Expert Systems with Applications 158 (2020) 113556

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Review

A comprehensive review of Branch-and-Bound algorithms: Guidelines and directions for further research on the flowshop scheduling problem

Caio Paziani Tomazella, Marcelo Seido Nagano *

Department of Production Engineering, São Carlos School of Engineering, University of São Paulo, 400 Trabalhador São Carlense Avenue, São Carlos, São Paulo, Brazil

ARTICLE INFO

Article history:
Received 15 June 2019
Revised 15 January 2020
Accepted 9 May 2020
Available online 29 May 2020

Keywords:
Scheduling
Flowshop
Branch-and-bound
Review

ABSTRACT

This article is a comprehensive review of Branch-and-Bound algorithms for solving flowshop scheduling problems, from the early works of Ignall and Schrage (1965) and Brown and Lomnicki (1966) to the recent approaches of Labidi et al. (2018) and Li et al. (2018). The first part of the article contains an overview of the Branch-and-Bound algorithm, how it is applied for scheduling problems and its different components. The literature review is focused on permutation flowshop problems, and shows the contribution of each article to the method itself and its application. The articles are divided according to the characteristics of the problem and summarized in tables for an easier viewing. The objectives of this review are to provide guidelines for future research in the application of the Branch-and-Bound algorithm for scheduling problems and also to be used as an index for authors to locate the articles for particular problems within the state-of-the-art literature.

1. Introduction

As an exact method, the Branch-and-Bound (B&B) algorithm is used to obtain optimal solutions to mathematical and computational problems which cannot be solved in polynomial time. Initially proposed by Land and Doig (1960) to solve integer programming problems, this algorithm divides the original problem into a series of subproblems using certain criteria (e.g. fixing the value of a variable) and, according to the properties of the subproblem (e.g. feasibility, solution value), a decision is made whether it is further explored or discarded. The methods used to generate the subproblems represent the "branching" steps of the algorithm, while the "bounding" phase consists of the elimination of subproblems.

The implicit enumeration nature of the B&B method means that it is a useful tool for solving combinatorial problems such as the Travelling Salesman Problem (Wolsey, 1998), Supplier Selection Problems with Discounts (Goossens, Maas, Spieksma, & van de Klundert, 2007) and the Permutation Flowshop Problem (PFSP) (Ilgnall & Schrage, 1965). In the PFSP, which is the focus of this article, the B&B uses criteria such as lower bounds and dominance rules in order to prune nodes that correspond to partial sequences. This way, in a PFSP with n jobs, only a fraction of all $n!$ possible solutions are explored.

A number of factors commonly motivate the application of the B&B in PFSPs. The first is the superior efficiency of the combinatorial B&B approach when compared to another traditional exact method, the Mixed Integer-Linear Programming (MILP) formulation (Maccarthy & Liu, 1993). For complex problems involving several flowshop constraints, solving the MILP formulation (which also involves applying the B&B albeit in a different manner) is not viable, and the B&B is the most efficient method for obtaining optimal solutions.

Another application of the B&B arises form the importance of using optimal solutions to validate the performance of heuristic methods. This analysis is done by calculating the relative error for the heuristic solutions when compared to the optimal ones (Allahverdi, 2000; Chung et al., 2011). Lastly, some authors use the B&B algorithm, totally or partially, inside another method such as a heuristic (Haouari & Ladhari, 2003) or a metaheuristic (Nagar, Heragu, & Haddock, 1996).

This article aims at providing an extensive literature review from the application of the B&B to flowshop scheduling problems, identifying the current state-of-the-art methods and which areas still need to be studied. This review contains articles published between 1965 and 2018 that either propose, improve or apply a B&B method to solve PFSPs.

The article is structured as follows: Section 2 gives an overview of the characteristics and the notation used to describe the PFSP;

* Corresponding author.
E-mail addresses: caio.tomazella@usp.br (C.P. Tomazella), drnagano@usp.br (M.S. Nagano).

https://doi.org/10.1016/j.eswa.2020.113556
0957-4174/

Section 3 describes B&B approach to solve combinatorial scheduling; Section 4 presents the methodology used in the review; Section 5 contains an extensive review of the two- and three-machine PFSPs; Section 6 of the m-machine PFSPs; lastly, Section 7 presents the conclusions drawn from the review and final remarks on the subject.

2. The permutation flowshop problem

In a PFSP, a set of n jobs is processed by a set of m machines. In a flowshop machine environment, each job follows the same route between the machines, and, due to the permutational characteristic, the machines process the jobs all in the same order. Examples of industries that operate as a PFSP are foundries (Saadani, Guinet, & Moallaa, 2003) and seamless steel tubes production (Tang & Huang, 2007).

Each operation is done once and each machine is capable of handling a single operation at a time. For the operation from job j in machine i to start, both operations from $j-1$ in i and j in $i-1$ must be finished. Other constraints may delay the start of an operation, such as the need of setup operations and absence of buffer storage (blocking). PFSPs are also non-preemptive, which means that when an operation starts, it cannot be interrupted until it is finished (Pinedo, 2008).

This article uses the notation from Graham, Lawler, Lenstra, and Kan (1979), in which a scheduling problem is presented in the form of $\alpha|\beta|\gamma : \alpha$ being the production environment; β the environments characteristics and constraints; and γ the performance criteria. The values of α, β and γ are extensively described by Pinedo (2008). A flowshop environment with m machines is written as $\alpha = F_m$.

The values of β are shown in Table 1. Release dates (r_j) occur when, due to external factors, a job is only available for processing after a given elapsed time (Tadei, Gupta, Croce, & Cortesi, 1998). The blocking constraint (*block*) appears in situations in which there is no buffer storage between machines, therefore a job must occupy machine m until $m+1$ is available, even if its operation in m is finished (Ronconi & Armentano, 2001). In case there is some buffer storage, but its size is limited so that it affects the PFSP operations, the limited buffer (*Lim.Buffer*) constraint is used instead (Knopf, 1985). Constraints on the amount of time a job spends between machines are also considered: no-wait (*nwt*) occurs when the allowed time is zero, and the parameter w_j is used to set minimum and maximum waiting times (Allahverdi & Aldowaisan, 2002a; Fondrevelle, Oulamara, & Portmann, 2006). Other cases do not allow for a machine to be idle after it starts processing the first job ($no-idle$).

The breakdown characteristic (*brkdwn*) appears in problems in which machines are not available during the entire time, duo to factors such as maintenance or prescheduling of other activities (Kubiak, Błażewicz, Formanowicz, Breit, & Schmidt, 2002). Cases in which the processing time of operation changes over time are indicated as r, in which times are reduced due to increased operational expertise (Lee & Wu, 2004), and λ, in which times are increased due to job deterioration (Wang, Ng, Cheng, & Liu, 2006). Lastly, job precedence constraints (*Prec.*) occur when external factors may demand that some jobs are scheduled before other (McMahon & Lim, 1993).

As previously mentioned, all articles in this review have the permutation characteristic (*prmu*), therefore it is omitted in order to avoid repetition. Unique characteristics that appear on specific articles are described as they appear.

Additionally, another constraint that is usually found in PFSPs is the occurrence of setup times between operations. Although following the notation from Graham et al. (1979) the setup constraints are written in the β field, in this article they are shown separately due to the importance of these times in building an efficient schedule (Allahverdi, 2015). The setup time for an operation can be identified as sequence independent when its duration is constant regardless of the previous job, or dependent when otherwise. Also, family setup operations can appear in problems in which the jobs are divided into different families and the operations must be done only when changing families. Table 2 shows the notation used to identify the setup types.

Table 3 shows the performance criteria of the literature problems. Criteria that are based on job completion times are makespan, which minimizes the completion time of the last job in the schedule, maximizing the efficiency of machine usage, and total flow time (TFT), which minimizes the sum of completion times of all jobs, reducing the work-in-process during the schedule and the average production lead time. Performance criteria related to due-dates often use as a measure tardiness and lateness. Both measure the amount of time elapsed between the completion time of a job and its corresponding due-date, with the difference being that lateness can have negative values when a job is delivered earlier while the tardiness minimum value is set to zero. On the other hand, the earliness criterion measures the amount of time a job is delivered before its due-date. Other performance criteria are the total late work, which minimizes the total operational time that is needed to finish a job after its due-date is past, and the number of tardy jobs, which minimizes the number of jobs finished after their due-date, regardless of the tardiness value. Lastly, two or more criteria can be used to measure performance, which is done by multiplying it by weighting constants. In this case, γ is written as $f(\gamma_1, \gamma_2, \ldots)$.

Table 1
Different values of β.

β	Description
r_j	Release dates
block	Machine blocking
Lim.Buffer	Limited Buffer Storage
nwt	No-wait
w_j	Limited job waiting time
$no-idle$	No-idle machines
brkdwn	Limited machine availability
r	Machine learning
λ	Job deterioration
Prec.	Job Precedence Constraints

Table 2
Different values of β.

Notation	Setup type
SI	Sequence independent
SI_f	Sequence independent, product family setup
SD	Sequence dependent
SD_f	Sequence dependent, product family setup

Table 3
Different values of γ.

γ	Description
C_{max}	Makespan
$\sum C$	Total Flow Time (TFT)
T_{max}	Maximum Tardiness
$\sum T$	Total Tardiness (TT)
L_{max}	Maximum Lateness
$\sum Y$	Total Late Work
$\sum U$	Number of Tardy Jobs
$\sum E$	Total Earliness
$\sum wX$	Weighted sum of a criteria

3. The Branch-and-Bound method

The B&B method was firstly introduced for scheduling problems by IIgnall and Schrage (1965) and Lomnicki (1965), addressing the makespan minimization in a three-machine PFSP and using proper criteria for branching and pruning nodes. Since then, studies improved this method using different components, which will be listed in this section. Consequently, a vast array of constraints were explored: PFSPs with blocking (Ronconi & Armentano, 2001); sequence-dependant setup times (Rios-Mercado & Bard, 1999); due dates and release dates (Grabowski, Skubalska, & Smutnicki, 1983; Tadei et al., 1998). Furthermore, different performance criteria were studied: Total Flow Time (Bansal, 1977); Total Tardiness (Chung, Flynn, & Kirca, 2006) and the weighted sum of different objectives (Nagar, Heragu, & Haddock, 1995).

A B&B for scheduling consists of an implicit enumeration of the solutions by creating partial sequences job per job and creating a tree that branches into the complete solutions. Each node of the tree corresponds to a partial solution, and has an intricate lower bound value which tells if the node can be pruned, narrowing the searching region in the tree and shortening the computational time.

Through the literature review, six main components of the B&B method were identified:

- **Lower Bound (LB)**: the smallest value of the performance criterion the partial sequence can achieve, also considering the non-sequenced jobs. The LB formula is calculated by relaxing the problem's formulation, and, since a PFSP can be written in different forms, the lower bound is the component that has most variations in the literature. The most common LB is calculated by relaxing the machines capacity constraints, considering all but one or two being infinite, turning the PFSP into a single or two-machine problem that can be easily solved. This LB is usually called machine-based and was introduced by IIgnall and Schrage (1965) and Lomnicki (1965). Other types of LBs can be developed by applying the linear (Croce, Gupta, & Tadei, 2000) or the Lagrangian relaxations (van de Velde, 1991) on the MILP formulation, or by relaxing β constraints of the problem (An, Kim, & Choi, 2016). Based on the B&B theory (Wolsey, 1998), the higher the LB value is the better, since it approaches the value of the optimum.
- **Upper Bound (UB)**: in a minimization problem, which is the case of PFSPs, the UB represents the known complete sequence which has the minimum performance criterion value (Land & Doig, 1960). The use of an UB allows the B&B to prune active nodes that have a bigger LB values, reducing the size of the search region and shortening the computer processing time. Usually, authors apply a heuristic method to find an initial UB before starting the B&B itself in order to improve its efficiency.
- **Search Strategy**: it defines how the algorithm chooses the next node of the tree to be branched among all active nodes. The first strategy, used by IIgnall and Schrage (1965) and Lomnicki (1965), is the *Best Bound*, which searches the entire tree for the active node with the smallest LB value, breaking ties by picking the one with the most jobs in the partial sequence. The second strategy, which was introduced by Potts (1980) and has been the most used since the 90s, is the *Depth First* rule. This rule searching space is limited to the deepest level of the tree with active nodes, that is, the nodes with the highest number of jobs in the partial sequence, and picks the one with the smallest LB value. By using this strategy, the algorithm limits the size of the active nodes tree to less than $\frac{n^2}{2}$ (the *Best Bound* strategy does not limit the size of the tree) and allows for a faster exploration when compared to the *Best Bound*.
- **Branching Strategy**: it defines how new nodes are created from another active node. In most cases, when the node has r jobs in its partial sequence, it is branched into $n - r$ nodes by adding each of the non-fixed jobs immediately after the partial sequence.
- **Dominance Rules**: these rules allow active nodes to be pruned by criteria other than their LBs. The first proposed rule compares two nodes with the same set of jobs in two different partial sequences. In case the sequences meet a property, one node can be pruned since it is mathematically guaranteed that it will not lead to a better solution than the other. This rule was introduced by IIgnall and Schrage (1965) and also used by Rios-Mercado and Bard (1999) and Chung, Flynn, and Kirca (2002). However, these rules are not often used since they require heavy processing times. Two other classes of rules are more often used, since they can be computed on a preprocessing stage and both consist of establishing precedence relations between jobs in an optimal solution. Authors such as van de Velde (1991) and Tadei et al. (1998) use rules that prune nodes when a job j generally precedes a job l in case their processing times suffice a property. On the other hand, Croce, Narayan, and Tadei (1996) and Chung et al. (2006) apply a more specific rule, analysing the relations between j and l when both follow the same partial sequence, and determining if the pair lj dominates jl or if jl dominates lj.
- **Stopping Criteria**: since the B&B is an implicit enumeration scheme, it tends to explore all $n!$ possible solutions except those pruned by the LB and the dominance rules. In order for the algorithm not to spend an indefinite amount of processing time, every application includes a stopping criterion which interrupts the B&B execution and the best UB is kept as a solution. In almost the entirety of articles reviewed, the authors used a time limit of 30 to 60 min of CPU time.

3.1. Numerical example of the B&B application

In order to illustrate how the B&B algorithm is applied to solve a PFSP, a numerical example of makespan minimization with four jobs and three machines is solved. This problem instance was shown by IIgnall and Schrage (1965) and was chosen since it effectively shows how the search strategy and dominance rule concepts are applied. The processing times of this instance are shown in Table 4. This example uses the LB formulation and the dominance rule from IIgnall and Schrage (1965) and the *Best Bound* search strategy, breaking ties by depth and numerical order. Examples of how the LB is calculated and of how the dominance rule is applied are shown after the step-by-step procedure. Fig. 1 shows the final tree created by the B&B algorithm to solve this problem, and highlights the path to the optimal solution. Each node contains its number, partial sequence and LB value.

- **Step 1**: The tree is initialized by creating *Node 0*.
- **Step 2**: *Node 0* is branched into four nodes, one for each available job. After calculating the LBs, *Nodes 1* and *2* have the smallest value (55).
- **Step 3**: *Node 1* is branched into three nodes with two jobs each in the partial sequence.

Table 4
Processing times of the example instance from IIgnall and Schrage (1965).

Job	1	2	3	4
Machine 1	13	7	26	2
Machine 2	3	12	9	6
Machine 3	12	16	7	1

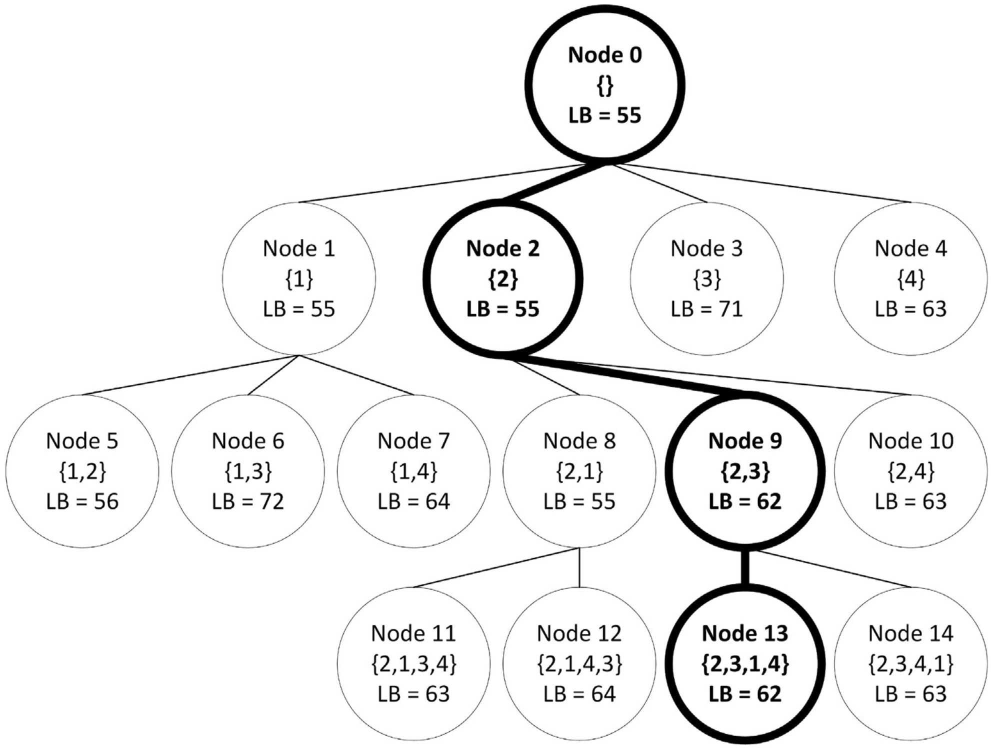

Fig. 1. Branch-and-Bound tree of the sample problem from IIgnall and Schrage (1965).

