International Journal of Industrial Engineering Computations 11 (2020) 469–480

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Solving the permutation flow shop problem with blocking and setup time constraints

Mauricio Iwama Takano[a] and Marcelo Seido Nagano[b*]

[a]*Federal Technological University - Paraná, Av. Alberto Carazzai, 1640, 86300-000, Cornélio Procópio, PR, Brazil*
[b]*University of São Paulo, School of Engineering of São Carlos, Department of Production Engineering, Av. Trabalhador São-carlense, 400, 13566-590, São Carlos, SP, Brazil*

C H R O N I C L E

Article history:
Received October 5 2019
Received in Revised Format November 6 2019
Accepted November 6 2019
Available online November 6 2019

Keywords:
Scheduling
Flow shop
Blocking
Setup time constraints
Mixed-integer programming model
Iterated Greedy

A B S T R A C T

In this paper, the flow shop with blocking and sequence and machine dependent setup time problem aiming to minimize the makespan is studied. Two mixed-integer programming models are proposed (TNZBS1 and TNZBS2) and two other mixed-integer programming models, originally proposed for the no setup problem, are adapted to the problem. Furthermore, an Iterated Greedy algorithm is proposed for the problem. The permutation flow shop with blocking and sequence and machine dependent setup time is an underexplored problem and the authors did not find the use of mixed-integer programming models for the problem in any other work. To compare the models, a database of 80 problems was generated, which vary in number of machines and jobs. For the small sized problems, the adapted MILP model obtained the best results. However, for bigger problems, both proposed MILP models obtained significantly better results compared to the adapted models, proving the efficiency of the new models. When comparing the Iterated Greedy algorithm with the MILP models, the former outperformed the latter.

1. Introduction

This paper addresses the permutation flow shop problem, which is a set of n jobs that must be processed in m machines, all of the jobs having the same flow in all the machines. In the presented problem, the sequence and machine dependent setup is also considered. The sequence and machine dependent setup time constraint can be used as a general case of the sequence dependent setup. This is because if one considers the setup time in all machines to be the same (only dependent on the sequence) it is the same as the sequence dependent setup constraint. Moreover, the blocking constraint with no buffer is considered between the machines (zero buffer), resulting in a higher possibility of blocking a machine after it finishes processing a job. As there is no buffer between machines, when a job j finishes being processed by machine k and machine (k+1) is still processing job (j-1) or is still being set up, the job remains in machine k, blocking it from receiving job (j+1). In this paper, the evaluation criterion is the minimization of the makespan. Papadimitriou and Kanellakis (1980) proved that the problem with a limited buffer of only one unit between machines is NP-HARD. Afterwards, Hall and Sriskandarajah

* Corresponding author Tel.: 55 16 3373-9428; Fax: 55 16 3373-9425
E-mail: drnagano@usp.br (M. S. Nagano)

2020 Growing Science Ltd.
doi: 10.5267/j.ijiec.2019.11.002

(1996), based on the results obtained by Papadimitriou and Kanellakis (1980), showed that the permutation flow shop with three work stations and blocking problem is strongly NP-complete. In the same paper, the authors related the main works developed in the literature. Recently, a literature review for the *m*-machine flow shop, ranging from 1970 up to 2019, can be seen in Miyata and Nagano (2019). The first study to address the flow shop problem with a limited buffer and sequence dependent setup found in the literature is Norman (1999). The evaluation criterion used in this study is the minimum makespan. A Tabu search and two adapted constructive heuristics methods (*NEH* and *PF*) were presented to solve the problem. A greedy improvement procedure was added to the constructive heuristics. Furthermore, 900 problems were generated to evaluate the proposed methods, varying setup times, buffer sizes and number of jobs and machines. Takano and Nagano (2019) evaluated 28 different heuristics for the zero buffer with sequence dependent setup times problem. The objective function considered was the minimization of the makespan. Each heuristic solved 480 problems, varying in the number of jobs, number of machines and setup times. Moslehi and Khorasanian (2013) addressed the permutation flow shop problem with zero buffer. They proposed two mixed integer linear programming (*MILP*), an initial upper bound generator and some lower bounds and dominance rules to be used in a branch-and-bound (B&B) algorithm to minimize the total completion time. The *MILP* models had some difficulties in solving instances with sizes (*n*,*m*) equal to (16,10), (18,7), and (18,10). The B&B model was able to solve 30 of the 120 instances from the Taillard (1993) database. Sanches et al. (2016) evaluated the efficiency of five different constructive heuristics to provide an initial solution for the B&B algorithm. A flow shop with a zero buffer environment was considered aiming to minimize the makespan. Results show that the constructive heuristic that obtained the best results will not necessarily be better for the algorithm. This is due to the computational time required to calculate the initial solution, that is, in some cases the computational time required to solve the heuristics (which provides the initial solution) seems to affect more the total computational time than the quality of the initial solution itself. Mixed Integer Linear Programming (*MILP*) models can be used to find the optimum solution for small and medium problems. Computational research in this field has grown considerably, see for example Pan (1997), Stafford (1988), Zhu and Heady (2000), Stafford, Tseng, and Gupta (2005), Ronconi and Birgin (2012). Despite this, the use of *MILP* models to optimize scheduling problems in a permutation flow shop with blocking environment is not yet widely reported due to the high computational time. Among the papers, only two studies applied *MILP* models to the problem with blocking. Ronconi and Birgin (2012) presented two *MILP* models for the problem, and four more models for the problem without blocking, all of them aiming to minimize the total earliness and tardiness. The models were tested in 320 problems that varied in the number of machines, jobs, and in the values of due dates. The results showed that only the number of binary variables of the model does not necessarily indicate the difficulty of solving it. Maleki-Darounkolaei et al. (2012) addressed the flow shop problem with three workstations, sequence dependent setup time, and blocking with two objectives (minimizing the makespan and the flow time). They developed a Simulated Annealing (SA) algorithm and a *MILP* model for the problem. In this paper, problems with more than nine jobs were not solved using the *MILP* model because of the elevated computational time. It is important to notice that despite the fact that Maleki-Darounkolaei et al. (2012) addressed the use of a *MILP* model to solve the blocking with a dependent setup time, they did not consider the zero buffer constraint, and the dependent setup time was considered only in the first work station. Furthermore, Maleki-Darounkolaei et al. (2012) consider a flexible flow shop environment with the objective function of minimizing both the makespan and the flow time, whereas in this paper a non-flexible flow shop environment aiming at minimizing the makespan is considered.

