Journal of Industrial and Management Optimization

https://www.aimspress.com/journal/jimo

JIMO, 22(10): 5275–5307.
DOI: 10.3934/jimo.2026182
Received: 01 August 2026
Revised: 30 August 2026
Accepted: 11 September 2026
Published: 21 September 2026

Research article

The no-wait flow shop problem with makespan objective and learning effects

Jia-Ming Gu, Lin Lin* and Ji-Bo Wang

School of Computer, Shenyang Aerospace University, Shenyang 110136, P.R. China

* **Correspondence:** Email: linlin@sau.edu.cn.

Abstract: The annealing operation of cold rolled coils follows a no-wait constraint between heating and rapid cooling, and manual operations generate learning effects on job processing times. This paper abstracts the annealing process of cold rolled coils as a two-machine no-wait flow shop scheduling problem with learning effects in which the actual processing time of a job depends on the sum of the logarithmic basic processing times of previous jobs. The optimization objective of this problem is minimizing the makespan to improve production efficiency for cold rolled coil annealing. We propose polynomial algorithms to solve two special dominating machine constraints and two special fixed processing times on the machine. For the problem, we propose a worst-case approximation ratio, a branch-and-bound algorithm and some heuristic algorithms. Computational results demonstrate that the genetic algorithm performs well for small-scale instances, while the NEH (Nawaz-Enscore-Ham) algorithm achieves better performance for large-scale instances.

Keywords: no-wait; flow shop scheduling; learning effect; branch-and-bound; heuristic
Mathematics Subject Classification: 90B35

1. Introduction

Cold rolled products are processed from hot rolled coils through a series of operations. Compared with hot rolled coils, cold rolled coils feature high dimensional accuracy, greater hardness, and a superior surface finish. As a critical procedure in cold rolling production, annealing consists of the heating operation and the rapid cooling operation with a no-wait constraint between them. That is, the coils must undergo rapid cooling immediately after heating (see Wu and Chen [1]). Moreover, handling abnormal working conditions and quality sampling inspections requires manual participation, which leads to learning effects (LE) in the production process. This paper abstracts the annealing process of cold rolled coils as a two-machine no-wait flow shop scheduling problem with learning effects.

In the classical scheduling models, it is assumed that processing times of jobs are fixed constants. However, there are some real-life settings in which the processing times of jobs are variable due to the LE

(e.g., Lu et al. [2], Zhang et al. [3], Jiang et al. [4], Yin [5], Azzouz et al. [6], Wu et al. [7]). More recently, Wang et al. [8] considered single-machine release date scheduling with the position-weighted LE. For the total completion time minimization, they proposed the branch-and-bound and heuristic algorithms. Zhao [9] studied single-machine setup times scheduling with general truncated LE. Ma et al. [10] considered online scheduling with position-based LE. Jiang et al. [11] addressed Seru scheduling with LE and multiple due window assignment. Deliktas [12] studied single-machine release time scheduling with LE. For bi-objective (i.e., makespan and total tardiness) optimization, the author proposed two self-adaptive memetic algorithms. Ren et al. [13] considered single-machine delivery times scheduling with LE. Luo [14] studied stochastic single-machine scheduling with LE or deterioration effects. Wang et al. [15] addressed single-machine resource allocation problems with LE. Mor et al. [16] considered single-machine job-rejection scheduling with LE and generalized due dates. Liu et al. [17] studied single-machine due window assignment scheduling with LE, job-rejection, and setup times. Gao et al. [18] addressed single-machine LE scheduling with deteriorating maintenance activity. Zhang et al. [19] studied single-machine resource allocation scheduling with the exponential time-dependent LE. Cheng et al. [20] considered single-machine scheduling with batch production and intermediate delivery. Under the batch-position-based LE of minimizing the makespan, they proposed some solution algorithms. Wang [21] studied due window assignment scheduling problems with the truncated LE and past-sequence-dependent setup times. Song et al. [22] considered the multiple due window assignment problems with LE and deteriorating effects. Cheng et al. [23] studied batch scheduling under energy constraint with truncated LE and aging effects.

