithm 2, the Pseudo-code construction phase is shown.

Algorithm 2
GRASP Construction Phase Pseudocode

```
Sets:
M: Machines {i=1,2,3,…,m}
J: Jobs {j=1,2,3,…,n}
$\theta$: Set of jobs not sequenced.
L: Positions in the sequence.

Parameters:
$d_j$: Due date for Job j.
$P_{ij}$: Processing time from job j in machine i.
α: Value [0,1] for the candidate’s selection for the sequence.
$w_j$: Job weight j.

Variables:
cont: Counter.
$S_{ij}$: Start time of job j on machine i.
$C_{ij}$: Completion time of job j on machine i.
time_mac(i): available time of machine i.
$WMDD_j$: Weighted Modified Due Date utility function for job j.
$WMDD_{min}$: Utility function WMDD minimum Value.
$WMDD_{max}$: Utility function WMDD maximum Value.
$ATC_j$: Aparent Tardiness Cost utility function.
$ATC_{min}$: Utility function ATC minimum Value.
$ATC_{max}$: Utility function ATC maximum Value.
SQ: Solution
TWT: Total weighted tardiness.
RCL: List of candidates for the solution.
t: System time in a given time.
k: Dependent constant for the Due date range from the ATC rule.
P: Average processing time from the waiting Jobs from the ATC rule.

START
Assign all jobs j to the set of jobs not sequenced θ. Initialize the final sequence SQ = ∅. Also do $C_{ij}$ = 0, $S_{ij}$ =
0, time_mac(i) = 0 ∀ i, cont = 0.
Initialize TWT=0
WHILE cont<n
3. Evaluate the utility function in each job $j \in \theta$.
4. Construct the RCL from the jobs $j \in \theta$ satisfying the following condition according to the utility function:

$ATC_j \geq ATC_{max} - \alpha (ATC_{max} - ATC_{min})$
$WMDD_j \leq WMDD_{min} + \alpha (WMDD_{max} - WMDD_{min})$

5. Select randomly job j from the RCL.
6. Include j selected at cont position from SQ and determine time_mac(i). Eliminate job j selected from $\theta$set.
7. Calculate TWT.
8. cont=cont+1
END WHILE
9. Return SQ solution
END
```

4.2 Local Search Phase

The local search used was proposed in 1958 by Croes, called 2-optimal and it was used to solve the traveling salesman problem. Besides, the main idea of the algorithm is to swap two jobs in the sequence (as shown in Fig. 1) and keep swapping if that change gets better results for the objective function. This exchange processes two jobs and compares them against the previous objective function value and this process is performed until all jobs are exchanged and it obtains the best objective function value. To explain the operation of this type of local search, we use the following example. For this example the sequence obtained from the construction phase is considered, in which the final sequence is SQ = {2,1}.

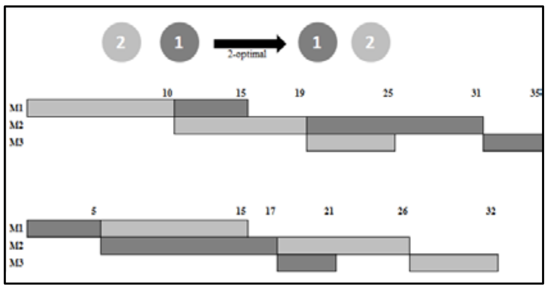

Fig. 1. 2-opt example

As shown in the example of Fig. 1, once the sequence (SQ) is obtained in the construction phase, the local search makes the exchange of each possible pair of jobs to find a sequence with better TWT. These exchanges are made until the maximum execution time is accomplished or until no improvement is found, that is a local optimum. However, only SQ is updated when the TWT is less in this exchange of two jobs. The jobs are exchanged in pairs until a better solution is found. The GRASP pseudo-code for this stage is shown in Algorithm 3.