In this paper, two *MILP* models are proposed, as well as the adaptation of the two models proposed for blocking problems by Ronconi and Birgin (2012). The objective of this paper is to compare, in relation to the computational time, the four *MILP* models to find a solution for scheduling problems in a permutation flow shop environment with a zero buffer and sequence and machine dependent setup time, with the objective function of minimizing the makespan. The four MILP models are then compared to an Iterated Greedy algorithm. This paper is organized as follows. In section 2, the proposed MILP models are presented for the problem. In Section 3, the adapted MILP models are discussed concerning the

problem. In Section 4, the adapted Iterated Greedy algorithm is presented and in Section 5, a comparison between the IG algorithm and the MILP models is made. The conclusions are drawn in Section 6.

2. Proposed Models

In a permutation flow shop problem with a zero buffer constraint, a machine remains blocked when it finishes processing a job and the next machine is not prepared to receive this job. Initially two *MILP* models are presented, which are proposed for the problem (TNZBS1 and TNZBS2). Afterwards, two other *MILP* models, adapted from Ronconi and Birgin (2012), are presented. In TNZBS1, the makespan equations are directly programmed into the model, using the $R_{\sigma k}$ and $C_{\sigma k}$ indexes to compute the completion time of the setup and processing of the jobs, respectively. In TNZBS2, the starting time of the processing of a job ($e_{\sigma k}$) is calculated and then added to the processing time of this job to obtain its completion time. Then the gap between two consecutive jobs ($B_{\sigma k}$) is calculated and then summed up with the completion time of the job to obtain its departure time. In RBZBS1, a structural property of the problem is used to connect all the variables of the problem (depicted in Fig. 2 and Fig. 3), and thus, calculates the departure time of the jobs. In RBZBS2, the makespan equations are directly programmed into the model, however without using the $R_{\sigma k}$ and $C_{\sigma k}$ indexes. The notations used for the models are:

n	Number of jobs;
m	Number of machines;
j	A job from the sequence;
i	The job that directly precedes job j in the sequence;
σ	A position in the sequence;
P_{jk}	Processing time of job j in machine k;
S_{ijk}	Setup time of machine k between the completion time of job i and the starting time of job j;
$R_{\sigma k}$	Completion time of the setup of machine k before the σth job in the sequence;
$C_{\sigma k}$	Completion time of processing the σth job in the sequence at machine k;
$D_{\sigma k}$	Departure time of the σth job in the sequence at machine k; i.e. time that the σth job in the sequence liberates machine k after finishing its processing. $D_{\sigma k} \geq C_{\sigma k}$: if there is no blocking $D_{\sigma k} = C_{\sigma k}$; otherwise $D_{\sigma k} > C_{\sigma k}$;
$x_{j\sigma}$	$\begin{cases}1 & \text{If job } j \text{ is the } \sigma\text{th job in the sequence;}\\ 0 & \text{otherwise;}\end{cases}$
$y_{ij\sigma}$	$\begin{cases}1 & \text{If job } i \text{ directly precedes job } j\text{, which is the σth job in the sequence;}\\ 0 & \text{otherwise;}\end{cases}$
$e_{\sigma k}$	Starting time of the processing of the σth job in the sequence in machine k;
$I_{\sigma k}$	Gap between the completion time of the setup of machine k to the σth job in the sequence and the starting time of its processing in the machine, in other words, is the idle time of machine k;
$B_{\sigma k}$	Gap between the completion time of the processing of the σth job in the sequence in machine k and its starting time in machine (k+1), in other words, the blocking time of machine k.

- **Model TNZBS1**

$$\text{Minimize: } D_{max} = D_{nm} \tag{1}$$
$$\sum_{j=1}^{n} x_{j\sigma} = 1 \qquad \sigma = 1, \dots, n \tag{2}$$
$$\sum_{\sigma=1}^{n} x_{j\sigma} = 1 \qquad j = 1, \dots, n \tag{3}$$
$$y_{ij\sigma} \geq x_{j\sigma} + x_{i,\sigma-1} - 1 \qquad i = 1, \dots, n; j = 1, \dots, n; \sigma = 2, \dots, n; i \neq j \tag{4}$$
$$\sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij\sigma} = 1 \qquad \sigma = 2, \dots, n \tag{5}$$
$$\sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij1} = 0 \tag{6}$$
$$R_{1k} = \sum_{i=1}^{n} \sum_{j=i} S_{ijk} \cdot x_{j1} \qquad k = 1, \dots, m \tag{7}$$
$$R_{\sigma k} = D_{\sigma-1,k} + \sum_{i=1}^{n} \sum_{j\neq i} S_{ijk} \cdot y_{ij\sigma} \qquad \sigma = 2, \dots, n; k = 1, \dots, m \tag{8}$$
$$D_{\sigma 1} \geq R_{\sigma 1} + \sum_{j=1}^{n} P_{j1} \cdot x_{j\sigma} \qquad \sigma = 1, \dots, n \tag{9}$$
$$D_{\sigma k} \geq R_{\sigma,k+1} \qquad \sigma = 1, \dots, n; k = 1, \dots, m-1 \tag{10}$$
$$D_{\sigma k} \geq D_{\sigma,k-1} + \sum_{j=1}^{n} P_{jk} \cdot x_{j\sigma} \qquad \sigma = 1, \dots, n; \mathrm{k} = 2, \dots, m \tag{11}$$

Constraints (2) and (3) guarantee that each job will be allocated in only one position in the sequence, and that each position in the sequence has only one job associated with it. Constraint (5) and constraint (4)

ensure that each job will have only one job that precedes it in the sequence. Constraint (6) guarantees that no job will precede the first job in the sequence. Constraints (7) and (8) are used to calculate the completion time of the setup of the machines. Constraint (7) is applied only to the first job in the sequence, whose completion time of the setup depends only on the setup time (represented by S_{ijk}). On the other hand, constraint (8) is the general formula of the completion time of the setup of the machines, in other words, the departure time of the (σ-1)th job in the sequence summed to the setup time of machine k between the processing of the (σ-1)th and the σth jobs in the sequence. Constraints (9-11) are used to calculate the departure time of the jobs in the machines, considering the possibility of blocking. Constraint (9), which is illustrated in Fig. 1a, is applied to all the jobs only in the first machine, where, because there is no idle time, the starting time of processing all jobs is equal to the completion time of the setup of the machine. The departure time of a job in a machine is the time when the job leaves the machine, and, as there is no buffer in between machines, a job cannot leave a machine until the proceeding machine is ready to start processing it. Therefore, if $D_{\sigma k} = R_{\sigma,k+1}$, blocking might have occurred and its value is greater than or equal to zero. Constraint (10) determines the value of $D_{\sigma k}$ when blocking occurs as illustrated in Fig. 1b. If $D_{\sigma k} > R_{\sigma,k+1}$ then a block has not occurred in the machine, and constraint (11) will determine the value of $D_{\sigma k}$, as illustrated in Fig. 1c.