- **Step 4**: *Node 2* is branched into three nodes. If the *Depth First* strategy was used, *Node 5* would be chosen instead of *Node 2*.
- **Step 5**: *Nodes 5* and *8* share the same jobs in the partial sequence. By applying the dominance rule, it is possible to prune *Node 5*, preventing its unnecessary exploration which would happen in case this rule was not used.
- **Step 6**: The next node to be branched is *Node 8*. Since only two jobs remain to be sequenced, both created nodes are complete solutions for the problem. Therefore, instead of an LB for the makespan, these nodes contain a solution value that can serve as a UB for the algorithm. *Nodes 11* and *12* have makespans of 63 and 64, respectively. 63 is set as the current UB.
- **Step 7**: *Nodes 3*, *6* and *7* are pruned since their LB values are higher than 63. In case only one optimal solution is needed, *Nodes 4* and *10* can also be pruned since their LBs are equal to 63.
- **Step 8**: *Node 9* is branched, creating two solutions. *Nodes 13* and *14* have a makespan of 62 and 63, respectively. Since there are no active nodes left on the tree, the algorithm is finished and sequence $\{2,3,1,4\}$ is an optimal solution with value 62.

An example of the calculation of the LB is given using *Node 1* of the B&B tree, containing partial sequence $\{1\}$.

The notation used for writing the formulation is the following: p_{kj} is the processing time of the operation from job j in machine k; $q_k(\sigma)$ is the completion time of the last job in partial sequence σ in machine k; $J'(\sigma)$ is the set of jobs that are not sequenced in σ.

Eq. (1) shows the formulation for the LB proposed by IIgnall and Schrage (1965), which is obtained by finding the highest solution among the m single-machine problems obtained from the machine capacity relaxation of the PFSP (as described in Section 3). The formulation (for any machine k) is a sum of: **(1)** the completion time of the partial sequence in machine k; **(2)** the sum of the processing times of the non-sequenced jobs in k; **(3)** the smallest time required to process an non-sequenced job from $k+1$ to m, which corresponds to the minimum time taken from the remaining machines to finish the last job.

$$LB = \max_{1 \leqslant k \leqslant m} \left(q_k(\sigma) + \sum_{j \in J'(\sigma)} p_{kj} + \min_{j \in J'(\sigma)} \left(\sum_{i=k+1}^{m} p_{ij} \right) \right) \tag{1}$$

The detailed calculations for the numerical example are shown below:

- $k = 1$
  - $q_1(\sigma) = 13$
  - $\sum_{j \in J'(\sigma)} p_{1j} = 7 + 26 + 2 = 35$
  - $\min_{j \in J'(\sigma)} \left(\sum_{i=2}^{3} p_{ij} \right) = \max(12 + 16, 9 + 7, 6 + 1) = 7$
  - Total: 55
- $k = 2$
  - $q_2(\sigma) = 16$
  - $\sum_{j \in J'(\sigma)} p_{2j} = 12 + 9 + 6 = 27$
  - $\min_{j \in J'(\sigma)} \left(\sum_{i=3}^{3} p_{ij} \right) = \max(16, 7, 1) = 1$
  - Total: 44
- $k = 3$
  - $q_3(\sigma) = 28$
  - $\sum_{j \in J'(\sigma)} p_{3j} = 16 + 7 + 1 = 24$
  - Total: 52

As a result, the LB associated with *Node 1* is $LB = \max(55, 44, 52) = 55$.

The dominance rule from IIgnall and Schrage (1965) states that, when two partial sequences, σ and σ' contain the same set of jobs in a different order and, for every machine k, $q_k(\sigma) \leqslant q_k(\sigma')$, partial sequence σ dominates σ', and the node containing σ' is pruned from the B&B tree. In Step 5 of the numerical example, *Nodes 5* and *8* are compared: the completion times in each machine for partial sequence $\{1,2\}$ (*Node 5*) are $(20, 32, 48)$ and for partial sequence $\{2,1\}$ (*Node 8*), $(20, 23, 47)$. Therefore, the dominance occurs and *Node 5* is pruned.

4. Review Methodology

The following criteria were set in order to the define the scope of this review and select which articles would be included. First, regarding the addressed PFSP, the production environment must be a permutation flowshop with or without a specific number of machines. Problems with flexible/hybrid and distributed flowshops, flowshops that function as an assembly line, process multiple jobs in form of batches and that, due to certain characteristics, are not permutation were not included in the review. Also, only problems with time-related performance criteria were considered (eg. makespan, TFT, TT).

Regarding the application of the B&B, the article either applies a novel B&B to optimally solve a PFSP, improves an existing method, studies the performance of different variations of B&B algorithm when solving a PFSP, or uses the B&B along with another algorithm, such as a heuristic or metaheuristic.

The articles were searched by using the keywords *flowshop* and *branch-and-bound*, within the scientific databases Web of Science and Scopus. Only articles written in English and published in indexed journals were included. Conference articles were considered only when the subject was not addressed by the authors in journal articles.

This review divides the article in two main groups: PFSPs with two and three machines ($\alpha = F_2, F_3$); and PFSPs with a generalized number of m machines. Inside each group, the articles are ordered by their performance criteria (γ); setup type; characteristics and constraints (β); and then in chronological order.

5. Two- and three-machine PFSPs

This section contains the literature on B&B applied to two- and three-machine PFSPs. Table 5 summarizes the reviewed articles in the same order as they appear in the text. The main characteristic of this group is a diversified list of studied problems, which is explained by the fact that, while formulating real industrial environments, authors usually apply premises and simplifications which lead to $\alpha = F_2$. This can be observed by the number of unique restrictions in Table 5. It is also important to remind that $F_2\|C_{max}$ is solved by Johnson's algorithm Johnson (1954), and its variation for the $no-idle$ case also can have its optimal solution found through the same rule (Saadani et al., 2003).

5.1. Makespan criterion

IIgnall and Schrage (1965) introduced the use of B&B for PFSPs by proposing a machine-based LB for $F_3\|C_{max}$ by relaxing the capacity of two machines and considering one bottleneck at a time, solving the problem as a single machine with job tails (the sum of the job's processing times on all the machines that follow the bottleneck). The authors also showed the first dominance rule for this problem, comparing sequences with the same jobs in different orderings. In the same year Lomnicki (1965), used graphs to formulate the problem and arrive at an LB formulation similar to the one from IIgnall and Schrage (1965). McMahon and Burton (1967) improved the method by combining a job-based LB to the formulation, which relaxed the operations precedence constraints inside each job. Special cases of dominant jobs and machines (with larger operation processing times than others) were shown in which this LB had a better performance.

Tsujimura, Park, Chang, and Gen (1993) and Temiz et al. (2004) both applied the B&B algorithm from IIgnall and Schrage (1965) for a three-machine PFSP with fuzzy processing times. Tsujimura et al. (1993) used the method to solve a numerical example, while Temiz et al. (2004) applied it to an industrial case of steel tubes processing.

Su and Chen (2011) addressed the $F_3\|C_{max}$ problem in which some jobs have no operations on the last machine. Two LBs for the B&B algorithm were developed by relaxing the capacity of one machine and applying Johnson's algorithm to the remaining pair.

Tadei et al. (1998) relaxed the machine capacity constraints and the condition that the jobs are non-preemptive in order to calculate five LBs and prove five dominance rules for $F_2|r_j|C_{max}$. The most dominant LB from these uses the first machine as a bottleneck and allows job preemption. In the same problem, Cheng, Steiner, and Stephenson (2002) used machine capacity relaxation to develop new LBs, and also presented a new dominance rule. The results of their B&B were not compared to Tadei et al. (1998) in order to show any improvement in the method. Cheng, Steiner, and Stephenson (2001) presented three lower bounds and three dominance rules for $F_3|r_j|C_{max}$. The lower bounds used different machine capacity relaxations and the experimentation showed that each formulation was suitable for respective problem sizes and release date distributions.

Lee, Shiau, Chen, and Wu (2010) conducted an extensive study on the $F_2|block, \lambda|C_{max}$ PFSP, proposing a series of dominance rules, three heuristics and two machine-based LBs. This was the only study containing the blocking restriction for two-machine PFSPs, and the B&B was applied in order to measure the heuristics efficiency.

Chihaoui, Kacem, Hadj-Alouane, Dridi, and Rezg (2011) calculated five different machine-based LBs for $F_2|nwt, r_j, brkdwn|C_{max}$, and the better results were found when relaxing the capacity of the first machine. Regarding the same problem, Labidi, Kooli, Ladhari, Gharbi, and Suryahatmaja (2018) presented several LBs based on different relaxations, along with a dominance rule. A heavy experimentation showed that the relaxation that the PFSP into a TSP-based problem was the tighter, and the use of the *Best Bound* search strategy was more efficient for the algorithm than *Depth First*.

Yang and Chern (1995) studied the complexity and optimality for problems in which the jobs have limited buffer times, $F_2|w_{max}|C_{max}$, proposing a machine-based LB and two dominance properties. The authors also shortly extended the studies for $m \geqslant 3$, showing where the presented concepts can be applied to formulate a proper LB for this case. Joo et al. (2009) improved the algorithm from Yang and Chern (1995) with seven additional dominance rules and new LBs. The LBs consisted of an enhancement of the formulation from Yang and Chern (1995) and adaptations of LBs from other authors: two based on optimal algorithms for simpler problems, and one obtained by solving the linear relaxation of the MILP.

Bouquard et al. (2006) calculated LBs for $F_2|w_{min}, w_{max}|C_{max}$ by relaxing the buffer constraints and by solving the linear relaxation of the problem's MILP. The tests on empty nodes showed that the LB based on Johnson's algorithm returned the best values for higher w_{max}, and the tests using the entire B&B algorithm showed that its application is more efficient than directly solving the MILP.

Kim and Lee (2017) addressed the three-machine PFSP with specific and overlapping waiting times between the first and second and the first and third machines, $F_3|w_{min}|C_{max}$, presenting a MILP formulation and a B&B algorithm. From the five proposed LBs, the most dominant ordered the non-sequenced jobs using Johnson's algorithm on machines two and three and relaxed the constraints of overlapping waiting times. The authors also showed that the use of the dominance rules from Joo et al. (2009) improved the algorithm's performance. However, no results were presented regarding the MILP implementation.

Moukrim, Rebaine, and Serairi (2014) presented machine-based LBs and two dominance rules for the two-machine PFSP with unitary processing times and minimum waiting times between

Table 5
Application of B&B in two- and three-machine PFSPs.

Criteria	Setup	Constraints	m	Reference	Lower bound relaxation	Additional contribution
C_{max}	–	–	3	Ilgnall and Schrage (1965)	Machine Capacity	–
C_{max}	–	–	3	Lomnicki (1965)	Machine Capacity	–
C_{max}	–	–	3	McMahon and Burton (1967)	Machine Capacity, Operation Precedence	–
C_{max}	–	–	3	Tsujimura et al. (1993)	–	Fuzzy processing times
C_{max}	–	–	3	Temiz et al. (2004)	–	Fuzzy processing times
C_{max}	–	–	3	Su and Chen (2011)	Machine Capacity	Optional operation on 3rd machine
C_{max}	–	r_j	2	Tadei et al. (1998)	Machine Capacity, Non-Preemption	–
C_{max}	–	r_j	2	Cheng et al. (2002)	Machine Capacity	–
C_{max}	–	r_j	3	Cheng et al. (2001)	Machine Capacity	–
C_{max}	–	$block, \lambda$	2	Lee et al. (2010)	Machine Capacity	–
C_{max}	–	$nwt, r_j, brkdwn$	2	Chihaoui et al. (2011)	Machine Capacity	–
C_{max}	–	$nwt, r_j, brkdwn$	2	Labidi et al. (2018)	Machine Capacity, Non-Preemption, Linear	Efficiency of search strategies
C_{max}	–	w_{max}	2	Yang and Chern (1995)	Machine Capacity	Extension for $m \geqslant 3$
C_{max}	–	w_{max}	2	Joo et al. (2009)	Machine Capacity, Linear	–
C_{max}	–	w_{min}, w_{max}	2	Bouquard et al. (2006)	Constraint, Linear	–
C_{max}	–	w_{min}	3	Kim and Lee (2017)	Machine Capacity, Constraint	–
C_{max}	–	$w_{min}, p_{ij} = 1$	2	Moukrim et al. (2014)	Machine Capacity	–
C_{max}	–	$no - idle$	3	Narain and Bagga (2003)	Machine Capacity	–
C_{max}	–	$brkdwn$	2	Kubiak et al. (2002)	Machine Capacity, Constraint	–
C_{max}	–	$brkdwn$	2	Liao and Tsai (2009)	Machine Capacity	–
C_{max}	–	$brkdwn$	2	Hnaien et al. (2015)	Machine Capacity	–
C_{max}	–	r	2	Cheng et al. (2013)	Machine Capacity	–
C_{max}	–	r	2	Wu et al. (2015)	Machine Capacity	–
C_{max}	–	r	3	Wang et al. (2016)	Machine Capacity	–
C_{max}	–	λ	2	Wagneur and Sriskandarajah (1993)	Machine Capacity	–
C_{max}	–	λ	3	Wang et al. (2010)	Machine Capacity	–
C_{max}	–	λ	3	Wang and Wang (2013a)	Machine Capacity	–
C_{max}	–	λ	3	Jafari et al. (2016)	–	Observation on Wang and Wang (2013a)
C_{max}	–	λ	3	Jafari et al. (2017)	–	Observation on Wang et al. (2010)
C_{max}	–	λ, r	2	Wang et al. (2012)	Machine Capacity, Constraint	–
C_{max}	–	λ, r	2	Li et al. (2018)	Machine Capacity, Constraint	–
C_{max}	–	$p_{ij} = 1$	2	Süral et al. (1992)	Constraint	Limited renewable processing resources
C_{max}	–	$p_{ij} = 1$	2	Tellache and Boudhar (2017)	Constraint, Linear	Job processing conflicts
C_{max}	–	$Prec.$	2	McMahon and Lim (1993)	Operation Precedence, Constraint	–
C_{max}	–	$Prec.$	2	Gim et al. (1994)	Constraint	–
C_{max}	–	$Prec.$	2	Lim and McMahon (1994)	Operation Precedence, Constraint	–
C_{max}	–	$Lim.Buffer$	2	Knopf (1985)	Machine Capacity, Constraint	–
C_{max}	–	$Lim.Buffer$	2	Agnetis et al. (1998)	Constraint, Processing Times	Batch scheduling
C_{max}	–	$Lim.Buffer$	2	Lin et al. (2009)	Machine Capacity, Constraint	Different precedence occurrences
C_{max}	–	$Lim.Buffer$	2	Kononov et al. (2012)	Machine Capacity, Constraint	–
C_{max}	SI_f	–	2	Yang (2002)	Setup Times	$B\&B$ use within a heuristic
C_{max}	SD	$w_{max}, s^1_{ij} = 0$	2	An et al. (2016)	Machine Capacity, Constraint, Linear	–
$\sum C$	–	–	2	Ilgnall and Schrage (1965)	Machine Capacity	–
$\sum C$	–	–	2	van de Velde (1991)	Lagrangian	–
$\sum C$	–	–	2	Croce et al. (1996)	Machine Capacity	–
$\sum C$	–	–	2	Pan and Wu (1996)	Machine Capacity	–
$\sum C$	–	–	2	Croce et al. (2002)	Lagrangian	–
$\sum C$	–	–	2	Akkan and Karabati (2004)	Linear, Lagrangian	–
$\sum C$	–	–	2	T'kindt et al. (2004)	–	Efficiency of search strategies
$\sum C$	–	–	2	Lin and Wu (2005)	Machine Capacity	–
$\sum C$	–	–	2	Hoogeveen et al. (2006)	Lagrangian	–
$\sum C$	–	–	2	Haouari and Kharbeche (2013)	Assignment Problem	Extension for non-linear functions of C
$\sum C$	–	r_j	2	Rakrouki et al. (2017)	Machine Capacity, Non-Preemption, Lagrangian	–
$\sum C$	–	w_{min}	2	Msakni et al. (2016)	Machine Capacity, Assignment Problem	–
$\sum C$	–	$no - idle$	2	Narain and Bagga (2005)	Machine Capacity	–
$\sum C$	–	r	2	Lee and Wu (2004)	Machine Capacity	–
$\sum C$	–	r	2	Wang and Liu (2009)	Machine Capacity	–
$\sum C$	–	r	2	Li et al. (2011)	Machine Capacity	–
$\sum C$	–	r	2	Wu et al. (2012)	Machine Capacity	–
$\sum C$	–	r	2	Lai et al. (2014)	Machine Capacity	–
$\sum xC$	–	$r, xT_j < M$	2	Shiau et al. (2015)	Machine Capacity	–
$\sum C$	–	λ	2	Wang et al. (2006)	Machine Capacity	–
$\sum C$	–	λ	2	Wu and Lee (2006)	Machine Capacity	–
$\sum C$	–	λ	2	Ng et al. (2010)	Machine Capacity	–
$\sum C$	–	λ, r	2	Wang and Liu (2009)	Machine Capacity	–
$\sum C$	–	$\lambda, min(C_{max})$	2	Cheng et al. (2014)	Machine Capacity, Linear	–
$\sum C$	–	$min(C_{max})$	2	Rajendran (1992)	–	Use of existing algorithms