Algorithm 3
GRASP Local Search Phase Pseudocode

```
Parameters:
M: Machines {i=1,2,3,…,$m_i$}
J: Jobs {j=1,2,3,…,$n_j$}
Max_it: Number of iterations.
$w_j$: Jobs weight j.
$d_j$: Due date for job j.
$P_{ij}$: Processing time for job j in machine i.

Variables:
f(): Objective function.
SQ: Solution.
SQ2: Solution that is constructed from the 2-optimal exchange.
Cont1: Counter.
Cont2: Counter.

START
cont1=0
WHILE cont1< j
  2. cont2=cont1+1
   WHILE cont2< j+1
   3. DO SQ2=SQ
   4. Modify SQ2 exchanging Jobs from position cont1 and cont2.
   5. Calculate the objective function f(SQ2).
   6. IF f(SQ2)< f(SQ) THEN
   SQ=SQ2
   cont1 = 0
   cont2 = cont1 + 1
   IF NOT
   cont2=cont2+1
   END IF
  END WHILE
  7. cont1=cont1+1
END WHILE
8. Return SQ y f(SQ)
END
```

5. Computational Results and Discussion

For this study the instances proposed by Vallada et al. (2008) were used. However since the objective function was to minimize total tardiness, so in order to minimize TWT; we need to generate the jobs weights (w_j) for those instances. This w_j generation is performed according to Osman et al. (2009), where the weights of the jobs are uniformly distributed in the in [1,10] interval. Furthermore, Vallada et al. (2008) generated nine different set of the instances where each contains 60 problems. Considering combinations of the parameters T (tardiness factor) and R (Due date range), the following combinations are proposed: $T = \{0.2, 0.4, 0.6\}$ and $R = \{0.2, 0.6, 1\}$, which results in nine combinations. In some cases, particularly for low T-values and big R-values, some Due Dates may be negative. When this happens, zero replaces the negative Due Dates. In terms of the number of jobs and machines, the following were selected: $N = \{50, 150, 250, 350\}$ and $m = \{10, 30, 50\}$. In total, there are 108 combinations of n, m, T and R, each of which has 5 different instances. Therefore, it has a total of 540 instances of which are available on the website: http://soa.iti.es/files/Eva_Instances.zip[1]. In addition, GRASP was implemented in C + + and run on a computer with Intel ® Core ™ i5 2400 CPU 3.10de GHz processor and 4 GB of RAM memory installed. In order to make an equivalent comparison in all instances, the stopping criterion of all metaheuristics was established from a maximum CPU elapsed time similar to the one presented by Vallada et al. (2008). The stopped criterion here is defined as m × n × 90 ms, twice the value the authors used in their article since the objective function in this case is the Total Weighted Tardiness and requires more computational effort.

5.1 Design of Experiments

An experimental design was carried out to assess the quality of the proposed solutions compared with two different utility functions. The features of the experiment are presented:

- Response Variable: GRASP percentage improvement in relation to the Weighted Earliest Due Date (WEDD) dispatching rule. WEDD (Weighted Earliest Due Date) indicates that jobs should be sequenced in ascending order by the ratio between the Due Date and the weights of the jobs. That is if $d_j/w_j < d_h/w_h$ then the job j must be sequenced before job h. The percentage improvement is calculated as Eq. (15):

$$\%\,\mathrm{Improvement_k} = \left(\frac{\mathrm{TWTWEDD_k} - \mathrm{TWTGRASP_k}}{\mathrm{TWTWEDD_k}}\right) \times 100\%\,, \tag{15}$$

  where $TWTWEDD_k$ is the total weighted tardiness of instance k jobs ordered under WEDD rule, and $TWTGRASP_k$ is the total weighted tardiness of the GRASP execution instance k.

- Factors: The analyzed factors and respective levels are shown in Table 2.

Table 2
Design of Experiment Factors and its levels

Factors	Levels
Number of Jobs	50 150 250 350
Number of Machines	10 30 50
Due Date Range (R)	0.2 0.6 1.0
Tardiness Factor (R)	0.2 0.4 0.6
Alphas	0.02 0.10
Utility Function	WMDD ATC

[1] Downloaded on March 2015

Multiple runs were executed with two utility functions and two alphas for each of the 540 instances, a total of 2160 executions were obtained. Table 3 presents the improvement percentage that was obtained with the GRASP in relation to the implementation of WEDD dispatching rule for each of the utility functions. Also it presents the percentage of solutions in which there was an improvement in the objective function.