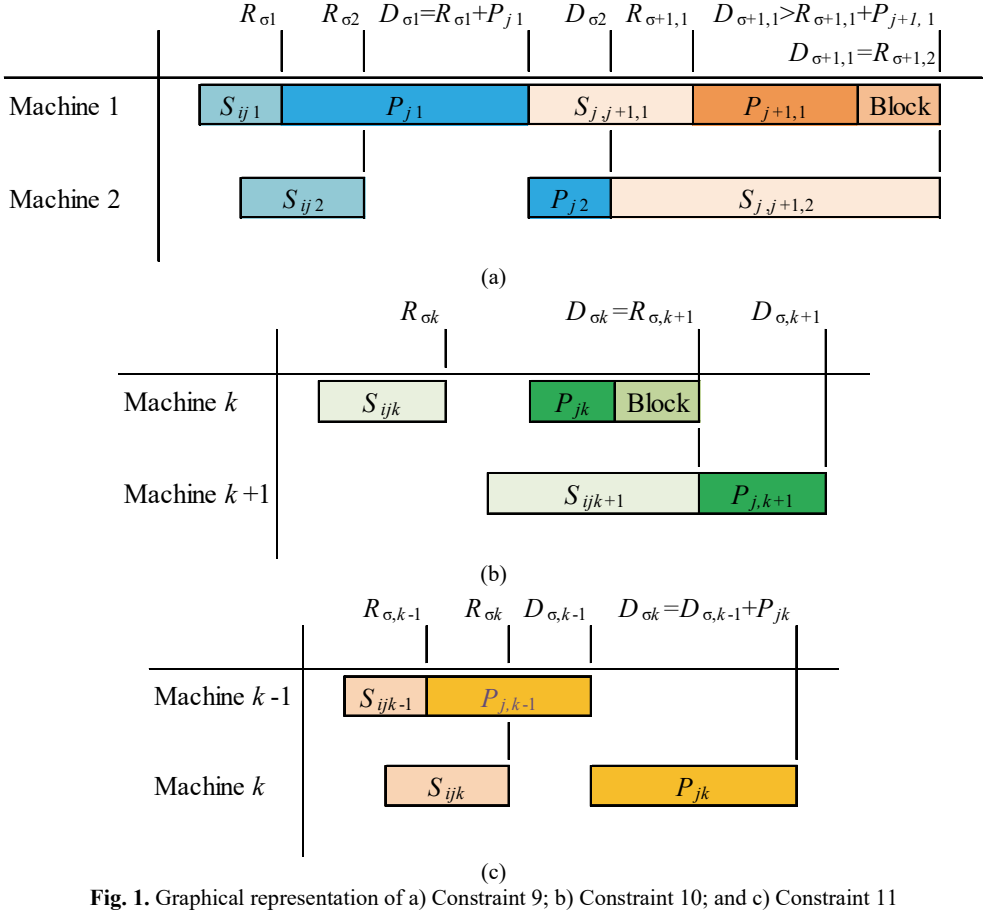

Fig. 1. Graphical representation of a) Constraint 9; b) Constraint 10; and c) Constraint 11

- **Model TNZBS2**

$$\text{Minimize: } D_{max} = D_{nm} \tag{12}$$

$$\sum_{j=1}^{n} x_{j\sigma} = 1 \qquad \sigma = 1, \dots, n \tag{13}$$

$$\sum_{\sigma=1}^{n} x_{j\sigma} = 1 \qquad j = 1, \dots, n \tag{14}$$

$$y_{ij\sigma} \geq x_{j\sigma} + x_{i,\sigma-1} - 1 \qquad i = 1, \dots, n; j = 1, \dots, n; \sigma = 2, \dots, n; i \neq j \tag{15}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij\sigma} = 1 \qquad \sigma = 2, \dots, n \tag{16}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij1} = 0 \tag{17}$$

$$R_{1k} = \sum_{i=1}^{n} \sum_{j=i} S_{ijk} \cdot x_{j1} \qquad k = 1, \dots, m \qquad (18)$$

$$R_{\sigma k} = D_{\sigma-1,k} + \sum_{i=1}^{n} \sum_{j \neq i} S_{ijk} \cdot y_{ij\sigma} \qquad \sigma = 2, \dots, n; k = 1, \dots, m \qquad (19)$$

$$e_{\sigma 1} = R_{\sigma 1} \qquad \sigma = 1, \dots, n \qquad (20)$$

$$e_{\sigma k} \geq D_{\sigma,k-1} \qquad \sigma = 1, \dots, n; k = 2, \dots, m-1 \qquad (21)$$

$$D_{\sigma,k-1} \geq R_{\sigma k} \qquad \sigma = 1, \dots, n; k = 2, \dots, m \qquad (22)$$

$$D_{\sigma k} = C_{\sigma k} + B_{\sigma k} \qquad \sigma = 1, \dots, n; k = 1, \dots, m \qquad (23)$$

$$B_{1k} \geq R_{1,k+1} - C_{1k} \qquad k = 1, \dots, m-1 \qquad (24)$$

$$B_{\sigma m} = 0 \qquad \sigma = 1, \dots, n \qquad (25)$$

$$C_{\sigma k} = e_{\sigma k} + \sum_{j=1}^{n} P_{jk} \cdot x_{j\sigma} \qquad \sigma = 1, \dots, n; k = 1, \dots, m \qquad (26)$$

Constraints (20) and (21) define the starting time of processing the jobs in the machines. Constraint (20) is only applied to the first machine, where there is no idle time between the completion time of the setup and the starting time of the processing. Therefore, the starting time of processing the jobs is equal to the completion time of the setup of the machine. Constraint (21) is the general formula to the starting time of processing the jobs, which is greater than or equal to the completion time of processing the job in the preceding machine. If $e_{\sigma k} = C_{\sigma,k-1}$, then there was no blocking in machine (k-1) nor idle in machine k, if $e_{\sigma k} > C_{\sigma,k-1}$, then blocking occurred in machine (k-1), or idle in machine k, or both (blocking in machine k-1 and idle in machine k). Constraints (22) and (23) determine the time that each job leaves a machine. Constraint (22) is applied to all machines except the first one, and guarantees that no job will leave a machine until the preceding machine is ready to receive it. Constraint (23) is applied to all machines and defines that the time that a job leaves a machines is equal to the completion time of processing the job in that machine plus the blocking of that machine. Constraints (24-25) are used to calculate the blocking time of the machines. Constraint (25) guarantees that there will be no blocking in the last machine, and constraint (24) is used to calculate the blocking time of the remaining machines for the first job in the sequence. Constraint (26) is used to calculate the completion time of processing the jobs in the machines, which are equal to the starting time of processing the jobs plus the processing time of jobs in the machines.

