Sādhanā (2020)45:255
https://doi.org/10.1007/s12046-020-01432-z

Branch-and-bound algorithms for scheduling in an *m*-machine no-wait flowshop

NARAYANAPRASAD MADHUSHINI[1] and CHANDRASEKHARAN RAJENDRAN[2,*]

[1]ExxonMobil Research and Engineering, Spring, TX, USA
[2]Department of Management Studies, Indian Institute of Technology Madras, Chennai 600036, India
e-mail: madhu_shanu@yahoo.com; craj@iitm.ac.in

MS received 28 November 2019; revised 28 April 2020; accepted 3 June 2020

Abstract. In this paper, we develop branch-and-bound algorithms for objectives such as sum of weighted flowtime, weighted tardiness and weighted earliness of jobs, for an $m-$machine no-wait (continuous) flowshop. We believe that there has been no prior work on exact algorithms for this problem setup with a variety of objective functions. For the interest of space, we confine our discussion to a subset of certain combination of these objectives and the extension to other objective combinations is quite straight-forward. We explore the active nodes of a branch-and-bound tree by deriving an assignment-matrix based lower bound, that ensures one-to-one correspondence of a job with its due date and weight. This idea is based on our earlier paper on general $m-$machine permutation flowshop (Madhushini et al. in J Oper Res Soc 60(7):991–1004, 2009) and here we exploit the intricate features of a no-wait flowshop to develop efficient lower bounds. Finally, we conclude our paper with the numerical evaluation of our branch-and-bound algorithms.

Keywords. Manufacturing; no-wait flowshops; weighted flowtime; weighted tardiness; weighted earliness; branch-and-bound algorithm.

1. Introduction

A general $m-$machine permutation flowshop is a manufacturing system that contains $n-$jobs to be scheduled among $m-$sequential machines. In this system, every machine performs a different operation on a job with an unlimited wait allowed between the operations. This basic assumption of wait between machines render the schedule unrealistic and impractical in certain scenarios, leading to a different class of flowshops known as no-wait flowshops, also known as continuous flowshops. Here, once the processing of a job begins in the first machine, subsequent processing of that job must be carried out without any delay in the passage of jobs between machines. If necessary, the start of a job on the first machine can be delayed to ensure the continuous flow of jobs in the subsequent machines. Applications of such no-wait flowshops can be found in many industries like chemical refineries, steel factories, pharmaceuticals, food processing, semiconductor manufacturing where the process cannot be interrupted in between. There is quite a significant amount of work done in the field of no-wait flowshop with makespan and total flowtime objectives. But we believe that the due-date based objectives are perhaps more important in real life and are gaining importance in no-wait scheduling practices. We proposed branch-and-bound algorithms for an $m-$machine permutation flowshop manufacturing system in [1] and we now extend our idea to study the no-wait flowshop manufacturing systems.

The main contributions of this article are summarized as follows.

(1) We focus on developing a generic branch-and-bound algorithm to handle a variety of composite and versatile objective functions.
(2) The proposed algorithm can be applied in a no-wait setting for minimizing makespan, (un-)weighted flowtime, (un-)weighted tardiness and (un-)weighted earliness of jobs, both in isolation and in different combinations.
(3) Firstly, we propose lower bounds that are efficient due to one-to-one correspondence between a job, and its due date and weight. This one-to-one correspondence enables our algorithm to be generic enough to handle several criteria.
(4) We then develop exact branch-and-bound algorithms with the proposed lower bounds for each of the objectives considered. To our best knowledge, we are the first to develop exact algorithms for weighted objectives in a no-wait flowshop.
(5) We finally present the empirical results of the branch-and-bound algorithms. We would like to emphasize that

*For correspondence
Published online: 12 October 2020

we do not claim superior performance over any specific objective function, rather we aim to develop a single generic exact algorithm that is applicable to a variety of completion-time based criteria.

The remainder of the article is organized as follows. Section 2 summarizes the related work on no-wait flowshops. Section 3 presents a detailed discussion on the proposed lower bounds, a review of branch-and-bound algorithms developed and the proofs for the proposed bounds. We present the numerical results in Section 4 and Section 5 presents the final summary of our work.

2. Literature review

As mentioned in Section 1, there is quite a significant amount of literature on no-wait flowshop with makespan and flowtime criterion. These problems were perhaps first addressed in [2] and [3]. While several algorithms are proposed in [2] to deal with makespan objective, branch-and-bound algorithms with flowtime objective are developed in [3] . Some special solvable cases like modeling no-wait flowshop with makespan criterion as traveling salesman problem are dealt in [4]. This problem is said to be NP-complete in a strong sense even for the two-machine case ([5]) and thus there are very few attempts on developing exact methods, even for unweighted objectives. An exact branch-and-bound algorithm to minimize the makespan in a general $m-$machine permutation flowshop with set-up times and blocking is presented in [6], in which a job i is blocked in machine $m-1$ until machine m is free to process the job i instantaneously. This is different from the no-wait condition considered in our paper, where the job needs to flow continously between machines and cannot wait in machine $m-1$ until the next machine m is ready. A number of mathematical models and enumeration based algorithms are developed in [7, 8] and [9] to deal the makespan objective in no-wait flowshop with hard due date constraints.

More details on the complexity of no-wait flowshop scheduling problems in general are discussed in [10, 11], while the theoretical aspects specific to flowtime criterion are addressed in [12]. There are several approximate algorithms for generic m-machine no-wait flowshop. To mention a few, [13–23] and [24]. There are some algorithms proposed in [25–29] and [30] to study the no-wait flowshop with setup times minimizing total flowtime and/or tardiness criterion.

An extensive review of heuristics and approximation algorithms for makepsan and due-date based objectives on a $m-$machine no-wait flowshop is presented in [31]. Also, a review and classification of constructive heuristics proposed for no-wait flowshops with additional constraints like setup times, release dates, machine downtimes and due dates, and several objectives like makepsan, flowtime and due date consideration is presented in [32]. There seem to be no attempts, hence we seek to develop exact algorithms for no-wait flowshops with weighted flowtime objective.

Considering the tardiness and earliness objectives, there is not much work done in no-wait flowshops compared to other flowshops. A mixed-integer progarm along with a number of metaheuristics are developed in [33] and [34] for weighted tardiness and weighted earliness objectives in no-wait flowshop. Few other heuristics for un-weighted tardiness objective are proposed in [35–38] and [39]. The common theme between some of these heuristics is that they first solve a due-date relaxed problem and then develop a no-wait feasible schedule using some heuristic/meta-heuristic methods.