In addition, flow shop scheduling plays an important role in modern manufacturing and engineering (see Shabtay and Oron [24], Zhao et al. [25], Sheikh et al. [26], Al-Behadili et al. [27], Wang et al. [28], Jin et al. [29], and Wu et al. [30]). Sun et al. [31] examined the total weighted completion time flow shop problem with positional weighted LE. Lv and Wang [32] addressed flow shop scheduling with LE and deterioration effects. Under the peak power consumption of minimizing the makespan, they proposed some algorithms. Li et al. [33] considered makespan minimization flow shop scheduling with truncated LE. Under the m-machine setting, they showed that the problem is NP-hard. They also proposed some heuristics and a branch-and-bound. Paredes-Astudillo et al. [34] examined flow shop scheduling truncated LE. For makespan minimization, they proposed some solution algorithms (including the simulated annealing and Nawaz-Enscore-Ham algorithms). Lv and Wang [35] studied two-machine flow shop scheduling with release dates and truncated LE. For minimizing the total completion time, they proposed some solution algorithms. Zhang et al. [36] considered flow shop scheduling with DeJong's LE. For makespan minimization, they proposed some algorithms.

No-wait flow shop scheduling is also a wide area which has been studied by many researchers (see Nagano and Miyata [37], Singh et al. [38], Utama et al. [39], Samarghandi and Behroozi [40], Ye et al. [41], Koulamas and Kyparisis [42], and Li et al. [43]). Liu and Feng [44] first considered no-wait flow shop scheduling with LE and convex research allocation. Then Gao et al. [45], and Liu and Jiang [46] studied the same scheduling model as Liu and Feng [44]. Under common due date assignment, they proved that some problems are polynomially solvable. Lv and Wang [47] and Sun et al. [48] addressed no-wait flow shop problems with LE and convex research allocation. Under some due window assignments, they also proved that some problems are polynomially solvable. Gu and Wang [49] considered the no-wait flow shop problem with LE and group technology. For minimizing the total completion time, they proposed some algorithms.

To our knowledge, there are currently few research on no-wait flow shop problems with LE, and they mainly focus on resource allocation and algorithm design. To date, a large number of scheduling studies incorporating LE introduce resource allocation. In such models, decision-makers are allowed to invest extra resources to compress the processing time of jobs, which often brings opportunities to simplify the problem's complexity. Nevertheless, there is a class of practical no-wait flow shop production environments, such as steel annealing and surface treatment workshops, where the resources are fixed in advance. No resource allocation operations can be performed during scheduling. In these fixed resource scenarios, processing times are only affected by the LE accumulated from previous jobs. Up to now, there has been no research on no-wait flow shop problems with LE in the property analysis and exact algorithms. In this paper, we will fill the research gap, and the main contributions of this paper are summarized as follows.

1. This paper studies the two-machine no-wait flow shop scheduling problem with LE based on the sum of the logarithm of processing times, proposes the mathematical expression for the makespan, four polynomial-solvable special cases with dominating machine constraints or fixed processing times on one machine's constraints, and the worst-case ratio of a simple heuristic algorithm.
2. For this problem, a branch-and-bound algorithm is developed. Several classic algorithms including the Nawaz-Enscore-Ham (NEH) algorithm, the simulated annealing algorithm, and the genetic algorithm are adopted to solve the problem.

The paper continues as follows: Section 2 presents the problem. Section 3 gives some basic results. Section 4 proposes a branch-and-bound algorithm. Section 5 presents some heuristic algorithms. Section 6 gives the computational results of the proposed algorithms. Section 7 presents the conclusions.

2. Problem description

The details of two-machine no-wait flow shop problem with LE is described as follows: A set of n jobs $\widetilde{J} = \{J_1, J_2, \ldots, J_n\}$ is processed on two machines M_1 and M_2. All jobs and machines are available at time 0. Each job must not be interrupted during processing. Each job must be processed first on M_1 and then on M_2. Job $J_j \in \widetilde{J}$ consists two operations, $\widetilde{O}_{1j}$ and $\widetilde{O}_{2j}$. The starting time of operation $\widetilde{O}_{2j}$ must be equal to the completion time of operation $\widetilde{O}_{1j}$; this is the no-wait characteristic (see Figure 1). Generally, in the classical position-based learning model and the learning model based on the sum of processing times, the actual processing time of a given job drops to zero precipitously since the number of jobs increases or the normal processing times of jobs are large. Cheng et al. [50] considered a single machine with the sum of logarithmic basic processing times of previous jobs $\widehat{P}_{j[r]} = \widehat{P}_j \left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{[h]}\right)^{\beta}$, where $\widehat{P}_j$ is the basic processing time of J_j, β is a learning factor, and $\widehat{P}_{[h]}$ is the basic processing time of some job is scheduled in the hth position. This model allows the learning process to be subject to the law of diminishing returns by the logarithm function. In this paper, we consider the two-machine flow shop scheduling based on this foundation; i.e., the actual processing time of operation $\widetilde{O}_{ij}$ is