3. Adapted models

Ronconi and Birgin (2012) proposed two models for the permutation flow shop problem with a zero buffer aiming to minimize the earliness and tardiness (MZB1 and MZB2). The first and second models adapted for the problem with both blocking and sequence and machine dependent setup constraints aiming to minimize the makespan (RBZBS1 and RBZBS2) are presented.

- **Model RBZBS1**

Minimize: $D_{max} = D_{nm}$ (27)

$$\sum_{j=1}^{n} x_{j\sigma} = 1 \qquad \sigma = 1, \dots, n \qquad (28)$$

$$\sum_{\sigma=1}^{n} x_{j\sigma} = 1 \qquad j = 1, \dots, n \qquad (29)$$

$$y_{ij\sigma} \geq x_{j\sigma} + x_{i,\sigma-1} - 1 \qquad i = 1, \dots, n; j = 1, \dots, n; \sigma = 2, \dots, n; i \neq j \qquad (30)$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij\sigma} = 1 \qquad \sigma = 2, \dots, n \qquad (31)$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij1} = 0 \qquad (32)$$

$$I_{\sigma 1} = 0 \qquad \sigma = 1, \dots, n \qquad (33)$$

$$B_{\sigma m} = 0 \qquad \sigma = 1, \dots, n \qquad (34)$$

$$\sum_{i=1}^{n} \sum_{j=i} S_{ijk} \cdot x_{j1} + I_{1k} + \sum_{j=1}^{n} P_{jk} \cdot x_{j1} + B_{1k} = \sum_{i=1}^{n} \sum_{j=i} S_{i,j,k+1} \cdot x_{j1} + I_{1,k+1} \qquad k = 1, \dots, m-1 \qquad (35)$$

$$\sum_{i=1}^{n}\sum_{j\neq i} S_{ijk}\cdot y_{i,j,\sigma+1} + \mathrm{I}_{\sigma+1,k} + \sum_{j=1}^{n} P_{jk}\cdot x_{j,\sigma+1} + B_{\sigma+1,k}$$
$$= \sum_{j=1}^{n} P_{j,k+1}\cdot x_{j,\sigma} + B_{\sigma,k+1} \qquad \sigma = 1,\dots,n-1; k = 1,\dots,m-1 \qquad (36)$$
$$+ \sum_{i=1}^{n}\sum_{j\neq i} S_{i,j,k+1}\cdot y_{i,j,\sigma+1} + I_{\sigma+1,k+1}$$

$D_{1m} \geq \sum_{i=1}^{n}\sum_{j=i} S_{i,j,1}\cdot x_{j1} + \sum_{k=1}^{m}\left(B_{1k} + \sum_{j=1}^{n}\left(P_{jk}\cdot x_{j1}\right)\right)$ (37)

$D_{\sigma m} \geq C_{\sigma-1,m} + \sum_{i=1}^{n}\sum_{j\neq i} S_{ijm}\cdot y_{ij\sigma} + I_{\sigma m} + \sum_{j=1}^{n} P_{jk}\cdot x_{j\sigma} \qquad \sigma = 2,\dots,n$ (38)

Constraints (33) and (34) guarantee that there will be no idle in the first machine nor blocking in the last machine, respectively. Constraints (35) and (36) establish a relation between the idle, blocking, processing time and setup time. Constraint (35) establishes this relation for the first job in the sequence, and is illustrated in Fig. 2. Constraint (36) establishes this relation for the other jobs except for the last job in the sequence and is illustrated in Fig. 3. Constraints (37) and (38) are used to calculate the completion time of processing the jobs in the last machine. Constraint (37) is used to calculate the completion time of processing the first job in the sequence in the last machine, which is equal to the sum of the setup time in the first machine, the blocking times of all machines, and the processing times in all machines. Constraint (38) is the general formula of the calculus of the completion time of processing the jobs in the last machine, which is the sum of the completion time of processing the preceding job in the machine, the setup time of the last machine, and the idle and processing time of the job.

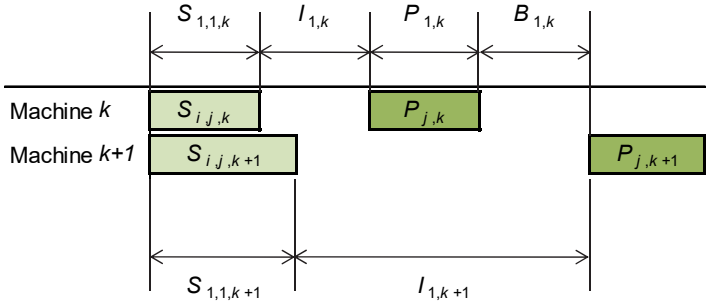

Fig. 2. Graphical representation of constraint (35), which expresses the relation between the setup time, the processing time, the idle time, and the blocking time of the first job in the sequence

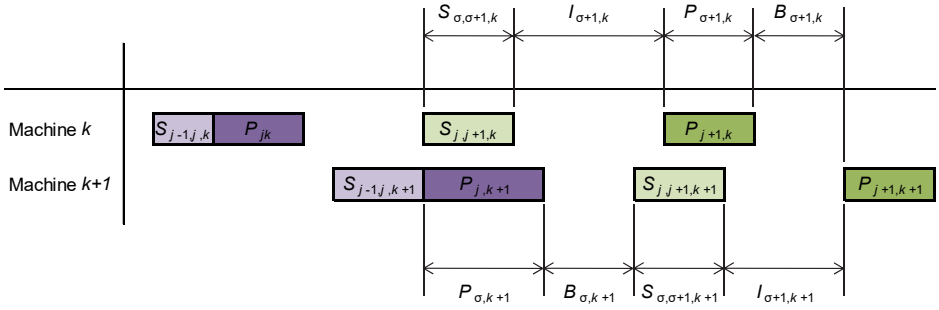

Fig. 3. Graphical representation of constraint (36), which express the relation between the setup time, the processing time, the idle time, and the blocking time to all other jobs.