From these literature, we can see that heuristics are the main-stream approaches for solving no-wait flowshop scheduling problems with completion-time and due-date based objective functions, and to the best of our knowledge there is no prior work on developing exact methods for criterion such as weighted flowtime and weighted lateness on no-wait flowshops. Additionally, the majority of the heuristics proposed in the literatures are quite specific to the criterion dealt in their papers and the extension of the methodology to other criteria is non-trivial.

Hence we believe that this work is a new attempt to consider the problem of no-wait flowshop scheduling with weighted objectives, and develop an exact branch-and-bound algorithm that can handle a variety of composite functions. For the sake of length of the paper, we confine our content to a subset of objectives such as the minimization of 1. sum of weighted flowtime, 2. sum of weighted tardiness, 3. sum of weighted flowtime and weighted tardiness and 4. sum of weighted flowtime, weighted tardiness and weighted earliness of jobs, and we strongly believe that the algorithm can easily be extended to other excluded objectives such as makespan and alternative combinations of aforementioned objectives.

It is important to note that we dealt with the weighted flowtime and weighted due-date based objective functions in the context of $m-$machine permutation flowshop in [1], and in this work we propose a new branch-and-bound algorithm by exploiting the special features of no-wait flowshops.

3. Proposed bounds on the objectives under study

For the sake of consistency with our prequel paper [1], we adapt the same problem set-up and some common notations appropriately.

Let σ denote the available partial sequence with n' jobs scheduled; Π denote the set of n'' unscheduled jobs such that total number of jobs $n = n' + n''$, $\sigma \cup \Pi = \Sigma$ (a complete sequence) and $\sigma \cap \Pi = \emptyset$, a null set. The following subsection presents in detail the computation of lower

bounds for every unscheduled job. This includes computation of a delay time matrix for every pair of jobs and objective function values with respect to partial sequence σ.

3.1 *Proposed lower bounds on completion time under no-wait condition*

As the definition goes, in a no-wait flowshop the start of the jobs can be delayed on the first machine if necessary for an uninterrupted flow of jobs. Thus we now start with a discussion on the bounds for the delay time between the start of jobs on the first machine. Let the delay between the start of job i and the start of job k (when job k follows job i under no-wait condition) on the first machine be denoted by d_{ik}, and p_{ij} denote the processing time of job i on machine j. The delay time $[d_{ik}]$ matrix for every pair of jobs i and k is computed by aggregating the wait times intended between job i and k along the fixed machine sequence and the steps to compute are given in detail as follows:

Do the following for every job $i = 1, 2, \ldots, n$.

{

Step 0.1: Compute completion time of job i on machine j, which is given by

$$C(i, j) = \sum_{j'=1}^{j} p_{ij'}, \text{ for } j = 1, 2, \ldots, m.$$

Step 0.2: Do the following for other jobs, $k \neq i$ and $k = 1, 2, \ldots, n$:

{

Step 0.2.1: Compute completion time of job k following job i on machine j

$$C(ik, 1) = C(i, 1) + p_{k1} \text{ and}$$

$$C(ik, j) = \max\{C(ik, j-1); C(i, j)\} + p_{kj},$$

$$\text{for } j = 2, 3, \ldots, m.$$

Step 0.2.2: Compute the delay time d_{ik} on first machine, when job k follows job i

$$d_{ik} = p_{i1} + \sum_{j=2}^{m} \max\{C(i, j) - C(ik, j-1); 0\} \qquad (1)$$

}

}

Once the delay time matrix between every pair of jobs is obtained, we proceed to compute the objective function values with respect to jobs scheduled in known partial sequence σ, under no-wait condition.

Given the due-date (D_i), flowtime weights (h_i), tardiness weights (τ_i) and earliness weights (e_i) of jobs $i = 1, 2, \ldots n$, let $C_w(\sigma_{[k]}, j)$ denote the completion time of job placed in k^{th} position in partial sequence σ on machine j, with all jobs in σ scheduled under no-wait condition.

The completion time of any job placed in first position of partial sequence σ on machine j is given by,

$$C_w(\sigma_{[1]}, j) = \sum_{j'=1}^{j} p_{\sigma_{[1]}j'}, \forall j = 1, 2, \ldots, m$$

and the completion times of jobs placed in positions $k = 2, 3, \ldots, n'$ of σ is given by,

$$C_w(\sigma_{[k]}, j) = \sum_{k'=2}^{k} d_{\sigma_{[k'-1]}\sigma_{[k']}} + \sum_{j'=1}^{j} p_{\sigma_{[k]}j'}, \quad \forall j = 1, 2, \ldots, m \qquad (2)$$

Finally, the sum of weighted flowtime (Z'_f), sum of weighted tardiness (Z'_t), sum of weighted flowtime and weighted tardiness (Z'_{ft}), and sum of weighted flowtime, weighted tardiness and weighted earliness (Z'_{fte}) of jobs in partial sequence σ are given by,

$$\begin{aligned} Z'_f(\sigma) &= \sum_{k=1}^{n'} C_w(\sigma_{[k]}, m) \times h_{\sigma_{[k]}} \\ Z'_t(\sigma) &= \sum_{k=1}^{n'} \max\{\{C_w(\sigma_{[k]}, m) - D_{\sigma_{[k]}}\}; 0\} \times \tau_{\sigma_{[k]}} \\ Z'_{ft}(\sigma) &= Z'_f(\sigma) + Z'_t(\sigma) \\ Z'_{fte}(\sigma) &= Z'_f(\sigma) + Z'_t(\sigma) \\ &\quad + \sum_{k=1}^{n'} \max\{\{D_{\sigma_{[k]}} - C_w(\sigma_{[k]}, m)\}; 0\} \times e_{\sigma_{[k]}} \end{aligned} \qquad (3)$$

Here, $h_{\sigma_{[k]}}, \tau_{\sigma_{[k]}}$ and $e_{\sigma_{[k]}}$ denote the respective weights of flowtime, tardiness and earliness of job k placed in k^{th} position in partial sequence σ. We now proceed to discuss the steps to compute the proposed lower bound on completion time for unscheduled jobs, under no-wait condition.

Let a be the last job in σ and $\text{LB}C_w(\sigma[1], m)$ denote a lower bound on the completion time of a job placed immediately following σ. Similarly, let the parameter $\text{LB}C_w(\sigma[1]\ldots[r'], m)$ denote a lower bound on the completion time of a job found in position r' and let $\text{LB}C_w(\sigma[1]\ldots[r']i, m)$ denote a lower bound on the completion time of job i, when placed in position $r'+1$ following σ, on machine m for any $r' = 2, 3, \ldots, |\Pi - 1|$.