Table 3
GRASP performance

Utility Function	Improvement percentage in TWT GRASP in relation to WEDD	Instances percentage that obtained improvement in the algorithm
ATC	10.96%	64.54%
WMDD	47.80%	98.24%
General Total	29.38%	81.39%

Clearly, there is an improvement in the results presented by the proposed utility functions with both algorithms, which is greater when the utility function based on the dispatching rule called WMDD is used. Formerly the assumption of normality of residuals was tested with Kolmogorov-Smirnov, the variances homogeneity through the Levene test, and independence through the run test. The tests were given with a confidence level of 95% and the results obtained were: the tests of normality of residuals and variances homogeneity were rejected with a p-value <0.01, while the test of independence was not rejected with p = 0.66. While there are two assumptions that are no accomplished, it is necessary to calculate the non-parametric Tamhane test to analyze the differences between means and corroborate the ANOVA results. However the R-Square achieved from the ANOVA was 84.21% and the adjusted R-Square is 80.27%, indicating that the variables used adequately to explain the GRASP improvement percentage in relation to WEDD dispatching rule.

Once the Tamhane tests is used, we obtained the same conclusions from the results previously obtained at the ANOVA, in terms of the factors and interactions that are meaningful. The conclusions obtained from the main effects plot (Fig. 2) and Tamhane non-parametric tests are presented in Table 4.

Table 4
Principal Factors Conclusions

Principal Factor	P-value	Conclusion
Utility Function	<0.001	The GRASP-WMDD, has an average improvement over the WEDD dispatching rule in 47.8%, while the GRASP-ATC has a 10.96% average improvement over the implementation WEDD rule. This corroborates that GRASP is a very good metaheuristic if the utility function chosen it is adequate, in this case is the adaptation of the WMDD rule.
Alpha	0.13	There is no significant difference between the GRASP average performance, using alpha of 0.02 and 0.1, so the GRASP has a statistically similar improvement rate over the WEDD rule with either alphas.
Number of Jobs	<0.001	In the 150 jobs instance is where the proposed GRASP has the highest improvement percentage over the WEDD rule.
Number of Machines	<0.001	In the 10 machines instance is where the proposed GRASP has the highest improvement percentage in relation to the WEDD dispatching rule.
Tardiness Factor	<0.001	In instances of 0.2 Tardiness Factor is where the proposed GRASP has the highest improvement percentage over the WEDD rule.
Due Date Range	<0.001	The instances with 0.2 and 0.6 Due Date Range have the highest improvement percentage in relation to the WEDD rule.

Fig. 3 shows all the double interactions effects plots. From these plots, from the Tamhame non-parametric tests and from the ANOVA, we can complete analysis of the best performance in terms of the interactions effects (Observe Table 5).