Table 5 (*continued*)

Criteria	Setup	Constraints	m	Reference	Lower bound relaxation	Additional contribution
$\sum C$	–	$min(C_{max})$	2	T'kindt et al. (2003)	Linear, Lagrangian	–
$\sum C$	–	$min(C_2)$	3	Yanai and Fujie (2006)	Constraint Relaxation	Efficiency of branching strategies
$\sum C$	SI	–	2	Allahverdi (2000)	Machine Capacity	–
$\sum C$	SI	–	2	Allahverdi et al. (2002b)	Machine Capacity	Setup and removal times
$\sum C$	SI	–	2	Wang and Cheng (2005)	Machine Capacity	Special cases
$\sum C$	SI	–	2	Gharbi et al. (2010)	Machine Capacity, Non-Preemption	–
$\sum C$	SI	–	2	Gharbi et al. (2013)	Machine Capacity, Non-Preemption, Linear, Lagrangian	–
$\sum C$	$-/SI$	–	2	Detienne et al. (2016)	Lagrangian	–
$\sum C$	SI	–	3	Allahverdi and Al-Anzi (2006)	Machine Capacity	–
$\sum C$	SI	nwt	2	Aldowaisan (2001)	Machine Capacity	–
$\sum C$	SI	nwt	2	Su and Lee (2008)	Machine Capacity	Single server setups
$\sum wC$	–	r	2	Wang and Wang (2013b)	Machine Capacity	–
$\sum wC$	–	λ	2	Yang and Wang (2011)	Machine Capacity	–
L_{max}	–	–	2	Su and Chen (2011)	Machine Capacity	Optional operation on 2nd machine
L_{max}	–	r_j	2	Haouari and Ladhari (2000)	Machine Capacity, Release and Due Dates	–
L_{max}	–	r_j	2	Cheng et al. (2002)	Machine Capacity	–
L_{max}	SI	–	2	Dileepan and Sen (1991)	Machine Capacity	–
L_{max}	SI	nwt	2	Fondrevelle et al. (2005)	Machine Capacity	Setup and removal times
T_{max}	–	r	2	Wu et al. (2007)	Machine Capacity, Due Dates	–
$\sum T$	–	–	2	Sen et al. (1989)	Machine Capacity	–
$\sum T$	–	–	2	Kim (1993)	Machine Capacity	–
$\sum T$	–	–	2	Pan and Fan (1997)	Machine Capacity	–
$\sum T$	–	–	2	Pan et al. (2002)	Machine Capacity	–
$\sum T$	–	–	2	Schaller (2005)	Machine Capacity	–
$\sum T$	–	–	2	Haouari and Kharbeche (2013)	Assignment Problem	–
$\sum T$	–	$brkdwn$	2	Lee and Kim (2017)	Machine Capacity, Due Dates	–
$\sum T$	–	λ	2	Bank et al. (2012)	Machine Capacity	–
$\sum T$	–	$Prec.$	2	Cheng et al. (2017)	Machine Capacity, Due Dates	–
$\sum xT$	–	$\sum xU = 0$	2	Lee et al. (2010)	Machine Capacity	–
$\sum U$	–	$d_j = d$	2	Croce et al. (2000)	Machine Capacity, Linear Relaxation	–
$\sum U$	–	r_j	2	Ardakan et al. (2014)	Machine Capacity	–
$\sum wU$	–	–	2	Bulfin and M'Hallah (2003)	Linear, Lagrangian	–
$\sum Y$	–	–	2	Lin et al. (2006)	Machine Capacity	–
$\sum I$	–	$\sum U = 0$	2	Ahmadi and Bagchi (1992)	Constraint	–
$f(C_{max}, \sum C)$	–	–	2	Nagar et al. (1995)	Machine Capacity	–
$f(C_{max}, \sum C)$	–	–	2	Nagar et al. (1996)	–	Combined application with a GA
$f(C_{max}, \sum C)$	–	–	2	Sivrikaya-Şerifoğlu and Ulusoy (1998)	Machine Capacity	Different branching strategies
$f(C_{max}, \sum C)$	–	–	2	Sayin and Karabati (1999)	–	Solves the criteria separately
$f(C_{max}, \sum C)$	–	–	2	Yeh (1999)	Machine Capacity	–
$f(C_{max}, \sum C)$	–	–	2	Yeh (2001)	Machine Capacity	–
$f(C_{max}, \sum C)$	–	–	2	Lin and Wu (2006)	Machine Capacity	–
$f(C_{max}, \sum C)$	–	–	3	Yeh and Allahverdi (2004)	Machine Capacity	–
$f(C_{max}, \sum C)$	–	nwt	2	Allahverdi and Aldowaisan (2002a)	Machine Capacity	–
$f(C_{max}, \sum C)$	–	λ	2	Cheng et al. (2015)	Machine Capacity	–
$f(C_{max}, L_{max})$	–	nwt	2	Allahverdi and Aldowaisan (2004)	Machine Capacity	–
$f(C_{max}, \sum C, L_{max})$	–	–	2	Allahverdi (2001)	Machine Capacity	–
$f(C_{max}, T_{max})$	–	–	2	Daniels and Chambers (1990)	Machine Capacity	–
$f(C_{max}, T_{max})$	–	–	2	Lin et al. (2013)	Machine Capacity	–
$f(C_{max}, T_{max})$	–	r	2	Chen et al. (2006)	Machine Capacity	–
$f(C_{max}, \sum T)$	–	–	2	Lin et al. (2013)	Machine Capacity	–
$f(C_{max}, \sum T)$	–	r	2	Liao et al. (1997)	Machine Capacity	–
$f(C_{max}, \sum U)$	–	r	2	Liao et al. (1997)	Machine Capacity	–
$f(C_{max}, E_{max})$	–	–	2	Toktaş et al. (2004)	Due Dates	–
$f(\sum C, \sum T)$	–	–	2	Lee and Wu (2001)	Operation Precedence	–
$T_{max} + E_{max}$	–	–	2	Moslehi et al. (2009)	Machine Capacity, Due Dates	–
$\sum T + \sum E$	–	d_{min}, d_{max}	2	Yeung et al. (2004)	Machine Capacity	–

machines. The experimentation evaluated the trade-off between the complexity and tightness of the presented LBs. Narain and Bagga (2003) worked with the no-idle constraint in the $F_3|no-idle|C_{max}$ problem, proposing an LB using an estimate of job waiting times when changing machines.

Kubiak et al. (2002) studied the $F_2|brkdwn|C_{max}$ PFSP, defining properties of an optimal solution in order to apply pruning rules during the B&B execution. Three LBs were calculated for the case in which the machine stoppage times are previously known: two by the machine capacity relaxation and one by relaxing the breakdown constraint. Liao and Tsai (2009) presented several LB calculations for this same problem, considering the breakdowns occurred due to preventive maintenance in one or both of the machines. Hnaien, Yalaoui, and Mhadhbi (2015) addressed the two-machine problem with scheduled breakdowns on the first machine. The authors showed that the B&B algorithm performed better for instances with the breakdowns occurring near the start or the end of the production horizon.

Cheng, Wu, Chen, Wu, and Cheng (2013) proved dominance rules and an LB for the two-machine PFSP with truncated learning effects. In a PFSP with learning effects, as the machine operates, the processing time of an operation is reduced at an exponential or linear rate. When truncation is applied, a minimum value is set to be used as the multiplier when the calculated rate is smaller than this minimum. The truncation is used when the machine has a learning capacity and to avoid unrealistic scenarios of low (or even negative) processing times. Wu et al. (2015) did a similar study for this problem, considering a more complex truncated learning formulation. For the three-machine variation with linear decreasing deterioration, Wang, Wei, and Lu (2016) proved two rules and two machine-based LBs: one that is an adaptation of the LB from IIgnall and Schrage (1965) that considers the learning effects and the other is based on Johnson's algorithm.

Wagneur and Sriskandarajah (1993) proposed a simple machine-based LB for the job deterioration problem, $F_2|\lambda|C_{max}$, in which the processing times in the second machine increase according to the waiting time of the job. Wang, Sun, Sun, and Wang (2010) addressed the three-machine problem with linear deterioration, presenting one dominance rule and an LB formulation. Wang and Wang (2013a) published a similar study on the problem with deterioration inflicted through a different formulation. In both cases the presented LBs were adaptations of the LB from IIgnall and Schrage (1965). Jafari, Khademi-Zare, Lotfi, and Tavakkoli-Moghaddam (2016, 2017) showed a counterexample and a correction to the dominance rules from Wang and Wang (2013a) and Wang et al. (2010), respectively.

Wang, Ji, Cheng, and Wang (2012) studied the problem with two machines and both deterioration and learning effects simultaneously. Four dominance rules were presented for three different cases and different LBs were obtained by relaxing the machine capacity and other constraints of the problem. Two heuristics for the problem were also developed in the article and the optimal solutions obtained from the B&B were used to evaluate their performance. Li, Jiang, and Ruiz (2018) addressed the two-machine problems with learning effects on both machines and also forgetting effects on the second caused by machine idle time. Several LB calculations were proposed and the results were used to analyse the performance of a series of algorithms.

Süral, Kondakci, and Erkip (1992) modelled the problem with three machines and unitary job processing times, in which each operation requires a number of resources which are replenished over time. Three LBs were calculated for a B&B algorithm, each one relaxing one of the problem's constraints. Tellache and Boudhar (2017) presented 3 MILP formulations and a B&B algorithm for the two-machine PFSP with unitary processing times ($p_{ij} = 1$) and conflict between jobs, which means that certain pairs of jobs cannot be processed simultaneously. Seven LBs were proposed in this article: two based on the graph representation of the conflicts; two based on relaxations of constraints; and three on the linear relaxations of each the MILP formulation. The experimentation showed that the most efficient LB was the one calculated by the conflict constraints relaxation.

McMahon and Lim (1993) addressed the problem with two machines and arbitrary precedence constraints between jobs. A job-based LB was presented, followed by an enhancement using the precedence constraint relaxation and graph formulation. The same problem was addressed by Gim, Curry, and Deuermeyer (1994), which used an LB by using Johnson's algorithm while ignoring the precedence constraints. However, this LB was not compared to the one from McMahon and Lim (1993). Lim and McMahon (1994) extended the LB formulation and the B&B algorithm from McMahon and Lim (1993) to the three-machine problem.

Knopf (1985) modelled a chemical facility as a two-machine PFSP with limited storage between the stages. A simple B&B algorithm was used, but the calculated LB was not explicitly specified. Agnetis, Rossi, and Gristina (1998) addressed the generalized two-machine PFSP with limited buffer capacity, calculating three different LBs for the C_{max} criterion and also a couple of dominance rules.

Lin, Hong, and Lin (2009) proved the *NP*-Hardness of multimedia scheduling problems, in which the buffer between the two machines is limited based on the size of the jobs and certain cases of precedence occur. The LBs were developed after constraint relaxations and the authors suggested a model for implementing the algorithm. Kononov, Hong, Kononova, and Lin (2012) improved the previous algorithm by presenting tighter LBs, and also showed a detailed analysis regarding the optimality of different cases of the problem.

Yang (2002) studied the properties of the two-machine flowshop problem with sequence-independent setup times. In this problem the jobs were divided into different groups for each machine. The authors proposed two heuristics which separated the jobs according to their groups, and applied a B&B to find optimal sequences within these groups.

An et al. (2016), did an extensive study for the problem with maximum waiting times and sequence dependent setup times on the second machine only, $F_2|w_{max}, s^2_{jl}|C_{max}$. They proposed a series of dominance rules and five LBs: three machine-based, one obtained by relaxing both constraints and one by the solving the linear relaxation of the MILP formulation. The B&B experimentation showed that the machine-based LBs were the tightest, and that the use of an initial UB using a high quality heuristic resulted in a significant gain of efficiency, while the use of dominance rules did not, due to their high required computational processing times.

5.2. Total Flow Time Criterion

In the same article in which they addressed the makespan minimization, IIgnall and Schrage (1965) also calculated an LB for the TFT in a two-machine flowshop ($F_2\|\sum C$) using similar concepts of machine capacity relaxation. van de Velde (1991) defined dominance criteria and presented a new LB based on the Lagrangian relaxation of the MILP formulation. The experimentation showed that the proposed LB outperformed the one from IIgnall and Schrage (1965) and that the use of the dominance rules was only viable for small-sized problems.

Croce et al. (1996) revised several LBs for the $F_2\|\sum C$ PFSPs: both formulations from IIgnall and Schrage (1965); an adaptation of the LB for $F_m\|\sum C$ from Ahmadi and Bagchi (1990); the LB from van de Velde (1991); and the one from Hoogeveen et al. (1995), also based on the Lagrangian relaxation of the problem, even though it was not tested on a B&B method. In sequence, the author presented two dominance rules, with one being an extension of a rule from van de Velde (1991); and two LBs, each one considering a machine as a bottleneck, which improved the efficiency of the B&B algorithm when compared to the other formulations.

Pan and Wu (1996) used the Shortest Processing Time (SPT) rule, which minimizes the TFT in a single-machine problem, in order to formulate an LB for the two-machine case. However, the efficiency of this formulation was not compared to the existing ones. As a sequence of the article from Croce et al. (1996) and Croce, Ghirardi, and Tadei (2002) improved the Lagrangian relaxation LBs and dominance rules, solving problems with higher number of jobs than in their previous article. Akkan and Karabati (2004) proposed two new LBs using relaxations on the network flow formulation for the problem: a linear relaxation LB to be applied on the root node and a Lagrangian relaxation LB for the other nodes. The experimentation showed that the LBs based on this formulation improved the performance of the algorithm.

T'kindt, Croce, and Esswein (2004) used the LB from Croce et al. (2002) in order to analyse the performance of the *Best Bound* and *Depht First* search strategies. The authors found that, for this problem, the *Best Bound* strategy was more efficient when no dominance rules were applied, while the *Depht First* was more efficient when using the dominance rules to prune nodes. Lin and Wu (2005) proposed an LB based on the SPT rules and the truncation of processing times in the second machine. While it was shown that this LB was not as tight as the ones from IIgnall and Schrage (1965), it performed better in a B&B algorithm with a larger number of jobs and distribution of processing times than the LB from Croce et al. (2002), due to its simpler computational complexity.

Hoogeveen, van Norden, and van de Velde (2006) presented a MILP formulation for the $F_2||\sum C$ problem focusing on the completion times of the positions rather than jobs. The given LB was obtained through the Lagrangian relaxation of this formulation, and, even though is was tighter than the existing ones from the literature, its higher computational complexity hindered its performance within a B&B algorithm. Haouari and Kharbeche (2013) modelled this PFSP as an assignment problem and used it to calculate an LB which could also be applied to non-linear functions of the completion times. With the experimentation, the authors showed that this new LB is tighter than the one from Hoogeveen et al. (2006) when calculated on root nodes.

Rakrouki, Kooli, Chalghoumi, and Ladhari (2017) formulated several LBs and two dominance rules for the TFT in a two-machine PFSP with release dates. By using the machine capacity, non-preemption, processing times and Lagrangian relaxations, the authors obtained LBs with different computational complexity levels, and showed that those obtained through the Lagrangian relaxation of the MILP formulations dominated the others. Msakni, Khallouli, Al-Salem, and Ladhari (2016) addressed the problem with minimum time delays between the two machines, showing four LBs based on the machine capacity relaxation and the formulation of the PFSP as an assignment problem. Furthermore, a dominance rule was proven, along with the suggestion of efficient methods for implementing it. As a follow-up for their previous study regarding the makespan criterion, Narain and Bagga (2005) proved the *NP*-completeness and calculated a machine-based LB for the $F_2|no-idle|\sum C$.