- **Model RBZBS2**

Minimize: $D_{max} = D_{nm}$ (39)

$\sum_{j=1}^{n} x_{j\sigma} = 1$ $\sigma = 1,\dots,n$ (40)

$\sum_{\sigma=1}^{n} x_{j\sigma} = 1$ $j = 1,\dots,n$ (41)

$y_{ij\sigma} \geq x_{j\sigma} + x_{i,\sigma-1} - 1$ $i = 1,\dots,n; j = 1,\dots,n; \sigma = 2,\dots,n; i \neq j$ (42)

$\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij\sigma} = 1$ $\sigma = 2,\dots,n$ (43)

$\sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij1} = 0$ (44)

$D_{11} \geq \sum_{i=1}^{n}\sum_{j=\mathrm{i}} S_{i,j,1}\cdot x_{j1} + \sum_{j=1}^{n} P_{j1}\cdot x_{j1}$ (45)

$D_{1k} \geq D_{1,k-1} + \sum_{j=1}^{n} P_{jk}\cdot x_{j1}$ $k = 2,\dots,m$ (46)

$$D_{1k} \geq \sum_{i=1}^{n} \sum_{j=i} S_{i,j,k+1} \cdot x_{j1} \qquad k = 1, \dots, m-1 \qquad (47)$$

$$D_{\sigma k} \geq D_{\sigma-1,k} + \sum_{i=1}^{n} \sum_{j \neq i} S_{ijk} \cdot y_{ij\sigma} + \sum_{j=1}^{n} P_{jk} \cdot x_{j\sigma} \qquad \sigma = 2, \dots, n; \mathrm{k} = 1, \dots, \mathrm{m} \qquad (48)$$

$$D_{\sigma k} \geq D_{\sigma,k-1} + \sum_{j=1}^{n} P_{jk} \cdot x_{j\sigma} \qquad \sigma = 2, \dots, n; \mathrm{k} = 2, \dots, m \qquad (49)$$

$$D_{\sigma k} \geq D_{\sigma-1,k+1} + \sum_{i=1}^{n} \sum_{j \neq i} S_{i,j,k+1} \cdot y_{ij\sigma} \qquad \sigma = 2, \dots, n; \mathrm{k} = 1, \dots, m-1 \qquad (50)$$

Constraints (45-50) are used to calculate the completion time of processing the jobs in the machines, considering the possibility of blocking in the machines. Constraint (45) is only applied to the first job in the sequence in the first machine, defining that the completion time of processing the job in the machine is greater than or equal to the setup time of the first job in the sequence in the first machine plus the processing time of that job in that machine. Constraint (46) is applied to the first job in the sequence in all machines, except for the first one, defining that the completion time of processing the job in the machine is greater than or equal to the completion time of the job in the preceding one plus the processing time of the job in the machine. Constraint (47) is applied to the first job in the sequence in the other machine, except for the last one, and defines the completion time of processing the job in the machines, which is greater than or equal to the setup time of the following machine. Altogether the constraints (45-47) define the completion time of processing the first job in the sequence of the machines, if the completion time of processing the first job in the sequence is limited by constraint (45), or if the completion time of processing the other jobs is limited by constraint (46), then no blocking has occurred. If the values of the completion times of processing the jobs are limited by constraint (47), then $B_{\sigma k} \geq 0$. Constraint (48) guarantees that the completion times of processing all the jobs, except for the first one, are greater than or equal to the completion time of processing the preceding job plus the setup time of the machine and the processing time of the job in the machine. Constraint (49) guarantees that the completion times of processing all the jobs, except for the first one, in machine two to the last one, are greater than or equal to the completion time of processing the job in the preceding machine plus the processing time of the job in the machine. Finally, constraint (50) guarantees that the completion times of processing all the jobs, except for the first one, in all machines, except the last one, are greater than or equal to the completion time of processing the preceding job in the following machine plus the setup time of the job in the following machine. If the completion time of processing the job in the machine is limited by constraints (48) or (49), then no blocking has occurred. On the other hand, if the completion time of the processing of the job in the machine is limited by constraint (50) then, $B_{\sigma k} \geq 0$. The characteristics of the four models presented for the permutation flow shop problem with a zero buffer and sequence and machine dependent setup time, aiming at minimizing the makespan, are listed in Table 1. The characteristics are related to the size of each model, expressed by the number of constraints, and number of binary and continuous variables.

Table 1
Number of variables and constraints of the models presented

Model		Binary variables	Continuous variables	Constraints
Proposed Model	TNZBS1	$n^3 + n^2$	$2nm + 1$	$n^3 - 2n^2 + 3n + 3nm$
Proposed Model	TNZBS2	$n^3 + n^2$	$5nm + 1$	$n^3 - 2n^2 + 3n + 5nm + m - 1$
Adapted Model	RBZBS1	$n^3 + n^2$	$2nm + n + 1$	$n^3 - 2n^2 + 6n + nm$
Adapted Model	RBZBS2	$n^3 + n^2$	$nm + 1$	$n^3 - 2n^2 + 2n + 3nm - m + 1$

4. Iterated Greedy

An Iterated Greedy (IG) algorithm proposed by Pan and Ruiz (2014) was adapted for the problem. The algorithm consists of two main steps: 1. An initial solution is built, usually by a constructive heuristic; 2. A destruction-reconstruction operator is applied to the initial solution until an end criterion is reached. The proposed IG starts by obtaining an initial solution using an improved $FRB4_k$ method (originally proposed by Rad *et al*., 2009). Then, the solution is improved by a Referenced Local Search (RLS) providing a sequence (π). After this, in the destruction phase, d jobs are randomly extracted from sequence π and inserted into a list of removed jobs π_R. Then, in the reconstruction phase, all jobs in π_R are reinserted, one by one, back into π using the NEH insertion procedure. Fig. 4 shows the IG procedure adapted from Pan and Ruiz (2014).

Procedure $IG\ (d,T)$
$\pi \coloneqq FRB4_1^*$
$\pi \coloneqq RLS(\pi)$
$\pi_b \coloneqq \pi$
while (termination criterion not satisfied) **do**
$\pi' \coloneqq \pi$
for $i \coloneqq 1$ to d **do** %Destruction phase
$\pi' \coloneqq$ remove one job at random from π' and insert it in π'_R
endfor
for $i \coloneqq 1$ to d **do** %Reconstruction phase
$\pi' \coloneqq$ Insert job $\pi'_{R(i)}$ in position p resulting in the best C_{max}
% Improved eDC operator
$\pi' \coloneqq$ Reinsert jobs $\pi'_{(p\pm1)}$ in positions resulting in the best C_{max}
endfor
$\pi'' \coloneqq RLS(\pi')$
if $C_{max}(\pi'') < C_{max}(\pi)$ **then** % Acceptance Criterion
$\pi \coloneqq \pi''$
if $C_{max}(\pi) < C_{max}(\pi_b)$ **then** %New best solution
$\pi_b \coloneqq \pi$
endif
elseif $\left(random \le e^{-\frac{(C_{max}(\pi'')-C_{max}(\pi))}{Temp}} \right)$ **then**
$\pi \coloneqq \pi''$
endif
endwhile
end