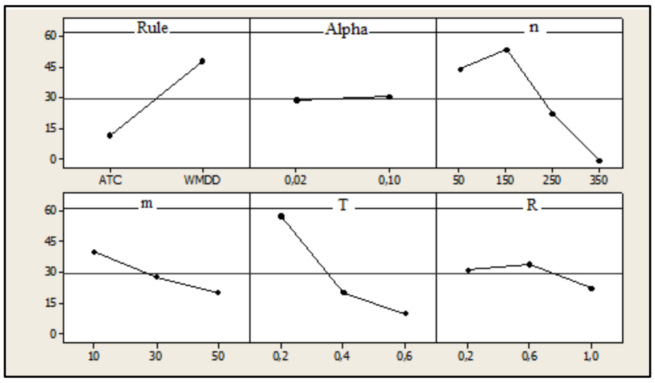

Fig. 2. Main Effects Plots

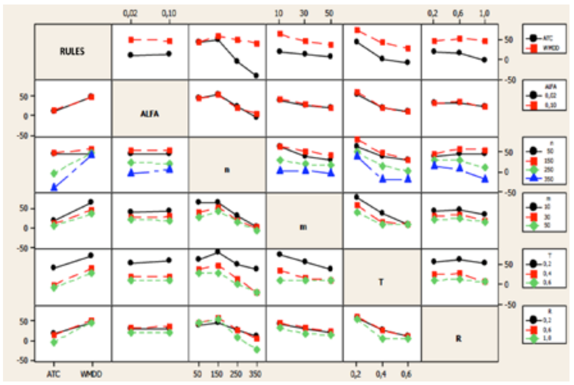

Fig. 3. Double Interactions Effects Plots

Table 5
Double Interactions Conclusions

Interaction	P-value	Conclusion
Number of jobs- Number of machines	<0.001	Both double interactions between 50 jobs with 10 machines as the interaction between 150 jobs with 10 machines are the instances when the proposed GRASP has the highest improvement percentage over the WEDD rule.
Number of jobs - Due Date Range	<0.001	Both double interactions between 150 jobs with 0.6 Due Date Range as the interaction between 150 jobs with 1.0 Due Date Range are the instances when the proposed GRASP has the highest improvement percentage over the WEDD rule.
Number of jobs - Tardiness Factor	<0.001	The instances with the double interactions between 150 jobs and 0.2 Tardiness Factor have the highest improvement percentage in relation to the WEDD rule.
Alpha- Number of jobs	<0.001	Both double interactions between 150 jobs with Alpha of 0.02 and Alpha of 0.1 are the instances when the proposed GRASP has the highest improvement percentage over the WEDD rule.
Utility Function- Number of jobs	<0.001	The instances with the double interactions between the WMDD utility function and 150 jobs are the ones that have the highest improvement percentage in relation to the WEDD rule.
Number of machines - Due Date Range	0.16	In this case three double interactions have the highest percentage of improvement over the WEDD rule. The first one is the double interaction between 10 machines and 0.6 Due Date Range, the next one is the double interaction between 10 machines and 0.2 Due Date Range, and finally between 30 machines and 0.6 Due Date Range.
Number of machines - Tardiness Factor	<0.001	The instances with the double interactions between 10 machines and 0.2 Tardiness Factor are the ones that have the highest improvement percentage in relation to the WEDD rule.
Alpha- Number of machines	<0.001	The instances with the double interactions between alpha of 0.1 and 10 machines are the ones that have the highest improvement percentage in relation to the WEDD rule.
Utility Function- Number of machines	<0.001	The instances with the double interactions between WMDD utility function and 10 machines are the ones that have the highest improvement percentage in relation to the WEDD rule.
Tardiness Factor- Due Date Range	<0.001	In this case three double interactions have the highest percentage of improvement over the WEDD rule. The first one is the double interaction between 0.2 Tardiness Factor and 0.2 Due Date Range, the next one is the double interaction between 0.2 Tardiness Factor and 0.6 Due Date Range, and finally between 0.2 Tardiness Factor and 1 Due Date Range.
Alpha- Due Date Range	0.20	The instances with the double interactions between alpha of 0.1 and 0.6 Due Date Range are the ones that have the highest improvement percentage in relation to the WEDD rule.
Utility Function- Due Date Range	<0.001	The instances with the double interactions between WMDD utility function and 0.6 Due Date Range are the ones that have the highest improvement percentage in relation to the WEDD rule.
Alpha- Tardiness Factor	<0.001	Both double interaction between 0.2 Tardiness Factor and with Alpha of 0.02 and Alpha of 0.1 are the instances when the proposed GRASP has the highest improvement percentage over the WEDD rule.
Utility Function- Tardiness Factor	<0.001	The instances with the double interactions between WMDD utility function and 0.2 Tardiness Factor are the ones that have the highest improvement percentage in relation to the WEDD rule.
Utility Function- Alpha	<0.001	The instances with the double interactions between WMDD utility function and Alpha of 0.02 are the ones that have the highest improvement percentage in relation to the WEDD rule.