Lee and Wu (2004) defined several dominance rules and adapted the LBs from IIgnall and Schrage (1965) for the $F_2|r|\sum C$ problem with an exponential learning curve. Wang and Liu (2009) also presented dominance rules and LB for the $F_2|r|\sum C$, though considering a different processing time reduction function, defined as linear decreasing deterioration. Li, Hsu, Wu, and Cheng (2011) formulated the problem with truncated machine learning, in which the effect was due to the position of the job in the schedule, and from that they proved three dominance rules and machine-based LB formulations using the SPT rule.

Still regarding the same problem with truncated machine learning effects, albeit considering the starting time of the job instead of its position, Wu, Wu, Hsu, and Lai (2012) applied the B&B with a dominance rule and LBs suitable to the learning function. Lai, Hsu, Ting, and Wu (2014) also presented a dominance rule and four LB calculations for the same problem, using the optimal results of the B&B to validate the performance of three simulated annealing algorithms. Shiau, Tsai, Lee, and Cheng (2015) addressed the $F_2|r|\sum C$ problem with two different groups of jobs, in which the TFT of a group must be minimized while the maximum tardiness of the other is kept below a bounded value. The learning function used was the same from Li et al. (2011), and four dominance rules and an LB were presented.

Wang et al. (2006) studied the problem with machine linear deterioration, $F_2|\lambda|\sum C$, and proposed a series of dominance rules and formulated two LBs for the problem, based on the concepts from IIgnall and Schrage (1965) and Lee and Wu (2004). Wu and Lee (2006) proved several dominance properties and a simple machine-based LB using the SPT rule for the same problem. Ng, Wang, Cheng, and Liu (2010) improved the efficiency of the algorithm from Wang et al. (2006) by enhancing its LB and also showing three additional dominance rules.

Wang and Liu (2009) addressed the problem with simultaneous effects of linear deterioration and exponential learning. The authors showed dominance rules and an LB set using similar approaches used by them in previous articles. Cheng, Tadikamalla, Shang, and Zhang (2014) formulated two LBs for the TFT in a two-machine PFSP with linear job deterioration subject to the minimum makespan, one using the machine capacity relaxation and the other obtained through the linear relaxation of the MILP formulation of the problem. As for the makespan, the authors used an algorithm that can find its optimal value in polynomial time for the type of deterioration considered.

Rajendran (1992) addressed the $F_2|min(C_{max})|\sum C$ PFSP, minimizing the total flow time while keeping the minimum makespan for the sequence. The B&B algorithm used the LB from IIgnall and Schrage (1965) and pruned nodes with a higher makespan than the one obtained by applying Johnson's algorithm. For the same PFSP, T'kindt, Gupta, and Billaut (2003) proved dominance rules and calculated two new LBs, using both linear and Lagrangian relaxations of the problems MILP formulation. Their B&B was shown to be more efficient than the one from Rajendran (1992) and two other exact methods from the literature, a dynamic programming algorithm and a MILP formulation direct solution. Yanai and Fujie (2006) the B&B for the problem of TFT minimization in a three-machine PFSP with minimum makespan on the second machine. The B&B consisted of four dominance rules and an LB obtained by relaxing the minimum makespan constraint. Furthermore, branching strategies were tested by applying different sets of dominance rules while creating the nodes.

Regarding the problem with sequence independent setup times, $F_2|s_j^i|\sum C$, Allahverdi (2000) calculated two LBs by merging the setup and operation times and applying the SPT rule, and also proved two dominance rules. Allahverdi et al. (2002b) proposed an LB, dominance rules and showed optimally solvable cases for the $F_2|s_j^i|\sum C$ problem with setup and removal times. Even though the authors did not implement the B&B algorithm, its use was suggested using the calculations proposed in the article. Wang and Cheng (2005) studied five special cases of this problem, with different relations between operation and setup times. Two of these cases were solvable in polynomial time, while the other three were *NP*-Hard, and the authors presented a B&B algorithm with a proper LB for each.

Gharbi, Ladhari, Msakni, and Serairi (2010) extended the studies regarding this problem by formulating five new LBs: three of them were machine-based, applying an LB on the waiting time of the jobs when changing machines; the other two were adaptations of existing single-machine LBs with the non-preemption constraints relaxed. The tests on root nodes showed that the more complex of the machine-based LBs were tighter than the others. Gharbi, Ladhari, Msakni, and Serairi (2013) improved their previous LBs, and proposed tighter formulations for all plus additional ones calculated by the linear and Lagrangian relaxations of the problems MILP. It was observed through the experimentation that the B&B algorithm had better results using the Lagrangian relaxation based LBs.

Detienne, Sadykov, and Tanaka (2016) developed an LB from the Lagrangian relaxation of a network flow formulation based on the one from Akkan and Karabati (2004). The formulation allowed for the LB to be calculated for the root node than updated in linear time for each other node. The authors adapted two existing dominance rules for this problem and proved three new ones. The B&B algorithm could be applied to cases with or without setup times, and in both it dominated the existing methods from the literature.

For the $F_3|s_j^i|\sum C$ PFSP, Allahverdi and Al-Anzi (2006) conducted a complete experimentation of the B&B Algorithm, developing a dominance rule and a machine-based LB. The results showed that the use of dominance rules and a heuristic to find an initial UB significantly reduce the number of nodes explored in the search tree.

Aldowaisan (2001) developed two dominance rules and an LB using the SPT rule for $F_2|nwt, s_j^i|\sum C$. Furthermore, Su and Lee (2008) conducted a study for the previous PFSP variation with a Single Server constraint, meaning that only one setup operation can be conducted at a time. The proposed B&B method dominated the methods from Aldowaisan (2001) adapted to this specific problem.

Wang and Wang (2013b) addressed the $F_2|r|\sum wC$ problem with linear decreasing deterioration. Several dominance rules were proven and the LBs were formulated using machine capacity relaxations and the SPT ordering. Finally, Yang and Wang (2011) showed dominance rules and a machine-based LB for the $F_2|\lambda|\sum wC$ problem with linear job deterioration.

5.3. Due-date related criteria

Su and Chen (2011) also addressed the $F_2||L_{max}$ problem in which some jobs are not processed on the second machine, proposing three LB calculations by mathematically transforming the problem into makespan minimization with job tails and using machine capacity relaxation and ordering rules.

Haouari and Ladhari (2000) developed three LBs for $F_2|r_j|L_{max}$, one by using machine capacity relaxation and two by fixing the release and due date values. The authors tested the LBs on root nodes and showed that the machine-based formulation dominated the others in most but not all cases. Cheng et al. (2002) applied the same concepts from their C_{max} algorithm to present a B&B algorithm for the L_{max} variation. Dileepan and Sen (1991) addressed the $F_2|s_j^i|\sum L_{max}$ problem, presenting a dominance rule and a lower bound. The experimentation showed that the B&B is more efficient for instances with tighter due dates.

Fondrevelle, Allahverdi, and Oulamara (2005) approached the two-machine, no-wait PFSP with sequence-independent setup and removal times, with the maximum lateness to be minimized, $F_2|nwt, s_j^i, sr_j^i|L_{max}$. The authors implemented a B&B algorithm with a proper LB, showing that it performs better for problems in which the setup and removal times are relatively short when compared to the processing times.

Wu, Lee, and Wang (2007) proved a series of dominance rules for various cases of job pairs in the $F_2|r|T_{max}$ problem with exponential learning effects. The authors formulated the LB by relaxing the machine capacity constraints and also the processing times and due dates by taking the minimum values of the unscheduled job sets.

Sen, Dileepan, and Gupia (1989) used the B&B algorithm to solve the $F_2||\sum T$ problem. The presented method included one dominance rule and an LB based on the capacity relaxation of the second machine. Kim (1993) applied dominance rules and a tighter LB to the same problem, which made the B&B algorithm more efficient compared to the previous one. Pan and Fan (1997) presented dominance rules and an LB for the $F_2||\sum T$ problem, which was an improvement on the formulation from Kim (1993).

Pan, Chen, and Chao (2002) contributed further to the problem by presenting additional dominance rules and tighter LB formulation. While the previous articles constructed the partial sequence back to front, this was based on the conventional branching that added a job in the first available position. Schaller (2005) improved the dominance rules and the LB from Pan et al. (2002) and also proved a new one. The experimentation also used a tightened LB, and the authors showed that the new formulation and rules significantly improved the B&B performance regarding computational time. Lastly, Haouari and Kharbeche (2013) also used the assignment modelling to calculate an LB for the $\sum T$ criterion. The tests on root nodes showed that it provides tighter values than the LB from Pan et al. (2002), albeit requiring more computational time.

Lee and Kim (2017) addressed the TT minimization in a two-machine PFSP with preventive maintenance in the first one, triggered by a set amount of operational time. The proposed B&B algorithm used an LB calculated using the machine capacity and due date relaxations and nine dominance rules. Bank, Ghomi, Jolai, and Behnamian (2012) formulated a machine-based LB and five dominance rules for the $F_2|\lambda|\sum T$ problem with job deterioration. Cheng et al. (2017) used the B&B to minimize the TT in a problem with job precedence constraints, using both machine capacity and due date relaxations, as well as 8 dominance rules.

Lee, kang Chen, and Wu (2010) presented a B&B algorithm for the problem in which the jobs are divided into two groups, with the objective of minimizing the TT for one group while keeping the jobs in the other group from being tardy. Several dominance rules and a machine-based LB were proposed.

Croce et al. (2000) presented a MILP formulation for the $F_2|d_j = d|\sum U$ problem, and formulated two machine-based LBs, stating that the LB obtained through the linear relaxation of the MILP is tighter than those. The authors also showed through experimentation that using the B&B algorithm with this LB is more efficient than solving the MILP formulation directly. Ardakan, Hakimian, and Rezvan (2014) used the algorithm from Moore (1968), which minimizes the number of tardy jobs in a single machine problem, to calculate an LB for $F_2|r_j|\sum U$. Along with the LB, a series of dominance rules were presented as well.

Bulfin and M'Hallah (2003) used the linear and Lagrangian relaxation of a MILP formulation in order to calculate LBs for the $F_2||\sum wU$ PFSP. Lin, Lin, and Lee (2006) addressed the $F_2||\sum Y$ problem, proving that the case with common due dates ($d_j = d$) is *NP*-Hard, calculating an LB considering infinite capacity on the second machine and proving a dominance criterion. Ahmadi and Bagchi (1992) presented three LB schemes and several dominance rules the problem that minimizes the total job idle time ($\sum I$) in a two-machine PFSP with the condition of no jobs being late ($\sum U = 0$). Two variations of the problem were addressed, with and without common due dates.

5.4. Multiple criteria

The B&B application for the PFSP with two weighted performance criteria, in this case $F_2||f(C_{max}, \sum C)$, was firstly studied by Nagar et al. (1995). The authors proposed an LB that considers the capacity of the first machine to be infinite, and showed that it performs better for instances in which the second machine is dominant ($p_{2j} \geqslant p_{1j}$). Nagar et al. (1996) combined their B&B with a genetic algorithm, showing that these techniques resulted in better sequences than those obtained by the algorithm alone. In the proposed method, the metaheuristic used information provided by the B&B regarding the pruned nodes in order to narrow its searching region.

Sivrikaya-Şerifoğlu and Ulusoy (1998) addressed the same problem by testing three different branching strategies: forwards, backwards or alternating both. The authors showed that the forward and alternate strategies work depending on the weight values. The LBs for each case were based on Johnson's algorithm and the LB formulation for TFT from IIgnall and Schrage (1965).

Sayin and Karabati (1999) presented a B&B algorithm which creates a separate tree for each of the performance criteria and uses a coordination mechanism to share information between both trees. The authors implemented three existing LBs from the literature for the TFT criterion and proved two dominance rules for the multiple criteria problem.

Yeh (1999) improved the B&B from Nagar et al. (1995) by applying an LB based on the waiting times on the second machine. Through the experimentation, it was shown that this method had a better performance on harder problems, that is, instances with non-trivial solutions which required more CPU time to be solved. In sequence, Yeh (2001) proposed new dominance rules and further enhancements to the LB formulation from Yeh (1999). Also, the authors noticed a gain in performance in instances with a higher weight for the TFT in the objective function.

Lin and Wu (2006) combined the LB from Lin and Wu (2005) and Johnson's algorithm to calculate a new LB the $F_2||f(C_{max}, \sum C)$ problem. The experimentation showed that the use of this lower bound improved the efficiency of the B&B when compared to the existing ones. Finally, Yeh and Allahverdi (2004) extended the LB formulation from Yeh (2001) to the three-machine case, also adapted the dominance rules from the same article.

Still regarding the $f(C_{max}, \sum C)$ criterion, Allahverdi and Aldowaisan (2002a) proposed LB and a dominance rule for the problem with two machines and the no-wait condition. The tests showed that the use of dominance rules slightly increased the B&B's efficiency. Cheng, Tadikamalla, Shang, and Zhang (2015) addressed this two-machine problem but with job deterioration, presenting a MILP formulation and B&B algorithm with four dominance rules and an LB using both the machine capacity relaxation and the SPT rule. The experimentation showed that the B&B outperformed the MILP direct solving in computational processing time, specially for the larger problems.

Similarly to Allahverdi and Aldowaisan (2002a) and Allahverdi and Aldowaisan (2004) studied the $F_2|nwt|f(C_{max}, L_{max})$ PFSP, proving a dominance rule and an LB. Allahverdi (2001) presented a machine-based LB and a dominance rule for the tricriteria problem $F_2||f(C_{max}, \sum C, L_{max})$. Through the experimentation, the author analysed the efficiency of the dominance rule in pruning nodes.

Daniels and Chambers (1990) used Johnson's algorithm and the Earliest Due Date (EDD) rule to calculate separate LBs for the makespan and the maximum tardiness in a two-machine PFSP, and applied it into a B&B algorithm with backwards partial sequence branching. The authors also proved two dominance rules for the problem.

Lin, Hong, and Lin (2013) modelled the sequencing of media objects as two two-machine PFSPs with limited buffer, one minimizing $\gamma = f(C_{max}, T_{max})$ and the other $\gamma = f(C_{max}, \sum T)$. The authors adapted the LB from Pan et al. (2002) for both due-date related criteria and developed a machine-based LB for the makespan including a term representing the buffering times.

Chen, Wu, and Lee (2006) addressed the $F_2|r|f(C_{max}, T_{max})$ problem with exponential learning, proving a series of dominance rules. The authors also derived a machine-based LB for the problem and the experimentation showed that the use of dominance rules had a significant impact on the algorithm's performance.

Liao, Yu, and Joe (1997) presented B&B algorithms for two bicriteria problems, one combining makespan and TT and the other, makespan and number of tardy jobs. The LB for the makespan can be easily calculated by Johnson's algorithm, while for the other criteria the author proposed variations of the EDD rule and Moore's algorithm Moore, 1968 combined with the machine capacity relaxation.

Toktaş, Azizoğlu, and Köksalan (2004) proposed an LB for the $F_2||f(C_{max}, E_{max})$ PFSP by using a due date relaxation and Johnson's algorithm, which was then applied in a B&B algorithm. Lee and Wu (2001) proposed several dominance rules and a lower bound based on both SPT and EDD rules for the $F_2||f(\sum C, \sum T)$ problem. The experimentation tested different parameters for criteria weight and due dates, and the LB was shown to be tighter for the completion times than the tardiness values.

Moslehi, Mirzaee, Vasei, Modarres, and Azaron (2009) studied the $F_2||T_{max} + E_{max}$ problem, proving dominance rules for specific instances and providing an LB based on a series of lemmas. The experimentation showed that the B&B was more efficient in the instances with stricter due dates, so that tardiness occurs more often than earliness. Lastly, for the two-machine PFSP with common due windows, Yeung, Oğuz, and Cheng (2004) proposed LB formulations for the sum of total tardiness and total earliness with machine capacity relaxations and also a series of dominance rules.

5.5. Considerations

The first observation regarding the articles in this section is that most focus on addressing a specific scheduling problem, usually motivated by the application in a real industrial scenario. These articles use the B&B algorithm as the exact method for obtaining optimal solutions to evaluate the performance of other algorithms that are also proposed, such as heuristics and metaheuristics. The B&B algorithm is often chosen since it can be applied to different classes of PFSPs without major changes in its structure, regardless of the complexity of the problem. Therefore, the main components the authors need to work on are the LB formulations and dominance rules.

Regarding the LB formulations, it is possible to observe that, when compared to the m-machine problems presented in the next section (Tables 5 and 6), a wider variety of relaxations are used in order to obtain LBs for the problems. This is highlighted by the number of articles using LBs that are based on linear of Lagrangian relaxation of the problems' MILP formulations, which are seldom used for m-machine PFSPs.