$$\widehat{P}_{ij[r]} = \widehat{P}_{ij} \left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta}, \tag{1}$$

where $\widetilde{O}_{ij}$ is executed at the rth position, $\widehat{P}_{ij}$ is the basic processing time of $\widetilde{O}_{ij}$ and $\widehat{P}_{ij} > 1$, $\beta \leq 0$ is a learning factor (see Biskup [51]), $[h]$ is the hth position in a sequence, and $\widehat{P}_{i[h]}$ is the basic processing time of some job scheduled in the hth position on machine M_i ($i = 1, 2$, $j/r = 1, \ldots, n$).

Let $\widetilde{C}_{ij}$ be the completion time of $\widetilde{O}_{ij}$, and $\widetilde{C}_j = \widetilde{C}_{2j}$ represents the completion time of J_j. The optimization objective is to find a sequence that minimizes $\widetilde{C}_{\max} = \max\{\widetilde{C}_j | j = 1, 2, \ldots, n\}$. We indicate $\widehat{P}_{1j}$ by $\widehat{a}_j$ and $\widehat{P}_{2j}$ by $\widehat{b}_j$, then, for a sequence $\phi = (J_1, J_2, \ldots, J_n)$, we have

$$\begin{aligned}
\widetilde{C}_{\max}(\phi) &= \widetilde{C}_{2n} \\
&= \widetilde{C}_{1n} + \widehat{b}_n \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta} \\
&= \widetilde{C}_{1n-1} + \max\left\{\widehat{b}_{n-1}\left(1 + \sum_{h=1}^{n-2} \ln \widehat{b}_{[h]}\right)^{\beta}, \widehat{a}_n \left(1 + \sum_{h=1}^{n-1} \ln \widehat{a}_{[h]}\right)^{\beta}\right\} + \widehat{b}_n \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta} \\
&= \widehat{a}_1 + \sum_{g=1}^{n-1} \max\left\{\widehat{b}_g \left(1 + \sum_{h=1}^{g-1} \ln \widehat{b}_{[h]}\right)^{\beta}, \widehat{a}_{g+1}\left(1 + \sum_{h=1}^{g} \ln \widehat{a}_{[h]}\right)^{\beta}\right\} \\
&\quad + \widehat{b}_n \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta}.
\end{aligned} \tag{2}$$

Using the three-field notation, this problem can be expressed as

$$F2 \left| no - wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij} \left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta} \right| \widetilde{C}_{\max}.$$

The Gantt chart of the two-machine no-wait flow shop problem is shown in Figure 1.

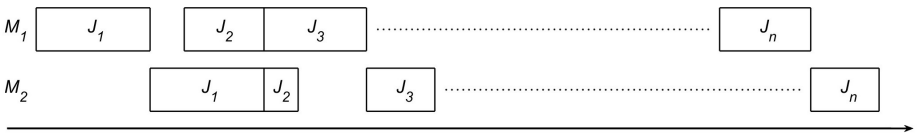

Figure 1. Gantt chart of the two-machine no-wait flow shop problem.

3. Basic results

Lemma 1 (Cheng et al. [50]). *For the single-machine problem*

$$1 \left| \widehat{P}_{j[r]} = \widehat{P}_j \left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{[h]}\right)^{\beta} \right| \widetilde{C}_{\max},$$

an optimal sequence is obtained by sequencing the jobs in non decreasing order of $\widehat{P}_j$ (i.e., the shortest processing time (SPT) rule).

According to Gilmore and Gomory [52], and Reddi and Ramamoorthy [53], $F2\,|no{-}wait|\,\widetilde{C}_{\max}$ can be solved in polynomial time (i.e., in $O(n^2)$ time). However, the computational complexity of $F2\left|no-wait, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$ is still open, so we guess that the problem $F2\left|no-wait, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$ is probably NP-hard.

3.1. Special cases

This subsection, we first consider two special cases, i.e., the dominating machine constraints.