To compute the proposed lower bound on completion time, for every unscheduled job $i \in \Pi$, we do the following steps:

Step 1: Compute the completion time of job i following last job a in partial sequence σ using,

$$C_w(\sigma_{[n']}i,m) = C'(\sigma,1) + d_{ai} + \sum_{j=1}^{m} p_{ij} \tag{4}$$

where $\sigma_{[n']} = a$, $C'(\sigma,1) = \sum_{\substack{k'=2 \\ k' \in \sigma}}^{n'} d_{\sigma_{[k'-1]}\sigma_{[k']}}$ for $n' \geq 2$ and $C'(\sigma,1) = 0$ for $n' < 2$.

Step 2: Do the following:

Step 2.1: Find the minimum delay among the unscheduled jobs following last job a in σ (denoted by d_a^*). Also find the minimum delay for every unscheduled job $i' \in \Pi$ with respect to other unscheduled jobs following job i' excluding job i (denoted by $d_{i'}^*$), and minimum delay for every unscheduled job $i' \in \Pi$ with respect to other unscheduled jobs following job i' including job i (denoted by $d_{i'}^{\#}$):

$$\begin{aligned} d_a^* &= \min_{i' \in \Pi - \{i\}} \{d_{ai'}\}, \\ d_{i'}^* &= \min_{\substack{i'' \in \Pi - \{i\} \\ i'' \neq i'}} \{d_{i'i''}\}, \forall i' \in \Pi - \{i\} \text{ and} \\ d_{i'}^{\#} &= \min_{\substack{i'' \in \Pi \\ i'' \neq i'}} \{d_{i'i''}\}, \forall i' \in \Pi - \{i\}. \end{aligned} \tag{5}$$

Step 2.2: Find the minimum delay among the unscheduled jobs preceding job i (denoted by b_i^*), minimum delay for every unscheduled job i' with respect to other unscheduled jobs preceding job i' (denoted by $b_{i'}^*$), and minimum delay for every unscheduled job i' with respect to last job a in partial sequence σ and other unscheduled jobs preceding job i' (denoted by $b_{i'}^{\#}$):

$$\begin{aligned} b_i^* &= \min_{i' \in \Pi - \{i\}} \{d_{i'i}\}, \\ b_{i'}^* &= \min_{\substack{i'' \in \Pi - \{i\} \\ i'' \neq i'}} \{d_{i''i'}\}, \forall i' \in \Pi - \{i\} \text{ and} \\ b_{i'}^{\#} &= \min_{\substack{i'' \in \Pi - \{i\} \\ i'' \neq i'}} \{d_{i''i'}; d_{ai'}\}, \forall i' \in \Pi - \{i\}. \end{aligned} \tag{6}$$

Step 3: Sort the values computed in Step 2 in the non-decreasing order given as follows,
Let $d_{[r']}^{**}$ and $b_{[r']}^{**}$ denote the ordered set of $d_{i'}^*$'s and $b_{i'}^*$'s respectively, such that $d_{[1]}^{**} \leq d_{[2]}^{**} \leq \ldots \leq d_{[|\Pi|-1]}^{**}$ and $b_{[1]}^{**} \leq b_{[2]}^{**} \leq \ldots \leq b_{[|\Pi|-1]}^{**}$.

Similarly, $d_{[r']}^{\#\#}$ and $b_{[r']}^{\#\#}$ denote the ordered set of $d_{i'}^{\#}$'s and $b_{i'}^{\#}$'s respectively, such that $d_{[1]}^{\#\#} \leq d_{[2]}^{\#\#} \leq \ldots \leq d_{[|\Pi|-1]}^{\#\#}$ and $b_{[1]}^{\#\#} \leq b_{[2]}^{\#\#} \leq \ldots \leq b_{[|\Pi|-1]}^{\#\#}$.

Step 4: Do the following to obtain a lower bound on completion time of an unscheduled job i when placed in positions $2, 3, \ldots, (r'+1)$, following σ:

{

Step 4.1: Set

$$\text{LB}C_w(\sigma[1]i,m) = C'(\sigma,1) + \max\{d_{ai}; d_a^* + b_i^*\} + \sum_{j=1}^{m} p_{ij}. \tag{7}$$

Step 4.2: Compute

$$\begin{aligned} \text{LB}C_w(\sigma[1]\ldots[r']i,m) = \; & C'(\sigma,1) + \max\{d_a^* \\ & + \max\{\sum_{r''=1}^{r'-1} d_{[r'']}^{**}, \sum_{r''=1}^{r'-1} b_{[r'']}^{**}\} + b_i^*; \\ & d_a^* + \sum_{r''=1}^{r'} d_{[r'']}^{\#\#}; \sum_{r''=1}^{r'} b_{[r'']}^{\#\#} + b_i^*; d_{ai}\} \\ & + \sum_{j=1}^{m} p_{ij}, \text{ for } r' = 2,3,\ldots,(|\Pi|-1). \end{aligned} \tag{8}$$

}

It is important to note that the optimal solution (in the class of permutation sequences) of a general flowshop is a lower bound to a no-wait flowshop, with the same objective function. Hence we include our lower bounds to a general permutation flowshop (denoted by the variable $\text{LB}q(\sigma[1][2]\ldots[r'],m)$ as defined in Equation (6) of article [1]) to the set of new bounds proposed in this work.

Step 5: Compute a lower bound on the weighted flowtime of job i, when placed in position r $\left(\text{given by} \left(a'_{ir}\right)^f\right)$ using,

$$\left(a'_{i1}\right)^f = C_w(\sigma_{[n']}i,m) \times h_i,$$

and

$$\left(a'_{ir}\right)^f = \max\{\mathrm{LB}q(\sigma[1]\ldots[r],m); \mathrm{LB}C_w(\sigma[1]\ldots[r-1]i,m)\} \times h_i, \quad \text{for } r = 2,3,\ldots,|\Pi|. \tag{9}$$

Step 6: Compute a lower bound on the weighted tardiness of job i, when placed in position r $\left(\text{given } by \left(a'_{ir}\right)^t\right)$ using,

$$\left(a'_{i1}\right)^t = \max\{C_w(\sigma_{[n']}i,m) - D_i; 0\} \times \tau_i,$$

and

$$\left(a'_{ir}\right)^t = \max\{\max\{\mathrm{LB}q(\sigma[1]\ldots[r],m); \mathrm{LB}C_w(\sigma[1]\ldots[r-1]i,m)\} - D_i; 0\} \times \tau_i, \quad \text{for } r = 2,3,\ldots,|\Pi|. \tag{10}$$

3.2 *Proposed upper bound on completion time to obtain a lower bound on earliness*

We propose an upper bound on completion time of every unscheduled job placed in every unscheduled position to obtain a lower bound on weighted earliness of jobs.