Fig. 4. Iterated Greedy Method (Adapted from Pan and Ruiz, 2014)

5. Computational results for the models

Due to the high computational time, the *MILP* model is recommended for small or medium sized classes of problems. Therefore, instead of using the Taillard (1993) database, which consists of relatively large problems, a new database was generated to compare the presented models, comprising 80 problems, which are separated into eight different classes that vary in the number of jobs (n) and number of machines (m). The problem classes are:

$$(n,m) = \{(5,3);(10,3);(10,7);(10,10);(15,3);(15,7);(15,10);(20,3)\}$$

Each problem class has 10 different problems. The processing times were uniformly distributed between 1 and 99 (as proposed by Taillard, 1993). The setup times for the machines were generated for these tests using the same method, i.e. the values were uniformly distributed between 1 and 99. By doing so, it is possible that the setup time might be lower, equal, or higher than the processing time, without a very large discrepancy on the values. Much lower values of setup times might generate problems where there is no blocking, and much higher values of the setup time might impose too many blocking occurrences. The MILP models were written in GAMS software and solved using CPLEX 12 and the IG algorithm was programmed in Python. The computational experiments were performed in a 2.3 GHz Inter® core i7 3610QM with 8 Gb DDR3 RAM memory and Windows 7 operational system. It was set out a limit for the computational time of 3600 seconds to solve each problem using each of the MILP models and the relative termination tolerance was set to zero for all problems and all models. The termination criterion of the IG algorithm was set by a predetermined elapsed CPU time according to the expression $t = n * (m/2) * \rho$ milliseconds, where ρ was set to 60. For the sake of comparison, the mean computational time (CPU time) was calculated in seconds (Table 2), the mean number of simplex iterations (Simplex it) is shown in Table 3, and the mean number of nodes is explored in the branch-and-bound tree (B&B nodes) shown in Table 4. As the computational time was limited to 3600 seconds, some models have not been able to obtain the optimum result for some of the problems. Therefore, the mean relative deviation (*MRD*) was also calculated of the obtained results (makespan) shown in Table 5. The mean relative deviation of the makespan indicates the quality of the obtained result; the lower the value, the better. It is obtained by Eq. (51).

$$MRD = \frac{1}{10} * \sum_{1\le i\le 10} (DM_i - DM_i^*)/DM_i^* * 100 \quad (51)$$

where,

i : A problem of the database;
DM_i : The result obtained by the algorithm for problem i;
DM_i^* : The best result obtained by all the algorithms for problem i.

Tables 2, 3, 4, and 5 show the results obtained by the models. Each cell of the table represents the mean value of a set of 10 problems (one class of problems). In the tables, the total mean value of the parameters were also calculated for all models. Moreover, in Tables 2, 3 and 4, the mean values of the parameters were calculated for all the models in which the "CPU time" of the MILP models is less than 3600 seconds. In Tables 3, 4 and 5, the mean values of the parameters only considering the models in which "CPU time" is greater than or equal to 3600 seconds were calculated. Tables 2 and 5 also show the results for the IG algorithm. The best results for each category are shown in bold. In Table 2, the best "CPU time" among the MILP models for each category is also highlighted.

Table 2
Comparison of the computational time

Size		CPU time (seconds)				
n	*m*	TNZBS1	TNZBS2	RBZBS1	RBZBS2	IG
5	3	0.146	0.169	**0.145**	0.151	**0.45**
10	3	213.1	221.0	**196.1**	209.5	**0.9**
10	7	477.7	553.3	**452.7**	548.0	**2.1**
10	10	453.3	421.1	**379.5**	477.7	**3.0**
15	3	3600	3600	3600	3600	**1.35**
15	7	3600	3600	3600	3600	**3.15**
15	10	3600	3600	3600	3600	**4.5**
20	3	3600	3600	3600	3600	**1.8**
Total mean		1943.03	1949.45	**1928.56**	1954.42	**2.156**
Mean ("CPU time" < 3600)		286.07	298.90	**257.11**	308.83	**2.7**

Table 3
Comparison of the number of simplex iterations of the *MILP* models

Size		Simplex it			
n	*m*	TNZBS1	TNZBS2	RBZBS1	RBZBS2
5	3	1245.9	1448.1	**1112.0**	1373.3
10	3	**4085221.8**	4136571.1	4161631.2	4089335.0
10	7	**7799378.5**	8893589.7	8252105.5	8692875.5
10	10	6691378.6	**5874142.0**	6218593.4	6722366.7
15	3	25935828.6	**25923635.6**	27008938.3	26541075.1
15	7	19538952.4	**18918844.4**	21884126.6	19362250.3
15	10	17142515.0	16296682.0	18918547.7	**15973099.9**
20	3	9093888.7	**8782921.6**	9040474.9	8992677.9
Total mean		11286051.2	**11103479.3**	11935691.2	11296881.7
Mean ("CPU time" >= 3600)		17927796.1750	**17480520.9000**	19213021.8750	17717275.8000
Mean ("CPU time" < 3600)		**4644306.2**	4726437.7	4658360.5	4876487.6

Table 4
Comparison of the number of explored nodes in the branch-and-bound tree of the *MILP* models

Size		B&B nodes			
n	*m*	TNZBS1	TNZBS2	RBZBS1	RBZBS2
5	3	55.6	62.4	**48.8**	56.1
10	3	129204.8	128825.0	120004.7	**110815.9**
10	7	167662.4	182453.1	172612.9	**167288.6**
10	10	116056.4	**97921.6**	104894.9	103742.2
15	3	279524.1	290851.1	294175.4	**160068.1**
15	7	135841.1	132454.4	146117.8	**61937.9**
15	10	102530.2	94324.1	102600.2	**52910.7**
20	3	47750.3	36449.4	35882.1	**17565.5**
Total mean		122328.1	120417.6	122042.1	**84298.1**
Mean ("CPU time" >= 3600)		141411.4250	138519.7500	144693.8750	**73120.5500**
Mean ("CPU time" < 3600)		103244.8	102315.5	99390.3	**95475.7**

Fig. 5 depicts the mean relative deviation of the makespan obtained by the models. Figs. 6-7 depict the number of simplex iterations and the number of explored nodes in the branch-and-bound tree, respectively.