Definition 3.1. *The machines form an increasing relationship of dominating machines (denoted idm), i.e., $M_1 < M_2$ (or M_2 dominates M_1), if and only if (iff)* $\max\{\widehat{a}_j|j=1,2,\ldots,n\}\le\min\{\widehat{b}_j|j=1,2,\ldots,n\}$.

Definition 3.2. *The machines form a decreasing relationship of dominating machines (denoted ddm), i.e., $M_1 > M_2$ (or M_1 dominates M_2), iff* $\max\{\widehat{b}_j|j=1,2,\ldots,n\}\le\min\{\widehat{a}_j|j=1,2,\ldots,n\}$.

For the *idm* constraint, for $\phi=(J_1,J_2,\ldots,J_n)$, by Eq (2), we know that

$$\begin{aligned}\widetilde{C}_{\max}(\phi)&=\widehat{a}_1+\sum_{g=1}^{n-1}\widehat{b}_g\left(1+\sum_{h=1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta}+\widehat{b}_n\left(1+\sum_{h=1}^{n-1}\ln\widehat{b}_{[h]}\right)^{\beta}\\&=\widehat{a}_1+\sum_{g=1}^{n}\widehat{b}_g\left(1+\sum_{h=1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta}\\&=\widehat{a}_1+\widehat{b}_1+\sum_{g=2}^{n}\widehat{b}_g\left(1+\sum_{h=1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta}.\end{aligned}\tag{3}$$

From Eq (3) and Lemma 1, the term $\sum_{g=1}^{n}\widehat{b}_g\left(1+\sum_{h=1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta}$ is minimized by the SPT rule. Thus, $\widetilde{C}_{\max}$ can minimized by keeping the relative position of SPT rule and constantly changing the job in the first position. Algorithm 1 below generates an optimal solution for $F2\left|no{-}wait, idm, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$.

Algorithm 1: Polynomial-time algorithm for $F2\left|no-wait, idm, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln \widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$

Input: n, β, $\widehat{P}_{ij}$

Output: The best job sequence ϕ^* and its makespan $\widetilde{C}_{max}$

Phase 1: Initialization

Sort jobs in non decreasing order of b_j to obtain a sequence ϕ^1 ;

Compute $\widetilde{C}_{max}(\phi^1)$ by Eq (3);

Let $\phi^* \leftarrow \phi^1$ and $\widetilde{C}_{max}(\phi^*) \leftarrow \widetilde{C}_{max}(\phi^1)$;

Phase 2: Insert improvement

$k \leftarrow 2$;

while $k \le n$ **do**

  For the sequence ϕ^1, insert the k-th jobs into Position 1 to obtain a new sequence ϕ^k;

  Compute $\widetilde{C}_{max}(\phi^k)$ by Eq (3);

  if $\widetilde{C}_{max}(\phi^k) < \widetilde{C}_{max}(\phi^*)$ **then**

    $\phi^* \leftarrow \phi^k$;

    $\widetilde{C}_{max}(\phi^*) \leftarrow \widetilde{C}_{max}(\phi^k)$;

  end

  $k \leftarrow k+1$;

end

return ϕ^* *and its makespan* $\widetilde{C}_{max}(\phi^*)$.

Theorem 1. *For the problem* $F2\left|no-wait, idm, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln \widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$, *Algorithm 1 generates an optimal solution.*

Proof : For any feasible solution $\phi' = ([1]', [2]', \ldots, [n]')$, let $[1]' = [k]$, where $[k]$ is the kth element of $\phi^1 = ([1], [2], \ldots, [n])$ obtained in Phase 1 of Algorithm 1. From Eq (3) and using the method of adjacent pairwise interchange, we have

$$\widetilde{C}_{max}(\phi') = \widehat{a}_{[k]} + \widehat{b}_{[k]} + \sum_{g=2}^{n}\widehat{b}_{[g]'}\left(1+\sum_{h=1}^{g-1}\ln \widehat{b}_{[h]'}\right)^{\beta},$$

$$\widetilde{C}_{max}(\phi^k) = \widehat{a}_{[k]} + \widehat{b}_{[k]} + \sum_{g=1}^{k-1}\widehat{b}_{[g]}\left(1+\ln \widehat{b}_{[k]}+\sum_{h=1}^{g-1}\ln \widehat{b}_{[h]}\right)^{\beta} + \sum_{g=k+1}^{n}\widehat{b}_{[g]}\left(1+\sum_{h=1}^{g-1}\ln \widehat{b}_{[h]}\right)^{\beta},$$

obviously, $\widetilde{C}_{max}(\phi') \ge \widetilde{C}_{max}(\phi^k) \ge \widetilde{C}_{max}(\phi^*)$.