Let $\mathrm{UB}C_w(\sigma[1],m)$ denote an upper bound on the completion time of a job placed immediately following σ and $\mathrm{UB}C_w(\sigma[1]\ldots[r'],m)$ denote an upper bound on the completion time of the job found in position r' following σ.

Step 7: Do the following to compute an upper bound on the completion time of job i when placed in various unscheduled positions:

Step 7.1: Find the maximum delay among the unscheduled jobs following last job a in σ (denoted by d^o_a), maximum delay for every unscheduled job $i' \in \Pi$ with respect to other unscheduled jobs following job i' excluding job i (denoted by $d^o_{i'}$), and maximum delay for every unscheduled job $i' \in \Pi$ with respect to other unscheduled jobs following job i' including job i (denoted by $d^{\&}_{i'}$):

$$d^o_a = \max_{i' \in \Pi - \{i\}} \{d_{ai'}\},$$
$$d^o_{i'} = \max_{\substack{i'' \in \Pi - \{i\} \\ i'' \neq i'}} \{d_{i'i''}\} \text{ and } d^{\&}_{i'} = \max_{\substack{i'' \in \Pi \\ i'' \neq i'}} \{d_{i'i''}\}, \forall i' \in \Pi - \{i\}. \tag{11}$$

Step 7.2: Find the maximum delay among the unscheduled jobs preceding job i (denoted by b^o_i), maximum delay for every unscheduled job i' with respect to other unscheduled jobs preceding job i' excluding job i (denoted by $b^o_{i'}$), and maximum delay for every unscheduled job i' with respect to last job a in partial sequence σ and other unscheduled jobs preceding job i' (denoted by $b^{\&}_{i'}$):

$$b^o_i = \max_{i' \in \Pi - \{i\}} \{d_{i'i}\},$$
$$b^o_{i'} = \max_{\substack{i'' \in \Pi - \{i\} \\ i'' \neq i'}} \{d_{i''i'}\}, \forall i' \in \Pi - \{i\} \text{ and}$$
$$b^{\&}_{i'} = \max_{\substack{i'' \in \Pi - \{i\} \\ i'' \neq i'}} \{d_{i''i'}; d_{ai'}\}, \forall i' \in \Pi - \{i\} \tag{12}$$

Step 8: Sort the values computed in Step 7 in the non-increasing order, as given below.

Let $d^{oo}_{[r']}$ and $b^{oo}_{[r']}$ denote the ordered set of $d^o_{i'}$'s and $b^o_{i'}$'s respectively, such that $d^{oo}_{[1]} \geq d^{oo}_{[2]} \geq \ldots \geq d^{oo}_{[|\Pi|-1]}$ and $b^{oo}_{[1]} \geq b^{oo}_{[2]} \geq \ldots \geq b^{oo}_{[|\Pi|-1]}$.

Similarly, $d^{\&\&}_{[r']}$ and $b^{\&\&}_{[r']}$ denote the ordered set of $d^{\&}_{i'}$'s and $b^{\&}_{i'}$'s respectively, such that

$$d^{\&\&}_{[1]} \geq d^{\&\&}_{[2]} \geq \ldots \geq d^{\&\&}_{[|\Pi|-1]}$$

and

$$b^{\&\&}_{[1]} \geq b^{\&\&}_{[2]} \geq \ldots \geq b^{\&\&}_{[|\Pi|-1]}.$$

Step 9: Do the following to obtain an upper bound on completion time of job i placed in various positions $2,3,\ldots,(r'+1)$, following σ:

{

Step 9.1: Set

$$\mathrm{UB}C_w(\sigma[1]i,m) = C'(\sigma,1) + d^o_a + b^o_i + \sum_{j=1}^{m} p_{ij}. \tag{13}$$

Step 9.2: Set

$$\mathrm{UB}C_w(\sigma[1]\ldots[r']i,m) = C'(\sigma,1) + \min\{d^o_a + \min\{\sum_{r''=1}^{r'-1} d^{oo}_{[r'']}, \sum_{r''=1}^{r'-1} b^{oo}_{[r'']}\} + b^o_i; d^o_a + \sum_{r''=1}^{r'} d^{\&\&}_{[r'']}; \sum_{r''=1}^{r'} b^{\&\&}_{[r'']} + b^o_i\} + \sum_{j=1}^{m} p_{ij}, \text{ for } r' = 2,3,\ldots,(|\Pi|-1). \tag{14}$$

}

Step 10: Compute a lower bound on the weighted earliness of job i, when placed in position r $\left(\text{given } by \left(a'_{ir}\right)^e\right)$ using,

$$\left(a'_{i1}\right)^e = \max\{D_i - C_w(\sigma_{[n']}i,m); 0\} \times e_i,$$

and

$$\left(a'_{ir}\right)^{e} = \max\{D_i - \mathrm{UB}C_w(\sigma[1]\ldots[r-1]i, m); 0\} \times e_i, \quad \text{for } r = 2, 3, \ldots, |\Pi|. \tag{15}$$

3.3 *Branch-and-bound algorithm using the proposed bounds - a brief review*

We develop a best-first branch and bound algorithm using lower bounds proposed in preceding Section 3.1 and Section 3.2. While a step-by-step description is presented in our paper [1], we provide here a short review of the algorithm for the sake of completion. For every active node N with an associated partial sequence σ in the branch-and-bound tree, we compute a lower bound on the respective objective function of every unscheduled job $i \in \Pi$ placed in every unscheduled position $r \in \{1, 2, \ldots n''\}$, resulting in a $n'' \times n''$ matrix. This square matrix is then solved as an assignment problem using labeling algorithm (as described in [40]). The solution obtained corresponds to an optimal assignment of unscheduled jobs to unscheduled positions, and the objective value corresponds to a lower bound which we denote as $Z'(\Pi)$. The overall lower bound on a node N is then given by $Z(\mathrm{N}) = Z'(\sigma) + Z'(\Pi)$.

We also employ suitable upper bounds to fathom the nodes and a node N' is considered active only when $Z(N') <$ heuristic upper bound. Likewise, we maintain a list of active nodes and enumerate the branch-and-bound tree from this list sorted in the non-decreasing order of objective function values.

Please refer to Appendix-A in [41] for the detailed step-by-step procedure of the heuristic methods employed. While the heuristic procedure depicted pertains to general $m-$machine flowshops over maximum flowtime/ lateness objectives, it is relatively straight-forward to employ the heuristic schemes for linear composite function under no-wait setting. The details of the heuristics are not provided here in order to limit the paper size and to confine to the main theme of our work.

3.4 *Proofs for the proposed lower bounds and upper bounds*

Recall that $C_w(\sigma_{[n']}b, m)$ denote the completion time of job b (when appended to the last job in σ) on machine m and $C_w(\sigma_{[n']}bi, m)$ denote the completion time of job i on machine m, when appended to σb, under the no-wait condition.