5.2 Performance of proposed GRASP-WMDD for FS|prmu|∑Tj

The GRASP-WMDD performance was also established for the problem FS|prmu|$\sum T_j$, in which there are best known and worst known solutions reported by Vallada et al. (2008). Therefore, the GRASP-WMDD was executed for the same 540 instances mentioned in the previous section, but in this case weights for jobs were not generated. In return, all job weights were set as w_j=1, i.e. it means to solve the FS|prmu|$\sum T_j$ problem. Consequently, GRASP–WMDD could be compared with 40 heuristic and metaheuristic methods evaluated by Vallada et al. (2008). Lastly, each instance was executed five times, as the authors did. Also we have used the same maximum time fixed by the authors (n×m/2×90ms). Table 6 presents the average and standard deviation of Relative Deviation Index (RDI), the percentage of best solutions found by GRASP-WMDD, and the percentage of solutions with a RDI ≤ 10%. The RDI given in Eq. (16) measures the quality of the solutions compared with the best and the worst known values for each instance. Hence, a value near to 0 represents a good solution, and a value close to 100% is the opposite.

$$Relative\ Deviation\ Index\ (RDI) = \frac{Method_{sol} - Best_{sol}}{Worst_{sol} - Best_{sol}} \times 100\% \qquad (16)$$

Table 6
RDI and percentage of best solutions given by GRASP for each instance size

Jobs	Machines	Average RDI	% best solutions	% solutions with RDI≤10%
50	10	2,92%	21,8%	100,0%
	30	6,95%	0,0%	98,7%
	50	7,83%	0,0%	93,8%
150	10	5,89%	24,0%	70,2%
	30	9,08%	1,3%	74,2%
	50	10,56%	0,0%	64,0%
250	10	10,07%	23,1%	57,3%
	30	21,72%	5,8%	12,9%
	50	25,94%	0,0%	0,9%
350	10	11,32%	22,2%	51,6%
	30	24,64%	11,1%	22,2%
	50	29,78%	0,0%	5,3%
Total general		13,89%	9,1%	54,3%

From the previous analysis, GRASP presents better results for instances with ten machines for different jobs combination. Nevertheless, the average RDI for GRASP-WMDD is 13,89% and gives the 15th position in the rank over 26 different methods compared by Vallada et al. (2008). It can be highlighted that for instances of 50 jobs in comparison to the 17 metaheuristics evaluated by the mentioned authors, GRASP-WMDD is ranked 2nd, outperforming 16 metaheuristics that includes GA, Simulated Annealing, TS and variants of them.

6. Conclusions and Future Research

This research has addressed the problem of minimizing the total weighted tardiness in a Permutation Flow Shop scheduling environment with deterministic processing times. For this purpose, a GRASP metaheuristic model algorithm was developed evaluating two utility functions (ATC and WMDD). With the statistical analysis, it was determined the best performance for all factors and double interactions evaluated according to the improvement percentage of improvement of the TWT in relation to the WEDD dispatching rule.

According to these analyzes, it was obtained that GRASP with the utility function based on the WMDD dispatching rule, the proposed method presented an average improvement of 47.8% over the WEDD dispatching rule. While GRASP with utility function based on the ATC dispatching rule has maintained an average improvement of 10.96% according to the WEDD rule. Also, for the five double interactions analysis, in which the utility function interacts with other factors, it was obtained for all cases that WMDD dispatching rule maintained the highest improvement percentage over the WEDD dispatching rule.

Likewise, the number of jobs factor in the double interactions obtained that the amount of jobs that has the highest improvement percentage in relation to the WEDD rule, are 150 jobs. When we performed the same analysis with the number of machines factor, we found that 10 machines had the highest improvement percentage according to the WEDD dispatching rule. Likewise, GRASP-WMDD was used to solve the total tardiness problem in order to compare it with other 40 heuristic and metaheuristic methods presented by Vallada et al. (2008). Results located GRASP in the 15 rank among those methods with a RDI of 13.89% and obtaining a 9% of best known solutions. Besides, GRASP shows again that its best performance was for instances with ten machines.