Theorem 2. *The problem* $F2\left|no-wait, idm, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln \widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$ *is polynomially solvable in* $O(n^2)$ *time by Algorithm* 1.

Proof : Phase 1 obtains the initial solution via sorting with $O(n\log n)$ time complexity. Phase 2 generates new solutions through $n-1$ insertion operations, each requiring makespan recalculation, leading to $O(n^2)$ time complexity. Hence, the total time of Algorithm 1 is $O(n^2)$.

For the *ddm* constraint, similarly, for $\phi = (J_1, J_2, \ldots, J_n)$, we know that

$$\begin{aligned}\widetilde{C}_{\max}(\phi) &= \widehat{a}_1 + \sum_{g=1}^{n-1} \widehat{a}_{g+1} \left(1 + \sum_{h=1}^{g} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{b}_n \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta} \\ &= \sum_{g=1}^{n} \widehat{a}_g \left(1 + \sum_{h=1}^{g-1} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{b}_n \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta} \\ &= \sum_{g=1}^{n-1} \widehat{a}_g \left(1 + \sum_{h=1}^{g-1} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{a}_n \left(1 + \sum_{h=1}^{n-1} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{b}_n \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta}.\end{aligned} \tag{4}$$

From Eq (4) and Lemma 1, the term $\sum_{g=1}^{n} \widehat{a}_g \left(1 + \sum_{h=1}^{g-1} \ln \widehat{a}_{[h]}\right)^{\beta}$ is minimized by the SPT rule. Thus, $\widetilde{C}_{\max}$ can minimized by keeping the relative position of the SPT rule and constantly changing the job in the last position. Algorithm 2 below generates an optimal solution for the problem $F2\left|no-wait, ddm, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta}\right| \widetilde{C}_{\max}$.

Algorithm 2: Polynomial-time algorithm for $F2\left|no-wait, ddm, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta}\right| \widetilde{C}_{\max}$

Input: n, β, $\widehat{P}_{ij}$

Output: The best job sequence ϕ^* and its makespan $\widetilde{C}_{max}$

Phase 1: Initialization
Sort jobs in non decreasing order of a_j to obtain a sequence ϕ^n ;
Compute $\widetilde{C}_{max}(\phi^n)$ by Eq (4);
Let $\phi^* \leftarrow \phi^n$ and $\widetilde{C}_{max}(\phi^*) \leftarrow \widetilde{C}_{max}(\phi^n)$;
Phase 2: Insert improvement
$k \leftarrow 1$;
while $k < n$ **do**
    For the sequence ϕ^n, insert the k-th jobs into position n to obtain a new sequence ϕ^k;
    Compute $\widetilde{C}_{max}(\phi^k)$ by Eq (4);
    if $\widetilde{C}_{max}(\phi^k) < \widetilde{C}_{max}(\phi^*)$ **then**
        $\phi^* \leftarrow \phi^k$;
        $\widetilde{C}_{max}(\phi^*) \leftarrow \widetilde{C}_{max}(\phi^k)$;
    end
    $k \leftarrow k + 1$;
end
return ϕ^* *and its makespan* $\widetilde{C}_{max}(\phi^*)$.

Theorem 3. *For the problem* $F2\left|no-wait, ddm, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta}\right| \widetilde{C}_{\max}$, *Algorithm 2 generates an optimal solution in* $O(n^2)$ *time.*

Example 1. *Consider the problem* $F2\left|no-wait, idm, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$, *where* $n=5$ *and the basic processing times* $\widehat{a}_j$ *and* $\widehat{b}_j$ *are* $(\widehat{a}_1,\widehat{a}_2,\widehat{a}_3,\widehat{a}_4,\widehat{a}_5)=(3,2,5,4,3)$ *and* $(\widehat{b}_1,\widehat{b}_2,\widehat{b}_3,\widehat{b}_4,\widehat{b}_5)=(8,10,6,9,7)$, *the learning rate of all jobs is* $\beta=-0.2$.