Table 5
Comparison of the mean relative deviation of the makespan

Size		Mean Relative Deviation of the makespan (%)				
n	*m*	TNZBS1	TNZBS2	RBZBS1	RBZBS2	IG
5	3	**0%**	**0%**	**0%**	**0%**	0.0000%
10	3	**0%**	**0%**	**0%**	**0%**	1.2490%
10	7	**0%**	**0%**	**0%**	**0%**	1.2244%
10	10	**0%**	**0%**	**0%**	**0%**	0.5489%
15	3	0.9370%	**0.8897%**	1.3542%	1.2041%	1.6165%
15	7	**0.5831%**	1.1842%	1.4246%	1.8271%	0.8074%
15	10	0.8086%	0.6929%	1.3649%	1.2882%	**0.5926%**
20	3	1.3060%	2.2139%	1.4629%	2.2029%	**0.6191%**
Total mean		**0.4543%**	0.6226%	0.7008%	0.8153%	0.8323%
Mean (“CPU time” >= 3600)		**0.9087%**	1.2452%	1.4016%	1.6306%	0.9089%

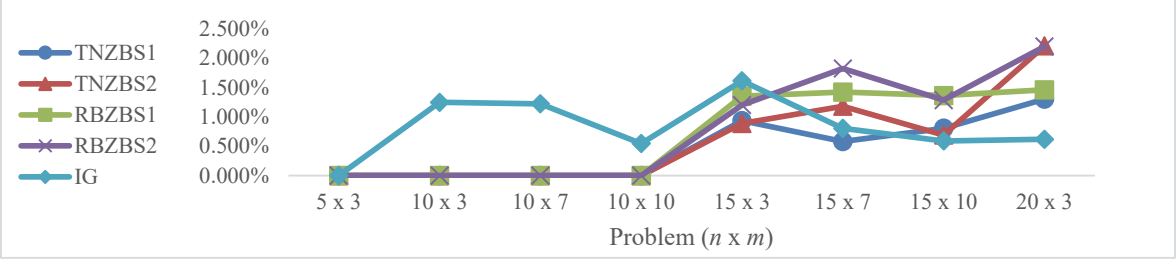

Fig. 5. Graphical representation of the mean relative deviation (MRD) of the makespan obtained by the models

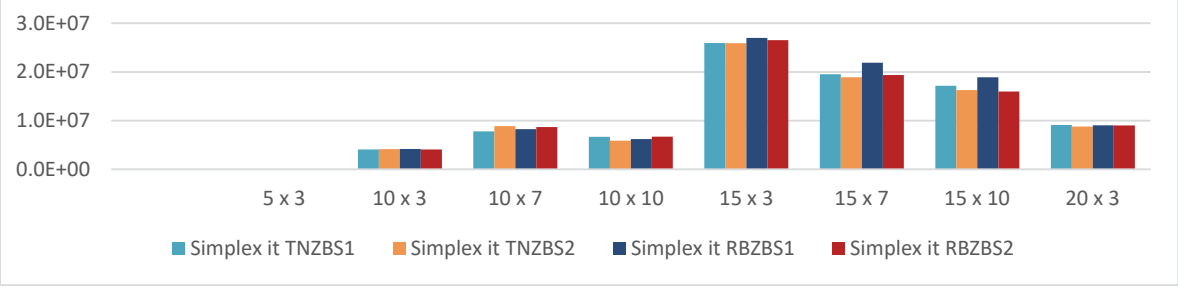

Fig. 6. Graphical representation of the number of simplex iterations

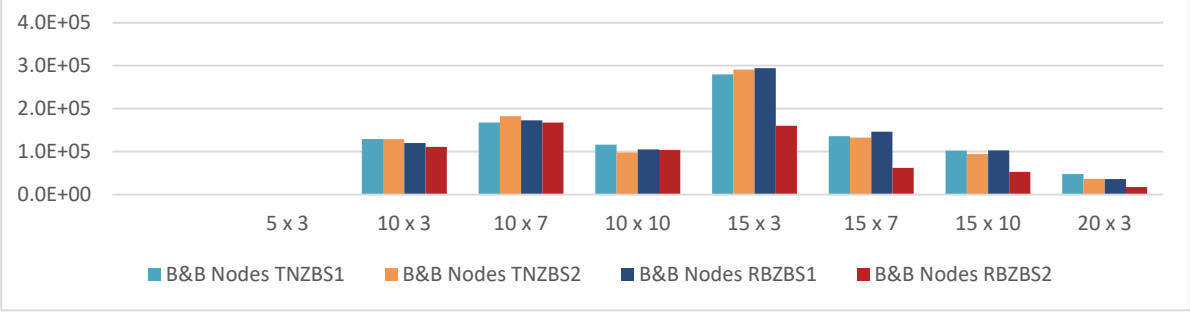

Fig. 7. Graphical representation of the number of explored nodes in the branch-and-bound tree

Table 6 shows the results of the makespan for problems with 15 jobs and 3 machines. In this class of problems, the smallest mean relative deviation of the problems was obtained by the TNZBS2 model, however in most of the problems the other models obtained better results. In all other classes of problems, the mean relative deviation of the makespan represents the model that obtained the best results in most of the problems. All results of computational time; mean relative deviation of the makespan; number of simplex iterations; and number of explored nodes in the branch-and-bound tree are shown in the online supplementary material. Table 2 shows that the problems with 15 or more jobs were not solved by the MILP models within the stipulated computational time limit. The number of binary variables of all the MILP models is equal and a variation occurs only in the number of continuous variables and the number of constraints. Table 2 also shows a certain similarity in the results of the MILP models, where the RBZBS1 model obtained the results a little faster than the others for small problems.

Table 6
Comparison of the makespan of the *MILP* models for 15×3 and 15×10 classes of problems

Size		Problem	Makespan				
n	*m*	#	TNZBS1	TNZBS2	RBZBS1	RBZBS2	IG
15	3	1	1528	1524	1576	**1523**	1530
		2	1576	1594	**1542**	1601	1587
		3	**1531**	1535	1552	1543	**1528**
		4	1602	**1591**	1622	1630	1608
		5	**1568**	1587	1587	1608	1604
		6	1458	**1429**	1432	1461	1458
		7	1543	1553	1549	**1531**	1609
		8	1629	1630	1661	**1622**	1644
		9	**1586**	1591	**1586**	1589	**1579**
		10	1419	1403	1403	**1379**	1392
15	10	1	2264	2250	2320	**2234**	2257
		2	2285	**2225**	2247	2274	2251
		3	2259	2253	2245	2277	**2206**
		4	2234	**2180**	2212	2230	2196
		5	2259	2275	2286	2308	**2216**
		6	**2168**	2188	2207	2180	2189
		7	**2212**	2230	2266	2279	2217
		8	2297	2309	2298	2311	**2259**
		9	**2168**	2216	2184	2181	2207
		10	2242	2236	2246	2221	**2200**