We have

$$C_w(\sigma_{[n']}bi, m) = C'(\sigma, 1) + d_{ab} + d_{bi} + \sum_{j=1}^{m} p_{ij} \tag{16}$$

We have from Equation (7) (omitting d_{ai} and without foregoing the validity of the proof)

$$\mathrm{LB}C_w(\sigma[1]i, m) = C'(\sigma, 1) + d^*_a + b^*_i + \sum_{j=1}^{m} p_{ij}. \tag{17}$$

Comparing Equations (16) and (17), it is evident that

$$\mathrm{LB}C_w(\sigma[1]i, m) \le C_w(\sigma_{[n']}bi, m) \tag{18}$$

because $b^*_i = \min_{i'' \in \Pi - \{i\}} \{d_{i''i}\}$ and $d^*_a = \min_{i' \in \Pi - \{i\}} \{d_{ai'}\}$.

In order to ensure that $\mathrm{LB}C_w(\sigma[1]i, m) \ge C_w(\sigma_{[n']}i, m)$, the term d_{ai} is introduced in Equation (7).

Let us now consider the next case wherein job c is scheduled after job b, and job i is scheduled thereafter.

We have

$$C_w(\sigma_{[n']}bci, m) = C'(\sigma, 1) + d_{ab} + d_{bc} + d_{ci} + \sum_{j=1}^{m} p_{ij} \tag{19}$$

From Equation (8) with $r' = 2$, we have (omitting d_{ai} and without foregoing the validity of the proof)

$$\begin{aligned}\mathrm{LB}C_w(\sigma[1][2]i, m) = & C'(\sigma, 1) + \max\{d^*_a + \max\{d^{**}_{[1]}; b^{**}_{[1]}\} \\ & + b^*_i; d^*_a + d^{\#\#}_{[1]} + d^{\#\#}_{[2]}; \\ & b^{\#\#}_{[1]} + b^{\#\#}_{[2]} + b^*_i\} + \sum_{j=1}^{m} p_{ij}.\end{aligned} \tag{20}$$

Equation (20) can be written as the maximum of four equations which are as follows:

$$\mathrm{LB}C'_w(\sigma[1][2]i, m) = C'(\sigma, 1) + d^*_a + d^{**}_{[1]} + b^*_i + \sum_{j=1}^{m} p_{ij}, \tag{21}$$

$$\mathrm{LB}C''_w(\sigma[1][2]i, m) = C'(\sigma, 1) + d^*_a + b^{**}_{[1]} + b^*_i + \sum_{j=1}^{m} p_{ij}, \tag{22}$$

$$\mathrm{LB}C'''_w(\sigma[1][2]i, m) = C'(\sigma, 1) + d^*_a + d^{\#\#}_{[1]} + d^{\#\#}_{[2]} + \sum_{j=1}^{m} p_{ij}, \tag{23}$$

and

$$\mathrm{LBC}''''_w(\sigma[1][2]i,m) = C'(\sigma,1) + b^{\#\#}_{[1]} + b^{\#\#}_{[2]} + b^*_i + \sum_{j=1}^{m} p_{ij}. \tag{24}$$

Comparing Equations (19) with (21), it is evident that

$$\mathrm{LBC}'_w(\sigma[1][2]i,m) \le C_w(\sigma_{[n']}bci,m) \tag{25}$$

This is so because $b^*_i = \min_{i''\in\Pi-\{i\}}\{d_{i''i}\}$, $d^*_a = \min_{i'\in\Pi-\{i\}}\{d_{ai'}\}$, and $d^{**}_{[1]} = \min_{i'\in\Pi-\{i\}}\{\min_{\substack{i''\in\Pi-\{i\}\\ i''\neq i'}}\{d_{i'i''}\}\}$.

Likewise, it is evident that

$\mathrm{LBC}''_w(\sigma[1][2]i,m) \le C_w(\sigma_{[n']}bci,m)$ because $b^{**}_{[1]} = \min_{i'\in\Pi-\{i\}}\{\min_{\substack{i''\in\Pi-\{i\}\\ i''\neq i'}}\{d_{i''i'}\}\}$,

$\mathrm{LBC}'''_w(\sigma[1][2]i,m) \le C_w(\sigma_{[n']}bci,m)$ because $d^{\#\#}_{[1]} = \min_{i'\in\Pi-\{i\}}\{\min_{\substack{i''\in\Pi\\ i''\neq i'}}\{d_{i'i''}\}\}$,

and $\mathrm{LBC}''''_w(\sigma[1][2]i,m) \le C_w(\sigma_{[n']}bci,m)$ because $b^{\#\#}_{[1]} = \min_{i'\in\Pi-\{i\}}\{\min_{\substack{i''\in\Pi-\{i\}\\ i''\neq i'}}\{d_{i''i'};d_{ai'}\}\}$.

By extending this argument, it is therefore evident that $\mathrm{LBC}_w(\sigma[1]\ldots[r']i,m)$ holds for $r' = 2,3,\ldots,(|\Pi|-1)$ in that $\mathrm{LBC}_w(\sigma[1]\ldots[r']i,m) \le C_w(\sigma[1]\ldots[r']i,m)$. It is also clear, by virtue of respective non-decreasing ordering of $d^{**}_{[r']}$'s, $b^{**}_{[r']}$'s, $d^{\#\#}_{[r']}$'s, and $b^{\#\#}_{[r']}$'s, that

$\mathrm{LBC}_w(\sigma[1]\ldots[r'-1]i,m) \le \mathrm{LBC}_w(\sigma[1]\ldots[r']i,m)$.

We finally need to prove that in any partial or complete sequence $\mathrm{UBC}_w(\sigma[1]\ldots[r']i,m) \ge C_w(\sigma[1]\ldots[r']i,m)$ for $r' = 1,2,\ldots,(|\Pi|-1)$, with all jobs scheduled under the no-wait condition.

Compare Equations (13) and (16). It is evident that

$$\mathrm{UBC}_w(\sigma[1]i,m) \ge C_w(\sigma_{[n']}bi,m)$$

This is so because $d^o_a = \max_{i'\in\Pi-\{i\}}\{d_{ai'}\}$ and $b^o_i = \max_{i'\in\Pi-\{i\}}\{d_{i'i}\}$.