Solution: According to the Algorithm 1, the optimal sequence is obtained as follows.
Step (1): The SPT order of $\widehat{b}_j$ generates an initial sequence $\phi^1=\{3,5,1,4,2\}$, $\widetilde{C}_{\max}(\phi^1)=35.13$.
Step (2): According to Step 2 in Algorithm 1, we have
$\phi^2=\{5,3,1,4,2\}$, $\widetilde{C}_{\max}(\phi^2)=33.27$. $\phi^3=\{1,3,5,4,2\}$, $\widetilde{C}_{\max}(\phi^3)=33.46$.
$\phi^4=\{4,3,5,1,2\}$, $\widetilde{C}_{\max}(\phi^4)=34.70$. $\phi^5=\{2,3,5,1,4\}$, $\widetilde{C}_{\max}(\phi^5)=32.98$.
Step (3): $\phi^5=\{2,3,5,1,4\}$ is the optimal sequence for Example 1.

Example 2. *Consider the problem* $F2\left|no-wait, ddm, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$, *where* $n=5$ *and the basic processing times* $\widehat{a}_j$ *and* $\widehat{b}_j$ *are* $(\widehat{a}_1,\widehat{a}_2,\widehat{a}_3,\widehat{a}_4,\widehat{a}_5)=(8,10,6,9,7)$ *and* $(\widehat{b}_1,\widehat{b}_2,\widehat{b}_3,\widehat{b}_4,\widehat{b}_5)=(3,2,5,4,3)$, *the learning rate of all jobs is* $\beta=-0.2$.

Solution: According to the Algorithm 2, the optimal sequence is obtained as follows.
Step (1): The SPT order of $\widehat{a}_j$ generates an initial sequence $\phi^5=\{3,5,1,4,2\}$, $\widetilde{C}_{\max}(\phi^5)=31.52$.
Step (2): According to Step 2 in Algorithm 1′, we have
$\phi^1=\{5,1,4,2,3\}$, $\widetilde{C}_{\max}(\phi^2)=34.10$. $\phi^2=\{3,1,4,2,5\}$, $\widetilde{C}_{\max}(\phi^2)=32.42$.
$\phi^3=\{3,5,4,2,1\}$, $\widetilde{C}_{\max}(\phi^3)=32.32$. $\phi^4=\{3,5,1,2,4\}$, $\widetilde{C}_{\max}(\phi^4)=33.00$.
Step (3): $\phi^5=\{3,5,1,4,2\}$ is the optimal sequence for Example 2.

Then we consider two other special cases, i.e., the fixed processing time on the machine constraints. Under this constraint, all job have the identical processing time on same machine, except for machine M_d (i.e., $\widehat{P}_{ij}=\widehat{P}_i, j=1,2,\ldots,n, i=1,2, i\neq d$). For the problem $F2\left|no-wait, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}, \widehat{P}_{2j}=\widehat{P}_2\right|\widetilde{C}_{\max}$, for $\phi=(J_1,J_2,\ldots,J_n)$, by Eq (2), we know that

$$\widetilde{C}_{\max}(\phi) = \widehat{a}_1+\sum_{g=1}^{n-1}\max\left\{\widehat{P}_2\left(1+\sum_{h=1}^{g-1}\ln\widehat{P}_2\right)^{\beta}, \widehat{a}_{g+1}\left(1+\sum_{h=1}^{g}\ln\widehat{a}_{[h]}\right)^{\beta}\right\}+\widehat{P}_2\left(1+\sum_{h=1}^{n-1}\ln\widehat{P}_2\right)^{\beta}.$$

Theorem 4. *The problem* $F2\left|no-wait, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}, \widehat{P}_{2j}=\widehat{P}_2\right|\widetilde{C}_{\max}$ *is polynomially solvable in* $O(n\log n)$ *time by sequencing the jobs in non decreasing order of* $\widehat{a}_j$ *(i.e., the SPT rule).*