Also, a higher number of dominance rules are proved for the flowshops with two- and three- machines, mainly for problems with waiting time constraints, and learning and deterioration factors. This highlights a preference for authors to address these rules for problems with a limited number of machines rather than in the generalized cases.

Lastly, a relatively small number of articles address problems with setup times, with the majority of them being sequence-independent. Given the importance of this topic due to its practical applications (Allahverdi, 2015), further research in this field is suggested, especially regarding sequence-dependent setup times.

6. m-machine PFSPs

This group contains articles related to the application of B&B in PFSPs with m machines ($\alpha = F_m$), and differs from the previous by having a smaller variety of studied problems. Table 6 lists the problems firstly according to their constraints (β) and then to their performance criterion (γ).

6.1. Makespan criterion

The first article regarding the B&B application to m-machine PFSPs was published by Brown and Lomnicki (1966), who extended the LB formulation from IIgnall and Schrage (1965) to the general case. Nabeshima (1967) used a similar relaxation, but leaving a pair of subsequent machines to calculate an LB by applying Johnson's algorithm on this pair. The author showed that this LB dominated the one from Brown and Lomnicki (1966) by solving

Table 6
Application of B&B in m-machine PFSPs.

Criteria	Setup	Constraints	Reference	Lower bound relaxation	Additional contribution
C_{max}	–	–	Brown and Lomnicki (1966)	Machine Capacity	–
C_{max}	–	–	Nabeshima (1967)	Machine Capacity	–
C_{max}	–	–	Panwalkar and Khan (1975)	–	–
C_{max}	–	–	Bestwick and Hastings (1976)	Machine Capacity	–
C_{max}	–	–	Lageweg et al. (1978)	Machine Capacity	–
C_{max}	–	–	Potts (1980)	Machine Capacity	Alternate branching rule
C_{max}	–	–	Carlier and Rebai (1996)	Machine Capacity	Different branching schemes
C_{max}	–	–	Cheng et al. (1997)	–	Fuzzy inference rule, not optimal
C_{max}	–	–	Cheng et al. (1997)	Machine Capacity	Automated flowshop
C_{max}	–	–	Companys (1999)	–	Pendular algorithm with parallel processing
C_{max}	–	–	Balasubramanian and Grossmann (2002)	Uncertainty of processing times	Uncertain processing times
C_{max}	–	–	Haouari and Ladhari (2003)	–	B&B used to find local optima in a heuristic
C_{max}	–	–	Ladhari and Haouari (2005)	Machine Capacity	LB calculation varying according to the partial sequence
C_{max}	–	–	Companys and Mateo (2007)	–	Pendular algorithm with parallel processing
C_{max}	–	–	Soylu et al. (2007)	Machine Capacity, Linear	Synchronous transfers
C_{max}	–	–	Drozdowski et al. (2012)	–	Parallel computing
C_{max}	–	–	Madhushini and Rajendran (2012)	Machine Capacity, Operation Precedence	–
C_{max}	–	–	Chakroun et al. (2013)	–	Parallel computing
C_{max}	–	–	Gharbi and Mahjoubi (2013)	Machine Capacity	–
C_{max}	–	–	Gmys et al. (2016)	–	Parallel computing
C_{max}	–	–	Ritt (2016)	–	Cyclical branching strategy
C_{max}	–	*block*	Suhami and Mah (1981)	–	Probability-based, not optimal, LB
C_{max}	–	*block*	Ronconi and Armentano (2001)	Machine Capacity	–
C_{max}	–	*block*	Ronconi (2005)	Machine Capacity	–
C_{max}	–	*block*	Companys and Mateo (2007)	–	Pendular algorithm with parallel processing
C_{max}	–	*block*	Sanches et al. (2016)	–	Efficiency of initial UBs
C_{max}	–	*block*	Toumi et al. (2017b)	Machine Capacity	–
C_{max}	–	r	Chung et al. (2011)	Machine Capacity	–
C_{max}	–	λ	Cheng and Janiak (2000)	Machine Capacity	–
C_{max}	–	w_{min}, w_{max}	Fondrevelle et al. (2006)	Machine Capacity	–
$\sum wC_{k,max}$	–	w_{min}, w_{max}	Fondrevelle et al. (2008)	Machine Capacity	–
C_{max}	–	$min(\sum C)$	Bagga and Bhambani (2002)	Machine Capacity	Special cases
C_{max}	SI	–	Proust et al. (1991)	Machine Capacity	Setup and removal times
C_{max}	SI_f	–	Schaller (2001)	Machine Capacity	Used within a heuristic
C_{max}	SI_f	–	Schaller (2005)	Machine Capacity	Two-phase *B&B* algorithm
C_{max}	SI	–	Kurihara et al. (2009)	Machine Capacity	–
C_{max}	SD	–	Rios-Mercado and Bard (1999)	Machine Capacity, Assignment Problem	–
C_{max}	SD_f	–	Schaller et al. (2000)	Machine Capacity, Assignment Problem	–
C_{max}	SD	–	Tang and Huang (2007)	Machine Capacity	–
C_{max}	SD	*block*	Takano and Nagano (2017)	Machine Capacity	–
$\sum C$	–	–	Bansal (1977)	Machine Capacity	–
$\sum C$	–	–	Ahmadi and Bagchi (1990)	Machine Capacity, Non-Preemption	–
$\sum C$	–	–	Karabati and Kouvelis (1993)	Lagrangian	–
$\sum C$	–	–	Gowrishankar et al. (2001)	–	Completion times variance
$\sum C$	–	–	Chung et al. (2002)	Machine Capacity	Extension for $\sum wC$
$\sum C$	–	–	Gharbi and Mahjoubi (2013)	Machine Capacity, Non-Preemption	–
$\sum C$	–	*block*	Moslehi and Khorasanian (2013)	Machine Capacity	–
$\sum C$	–	*block*	Toumi et al. (2013a)	Machine Capacity	–
$\sum C$	–	*block*	Toumi et al. (2013b)	Machine Capacity	–
$\sum C$	–	r_j	Bai et al. (2017)	Machine Capacity	k-power completion times
$\sum C$	SI_f	–	Gupta and Schaller (2006)	Machine Capacity	–
$\sum wC$	–	–	Toumi et al. (2013c)	Machine Capacity	–
$\sum wC$	–	–	Ren et al. (2017)	Machine Capacity	2-power weighted completion times
L_{max}	–	r_j	Grabowski et al. (1983)	Machine Capacity	–
L_{max}	–	r_j	Haouari and Ladhari (2007)	Machine Capacity, Non-Preemption	–
L_{max}	–	w_j	Fondrevelle et al. (2009)	Machine Capacity	–
L_{max}	–	r	He (2016)	Machine Capacity	–
T_{max}	–	–	Townsend (1977)	Machine Capacity	–
$\sum T$	–	–	Kim (1995)	Machine Capacity	–
$\sum T$	–	$d_j = d$	Gowrishankar et al. (2001)	–	2-power tardiness
$\sum T$	–	–	Chung et al. (2006)	Machine Capacity	–
$\sum T$	–	*block*	Ronconi and Armentano (2001)	Machine Capacity	–
$\sum T$	–	*block*	Toumi et al. (2017a)	Machine Capacity	Extension for $\sum wT$
$\sum T$	–	r	Lee and Chung (2013)	Machine Capacity	–

Table 6 (*continued*)

Criteria	Setup	Constraints	Reference	Lower bound relaxation	Additional contribution
$\sum T$	–	λ	Lee et al. (2014)	Machine Capacity	–
$\sum U$	–	–	Hariri and Potts (1989)	Machine Capacity	–
$f(C_{max}, \sum C)$	–	r	Chung and Tong (2012)	Machine Capacity	–
$f(C_{max}, \sum C)$	–	r	Wang and Zhang (2015)	Machine Capacity	–
$f(C_{max}, \sum C)$	SD_f	–	Hendizadeh et al. (2007)	Machine Capacity	–
$\sum wC, wT, wE$	–	–	Madhushini et al. (2009)	Machine Capacity, Operation Precedence	Different combinations of criteria
$f(wC_{max}, wT_{max}, wE_{max})$	–	–	Madhushini and Rajendran (2011)	–	Different combinations of criteria

three numerical examples. Panwalkar and Khan (1975) improved the B&B algorithm by proving and applying a new dominance rule.

Bestwick and Hastings (1976) calculated an LB using the machine capacity relaxation and a job property named *overrun*. The proposed LB was shown to be more efficient than the one from Brown and Lomnicki (1966) for problems in which some of the processing times are equal to zero. Lageweg, Lenstra, and Kan (1978) studied machine capacity relaxations which turn the PFSP into single-machine or two-machine flowshop problems with different levels of complexity, then used these relaxations to propose LBs for $F_m||C_{max}$. The most dominant LB considers two bottleneck machines with intermediate waiting times.

Potts (1980) adapted LBs and dominance rules for a B&B in which the branching is done by adding a job in the first and last positions of the partial sequence alternatively. Carlier and Rebai (1996) proposed two B&B algorithms: one was an adaptation of job shop concepts with immediate selection and the other using an alternate branching strategy similar to Potts (1980). The LBs were calculated by relaxing machine capacity constraints, and the authors stated that the first method dominated the one from Lageweg et al. (1978).

Cheng, Kise, and Matsumoto (1997) used fuzzy inference to apply a dominance rule in a B&B using the two-machine relaxation LB from Lageweg et al. (1978). The experimentation showed a gain of efficiency for the algorithm, and allowed it to obtain near-optimal solutions for large instances. In a similar article, Cheng, Kise, and Karuno (1997) proposed a B&B with fuzzy inference and a machine based LB for it in order to solve an automated PFSP, with a vehicle moving through all machines in constant speed cycles, being responsible for transferring the jobs between operations.

Companys (1999) analysed the performance of the B&B with parallel processing, with the algorithm being applied in the original and the reversed problem. The proposed method used existing LB formulations and performed better than the regular B&B algorithm. Balasubramanian and Grossmann (2002) used a modified B&B algorithm to minimize the makespan in a PFSP with uncertain processing times. In each node, the uncertainty is removed as the jobs are scheduled, and the LB value is calculated by doing explicit enumeration with the remaining jobs, considering the mean values as the operation times.

Haouari and Ladhari (2003) used the B&B from Potts (1980) as a local search to find an optimal solution within the neighbourhood of a NEH heuristic Nawaz, Enscore, and Ham (1983) solution. Ladhari and Haouari (2005) used three different LBs presented by Lageweg et al. (1978), and applied them in the B&B according to their complexity. For nodes on the first third of the three (less than $n/3$ jobs on the partial sequence) the method used the LB from Brown and Lomnicki (1966). For the second third, an LB with two bottleneck machines and waiting times, proposed by Lageweg et al. (1978). Lastly, for the last third, the LB was formulated from a two-machine *NP*-Hard problem, needing an exclusive B&B for its calculation. Companys and Mateo (2007) improved the parallel algorithm from Companys (1999), solving larger problems than the previous one.

Soylu, Kirca, and Azizoglu (2007) developed a B&B for a m-machine PFSP in which the jobs are release simultaneously by all machines, synchronous transfers. Two LBs are calculated, a machine-based and a MILP linear relaxation, and the latter was shown to be dominant. Drozdowski, Marciniak, Pawlak, and Płaza (2012) tested the use of B&B in a grid computational system, in which the nodes exploration is done in parallel by multiple processors. The LBs used were calculated by using one- and two- machine relaxations from Lageweg et al. (1978). Madhushini and Rajendran (2012) used both machine capacity and operation precedence relaxations to estimate the completion time of each job, and used the Bottleneck Assignment Problem in order to calculate an LB for the makespan. The tests showed that this LB dominated both from Brown and Lomnicki (1966) and McMahon and Burton (1967).

Gharbi and Mahjoubi (2013) improved the *NP*-Hard LB from Ladhari and Haouari (2005), in which each relaxed subproblem must be solved through a B&B. In this method, the authors use infeasibility properties from a subproblem to prune nodes in another, tightening the value of the LB. The tests in empty nodes showed that it consistently dominated the initial LB from Ladhari and Haouari (2005).

Both Chakroun, Melab, Mezmaz, and Tuyttens (2013) and Gmys, Mezmaz, Melab, and Tuyttens (2016) studied the integrated use of CPU and GPU for parallel computing of the B&B operations. In the two studies the authors used the LB formulations from Lageweg et al. (1978) and showed a significant reduction in the total processing time of the algorithm when using the proposed techniques.

In the last article that studied the $F_m||C_{max}$ PFSP, Ritt (2016) proposed new branching and node search strategies: the former created partial sequences by adding a job in both first and last positions and keeping it in the one with the smallest LB value; the latter was called cyclic best-first search, branching each node with the smallest LB in every level of the tree. The algorithm successfully solved more problem instances than the algorithms from Ladhari and Haouari (2005) and Companys and Mateo (2007). Furthermore, it was also stated in this article that the use of both dominance rules and *NP*-Hard LB from Ladhari and Haouari (2005) (instead of the two-machine polynomial formulation) have no significant effects in improving the efficiency of the B&B for this problem.

Suhami and Mah (1981) calculated an LB for $F_m|block|C_{max}$ through the averages and standard deviations of the partial sequences makespan. However, due to its probabilistic characteristic, the algorithm resulted in solutions in a 20% range from the optimal values. Ronconi and Armentano (2001) used properties from the problem to propose a machine-based LB, which calculated estimates for the idle time in each bottleneck machine caused by the blocking constraint. In sequence, Ronconi (2005) improved this LB, consistently solving larger problems with the B&B algorithm.

Companys and Mateo (2007) extended the parallel processing method from Companys (1999) for the $F_m|block|C_{max}$ PFSP, using the LB formulation from Ronconi and Armentano (2001). Sanches, Takano, and Nagano (2016) analysed the use of different heuristics in order to obtain an initial UB for the B&B from Ronconi (2005). Toumi, Jarboui, Eddaly, and Rebaï (2017b) enhanced the formulation of the LB from Ronconi (2005) and showed an improvement in the B&B performance in both computational time and number of explored nodes.

Chung et al. (2011) addressed the PFSP with exponential learning effects, and using concepts from Chung et al. (2002) the authors proved two dominance rules and formulated an LB for this problem. Cheng and Janiak (2000) presented two dominance properties and LB using machine capacity relaxation for a PFSP with convex job deterioration caused by limited availability of resources.

Fondrevelle et al. (2006) presented an LB for the problem with minimum and maximum waiting times between jobs. It was calculated by relaxing the problem into two-machine subproblems with time lags, which are optimally solved using a variation of Johnson's algorithm. Fondrevelle, Oulamara, and Portmann (2008) adapted the LB from Fondrevelle et al. (2006) to a similar problem, with the performance criterion being the minimization of the weighted sum of machine completion times ($\sum_{k=1}^{m} w_k C_{k,n}$). Bagga and Bhambani (2002), using the B&B algorithm, solved three special cases of the makespan minimization PFSP subject to the minimum TFT, using a particular LB calculation for each.

Proust, Gupta, and Deschamps (1991) adapted the LB from Brown and Lomnicki (1966) for the problem in which setup and removal operations are separated from the processing time and sequence independent. Schaller (2001) used the B&B as a starting point to a heuristic method for the PFSP with family sequence independent setup times. The algorithm was used to obtain an optimal sequence for the families, while the heuristic ordered the jobs inside each one. Schaller (2005) improved the previous algorithm by decomposing it in two phases: the first phase is similar to Schaller (2001), optimally sequencing the families as a whole; in the second phase, the B&B is used to find the optimal sequence within each family.

Kurihara, Li, Nishiuchi, and Masuda (2009) first directly adapted the LB from Brown and Lomnicki, 1966 to the problem with sequence independent setup times, considering the setups as regular operations. The authors then changed the formulation to make it suitable for the problem, considering that the setup activities for a job can be done while it is still being processed in the upstream machines.

Rios-Mercado and Bard (1999) addressed the $F_m|s_{jl}^{i}|C_{max}$ PFSP, testing machine capacity relaxations and showing that an LB considering a single machine as a bottleneck is more efficient than using a pair. In order to compute the setup times into the LB, the authors considered these operations as distances in an Asymmetric Travelling Salesman Problem (ATSP), and calculated the optimal value of its relaxation into an assignment problem. Schaller, Gupta, and Vakharia (2000) proposed an LB calculation for the family sequence dependent setup times by using the machine capacity relaxation of the PFSP to both single- and two-machine problems. In order to include the setup times in the LB, the authors used the same method of Rios-Mercado and Bard (1999), applying the assignment problem relaxation into the ATSP matrix of the family setup times.

Tang and Huang (2007) used the B&B in the same problem in order to apply it to a seamless steel tube production environment. The LB is similar to the one proposed by Rios-Mercado and Bard (1999), with a difference in the setup term formulation. Takano and Nagano (2017) calculated 4 LBs for the makepan in a PFSP with blocking and sequence-dependent setup times. The authors used a less complex formulation for the setup times compared to the previous two, and also added terms related to the blocking and job waiting times.