Now comparing Equations (14) and (19). From Equation (14) with $r' = 2$ we have

$$\begin{aligned}\mathrm{UBC}_w(\sigma[1][2]i,m) = C'(\sigma,1) + \min\{&d^o_a + \max\{d^{oo}_{[1]};b^{oo}_{[1]}\} + b^o_i; d^o_a + d^{\&\&}_{[1]} + d^{\&\&}_{[2]};\\ &b^{\&\&}_{[1]} + b^{\&\&}_{[2]} + b^o_i\} + \sum_{j=1}^{m} p_{ij}\end{aligned} \tag{26}$$

Equation (26) can be written as the minimum of four equations which are given by, follows:

$$\mathrm{UBC}'_w(\sigma[1][2]i,m) = C'(\sigma,1) + d^o_a + d^{oo}_{[1]} + b^o_i + \sum_{j=1}^{m} p_{ij}, \tag{27}$$

$$\mathrm{UBC}''_w(\sigma[1][2]i,m) = C'(\sigma,1) + d^o_a + b^{oo}_{[1]} + b^o_i + \sum_{j=1}^{m} p_{ij}, \tag{28}$$

$$\mathrm{UBC}'''_w(\sigma[1][2]i,m) = C'(\sigma,1) + d^o_a + d^{\&\&}_{[1]} + d^{\&\&}_{[2]} + \sum_{j=1}^{m} p_{ij}, \tag{29}$$

and

$$\mathrm{UBC}''''_w(\sigma[1][2]i,m) = C'(\sigma,1) + b^{\&\&}_{[1]} + b^{\&\&}_{[2]} + b^o_i + \sum_{j=1}^{m} p_{ij} \tag{30}$$

Comparing Equations (19) with Equation (27), it is evident that

$$\mathrm{UBC}'_w(\sigma[1][2]i,m) \ge C_w(\sigma_{[n']}bci,m). \tag{31}$$

This is so because $d^{oo}_{[1]} = \max_{i'\in\Pi-\{i\}}\{\max_{\substack{i''\in\Pi-\{i\}\\ i''\neq i'}}\{d_{i'i''}\}\}$, $d^o_a = \max_{i'\in\Pi-\{i\}}\{d_{ai'}\}$, and $b^o_i = \max_{i''\in\Pi-\{i\}}\{d_{i''i}\}$.

Likewise, it can be seen from Equations (19) and (28) that $\mathrm{UBC}''_w(\sigma[1][2]i,m) \ge C_w(\sigma_{[n']}bci,m)$ and so on up to $\mathrm{UBC}''''_w(\sigma[1][2]i,m) \ge C_w(\sigma_{[n']}bci,m)$.

By extending this argument, it is therefore evident that $\mathrm{UBC}_w(\sigma[1]\ldots[r']i,m)$ holds for $r' = 2,3,\ldots,(|\Pi|-1)$ in that $\mathrm{UBC}_w(\sigma[1]\ldots[r']i,m) \ge C_w(\sigma[1]\ldots[r']i,m)$. It is also clear, by virtue of respective non-increasing ordering of $d^{oo}_{[r']}$'s, $b^{oo}_{[r']}$'s, $d^{\&\&}_{[r']}$'s, and $b^{\&\&}_{[r']}$'s, that

$\mathrm{UBC}_w(\sigma[1]\ldots[r'-1]i,m) \ge \mathrm{UBC}_w(\sigma[1]\ldots[r']i,m)$.

4. Computational evaluation of the proposed algorithms

We generate 30 problem instances for every set $n\times m$ where number of jobs $n = \{9,10,12,14,16,18\}$, and number of machines $m = \{5,10,15,20\}$. All the input data are sampled from a rectangular distribution where processing times of jobs range between 1 to 99, due dates range between $\sum_{j=1}^{m} t_{ij}$ to $\sum_{j=1}^{m} t_{ij} + 50(n-1)$, and the weights for flowtime, earliness and tardiness range between 11 to 20, 21 to 40 and 41 to 80 respectively. The rationale behind this parameter setting is well explained in the computational evaluation section of our paper on general $m-$machine permutation flowshop [1], and we report the numerical results of our no-wait

branch-and-bound algorithm by solving the same instances in C++.

The performance of the algorithms is evaluated using the metrics such as minimum, maximum and mean number of active nodes explored in the branch-and-bound tree, and we also present $p^{\star}$ the proportion of problem instances for which the number of active nodes created in the scheduling tree is less than or equal to the mean number of nodes with respect to every problem set. We also note the mean CPU time for a problem set in seconds.

The numerical results are presented to be indicative of the implementability of the proposed algorithms and to illustrate the metrics for different problem sizes. The focus of this work is to develop an exact generic algorithm to handle a variety of weighted objectives in a no-wait flowshop. We believe analyzing which heursitic algorithm(s) performs closer to our exact algorithm, under which conditions, can be an interesting avenue for future work.

We present the metrics for the minimization of sum of weighted flowtime and sum of weighted tardiness of jobs in Table 1 and Table 2 respectively. Following these are the results of the minimization of sum of weighted flowtime and weighted tardiness, and the sum of weighted flowtime, weighted tardiness and weighted earliness of jobs in Table 3 and Table 4 respectively. It is important to note that the algorithms are forced to terminate when the count of evaluated active nodes exceeds 20×10^5 or when the execution time exceeds 6 hours (whichever is reached earlier). While columns 1 and 2 in the tables indicate the number of jobs and machines in the problem set, column 3 indicate the number of problems out of 30 instances solved to optimality without exceeding the node and CPU time limit. Column p^* for a problem set indicates the proportion of problem instances for which the count of nodes enumerated to reach optimality does not exceed the mean number of nodes reported for that problem set. For example, in the first row of Table 1, $p^* = 0.6$ indicates that, out of 30 instances evaluated, 18 instances enumerated less than or equal to 165 nodes to acheive the optimal solution. A larger $p^{\star}$ value for a problem set indicates that only a small proportion of instances demand computational effort larger than the mean number of nodes to reach optimality. Our proposed algorithms are able to solve up to 18 jobs with respect to every objective considered and larger p^* values in the tables indicate that our proposed branch-and-bound algorithms are performing quite well in terms of number of nodes enumerated.

5. Summary

Our paper is possibly the first attempt to consider a combination of weighted objectives and develop exact algorithms for a no-wait flowshop manufacturing system. The main contribution of our present work is to exploit the features of a no-wait flowshop and extend our novel idea,

Table 1. Computational experience for the sum of weighted flowtime of jobs.