Proof : Suppose that ϕ and ϕ' are two job sequences where the difference between ϕ and ϕ' is a pairwise interchange of two adjacent jobs J_s and J_t; that is, $\phi=[\eta_1,J_s,J_t,\eta_2]$ and $\phi'=[\eta_1,J_t,J_s,\eta_2]$, where η_1 and η_2 denote a partial sequence and η_1 has k jobs (note that k may be zero). Furthermore, T denotes the identical value in ϕ and ϕ', and we suppose $\widehat{a}_s\geq\widehat{a}_t$. Where $k=0$, the completion time of ϕ is

$$\begin{aligned}\widetilde{C}_{\max}(\phi) &= T+\widehat{a}_s+\max\{0,\widehat{P}_2-\widehat{a}_t(1+\ln\widehat{a}_s)^{\beta}\}+\widehat{a}_t(1+\ln\widehat{a}_s)^{\beta}\\ &= T+\widehat{a}_s+\max\{\widehat{a}_t(1+\ln\widehat{a}_s)^{\beta},\widehat{P}_2\}.\end{aligned}$$

Similarly, the completion time of ϕ' is

$$\begin{aligned}\widetilde{C}_{\max}(\phi') &= T + \widehat{a}_t + \max\{0, \widehat{P}_2 - \widehat{a}_s(1+\ln\widehat{a}_t)^\beta\} + \widehat{a}_s(1+\ln\widehat{a}_t)^\beta \\ &= T + \widehat{a}_t + \max\{\widehat{a}_s(1+\ln\widehat{a}_t)^\beta, \widehat{P}_2\}.\end{aligned}$$

Hence, we have

$$\begin{aligned}\widetilde{C}_{\max}(\phi) - \widetilde{C}_{\max}(\phi') &= \widehat{a}_s - \widehat{a}_t + \max\{\widehat{a}_t(1+\ln\widehat{a}_s)^\beta, \widehat{P}_2\} - \max\{\widehat{a}_s(1+\ln\widehat{a}_t)^\beta, \widehat{P}_2\} \\ &\geq \widehat{a}_s - \widehat{a}_t + \widehat{a}_t(1+\ln\widehat{a}_s)^\beta - \widehat{a}_s(1+\ln\widehat{a}_t)^\beta,\end{aligned}$$

using the similar method of derivative [50], we can know $\widetilde{C}_{\max}(\phi) - \widetilde{C}_{\max}(\phi') \geq 0$ in $k = 0$. Where $0 < k \leq n-2$, the completion time of ϕ is

$$\begin{aligned}\widetilde{C}_{\max}(\phi) &= T + \max\{0, \widehat{P}_2(1+\sum_{h=1}^{k-1}\ln\widehat{P}_2)^\beta - \widehat{a}_s(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]})^\beta\} + \widehat{a}_s(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]})^\beta \\ &\quad + \max\{0, \widehat{P}_2(1+\sum_{h=1}^{k}\ln\widehat{P}_2)^\beta - \widehat{a}_t(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]}+\ln\widehat{a}_s)^\beta\} + \widehat{a}_t(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]}+\ln\widehat{a}_s)^\beta \\ &= T + \max\{\widehat{a}_s(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]})^\beta, \widehat{P}_2(1+\sum_{h=1}^{k-1}\ln\widehat{P}_2)^\beta\} \\ &\quad + \max\{\widehat{a}_t(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]}+\ln\widehat{a}_s)^\beta, \widehat{P}_2(1+\sum_{h=1}^{k}\ln\widehat{P}_2)^\beta\}.\end{aligned}$$

Similarly, the completion time of ϕ' is

$$\begin{aligned}\widetilde{C}_{\max}(\phi') &= T + \max\{\widehat{a}_t(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]})^\beta, \widehat{P}_2(1+\sum_{h=1}^{k-1}\ln\widehat{P}_2)^\beta\} \\ &\quad + \max\{\widehat{a}_s(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]}+\ln\widehat{a}_t)^\beta, \widehat{P}_2(1+\sum_{h=1}^{k}\ln\widehat{P}_2)^\beta\}.\end{aligned}$$

Hence, we have

$$\begin{aligned}\widetilde{C}_{\max}(\phi) - \widetilde{C}_{\max}(\phi') &= \max\{\widehat{a}_s(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]})^\beta, \widehat{P}_2(1+\sum_{h=1}^{k-1}\ln\widehat{P}_2)^\beta\} \\ &\quad - \max\{\widehat{a}_t(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]})^\beta, \widehat{P}_2(1+\sum_{h=1}^{k-1}\ln\widehat{P}_2)^\beta\} \\ &\quad + \max\{\widehat{a}_t(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]}+\ln\widehat{a}_s)^\beta, \widehat{P}_2(1+\sum_{h=1}^{k}\ln\widehat{P}_2)^\beta\} \\ &\quad - \max\{\widehat{a}_s(1+\sum_{h=1}^{k}\ln\widehat{a}_{[h]}+\ln\widehat{a}_t)^\beta, \widehat{P}_2(1+\sum_{h=1}^{k}\ln\widehat{P}_2)^\beta\}\end{aligned}$$