6.2. Total flow time criterion

Bansal (1977) calculated the first LB for $F_m||\sum C$ by adapting the formula from Ilgnall and Schrage (1965) to the general problem and using the SPT rule to sequence the jobs in the bottleneck machine. Ahmadi and Bagchi (1990) proved mathematically that a machine-based LB with the non-preemptive condition also relaxed dominates the LB from Bansal (1977). The proof was validated by tests on root nodes. Karabati and Kouvelis (1993) modelled this PFSP as a designation problem and calculated an LB based on the Lagrangian relaxation of this formulation. This LB was shown to be more efficient than the one from Bansal (1977) when used in the B&B algorithm.

Gowrishankar, Rajendran, and Srinivasan (2001) used the LB formulations from Brown and Lomnicki (1966) and McMahon and Burton (1967) to calculate an LB for the completion times variance in a m-machine PFSP. Chung et al. (2002) proposed dominance rules and an LB for the $F_m||\sum C$ problem. The formulation was shown to be tighter than both LBs from Bansal (1977) and Ahmadi and Bagchi (1990), had a smaller computational complexity than the latter and could also be adapted to the problem with weighted TFT ($\sum wC$).

Gharbi and Mahjoubi (2013) applied the same concepts from the makespan LB to enhance the formulation from Croce and T'kindt (2003), which used the non-preemption relaxation to calculate an LB for the TFT in a single-machine problem, adapted for the m-machine case, also validating the results through experimentation on root nodes. Moslehi and Khorasanian (2013) presented two MILP formulations and a B&B algorithm for $F_m|block|\sum C$, the later containing three LBs and dominance rules. The B&B was shown to be more efficient than an adaptation of the algorithm from Ronconi and Armentano (2001) to $\sum C$ and both MILPs being solved directly.

Toumi, Jarboui, Eddaly, and Rebaï (2013a) and Toumi, Jarboui, Eddaly, and Rebaï (2013b) proposed a mathematical improvement on the $F_m|block|\sum C$ LB formulation from Ronconi and Armentano (2001). This LB was then extended for the weighted criteria ($\sum wC$) by Toumi, Jarboui, Eddaly, and Rebaï (2013c). In both cases, the LB was tested within a B&B algorithm, but its efficiency was not compared to other formulations from the literature.

Bai, Liang, Liu, Tang, and Zhang (2017) developed a MILP formulation and a B&B algorithm for the $F_m|r_j|\sum C^k$ PFSP, in which the job flow times are elevated to a power $k \geqslant 2$. The results showed that the B&B had a better performance than the MILP, which could only solve instances with $k = 2$ due to the high numerical value of the performance criterion. The authors also showed that a particular branching strategy considering the release dates in the node creation ordering reduces the computational time from the B&B.

Gupta and Schaller (2006) extended the LB from Bansal (1977) for the problem with sequence independent family setup times. The authors used an exploration strategy using two types of nodes, representing each family and job, and the branching was done by allocating the families into the partial sequence and then sequencing the jobs inside their families.

Lastly, Ren et al. (2017) formulated an LB for the sum of weighted quadratic completion times of a schedule. The authors used the machine capacity relaxation to turn the PFSP into a single-machine problem and the Weighted Shortest Processing Time Rule (WSPT) from Smith (1956) to solve it and calculate the LB value.

6.3. Due-date related criteria

Grabowski et al. (1983) tested two machine-based LBs for the $F_m|r_j|L_{max}$ PFSP. The one considering a single bottleneck machine outperformed the one considering a pair when applied to a B&B algorithm. Haouari and Ladhari (2007) presented additional LBs for the problem using machine capacity and non-preemption relaxations. The LBs were used in the B&B with the alternated branching strategy, and also proposed a heuristic that uses the first solution found by the B&B with a *Best Bound* search strategy as a result.

Based on the EDD rule, Fondrevelle, Oulamara, Portmann, and Allahverdi (2009) calculated an LB for the maximum lateness on a PFSP with job waiting times. Along with the B&B algorithm, the authors also proved a couple of dominance rules which are more efficient for two-machine problems but can be applied to the generalized case. He (2016) proved two dominance rules and formulated a machine-based LB for the PFSP with exponential learning effect.

Townsend (1977) presented an LB for the maximum tardiness by relaxing the PFSP and applying a SP-like rule, ordering the jobs in non-decreasing order of a modified due date, which is the proper due date minus the tail between the bottleneck job and the m-th machine.

Kim (1995) addressed the $F_m\|\sum T$ PFSP, presenting an LB and a pruning scheme based on job dominance. This B&B, when used to solve two-machine problems, was more efficient than the method from Sen et al. (1989). Similar to the calculations of the TFT variance, used the LBs from Brown and Lomnicki (1966) and McMahon and Burton (1967) to calculate the LB for the total squared tardiness for the PFSP with a common due date. Chung et al. (2006) adapted their LB for $F_m\|\sum C$ to $F_m\|\sum T$ by ordering the non-sequenced job tails according a due date based index. Two dominance rules were also proved, and the results of the B&B application showed that the proposed LB dominated the previous formulation from Kim (1995).

Ronconi and Armentano (2001), in the same article that proposed an LB for the $F_m|block|C_{max}$ PFSP, also used the same concepts to formulate an LB for the TT. The experiments showed that the B&B was able to solve more efficiently problems with looser due dates, and that the LB is tighter for problems with a small number of machines. Toumi, Jarboui, Eddaly, and Rebaï (2017a) presented an enhancement on the LB from Ronconi and Armentano (2001) by adding the concepts from Chung et al. (2006) to the formulation. The authors also extended the LB for the problem with total weighted tardiness.

Lee and Chung (2013) proposed a dominance rule and an LB formulation for the TT in a PFSP with exponential learning effects. With similar calculations, Lee, Yeh, and Chung (2014) also presented a dominance rule and an LB for the problem with linear deterioration. Hariri and Potts (1989) formulated three LBs for the $F_m\|\sum U$ PFSP, and the experiments showed that the most dominant LB used consistent sequences in the single-machine relaxations. In this case, a consistent sequence is a sequence in which jobs are not late when ordered according to a specific index.

6.4. Multiple criteria

Regarding the problem with weighted sum of makespan and TFT and learning effects, Chung and Tong (2012) presented an LB for the case with an exponential learning function, while Wang and Zhang (2015) formulated one for the case in which the learning is a function of both the time the machine has spent on previous operations and the position of the job. Hendizadeh, ElMekkawy, and Wang (2007) addressed the bi-criteria problem with family sequence dependent setup times, using one LB scheme for each calculation. The authors used the formulation from Schaller et al. (2000) for the makespan and developed a novel one for the TFT.

Madhushini, Rajendran, and Deepa (2009) developed a calculation for the LB on the completion time of each job by using both machine capacity and operation precedence relaxations. This values were then used to calculate the LB of the weighted sums of flow time, tardiness and earliness. The B&B algorithm was applied to solve problems with different combinations of these criteria, and it was also shown that the LB for $\sum wT$ dominated the adaptation of the LB from Kim (1995) to the weighted criterion. Madhushini and Rajendran (2011) used the same calculations to find LBs for the maximum weighted completion time, tardiness and earliness of a sequence, applying it to problems with different combinations of these values.

6.5. Considerations

Table 6 is organized similarly to Table 5. The most frequent problem is the makespan minimization in a conventional permutation flowshop, $F_m\|C_{max}$, addressed in 19 different articles. Since this problem is heavily explored, authors often use it to test new features for the B&B method itself, such as parallel processing (Companys & Mateo, 2007) and a new search strategy (Ritt, 2016). For future research, the use of these techniques in other problems with different restrictions and performance criteria is suggested in order to validate them generally.

A considerable number of articles addressed problems with blocking constraints ($\beta = block$), highlighting an importance of developing efficient methods to solve problems with limited storage space that occur in several industrial environments, as stated by Ronconi and Armentano (2001).

Regarding the comparison made in Subsection 5.5, m-machine PFSP class see the majority of the proposed LBs being based on machine capacity relaxations, showing that a more generalized problem makes it more difficult for proposing different formulations. The same happens with dominance rules, with most of them proposed to a small number of problems and constraints, and none proposed for m-machine PFSPs with setup times.

Regarding setup activities, only seven articles proposed methods for PFSPs with this characteristic. As observed in these articles, the main challenge is to formulate an LB which is both tight enough to prune a number of nodes that make the B&B implementation viable while keeping the computational cost low. By proposing a machine-based LB, the setup times turn into a ATSP, which is *NP*-Hard (Rios-Mercado & Bard, 1999), thus the term of the LB formulation which corresponds to the setup times must be an LB itself, weakening the formulations value. It is, therefore, suggested that additional studies are made aiming at improving the LBs in terms of both value and computational complexity, thus making the use of the B&B viable for PFSP with sequence-dependent setup times.

7. Final remarks and guidelines for future research

This article presented an extensive review on the application of the B&B method to scheduling problems in permutation flowshop environments. The aspects considered in the evaluation of each article were how the B&B algorithm was applied and the use of novel techniques related to the addressed problem, especially the LB formulations and dominance rules.

Firstly, there is an overall more focused approach from the authors on proposing a B&B method to different problems rather than improving the existing methods by using new techniques and formulations. This is more evident for two-machine PFSPs, in which a wider range of constraints and performance criteria combinations can be more easily applied to a real industrial case.

It was also noticed during the review a shortage of articles that focus on the method itself, and how it can be more efficiently applied. Examples that stood out were: Potts (1980), who proposed a branching strategy; T'kindt et al. (2004) and Ritt (2016), who studied the performance of different branching strategies; Chakroun et al. (2013) and Gmys et al. (2016), who applied the B&B with parallel processing. Therefore, in order to widen the range of problems that can be solved by the B&B algorithm, both in terms of problem size and complexity, further research is suggested in the following areas: analysis of the most suitable applications of branching and searching strategies; use of parallel processing to explore the problem's tree. Another suggestion is the use of adaptive algorithms that change its strategies according to the instance and its current performance during a solve.

In the past decade, some papers have proposed the use of Constraint Programming (CP) to solve PFSPs. Zeballos, Castro, and Mèndez (2011), Samarghandi and Behroozi (2017) and Samarghandi (2019) propose CP models and compare their efficient to MILP formulations. To this day, no comparison has yet been made between CP modelling and the B&B algorithm, neither both techniques were applied together. However, Hamdi and Loukil, 2015 proposed a decomposition method to calculate an LB for the $F_m|w_{min}|\sum U$ problem which involves the application of a CP model. Regarding this topic, a suggestion is to analyse the usage of CP as an alternative to calculate LBs for PFSPs inside a B&B algorithm.

As for problem restrictions, it was noticed a increasing number of articles addressing problems with blocking flowshops, suggesting an importance of studying this topic. Research on flowshops with other properties was not extensively conducted, such as no-wait and no-idle, which were both explored in two- or three-machine flowshops, but no B&B neither LB were proposed for the generalized m-machine PFSP.

Most importantly, only 15 out of the 100 unique problems in all sections consider setup times, totalling 23 articles in this review. Allahverdi (2015), in his third extensive review on scheduling problems with setup times and costs, highlights the importance of its study due to its applicability in real cases and shows a variety of heuristics and metaheuristics that generate high quality results. However, the number of articles that contain a B&B algorithm for this class of problems is considerably small when compared to the total. As stated by Rios-Mercado and Bard (1999), authors must consider trade-off between the time required to compute the LB and its tightness. This challenge is evident for PFSPs with setup times, since the terms in the LB formulation that correspond to the setup times usually are the most computationally complex, heavily affecting the performance of the LB inside the B&B algorithm.

Therefore, as the main guideline for studies, we suggest that future research is focused on developing LBs for PFSPs with sequence-dependant setup times, aiming at finding formulations that increase the value of the setup times which are incorporated to the LB value whilst keeping the computational costs low. This would allow for B&B methods to be applied into larger and more complex problem sets, and become a viable option to find optimal solutions to a wider variety of relevant problems at an industrial level.

8. Funding sources

This work was supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) – Brazil under Grant Nos. 306075/2017-2 and 430137/2018-4.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Agnetis, A., Rossi, F., & Gristina, G. (1998). An exact algorithm for the batch sequencing problem in a two-machine flow shop with limited buffer. *Naval Research Logistics, 45*(2), 141–164.

Ahmadi, R. H., & Bagchi, U. (1990). Improved lower bounds for minimizing the sum of completion times of n jobs over m machines in a flow shop. *European Journal of Operational Research, 44*(3), 331–336.

Ahmadi, R. H., & Bagchi, U. (1992). Minimizing job idleness in deadline constrained environments. *Operations Research, 40*(5), 972–985.

Akkan, C., & Karabati, S. (2004). The two-machine flowshop total completion time problem: Improved lower bounds and a branch-and-bound algorithm. *European Journal of Operational Research, 159*(2), 420–429.

Aldowaisan, T. (2001). A new heuristic and dominance relations for no-wait flowshops with setups. *Computers & Operations Research, 28*(6), 563–584.

Allahverdi, A. (2000). Minimizing mean flowtime in a two-machine flowshop with sequence-independent setup times. *Computers & Operations Research, 27*(2), 111–127.

Allahverdi, A. (2001). The tricriteria two-machine flowshop scheduling problem. *International Transactions in Operational Research, 8*(4), 403–425.

Allahverdi, A. (2015). The third comprehensive survey on scheduling problems with setup times/costs. *European Journal of Operational Research, 246*(2), 345–378.

Allahverdi, A., & Al-Anzi, F. S. (2006). A branch-and-bound algorithm for three-machine flowshop scheduling problem to minimize total completion time with separate setup times. *European Journal of Operational Research, 169*(3), 767–780.

Allahverdi, A., & Aldowaisan, T. (2002a). No-wait flowshops with bicriteria of makespan and total completion time. *The Journal of the Operational Research Society, 53*(9), 1004–1015.

Allahverdi, A., & Aldowaisan, T. (2002b). Two-machine flowshop scheduling to minimize total completion time with separate setup and removal times. *International Journal of Industrial Engineering: Theory Applications and Practice, 9*, 275–286.

Allahverdi, A., & Aldowaisan, T. (2004). No-wait flowshops with bicriteria of makespan and maximum lateness. *European Journal of Operational Research, 152*(1), 132–147.

An, Y.-J., Kim, Y.-D., & Choi, S.-W. (2016). Minimizing makespan in a two-machine flowshop with a limited waiting time constraint and sequence-dependent setup times. *Computers & Operations Research, 71*, 127–136.

Ardakan, M. A., Hakimian, A., & Rezvan, M. T. (2014). A branch-and-bound algorithm for minimising the number of tardy jobs in a two-machine flow-shop problem with release dates. *International Journal of Computer Integrated Manufacturing, 27*(6), 519–528.

Bagga, P. C., & Bhambani, A. (2002). Bicriteria in special flowshop problems. *Indian Journal of Pure and Applied Mathematics, 33*(4), 435–442.

Bai, D., Liang, J., Liu, B., Tang, M., & Zhang, Z.-H. (2017). Permutation flow shop scheduling problem to minimize nonlinear objective function with release dates. *Computers & Industrial Engineering, 112*(C), 336–347.

Balasubramanian, J., & Grossmann, I. (2002). A novel branch and bound algorithm for scheduling flowshop plants with uncertain processing times. *Computers & Chemical Engineering, 26*(1), 41–57.

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

Bansal, S. P. (1977). Minimizing the sum of completion times of n jobs over m machines in a flowshop: A branch and bound approach. *AIIE Transactions, 9*(3), 306–311.

Bestwick, P. F., & Hastings, N. A. (1976). A new bound for machine scheduling. *Operational Research Quarterly, 27*(2), 479–487.

Bouquard, J.-L., & Lenté, C. (2006). Two-machine flow shop scheduling problems with minimal and maximal delays. *4OR, 4*(1), 15–28.

Brown, A. P. G., & Lomnicki, Z. A. (1966). Some applications of the branch-and-bound algorithm to the machine scheduling problem. *Journal of the Operational Research Society, 17*(2), 173–186.

Bulfin, R., & M'Hallah, R. (2003). Minimizing the weighted number of tardy jobs on a two-machine flow shop. *Computers & Operations Research, 30*(12), 1887–1900.

Carlier, J., & Rebai, I. (1996). Two branch and bound algorithms for the permutation flow shop problem. *European Journal of Operational Research, 90*(2), 238–251.

Chakroun, I., Melab, N., Mezmaz, M., & Tuyttens, D. (2013). Combining multi-core and gpu computing for solving combinatorial optimization problems. *Journal of Parallel and Distributed Computing, 73*(12), 1563–1577.

Chen, P., Wu, C. C., & Lee, W. C. (2006). A bi-criteria two-machine flowshop scheduling problem with a learning effect. *The Journal of the Operational Research Society, 57*(9), 1113–1125.