With the previously obtaining results, it is suggested to use a stopping criterion as the one used at this research. But, for instances with more jobs (250 and 350 jobs) and higher number of machines (50 machines), it requires a longer elapsed time. This ensures that at the GRASP local search phase, in this longer time it can make as many 2-optimal exchanges as it is possible in this additional time and it will achieve better results. That is the reason why it can be explained the best performance of the algorithm with the instances that are composed with fewer number of jobs and machines.

References

Anandaraman, C. (2011). An improved sheep flock heredity algorithm for job shop scheduling and flow shop scheduling problems. *International Journal of Industrial Engineering Computations*, *2*(4), 749–764.

Araújo, D. C., & Nagano, M. S. (2011). A new effective heuristic method for the no-wait flowshop with sequence-dependent setup times problem. *International Journal of Industrial Engineering Computations*, *2*(1), 155–166.

Armentano, V. A., & Araujo, O. C. B. de. (2006). Grasp with memory-based mechanisms for minimizing total tardiness in single machine scheduling with setup times. *Journal of Heuristics*, *12*(6), 427–446.

Armentano, V. A., & de França Filho, M. F. (2007). Minimizing total tardiness in parallel machine scheduling with setup times: An adaptive memory-based GRASP approach. *European Journal of Operational Research*, *183*(1), 100–114.

Arroyo, J. E. C., & de Souza Pereira, A. A. (2010). A GRASP heuristic for the multi-objective permutation flowshop scheduling problem. *The International Journal of Advanced Manufacturing Technology*, *55*(5-8), 741–753.

Babaei, M., Mohammadi, M., Ghomi, S. M. T. F., & Sobhanallahi, M. a. (2012). Two parameter-tuned metaheuristic algorithms for the multi-level lot sizing and scheduling problem. *International Journal of Industrial Engineering Computations*, *3*(5), 751–766.

Baker, K. R. (2013). Computational results for the flowshop tardiness problem. *Computers & Industrial Engineering*, *64*(3), 812–816.

Bank, M., Fatemi Ghomi, S. M. T., Jolai, F., & Behnamian, J. (2012). Two-machine flow shop total tardiness scheduling problem with deteriorating jobs. *Applied Mathematical Modelling*, *36*(11), 5418–5426.

Bhongade, A., & Khodke, P. M. (2012). Heuristics for production scheduling problem with machining and assembly operations. *International Journal of Industrial Engineering Computations*, *3*(2), 185–198.

Caballero-Villalobos, J. P., & Alvarado-Valencia, J. A. (2010). Greedy Randomized Adaptive Search Procedure (GRASP): A Valuable Alternative for Minimizing Machine Total Weighted Tardiness. *Ingeniería Y Universidad*, *14*(2), 275–295.

Campbell, H. G., Dudek, R. A., & Smith, M. L. (1970). A heuristic algorithm for n job, m machine sequencing problem. *Management Science*, *16*(10), 630–637.

Chandra, P., Mehta, P., & Tirupati, D. (2009). Permutation flow shop scheduling with earliness and tardiness penalties. *International Journal of Production Research*, *47*(20), 5591–5610.

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.

Du, J., & Leung, J. Y.-T. (1990). Minimizing Total Tardiness on One Machine Is NP-Hard. *Mathematics of Operations Research*, *15*(3), 483–495.

Dubois-Lacoste, J., López-Ibáñez, M., & Stützle, T. (2011). A hybrid TP+PLS algorithm for bi-objective flow-shop scheduling problems. *Computers & Operations Research*, *38*(8), 1219–1236.

El-Bouri, A. (2012). A cooperative dispatching approach for minimizing mean tardiness in a dynamic flowshop. *Computers & Operations Research*, *39*(7), 1305–1314.

Framinan, J. M., & Leisten, R. (2008). Total tardiness minimization in permutation flow shops: a simple approach based on a variable greedy algorithm. *International Journal of Production Research*, *46*(22), 6479–6498.