Number of jobs	Number of machines	Number of problem instances	Number of active nodes created			p^*	Mean execution time (in secs)
			Mean	Minimum	Maximum		
9	5	30	165	7	524	0.60	0.13
	10	30	289	67	836	0.60	0.40
	15	30	265	32	866	0.63	0.57
	20	30	349	65	937	0.57	1.06
10	5	30	489	9	1874	0.67	0.50
	10	30	996	225	4198	0.63	1.46
	15	30	789	81	2558	0.57	2.22
	20	30	1315	203	5702	0.67	4.48
12	5	30	4225	112	33970	0.83	7.47
	10	30	7934	1491	42973	0.63	22.14
	15	30	9781	1041	33311	0.57	42.28
	20	30	12151	841	48991	0.67	75.59
14	5	30	51027	905	440134	0.80	546.54
	10	30	118960	9296	466720	0.70	1493.54
	15	30	112202	8653	313703	0.57	1385.87
	20	30	156359	13696	617716	0.63	2862.91
16	5	27	309821	11783	1326560	0.67	10590.64
	10	12	416278	86060	852489	0.50	12180.62
	15	13	423731	80561	831727	0.46	14234.22
18	5	5	183067	11112	508905	0.60	4222.76

Table 2. Computational experience for the sum of weighted tardiness of jobs.

Number of jobs	Number of machines	Number of problem instances	Number of active nodes created			p^*	Mean execution time (in secs)
			Mean	Minimum	Maximum		
9	5	30	138	29	332	0.53	0.15
	10	30	266	78	827	0.63	0.39
	15	30	247	65	634	0.60	0.59
	20	30	296	56	780	0.57	1.03
10	5	30	417	31	937	0.60	0.49
	10	30	817	110	2137	0.53	1.63
	15	30	837	169	2440	0.67	2.37
	20	30	885	149	2005	0.50	3.98
12	5	30	3507	186	10470	0.57	6.80
	10	30	9117	1498	47474	0.70	27.83
	15	30	8382	1708	31277	0.70	40.59
	20	30	9802	1655	32096	0.67	61.03
14	5	30	33840	3322	132855	0.60	174.18
	10	30	94144	108	309342	0.53	991.73
	15	30	134692	7601	730091	0.80	2607.80
	20	30	117860	19381	367520	0.67	2001.44
16	5	28	303883	28754	1105764	0.68	7849.77
	10	15	438192	66641	848539	0.47	14541.19
	15	11	439675	197533	778267	0.64	15639.84
18	5	5	422913	53848	763841	0.60	13874.08

Table 3. Computational experience for the sum of weighted flowtime and weighted tardiness of jobs.

Number of jobs	Number of machines	Number of problem instances	Number of active nodes created			p^*	Mean execution time (in secs)
			Mean	Minimum	Maximum		
9	5	30	137	21	378	0.60	0.13
	10	30	248	87	789	0.63	0.38
	15	30	256	66	767	0.67	0.59
	20	30	300	58	635	0.57	1.03
10	5	30	420	42	1033	0.63	0.58
	10	30	742	158	2407	0.57	1.53
	15	30	735	210	2234	0.70	2.41
	20	30	870	144	2098	0.57	4.23
12	5	30	3225	265	10726	0.67	7.29
	10	30	7208	1598	23127	0.63	25.89
	15	30	8431	2178	26274	0.67	45.06
	20	30	8905	1729	28602	0.70	67.40
14	5	30	35100	3454	187903	0.63	206.98
	10	30	104068	14814	298824	0.60	1122.58
	15	30	107179	7809	375171	0.60	1526.84
	20	30	119685	23257	464947	0.63	2133.34
16	5	29	385025	16837	1832227	0.69	16272.08
	10	13	352036	54186	751332	0.69	9791.92
	15	11	443938	200940	784940	0.55	14649.56
18	5	5	421638	104906	747667	0.60	13938.91

Table 4. Computational experience for the sum of weighted flowtime, weighted tardiness and weighted earliness of jobs.

Number of jobs	Number of machines	Number of problem instances	Number of active nodes created			p^*	Mean execution time (in secs)
			Mean	Minimum	Maximum		
9	5	30	157	24	463	0.60	0.19
	10	30	275	67	946	0.73	0.45
	15	30	226	70	619	0.60	0.60
	20	30	268	53	596	0.63	1.04
10	5	30	415	38	903	0.43	0.62
	10	30	827	88	2710	0.63	1.77
	15	30	744	187	1956	0.60	2.44
	20	30	866	162	1977	0.60	4.28
12	5	30	3008	209	7448	0.60	8.46
	10	30	7452	1428	20433	0.53	28.22
	15	30	8423	1433	42853	0.67	46.22
	20	30	8713	1524	23901	0.57	67.79
14	5	30	29930	3353	146658	0.63	181.04
	10	30	89278	10437	392632	0.60	1045.71
	15	30	98713	6137	275186	0.67	1361.89
	20	30	123995	20617	293450	0.60	2175.56
16	5	30	300637	12288	1362038	0.80	10367.15
	10	17	438178	65495	1258061	0.65	15152.03
	15	13	442354	238434	657710	0.54	14161.86
18	5	5	407541	56097	772154	0.60	14153.66

presented in [1], of ensuring one-to-one matching of the lower bound of a job with its corresponding due-date and weight. This one-to-one correspondence enables our lower bounds to be applied to a generic set of composite functions such as, but not limited to, makespan, weighted flowtime, weighted tardiness and weighted earliness of jobs. In addition, solving an assignment model to obtain an overall lower-bound renders the bound quite tight. Our experimental investigations also show that our proposed branch-and-bound algorithms perform quite well in the no-wait setting, however we do not claim superiority over any existing algorithms.

For future works, we propose developing dominance rules for the branch-and-bound algorithm to reduce the number of nodes evaluated. There is also potential to develop more efficient lower bounds on weighted earliness and one can also consider developing exact algorithms to obtain a Pareto-optimal front for multiple objectives in no-wait flowshops. Additionally, the use of more efficient heuristics can be studied and adapted to the problem as an upper bound for the proposed branch-and-bound algorithms.

Acknowledgements

We are grateful to the referees and the corresponding editor for their constructive comments and suggestions to improve our earlier version of the paper.