Cheng, J., Kise, H., & Matsumoto, H. (1997). A branch-and-bound algorithm with fuzzy inference for a permutation flowshop scheduling problem. *European Journal of Operational Research, 96*(3), 578–590.

Cheng, J., Steiner, G., & Stephenson, P. (2001). A computational study with a new algorithm for the three-machine permutation flow-shop problem with release times. *European Journal of Operational Research, 130*(3), 559–575.

Cheng, J., Steiner, G., & Stephenson, P. (2002). Fast algorithms to minimize the makespan or maximum lateness in the two-machine flow shop with release times. *Journal of Scheduling, 5*(1), 71–92.

Cheng, J.-L., Kise, H., & Karuno, Y. (1997). Optimal scheduling for an automated m-machine flowshop. *Journal of the Operations Research Society of Japan, 40*(3), 356–372.

Cheng, M., Tadikamalla, P. R., Shang, J., & Zhang, B. (2015). Two-machine flow shop scheduling with deteriorating jobs: minimizing the weighted sum of makespan and total completion time. *Journal of the Operational Research Society, 66*(5), 709–719.

Cheng, M., Tadikamalla, P. R., Shang, J., & Zhang, S. (2014). Bicriteria hierarchical optimization of two-machine flow shop scheduling problem with time-dependent deteriorating jobs. *European Journal of Operational Research, 234*(3), 650–657.

Cheng, S.-R., Yin, Y., Wen, C.-H., Lin, W.-C., Wu, C.-C., & Liu, J. (2017). A two-machine flowshop scheduling problem with precedence constraint on two jobs. *Soft Computing, 21*(8), 2091–2103.

Cheng, T., & Janiak, A. (2000). A permutation flow-shop scheduling problem with convex models of operation processing times. *Annals of Operations Research, 96*, 39–60.

Cheng, T., Wu, C.-C., Chen, J.-C., Wu, W.-H., & Cheng, S.-R. (2013). Two-machine flowshop scheduling with a truncated learning function to minimize the makespan. *International Journal of Production Economics, 141*(1), 79–86.

Chihaoui, F. B., Kacem, I., Hadj-Alouane, A. B., Dridi, N., & Rezg, N. (2011). No-wait scheduling of a two-machine flow-shop to minimise the makespan under non-availability constraints and different release dates. *International Journal of Production Research, 49*(21), 6273–6286.

Chung, C. S., Flynn, J., & Kirca, O. (2002). A branch and bound algorithm to minimize the total flow time for m-machine permutation flowshop problems. *International Journal of Production Economics, 79*(3), 185–196.

Chung, C. S., Flynn, J., & Kirca, O. (2006). A branch and bound algorithm to minimize the total tardiness for m-machine permutation flowshop problems. *European Journal of Operational Research, 174*(1), 1–10.

Chung, Y.-H., & Tong, L.-I. (2011). Makespan minimization for m-machine permutation flowshop scheduling problem with learning considerations. *The International Journal of Advanced Manufacturing Technology, 56*(1), 355–367.

Chung, Y. H., & Tong, L. I. (2012). Bi-criteria minimization for the permutation flowshop scheduling problem with machine-based learning effects. *Computers & Industrial Engineering, 63*(1), 303–312.

Companys, R. (1999). Note on an improved branch-and-bound algorithm to solve n/m/p/fmax problems. *Top, 7*(1), 25–31.

Companys, R., & Mateo, M. (2007). Different behaviour of a double branch-and-bound algorithm on fm—prmu—cmax and fm—block—cmax problems. *Computers & Operations Research, 34*(4), 938–953.

Croce, F. D., Ghirardi, M., & Tadei, R. (2002). An improved branch-and-bound algorithm for the two machine total completion time flow shop problem. *European Journal of Operational Research, 139*(2), 293–301.

Croce, F. D., Gupta, J. N., & Tadei, R. (2000). Minimizing tardy jobs in a flowshop with common due date. *European Journal of Operational Research, 120*(2), 375–381.

Croce, F. D., Narayan, V., & Tadei, R. (1996). The two-machine total completion time flow shop problem. *European Journal of Operational Research, 90*(2), 227–237.

Croce, F. D., & T'kindt, V. (2003). Improving the preemptive bound for the one-machine dynamic total completion time scheduling problem. *Operations Research Letters, 31*(2), 142–148.

Daniels, R. L., & Chambers, R. J. (1990). Multiobjective flow-shop scheduling. *Naval Research Logistics, 37*(6), 981–995.

Detienne, B., Sadykov, R., & Tanaka, S. (2016). The two-machine flowshop total completion time problem: Branch-and-bound algorithms based on network-flow formulation. *European Journal of Operational Research, 252*(3), 750–760.

Dileepan, P., & Sen, T. (1991). Job lateness in a two-machine flowshop with setup times separated. *Computers & Operations Research, 18*(6), 549–556.

Drozdowski, M., Marciniak, P., Pawlak, G., & Płaza, M. (2012). *Grid Branch-and-Bound for Permutation Flowshop* (pp. 21–30). Berlin Heidelberg, Berlin, Heidelberg: Springer.

Fondrevelle, J., Allahverdi, A., & Oulamara, A. (2005). Two-machine, no-wait flowshop scheduling problem to minimize maximum lateness with separate set-up and removal times. *International Journal of Agile Manufacturing, 8*, 165–174.

Fondrevelle, J., Oulamara, A., & Portmann, M. C. (2006). Permutation flowshop scheduling problems with maximal and minimal time lags. *Computers & Operations Research, 33*(6), 1540–1556.

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

Fondrevelle, J., Oulamara, A., Portmann, M. C., & Allahverdi, A. (2009). Permutation flowshops with exact time lags to minimize maximum lateness. *International Journal of Production Research, 47*(23), 6759–6775.

Gharbi, A., Ladhari, T., Msakni, M. K., & Serairi, M. (2010). Polynomial lower bounds for the two-machine flowshop problem with sequence-independent setup times. *Electronic Notes in Discrete Mathematics, 36*, 1089–1096.

Gharbi, A., Ladhari, T., Msakni, M. K., & Serairi, M. (2013). The two-machine flowshop scheduling problem with sequence-independent setup times: New lower bounding strategies. *European Journal of Operational Research, 231*(1), 69–78.

Gharbi, A., & Mahjoubi, A. (2013). New lower bounds for flow shop scheduling. *International Journal of Humanities and Management Sciences, 1*(1), 28–32.

Gim, B., Curry, G. L., & Deuermeyer, B. L. (1994). Two-machine flow-shop sequencing with sparse precedence constraints. *Computers & Industrial Engineering, 26*(1), 173–180.

Gmys, J., Mezmaz, M., Melab, N., & Tuyttens, D. (2016). A gpu-based branch-and-bound algorithm using integer–vector–matrix data structure. *Parallel Computing, 59*, 119–139.

Goossens, D. R., Maas, A. J. T., Spieksma, F. C. R., & van de Klundert, J. J. (2007). Exact algorithms for procurement problems under a total quantity discount structure. *European Journal of Operational Research, 178*(2), 603–626.

Gowrishankar, K., Rajendran, C., & Srinivasan, G. (2001). Flow shop scheduling algorithms for minimizing the completion time variance and the sum of squares of completion time deviations from a common due date. *European Journal of Operational Research, 132*(3), 643–665.

Grabowski, J., Skubalska, E., & Smutnicki, C. (1983). On flow shop scheduling with release and due dates to minimize maximum lateness. *The Journal of the Operational Research Society, 34*(7), 615–620.

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

Gupta, J. N. D., & Schaller, J. E. (2006). Minimizing flow time in a flow-line manufacturing cell with family setup times. *Journal of the Operational Research Society, 57*(2), 163–176.

Hamdi, I., & Loukil, T. (2015). Upper and lower bounds for the permutation flowshop scheduling problem with minimal time lags. *Optimization Letters, 9*(3), 465–482.

Haouari, M., & Kharbeche, M. (2013). An assignment-based lower bound for a class of two-machine flow shop problems. *Computers & Operations Research, 40*(7), 1693–1699.

Haouari, M., & Ladhari, T. (2000). Minimising maximum lateness in a two-machine flowshop. *The Journal of the Operational Research Society, 51*(9), 1100–1106.

Haouari, M., & Ladhari, T. (2003). A branch-and-bound-based local search method for the flow shop problem. *The Journal of the Operational Research Society, 54*(10), 1076–1084.

Haouari, M., & Ladhari, T. (2007). Minimizing maximum lateness in a flow shop subject to release dates. *The Journal of the Operational Research Society, 58*(1), 62–72.

Hariri, A., & Potts, C. (1989). A branch and bound algorithm to minimize the number of late jobs in a permutation flow-shop. *European Journal of Operational Research, 38*(2), 228–237.

He, H. (2016). Minimization of maximum lateness in an m-machine permutation flow shop with a general exponential learning effect. *Computers & Industrial Engineering, 97*, 73–83.

Hendizadeh, S. H., ElMekkawy, T. Y., & Wang, G. G. (2007). Bi-criteria scheduling of a flowshop manufacturing cell with sequence dependent setup times. *European Journal of Industrial Engineering, 1*(4), 391–413.

Hnaien, F., Yalaoui, F., & Mhadhbi, A. (2015). Makespan minimization on a two-machine flowshop with an availability constraint on the first machine. *International Journal of Production Economics, 164*, 95–104.

Hoogeveen, H., van Norden, L., & van de Velde, S. (2006). Lower bounds for minimizing total completion time in a two-machine flow shop. *Journal of Scheduling, 9*(6), 559–568.

Hoogeveen, J. A., & van de Velde, S. L. (1995). Stronger lagrangian bounds by use of slack variables: Applications to machine scheduling problems. *Mathematical Programming, 70*(1), 173–190.

Ignall, E., & Schrage, L. (1965). Application of the branch and bound technique to some flow-shop scheduling problems. *Operations Research, 13*(3), 400–412.

Jafari, A.-A., Khademi-Zare, H., Lotfi, M., & Tavakkoli-Moghaddam, R. (2016). A note on "minimizing makespan in three machine flow shop with deteriorating jobs". *Computers & Operations Research, 72*, 93–96.

Jafari, A.-A., Khademi-Zare, H., Lotfi, M., & Tavakkoli-Moghaddam, R. (2017). A note on "on three-machine flow shop scheduling with deteriorating jobs". *International Journal of Production Economics, 191*, 250–252.

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

Joo, B.-J., & Kim, Y.-D. (2009). A branch-and-bound algorithm for a two-machine flowshop scheduling problem with limited waiting time constraints. *Journal of the Operational Research Society, 60*(4), 572–582.

Karabati, S., & Kouvelis, P. (1993). The permutation flow shop problem with sum-of-completion times performance criterion. *Naval Research Logistics, 40*(6), 843–862.

Kim, H.-J., & Lee, J.-H. (2017). A branch and bound algorithm for three-machine flow shop with overlapping waiting time constraints. *IFAC-PapersOnLine, 50*(1), 1101–1105.

Kim, Y.-D. (1993). A new branch and bound algorithm for minimizing mean tardiness in two-machine flowshops. *Computers & Operations Research, 20*(4), 391–401.

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

Knopf, F. C. (1985). Sequencing a generalized two-stage flowshop with finite intermediate storage. *Computers & Chemical Engineering, 9*(3), 207–221.

Kononov, A., Hong, J.-S., Kononova, P., & Lin, F.-C. (2012). Quantity-based buffer-constrained two-machine flowshop problem: active and passive prefetch models for multimedia applications. *Journal of Scheduling, 15*(4), 487–497.

Kubiak, W., Błażewicz, J., Formanowicz, P., Breit, J., & Schmidt, G. (2002). Two-machine flow shops with limited machine availability. *European Journal of Operational Research, 136*(3), 528–540.

Kurihara, K., Li, Y. L., Nishiuchi, N., & Masuda, K. (2009). Flow shop scheduling for separation model of set-up and net process based on Branch-and-Bound method. *Computers & Industrial Engineering, 57*(2), 550–562.

Labidi, M., Kooli, A., Ladhari, T., Gharbi, A., & Suryahatmaja, U. S. (2018). A computational study of the two-machine no-wait flow shop scheduling problem subject to unequal release dates and non-availability constraints. *IEEE Access, 6*, 16294–16304.

Ladhari, T., & Haouari, M. (2005). A computational study of the permutation flow shop problem based on a tight lower bound. *Computers & Operations Research, 32*(7), 1831–1847.

Lageweg, B. J., Lenstra, J. K., & Kan, A. H. G. R. (1978). A general bounding scheme for the permutation flow-shop problem. *Operations Research, 26*(1), 53–67.

Lai, K., Hsu, P.-H., Ting, P.-H., & Wu, C.-C. (2014). A truncated sum of processing-times-based learning model for a two-machine flowshop scheduling problem. *Human Factors and Ergonomics in Manufacturing & Service Industries, 24*(2), 152–160.

Land, A. H., & Doig, A. G. (1960). An automatic method of solving discrete programming problems. *Econometrica, 28*(3), 497–520.

Lee, J.-Y., & Kim, Y.-D. (2017). Minimizing total tardiness in a two-machine flowshop scheduling problem with availability constraint on the first machine. *Computers & Industrial Engineering, 114*, 22–30.

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., kang Chen, S., & Wu, C.-C. (2010). Branch-and-bound and simulated annealing algorithms for a two-agent scheduling problem. *Expert Systems with Applications, 37*(9), 6594–6601.

Lee, W.-C., Shiau, Y.-R., Chen, S.-K., & Wu, C.-C. (2010). A two-machine flowshop scheduling problem with deteriorating jobs and blocking. *International Journal of Production Economics, 124*(1), 188–197.

Lee, W.-C., & Wu, C.-C. (2001). Minimizing the total flow time and the tardiness in a two-machine flow shop. *International Journal of Systems Science, 32*(3), 365–373.

Lee, W.-C., & Wu, C.-C. (2004). Minimizing total completion time in a two-machine flowshop with a learning effect. *International Journal of Production Economics, 88*(1), 85–93.

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, D.-C., Hsu, P.-H., Wu, C.-C., & Cheng, T. (2011). Two-machine flowshop scheduling with truncated learning to minimize the total completion time. *Computers & Industrial Engineering, 61*(3), 655–662.

Li, X., Jiang, Y., & Ruiz, R. (2018). Methods for scheduling problems considering experience, learning, and forgetting effects. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 48*(5), 743–754.

Liao, C.-J., Yu, W.-C., & Joe, C.-B. (1997). Bicriterion scheduling in the two-machine flowshop. *Journal of the Operational Research Society, 48*(9), 929–935.

Liao, L.-M., & Tsai, C.-H. (2009). Heuristic algorithms for two-machine flowshop with availability constraints. *Computers & Industrial Engineering, 56*(1), 306–311.

Lim, C. J., & McMahon, G. B. (1994). The three-machine flow-shop problem with arbitrary precedence relations. *European Journal of Operational Research, 78*(2), 216–223.

Lin, B. M. T., Lin, F. C., & Lee, R. C. T. (2006). Two-machine flow-shop scheduling to minimize total late work. *Engineering Optimization, 38*(4), 501–509.

Lin, B. M. T., & Wu, J. M. (2005). A simple lower bound for total completion time minimization in a two-machine flowshop. *Asia-Pacific Journal of Operational Research, 22*(03), 391–407.

Lin, B. M. T., & Wu, J. M. (2006). Bicriteria scheduling in a two-machine permutation flowshop. *International Journal of Production Research, 44*(12), 2299–2312.

Lin, F.-C., Hong, J.-S., & Lin, B. M. (2009). A two-machine flowshop problem with processing time-dependent buffer constraints—an application in multimedia presentations. *Computers & Operations Research, 36*(4), 1158–1175.

Lin, F.-C., Hong, J.-S., & Lin, B. M. (2013). Sequence optimization for media objects with due date constraints in multimedia presentations from digital libraries. *Information Systems, 38*(1), 82–96.

Lomnicki, Z. A. (1965). Branch-and-bound algorithm for the exact solution of the three-machine scheduling problem. *Journal of the Operational Research Society, 16*(1), 89–100.

Maccarthy, B. L., & Liu, J. (1993). Addressing the gap in scheduling research: A review of optimization and heuristic methods in production scheduling. *International Journal of Production Research, 31*(1), 59–79.

Madhushini, N., & Rajendran, C. (2011). Branch-and-bound algorithms for scheduling in an m-machine permutation flowshop with a single objective and with multiple objectives. *European Journal of Industrial Engineering, 5*(4), 361.

Madhushini, N., & Rajendran, C. (2012). A bottleneck-assignment based branch-and-bound algorithm to minimize the makespan in an m-machine permutation flowshop. *Technology Operation Management, 3*(1), 1–10.