Gupta, A., & Chauhan, S. (2015). A heuristic algorithm for scheduling in a flow shop environment to minimize makespan. *International Journal of Industrial Engineering Computations*, *6*(2), 173–184. Retrieved from http://growingscience.com/beta/ijiec/1853-a-heuristic-algorithm-for-scheduling-in-a-flow-shop-environment-to-minimize-makespan.html

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

Kanet, J. J., & Li, X. (2004). A Weighted Modified Due Date Rule for Sequencing to Minimize Weighted Tardiness. *Journal of Scheduling*, *7*(4), 261–276.

Khalili, M., & Tavakkoli-Moghaddam, R. (2012). A multi-objective electromagnetism algorithm for a bi-objective flowshop scheduling problem. *Journal of Manufacturing Systems*, *31*(2), 232–239.

Kim, Y.-D. (1995). Minimizing total tardiness in permutation flowshops. *European Journal of Operational Research*, *85*(3), 541–555.

Laha, D., & Chakraborty, U. K. (2007). An efficient heuristic approach to total flowtime minimization in permutation flowshop scheduling. *The International Journal of Advanced Manufacturing Technology*, *38*(9-10), 1018–1025.

Lee, W.-C., & Chung, Y.-H. (2013). Permutation flowshop scheduling to minimize the total tardiness with learning effects. *International Journal of Production Economics*, *141*(1), 327–334.

Lee, W.-C., Yeh, W.-C., & Chung, Y.-H. (2014). Total tardiness minimization in permutation flowshop with deterioration consideration. *Applied Mathematical Modelling*, *38*(13), 3081–3092.

Li, X., Chen, L., Xu, H., & Gupta, J. N. D. (2015). Trajectory Scheduling Methods for minimizing total tardiness in a flowshop. *Operations Research Perspectives*, *2*, 13–23.

Liao, C. J., Liao, L. M., & Tseng, C. T. (2006). A performance evaluation of permutation vs. non-permutation schedules in a flowshop. *International Journal of Production Research*, *44*(20), 4297–4309.

Liu, Q., Ullah, S., & Zhang, C. (2011). An improved genetic algorithm for robust permutation flowshop scheduling. *The International Journal of Advanced Manufacturing Technology*, *56*(1-4), 345–354.

M'Hallah, R. (2014). Minimizing total earliness and tardiness on a permutation flow shop using VNS and MIP. *Computers & Industrial Engineering*, *75*, 142–156.

Naderi-Beni, M., Tavakkoli-Moghaddam, R., Naderi, B., Ghobadian, E., & Pourroust, A. (2012). A two-phase fuzzy programming model for a complex bi-objective no-wait flow shop scheduling. *International Journal of Industrial Engineering Computations*, *3*(4), 617–626.

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

Osman, I., Belouadah, H., Fleszar, K., & Saffar, M. (2009). Hybrid of the weighted minimum slack and shortest processing time dispatching rules for the total weighted tardiness single machine scheduling problem with availability constraints. In *Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA)* (pp. 202–215). Dublin, Ireland.

Palmer, D. S. (1965). Sequencing Jobs Through a Multi-Stage Process in the Minimum Total Time -- A Quick Method of Obtaining a Near Optimum. *Operational Research Quarterly*, *16*(1), 101–107. Retrieved from http://www.jstor.org/discover/10.2307/3006688?uid=3737808&uid=2134&uid=2&uid=70&uid=4&sid=21103754534753

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.

Pinedo, M. L. (2012). *Scheduling: Theory, Algorithms and Systems* (4th ed.). New York: Springer Science & Business Media.

Prabhaharan, G., Khan, B. S. H., & Rakesh, L. (2005). Implementation of grasp in flow shop scheduling. *The International Journal of Advanced Manufacturing Technology*, *30*(11-12), 1126–1131.

Proth, J.-M. (2007). Scheduling: New trends in industrial environment. *Annual Reviews in Control*, *31*(1), 157–166.

Rajkumar, M., Asokan, P., Anilkumar, N., & Page, T. (2011). A GRASP algorithm for flexible job-shop scheduling problem with limited resource constraints. *International Journal of Production Research*, *49*(8), 2409–2423.