References

[1] Madhushini N, Rajendran C, and 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. *J. Oper. Res. Soc.* 60(7): 991–1004

[2] Reddi S S and Ramamoorthy C V 1972 On the flow-shop sequencing problem with no wait in process. *Oper. Res. Q. (1970–1977)* 23(3): 323

[3] Van Deman J M, and Baker K R 1974 Minimizing mean flowtime in the flow shop with no intermediate queues. *A I I E Trans.* 6(1): 28–34

[4] Van der Veen J A, and van Dal R 1991 Solvable cases of the no-wait flow-shop scheduling problem. *J. Oper. Res. Soc.* 42(11): 971

[5] Hall N G and Sriskandarajah C 1996 A survey of machine scheduling problems with blocking and no-wait in process. *Oper. Res.* 44(3): 510–525

[6] Takano M I and Nagano M S 2017 A branch-and-bound method to minimize the makespan in a permutation flow shop with blocking and setup times. *Cogent Eng.* 4(1): 1389638

[7] Samarghandi H and Behroozi M 2016 An enumeration algorithm for the no-wait flow shop problem with due date constraints. *IFAC-PapersOnLine* 49(12): 1803–1808

[8] Samarghandi H and Behroozi M 2017 On the exact solution of the no-wait flow shop problem with due date constraints. *Comput. Oper. Res.* 81: 141–159

[9] Ying K C, Lu C C, and Lin S W 2018 Improved exact methods for solving no-wait flowshop scheduling problems with due date constraints. *IEEE Access* 6: 30702–30713
[10] Sahni S and Cho Y 1979 Complexity of scheduling shops with no wait in process. *Math. Oper. Res.* 4(4): 448–457
[11] Kubiak W, Sriskandarajah C, and Zaras K 1991 A note on the complexity of openshop scheduling problems. *INFOR Inf. Syst. Oper. Res.* 29(4): 284–294
[12] Adiri I and Pohoryles D 1982 Flowshop/no-idle or no-wait scheduling to minimize the sum of completion times. *Nav. Res. Logist. Q.* 29(3): 495–504
[13] Bonney M C and Gundry S W 1976 Solutions to the constrained flowshop sequencing problem. *J. Oper. Res. Soc.* 27(4): 869–883
[14] Rajendran C and Chaudhuri D 1990 Heuristic algorithms for continuous flow-shop problem. *Nav. Res. Logist.* 37(5): 695–705
[15] Gangadharan R and Rajendran C 1993 Heuristic algorithms for scheduling in the no-wait flowshop. *Int. J. Prod. Econ.* 32(3): 285–290
[16] Rajendran C 1993 Heuristic algorithm for scheduling in a flowshop to minimize total flowtime. *Int. J. Prod. Econ.* 29(1): 65–73
[17] Chen C L, Neppalli R V, and Aljaber N 1996 Genetic algorithms applied to the continuous flow shop problem. *Comput. Ind. Eng.* 30(4): 919–929
[18] Bertolissi E 2000 Heuristic algorithm for scheduling in the no-wait flow-shop. *J. Mater. Process. Technol.* 107(1): 459–465
[19] Aldowaisan T and Allahverdi A 2004 New heuristics for m-machine no-wait flowshop to minimize total completion time. *Omega* 32(5): 345–352
[20] Framinan J M, Nagano M S, and Moccellin J V 2010 An efficient heuristic for total flowtime minimisation in no-wait flowshops. *Int. J. Adv. Manuf. Technol.* 46(9): 1049–1057
[21] Chaudhry I A, Ahmed R, and Khan A M 2014 Genetic algorithm to minimize flowtime in a no-wait flowshop scheduling problem. *IOP Conf. Ser. Mater. Sci. Eng.* 65: 012007
[22] Ding J, Song S, Zhang R, and Wu C 2014 Minimizing makespan for a no-wait flowshop using tabu mechanism improved iterated greedy algorithm. In *2014 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, pp. 1906–1911
[23] Laha D and Sapkal S U 2014 An improved heuristic to minimize total flow time for scheduling in the m-machine no-wait flow shop. *Comput. Ind. Eng.* 67: 36–43
[24] Zhu X and Li X 2014 Iterative search method for total flowtime minimization no-wait flowshop problem. *Int. J. Mach. Learn. Cybern.* 6(5): 747–761
[25] Aldowaisan T and Allahverdi Ali 1998 Total flowtime in no-wait flowshops with separated setup times. *Comput. Oper. Res.* 25(9): 757–765
[26] Aldowaisan T 2001 A new heuristic and dominance relations for no-wait flowshops with setups. *Comput. Oper. Res.* 28(6): 563–584
[27] Allahverdi A and Aydilek H 2014 Total completion time with makespan constraint in no-wait flowshops with setup times. *Eur. J. Oper. Res.* 238(3): 724–734
[28] Nagano M S, Miyata H H, and Arajo D C 2015 A constructive heuristic for total flowtime minimization in a no-wait flowshop with sequence-dependent setup times. *J. Manuf. Syst.* 36: 224–230
[29] Cheng C Y, Ying K C, Li S F, and Hsieh Y C 2019 Minimizing makespan in mixed no-wait flowshops with sequence-dependent setup times. *Comput. Ind. Eng.* 130:338–347
[30] Allahverdi A, Aydilek H, and Aydilek A 2020 No-wait flowshop scheduling problem with separate setup times to minimize total tardiness subject to makespan. *Appl. Math. Comput.* 365: 124688
[31] Allahverdi A 2016 A survey of scheduling problems with no-wait in process. *Eur. J. Oper. Res.* 255(3): 665–686
[32] Nagano M S and Miyata H H 2016 Review and classification of constructive heuristics mechanisms for no-wait flow shop problem. *Int. J. Adv. Manuf. Technol.* 86(5-8): 2161–2174
[33] Arabameri S and Salmasi N 2013 Minimization of weighted earliness and tardiness for no-wait sequence-dependent setup times flowshop scheduling problem. *Comput. Ind. Eng.* 64(4): 902–916
[34] Samarghandi H 2015 A particle swarm optimisation for the no-wait flow shop problem with due date constraints. *Int. J. Prod. Res.* 53(9): 2853–2870
[35] Rahimi-Vahed A R, Rabbani M, Javadi B, and Tavakkoli-Moghaddam R 2008 A multi-objective scatter search for a bi-criteria no-wait flow shop scheduling problem. *Eng. Optim.* 40(4): 331–346
[36] Aldowaisan T and Allahverdi A 2012 Minimizing total tardiness in no-wait flowshops. *Found. Comput. Decis. Sci.* 37(3): 149–162
[37] Liu G, Song S, and Wu C 2013 Some heuristics for no-wait flowshops with total tardiness criterion. *Comput. Oper. Res.* 40(2): 521–525
[38] Gupta U and Kumar S 2015 Minimization of weighted sum of total tardiness and make span in no wait flow shop scheduling using different heuristic algorithm: a review. *Int. J. Adv. Eng. Sci.* 5(4)
[39] Lin S W, Lu C C, and Ying K C 2018 Minimizing the sum of makespan and total weighted tardiness in a no-wait flowshop. *IEEE Access* 6: 78666–78677
[40] Lotfi V 1989 A labeling algorithm to solve the assignment problem. *Comput. Oper. Res.* 16(5): 397–408
[41] Madhushini N and Rajendran Chandrasekharan 2011 Branch-and-bound algorithms for scheduling in an m-machine permutation flowshop with a single objective and with multiple objectives. *Eur. J. Ind. Eng.* 5(4): 361–387