Madhushini, N., Rajendran, C., & Deepa, Y. (2009). Branch-and-bound algorithms for scheduling in permutation flowshops to minimize the sum of weighted flowtime/sum of weighted tardiness/sum of weighted flowtime and weighted tardiness/sum of weighted flowtime, weighted tardiness and weighted earliness of jobs. *The Journal of the Operational Research Society, 60*(7), 991–1004.

McMahon, G. B., & Burton, P. G. (1967). Flow-shop scheduling with the branch-and-bound method. *Operations Research, 15*(3), 473–481.

McMahon, G. B., & Lim, C.-J. (1993). The two-machine flow shop problem with arbitrary precedence relations. *European Journal of Operational Research, 64*(2), 249–257 (project Management and Scheduling).

Moore, J. M. (1968). An n job, one machine sequencing algorithm for minimizing the number of late jobs. *Management Science, 15*(1), 102–109.

Moslehi, G., & Khorasanian, D. (2013). Optimizing blocking flow shop scheduling problem with total completion time criterion. *Computers & Operations Research, 40*(7), 1874–1883.

Moslehi, G., Mirzaee, M., Vasei, M., Modarres, M., & Azaron, A. (2009). Two-machine flow shop scheduling to minimize the sum of maximum earliness and tardiness. *International Journal of Production Economics, 122*(2), 763–773.

Moukrim, A., Rebaine, D., & Serairi, M. (2014). A branch and bound algorithm for the two-machine flowshop problem with unit-time operations and time delays. *RAIRO - Operations Research, 48*(2), 235–254.

Msakni, M. K., Khallouli, W., Al-Salem, M., & Ladhari, T. (2016). Minimizing the total completion time in a two-machine flowshop problem with time delays. *Engineering Optimization, 48*(7), 1164–1181.

Nabeshima, I. (1967). On the bound of makespan and its application in m machine-schedule problem. *Journal of the Operations Research Society of Japan, 9*(3 & 4), 98–136.

Nagar, A., Heragu, S. S., & Haddock, J. (1995). A branch-and-bound approach for a two-machine flowshop scheduling problem. *The Journal of the Operational Research Society, 46*(6), 721–734.

Nagar, A., Heragu, S. S., & Haddock, J. (1996). A combined branch-and-bound and genetic algorithm based approach for a flowshop scheduling problem. *Annals of Operations Research, 63*(3), 397–414.

Narain, L., & Bagga, P. C. (2003). Minimizing total elapsed time subject to zero total idle time of machines in n x 3 flowshop problem. *Indian Journal of Pure and Applied Mathematics, 34*(2), 219–228.

Narain, L., & Bagga, P. C. (2005). Flowshop no idle scheduling to minimise the mean flowtime. *The ANZIAM Journal, 47*(2), 265–275.

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.

Ng, C., Wang, J.-B., Cheng, T., & Liu, L. (2010). A branch-and-bound algorithm for solving a two-machine flow shop problem with deteriorating jobs. *Computers & Operations Research, 37*(1), 83–90.

Pan, C.-H., & Wu, C.-C. (1996). An asymptotic two-phase algorithm to minimize total flow time for a two-machine flowshop. *International Journal of Systems Science, 27*(10), 925–930.

Pan, J. C.-H., Chen, J.-S., & Chao, C.-M. (2002). Minimizing tardiness in a two-machine flow-shop. *Computers & Operations Research, 29*(7), 869–885.

Pan, J. C.-H., & Fan, E.-T. (1997). Two-machine flowshop scheduling to minimize total tardiness. *International Journal of Systems Science, 28*(4), 405–414.

Panwalkar, S. S., & Khan, A. W. (1975). An improved branch and bound procedure for n x m flow shop problems. *Naval Research Logistics Quarterly, 22*(4), 787–790.

Pinedo, M. L. (2008). *Scheduling: Theory, algorithms, and systems* (3rd Edition). Springer Publishing Company, Incorporated.

Potts, C. N. (1980). An adaptive branching rule for the permutation flow-shop problem. *European Journal of Operational Research, 5*(1), 19–25.

Proust, C., Gupta, J. N. D., & Deschamps, V. (1991). Flowshop scheduling with set-up, processing and removal times separated. *International Journal of Production Research, 29*(3), 479–493.

Rajendran, C. (1992). Two-stage flowshop scheduling problem with bicriteria. *The Journal of the Operational Research Society, 43*(9), 871–884.

Rakrouki, M. A., Kooli, A., Chalghoumi, S., & Ladhari, T. (2017). A branch-and-bound algorithm for the two-machine total completion time flowshop problem subject to release dates. *Operational Research*.

Ren, T., Zhao, P., Zhang, D., Liu, B., Yuan, H., & Bai, D. (2017). Permutation flow-shop scheduling problem to optimize a quadratic objective function. *Engineering Optimization, 49*(9), 1589–1603.

Rios-Mercado, R. Z., & Bard, J. (1999). A branch-and-bound algorithm for permutation flow shops with sequence-dependent setup times. *IIE Transactions, 31*(8), 721–731.

Ritt, M. (2016). A branch-and-bound algorithm with cyclic best-first search for the permutation flow shop scheduling problem. In *2016 IEEE international conference on automation science and engineering (CASE)* (pp. 872–877).

Ronconi, D. P. (2005). A branch-and-bound algorithm to minimize the makespan in a flowshop with blocking. *Annals of Operations Research, 138*(1), 53–65.

Ronconi, D. P., & Armentano, V. A. (2001). Lower bounding schemes for flowshops with blocking in-process. *Journal of the Operational Research Society, 52*(11), 1289–1297.

Saadani, N. E. H., Guinet, A., & Moallaa, M. (2003). Three stage no-idle flow-shops. *Computers & Industrial Engineering, 44*(3), 425–434.

Samarghandi, H. (2019). Minimizing the makespan in a flow shop environment under minimum and maximum time-lag constraints. *Computers & Industrial Engineering, 136*, 614–634.

Samarghandi, H., & Behroozi, M. (2017). On the exact solution of the no-wait flow shop problem with due date constraints. *Computers and Operations Research, 81*, 141–159.

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), 321–326.

Sayin, S., & Karabati, S. (1999). Theory and methodology a bicriteria approach to the two-machine flow shop scheduling problem. *European Journal of Operational Research, 113*(2), 435–449.
Schaller, J. (2001). A new lower bound for the flow shop group scheduling problem. *Computers & Industrial Engineering, 41*(2), 151–161.
Schaller, J. (2005). An improved branch and bound procedure for scheduling a flow line manufacturing cell. *International Journal of Production Research, 43*(22), 4697–4720.
Schaller, J. (2005). Note on minimizing total tardiness in a two-machine flowshop. *Computers & Operations Research, 32*(12), 3273–3281.
Schaller, J. E., Gupta, J. N., & Vakharia, A. J. (2000). Scheduling a flowline manufacturing cell with sequence dependent family setup times. *European Journal of Operational Research, 125*(2), 324–339.
Sen, T., Dileepan, P., & Gupia, J. N. (1989). The two-machine flowshop scheduling problem with total tardiness. *Computers & Operations Research, 16*(4), 333–340.
Shiau, Y.-R., Tsai, M.-S., Lee, W.-C., & Cheng, T. (2015). Two-agent two-machine flowshop scheduling with learning effects to minimize the total completion time. *Computers & Industrial Engineering, 87*, 580–589.
Sivrikaya-Şerifoğlu, F., & Ulusoy, G. (1998). A bicriteria two-machine permutation flowshop problem. *European Journal of Operational Research, 107*(2), 414–430.
Smith, W. E. (1956). Various optimizers for single-stage production. *Naval Research Logistics Quarterly, 3*(1–2), 59–66.
Soylu, B., Kirca & Azizoglu, M. (2007). Flow shop-sequencing problem with synchronous transfers and makespan minimization. *International Journal of Production Research, 45*(15), 3311–3331.
Su, L.-H., & Chen, J. C. (2011). Two- and three-machine flowshop scheduling problems with optional final operation. *Journal of the Chinese Institute of Industrial Engineers, 28*(1), 55–71.
Su, L.-H., & Lee, Y.-Y. (2008). The two-machine flowshop no-wait scheduling problem with a single server to minimize the total completion time. *Computers & Operations Research, 35*(9), 2952–2963.
Suhami, I., & Mah, R. S. (1981). An implicit enumeration scheme for the flowshop problem with no intermediate storage. *Computers & Chemical Engineering, 5*(2), 83–91.
Süral, H., Kondakci, S., & Erkip, N. (1992). Scheduling unit-time tasks in renewable resource constrained flowshops. *ZOR Zeitschrift fü Operations Research Methods and Models of Operations Research, 36*, 497–516.
Tadei, R., Gupta, J. N. D., Croce, F. D., & Cortesi, M. (1998). Minimising makespan in the two-machine flow-shop with release times. *The Journal of the Operational Research Society, 49*(1), 77–85.
Takano, M. I., & Nagano, M. S. (2017). A branch-and-bound method to minimize the makespan in a permutation flow shop with blocking and setup times. *Cogent Engineering, 4*(1).
Tang, L., & Huang, L. (2007). Optimal and near-optimal algorithms to rolling batch scheduling for seamless steel tube production. *International Journal of Production Economics, 105*(2), 357–371.
Tellache, N. E. H., & Boudhar, M. (2017). Two-machine flow shop problem with unit-time operations and conflict graph. *International Journal of Production Research, 55*(6), 1664–1679.
Temiz, I., & Erol, S. (2004). Fuzzy branch-and-bound algorithm for flow shop scheduling. *Journal of Intelligent Manufacturing, 15*(4), 449–454.
T'kindt, V., Croce, F. D., & Esswein, C. (2004). Revisiting branch and bound search strategies for machine scheduling problems. *Journal of Scheduling, 7*(6), 429–440.
T'kindt, V., Gupta, J. N., & Billaut, J.-C. (2003). Two-machine flowshop scheduling with a secondary criterion. *Computers & Operations Research, 30*(4), 505–526.
Toktaş, B., Azizoğlu, M., & Köksalan, S. K. (2004). Two-machine flow shop scheduling with two criteria: Maximum earliness and makespan. *European Journal of Operational Research, 157*(2), 286–295.
Toumi, S., Jarboui, B., Eddaly, M., & Rebaï, A. (2017a). Branch and bound algorithm for solving blocking flowshop scheduling problem with total tardiness and total weighted tardiness criteria. *International Journal of Operational Research, 30*(4), 441–459.
Toumi, S., Jarboui, B., Eddaly, M., & Rebaï, A. (2017b). Branch-and-bound algorithm for solving blocking flowshop scheduling problems with makespan criterion. *International Journal of Mathematics in Operational Research, 10*(1), 34.
Toumi, S., Jarboui, B., Eddaly, M., & Rebaï, A. (2013a). Branch and bound algorithm for solving blocking flowshop scheduling problem with total completion time. In *2013 International conference on control, decision and information technologies (CoDIT)* (pp. 746–749).
Toumi, S., Jarboui, B., Eddaly, M., & Rebaï, A. (2013b). New lower bounds for the blocking flowshop scheduling problem to minimize the total completion time criterion. In *2013 5th international conference on modeling, simulation and applied optimization (ICMSAO)* (pp. 1–5).
Toumi, S., Jarboui, B., Eddaly, M., & Rebaï, A. (2013c). Solving blocking flowshop scheduling problem with branch and bound algorithm. In *2013 International conference on advanced logistics and transport* (pp. 411–416).
Townsend, W. (1977). Sequencing n jobs on m machines to minimise maximum tardiness: A branch- and-bound solution. *Management Science, 23*(9), 1016–1019.
Tsujimura, Y., Park, S. H., Chang, I. S., & Gen, M. (1993). An effective method for solving flow shop scheduling problems with fuzzy processing times. *Computers & Industrial Engineering, 25*(1), 239–242.
van de Velde, S. L. (1991). Minimizing the sum of the job completion times in the two-machine flow shop by lagrangian relaxation. *Annals of Operations Research, 26*(1–4), 257–268.
Wagneur, E., & Sriskandarajah, C. (1993). The two-machine permutation flow shop with state-dependent processing times. *Naval Research Logistics, 40*(5), 697–717.
Wang, J.-B., Ji, P., Cheng, T. C. E., & Wang, D. (2012). Minimizing makespan in a two-machine flow shop with effects of deterioration and learning. *Optimization Letters, 6*(7), 1393–1409.
Wang, J.-B., & Liu, L.-L. (2009). Two-machine flow shop problem with effects of deterioration and learning. *Computers & Industrial Engineering, 57*(3), 1114–1121.
Wang, J.-B., & Liu, L.-L. (2009). Two-machine flow shop scheduling with linear decreasing job deterioration. *Computers & Industrial Engineering, 56*(4), 1487–1493.
Wang, J.-B., Ng, C. D., Cheng, T., & Liu, L.-L. (2006). Minimizing total completion time in a two-machine flow shop with deteriorating jobs. *Applied Mathematics and Computation, 180*(1), 185–193.
Wang, J.-B., & Wang, M.-Z. (2013a). Minimizing makespan in three-machine flow shops with deteriorating jobs. *Computers & Operations Research, 40*(2), 547–557.
Wang, J.-B., & Wang, M.-Z. (2013b). Solution algorithms for the total weighted completion time minimization flow shop scheduling with decreasing linear deterioration. *The International Journal of Advanced Manufacturing Technology, 67*(1), 243–253.
Wang, J. J., & Zhang, B. H. (2015). Permutation flowshop problems with bi-criterion makespan and total completion time objective and position-weighted learning effects. *Computers & Operations Research, 58*, 24–31.
Wang, L., Sun, L.-Y., Sun, L.-H., & Wang, J.-B. (2010). On three-machine flow shop scheduling with deteriorating jobs. *International Journal of Production Economics, 125*(1), 185–189.
Wang, X., & Cheng, T. (2005). Two-machine flowshop scheduling with job class setups to minimize total flowtime. *Computers & Operations Research, 32*(11), 2751–2770.
Wang, Z., Wei, C.-M., & Lu, Y.-Y. (2016). Permutation flow shop problem with shortening job processing times. *Asia-Pacific Journal of Operational Research, 33*(04), 1650032.
Wolsey, L. A. (1998). *Integer programming*. New York, NY, USA: Wiley-Interscience.
Wu, C.-C., & Lee, W.-C. (2006). Two-machine flowshop scheduling to minimize mean flow time under linear deterioration. *International Journal of Production Economics, 103*(2), 572–584.
Wu, C.-C., Lee, W.-C., & Wang, W.-C. (2007). A two-machine flowshop maximum tardiness scheduling problem with a learning effect. *The International Journal of Advanced Manufacturing Technology, 31*(7), 743–750.
Wu, C.-C., Wu, W.-H., Hsu, P.-H., & Lai, K. (2012). A two-machine flowshop scheduling problem with a truncated sum of processing-times-based learning function. *Applied Mathematical Modelling, 36*(10), 5001–5014.
Wu, W. H., Wu, W. H., Chen, J. C., Lin, W. C., Wu, J., & Wu, C. C. (2015). A heuristic-based genetic algorithm for the two-machine flowshop scheduling with learning consideration. *Journal of Manufacturing Systems, 35*, 223–233.
Yanai, S., & Fujie, T. (2006). A three-machine permutation flow-shop problem with minimum makespan on the second machine. *The Journal of the Operational Research Society, 57*(4), 460–468.
Yang, D.-L., & Chern, M.-S. (1995). A two-machine flowshop sequencing problem with limited waiting time constraints. *Computers & Industrial Engineering, 28*(1), 63–70.
Yang, S.-H., & Wang, J.-B. (2011). Minimizing total weighted completion time in a two-machine flow shop scheduling under simple linear deterioration. *Applied Mathematics and Computation, 217*(9), 4819–4826.
Yang, W. H. (2002). Group scheduling in a two-stage flowshop. *The Journal of the Operational Research Society, 53*(12), 1367–1373.
Yeh, W.-C. (1999). A new branch-and-bound approach for the n/2/flowshop/n/2/flowshop/αf + βc_max flowshop scheduling problem. *Computers & Operations Research, 26*(13), 1293–1310.
Yeh, W.-C. (2001). An efficient branch-and-bound algorithm for the two-machine bicriteria flowshop scheduling problem. *Journal of Manufacturing Systems, 20*(2), 113–123.
Yeh, W.-C., & Allahverdi, A. (2004). A branch-and-bound algorithm for the three-machine flowshop scheduling problem with bicriteria of makespan andtotal flowtime. *International Transactions in Operational Research, 11*(3), 323–339.
Yeung, W., Oğuz, C., & Cheng, T. E. (2004). Two-stage flowshop earliness and tardiness machine scheduling involving a common due window. *International Journal of Production Economics, 90*(3), 421–434 (production Control and Scheduling).
Zeballos, L. J., Castro, P. M., & Mèndez, C. A. (2011). Integrated constraint programming scheduling approach for automated wet-etch stations in semiconductor manufacturing. *Industrial and Engineering Chemistry Research, 50*(3), 1705–1715.