Resende, M. C., & Ribeiro, C. (2003). Greedy Randomized Adaptive Search Procedures. In F. Glover & G. Kochenberger (Eds.), *Handbook of Metaheuristics SE - 8* (Vol. 57, pp. 219–249). Springer US.

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

Ruiz, R., Maroto, C., & Alcaraz, J. (2005). Solving the flowshop scheduling problem with sequence dependent setup times using advanced metaheuristics. *European Journal of Operational Research*, *165*(1), 34–54.

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.

Ruiz, R., & Stützle, T. (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), 1143–1159.

Sayadi, M. K., Ramezanian, R., & Ghaffari-Nasab, N. (2010). A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems. *International Journal of Industrial Engineering Computations*, *1*(1), 1–10.

Schaller, J., & Valente, J. M. S. (2013). A comparison of metaheuristic procedures to schedule jobs in a permutation flow shop to minimise total earliness and tardiness. *International Journal of Production Research*, *51*(3), 772–779.

Shahsavari Pour, N., Tavakkoli-Moghaddam, R., & Asadi, H. (2013). Optimizing a multi-objectives flow shop scheduling problem by a novel genetic algorithm. *International Journal of Industrial Engineering Computations*, *4*(3), 345–354.

Shahul Hamid Khan, B., Prabhaharan, G., & Asokan, P. (2007). A grasp algorithm for m-machine flowshop scheduling problem with bicriteria of makespan and maximum tardiness. *International Journal of Computer Mathematics*, *84*(12), 1731–1741.

Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, *3*(1-2), 59–66.

Sun, Y., Zhang, C., Gao, L., & Wang, X. (2010). Multi-objective optimization algorithms for flow shop scheduling problem: a review and prospects. *The International Journal of Advanced Manufacturing Technology*, *55*(5-8), 723–739.

Swaminathan, R., Pfund, M. E., Fowler, J. W., Mason, S. J., & Keha, A. (2007). Impact of permutation enforcement when minimizing total weighted tardiness in dynamic flowshops with uncertain processing times. *Computers & Operations Research*, *34*(10), 3055–3068.

Swaminathan, R., Pfund, M. E., Fowler, J. W., Mason, S. J., & Keha, A. (2007). Impact of permutation enforcement when minimizing total weighted tardiness in dynamic flowshops with uncertain processing times. *Computers & Operations Research*, *34*(10), 3055–3068.

Tasgetiren, M. F., Pan, Q.-K., Suganthan, P. N., & Oner, A. (2013). A discrete artificial bee colony algorithm for the no-idle permutation flowshop scheduling problem with the total tardiness criterion. *Applied Mathematical Modelling*, *37*(10-11), 6758–6779.

Vallada, E., & Ruiz, R. (2011). A genetic algorithm for the unrelated parallel machine scheduling problem with sequence dependent setup times. *European Journal of Operational Research*, *211*(3), 612–622.

Vallada, E., Ruiz, R., & Minella, G. (2008). Minimising total tardiness in the m-machine flowshop problem: A review and evaluation of heuristics and metaheuristics. *Computers & Operations Research*, *35*(4), 1350–1373.

Vega-Mejía, C. A., & Caballero-Villalobos, J. P. (2010). Combined Use of GRASP and Path-Relinking during Production Scheduling in order to Minimize Total Weighted Tardiness in a Machine. *Ingeniería Y Universidad*, *14*(1), 79–96.

Vepsalainen, A., & Morton, T. E. (1987). Priority rules and lead time estimation for job shop scheduling with weighted tardiness costs. *Management Science*, *33*(8), 1036–1047.

Wu, C.-C., & Lee, W.-C. (2009). A note on the total completion time problem in a permutation flowshop with a learning effect. *European Journal of Operational Research*, *192*(1), 343–347.

Xiao, Y.-Y., Zhang, R.-Q., Zhao, Q.-H., & Kaku, I. (2012). Permutation flow shop scheduling with order acceptance and weighted tardiness. *Applied Mathematics and Computation*, *218*(15), 7911–7926.

Yenisey, M. M., & Yagmahan, B. (2014). Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. *Omega*, *45*, 119–135.