However, in the bigger problems, in which the computational time exceeded 3600 seconds, it can be analysed from Table 5 that the TNZBS1 model was able to achieve slightly better results than the other MILP models. It can be observed by the mean relative deviation of the makespan of classes that the computational time to solve the problems was higher than 3600 seconds (last line in Table 5). Table 6 shows that in the class of problems with 15 jobs and 3 machines, the MILP model that obtained the best makespan most of the time was model RBZBS2; and in problems with 15 jobs and 10 machines, the MILP model that obtained the best makespan in most cases was model TNZBS1. However, in both cases, model TNZBS2 obtained the best mean relative deviation of the makespan among the MILP models. This is due to the fact that the makespan obtained by the TNZBS2 model was never too distant from those obtained by the best model in each problem. All MILP models have the same number of binary variables, but model RBZBS2 has the smallest number of continuous variables (with an average of 80% smaller than model TNZBS2, which is the model with the greatest number of continuous variables, for the classes of problems proposed). Model RBZBS1 has the smallest number of constraints (with an average of 17% smaller than TNZBS2, which is the model with the greatest number of continuous variables, for the classes of problems proposed). However, the number of variables and the number of constraints did not seem to influence the computational time to solve the models, since the model with the greatest number of constraints and variables (TNZBS2) obtained results faster than the model with the smallest number of variables and the second smallest number of constraints (RBZBS2). From Tables 2, 3, and 4, it can be observed that the number of simplex iterations and the number of explored nodes in the branch-and-bound tree to solve the problems do not seem to influence the computational time to solve the model as the model with the greatest number of simplex iterations and the second highest number of explored nodes in the branch-and-bound tree (RBZBS1) is the model that obtained the results in the smallest computational time. In Table 2, it can be observed that the IG algorithm obtained considerably shorter computational times than any of the MILP models. In Table 5 and Fig. 5, it is shown that the IG algorithm had a slightly larger MRD than the best MILP model in some of the problem classes, however for larger problems, the IG algorithm became much smaller MRD.

6. Conclusions

Despite being one of the factors that influences the difficulty of solving a problem, the number of variables and constraints alone do not determine how fast a model can solve a problem. This can be observed by the results obtained. Model RBZBS2, for example, has the smallest number of variables and the second smallest number of constraints (observed in Table 1) and still obtained the largest computational times to solve the problems. Analysing Fig. 6 and Fig. 7, it can be noted that the number of simplex iterations and the number of explored nodes in the branch-and-bound tree did not influence the speed to solve a problem. It can be observed that the model with the smallest number of explored nodes in the branch-and-bound tree and the third smallest number of simplex iterations was the model that obtained the overall greater computational times to solve the problems. Considering the results, it can be observed that the adapted RBZBS1 model obtained the smallest computational time for small problems, however for the bigger problems, the proposed TNZBS1 model obtained the best results (observed by the mean relative deviation of the makespan, in Table 5 and Fig. 5) within the stipulated computation time limit.

The mean number of simplex iterations is smaller for the TNZBS1 than for the RBZBS1 for all the problems. Moreover, the mean number of explored nodes in the branch-and-bound tree is smaller for the TNZBS1 than for the RBZBS1 for the problems where the computational time exceeded 3600 seconds. All these combined might suggest that the TNZBS1 model can converge to better results more quickly for bigger problems. Even though the IG algorithm had a larger MRD in some classes of problem, the computational time needed to solve a problem was much smaller. We consider that the "CPU Time" compensates the larger MRD of the makespan, especially considering that for bigger problems the mean relative deviation of the makespan of the IG algorithm is smaller than that of the other methods. This indicates that for bigger problems, the IG algorithm tends to obtain better results than the MILP model in a short amount of time. As the performance of the TNZBS1 seems to improve as the problem becomes bigger, it is suggested to run the models in a considerably larger database (e.g. Taillard, 1993 database). The MILP models can also be compared to a branch-and-bound algorithm to analyse which is better for larger problems. The ρ parameter from the termination criterion of the IG algorithm can be calibrated using the Taillard (1993) database. Another parameter of the IG algorithm that can be calibrated is the method to obtain the initial solution, some other constructive heuristics (e.g. MM, PF, PW, wPF) can be tested and the overall performance of the algorithm can be compared. Additionally, the IG algorithm can be tested with other metaheuristics (e.g. Genetic Algorithm, Simulated Annealing, Cluster Search) in order to analyse its efficiency.

Acknowledgments

The authors acknowledge the partial research support from the *Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) – Brazil* (306075/2017-2 and 430137/2018-4).

References

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

Maleki-Darounkolaei, A., Modiri, M., Tavakkoli-Moghaddam, R., & Seyyedi, I. (2012). A three-stage assembly flow shop scheduling problem with blocking and sequence-dependent set up times. *Journal of Industrial Engineering International*, 8-26.

Miyata, H. H., & Nagano, M. S. (2019). The blocking flow shop scheduling problem: A comprehensive and conceptual review. *Expert Systems with Applications*, 137, 130-156.

Norman, B. A. (1999). Scheduling flowshops with finite buffers and sequence-dependent setup times. *Computer & Industrial Engineering, 16*(1), 163-177.

Pan, C. H. (1997). A study of integer programming formulations for scheduling problems. *International Journal of Systems Science, 28*, 33-41.

Pan, Q.-K., & Ruiz, R. (2014). An effective iterated greedy algorithm for the mixed no-idle permutationflowshop scheduling problem. *Omega, 44*, 41-50.

Papadimitriou, C., & Kanellakis, P. (1980). Flow-shop scheduling with limited temporary storage. *Journal of the Association for Computing Machinery, 27*(3), 533-549.

Rad, S. F., Ruiz, R., & Boroojerdiana, N. (2009). New high performing heuristics for minimizing makespan in permutation flowshops. *Omega, 37*(2), 331-345.

Ronconi, D. P., & Birgin, E. G. (2012). Mixed-integer programming models for flowshop scheduling problems minimizing the total earliness and tardiness. *Just-in-Time Systems, 61*, 91-105.

Sanches, F. B., Takano, M. I., & Nagano, M. S. (2016). Evaluation of heuristics for a branch and bound algorithm to minimize the makespan in a flowshop with blocking. *Acta Scientiarum, 38*(3), pp. 321-326.

Stafford, E. F. (1988). On the Development of a Mixed-Integer Linear Programming Model for the Flowshop Sequencing Problem. *Journal of the Operational Research Society, 39*, 1163-1174.

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

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research, 64*(2), 278-285.

Takano, M. I., & Nagano, M. S. (2019). Evaluating the performance of constructive heuristics for the blocking flow shop scheduling. *International Journal of Industrial Engineering Computations, 10*, pp. 37-50.

Zhu, Z., & Heady, R. B. (2000). Minimizing the sum of earliness/tardiness in multimachine scheduling: a mixed integer programming approach. *Computers & Industrial Engineering, 38*, 297-305.