$$\geq (\widehat{a}_s - \widehat{a}_t)(1 + \sum_{h=1}^{k} \ln \widehat{a}_{[h]})^{\beta} + \widehat{a}_t(1 + \sum_{h=1}^{k} \ln \widehat{a}_{[h]} + \ln \widehat{a}_s)^{\beta}$$
$$-\widehat{a}_s(1 + \sum_{h=1}^{k} \ln \widehat{a}_{[h]} + \ln \widehat{a}_t)^{\beta},$$

using the similar method of derivative [50], we can know $\widetilde{C}_{\max}(\phi) - \widetilde{C}_{\max}(\phi') \geq 0$ in $0 < k \leq n-2$. Based on this interchange argument, we conclude that an optimal schedule can be obtained by sequencing the jobs in non decreasing order of $\widehat{a}_j$.

Theorem 5. *For the problem* $F2\left|no-wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta}, \widehat{P}_{1j} = \widehat{P}_1\right| \widetilde{C}_{\max}$ *is polynomially solvable in* $O(n \log n)$ *time by sequencing the jobs in non decreasing order of* $\widehat{b}_j$ *(i.e., the SPT rule).*

3.2. General case

Let $\widehat{D}_j = \widehat{a}_j + \widehat{b}_j$, $j = 1, 2, \ldots, n$; $\ln \widehat{P}_{\max} = \max\left(\sum_{h=1}^{n} \ln \widehat{a}_h - \ln \widehat{a}_{min}, \sum_{h=1}^{n} \ln \widehat{b}_h - \ln \widehat{b}_{min}\right)$, where $\ln \widehat{a}_{\min} = \min\{\ln \widehat{a}_j | j = 1, 2, \ldots, n\}$; and $\ln \widehat{b}_{\min} = \min\{\ln \widehat{b}_j | j = 1, 2, \ldots, n\}$. From the Lemma 1, we can use the SPT (i.e., in order of non decreasing $\widehat{D}_j$) rule as an approximate algorithm for

$$F2\left|no-wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta}\right| \widetilde{C}_{\max}.$$

Theorem 6. *Let* ϕ^* *(respectively.* ϕ*) be an optimal sequence (respectively. an SPT sequence) for*

$$F2\left|no-wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta}\right| \widetilde{C}_{\max},$$

then $\frac{\widetilde{C}_{\max}(\phi)}{\widetilde{C}_{\max}(\phi^*)} \leq \frac{2}{(1+\ln \widehat{P}_{\max})^{\beta}}$*, and this bound is tight.*

Proof: Without loss of generality, it is assumed that $\widehat{D}_1 \leq \widehat{D}_2 \leq \ldots \leq \widehat{D}_n$ and $\phi = (J_1, J_2, \ldots, J_n)$, then

$$\begin{aligned}
\widetilde{C}_{\max}(\phi) &\leq \widehat{D}_1 + \widehat{D}_2\left[1 + \min(\ln \widehat{a}_1, \ln \widehat{b}_1)\right]^{\beta} + \widehat{D}_3\left[1 + \min(\ln \widehat{a}_1 + \ln \widehat{a}_2, \ln \widehat{b}_1 + \ln \widehat{b}_2)\right]^{\beta} + \ldots \\
&+ \widehat{D}_n\left[1 + \min(\sum_{h=1}^{n-1} \ln \widehat{a}_h, \sum_{h=1}^{n-1} \ln \widehat{b}_h)\right]^{\beta} \\
&\leq \widehat{D}_1 + \widehat{D}_2 + \ldots + \widehat{D}_n.
\end{aligned}$$

Thus

$$\widetilde{C}_{\max}(\phi) \leq \sum_{h=1}^{n} \widehat{D}_h. \tag{5}$$

Let $\phi^* = (J_{[1]}, J_{[2]}, \ldots, J_{[n]})$ be the optimal sequence, then

$$\begin{aligned}\widetilde{C}_{1[k]} &\geq a_{[1]} + a_{[2]}(1+\ln a_{[1]})^{\beta} + \ldots + a_{[k]}(1+\ln a_{[1]} + \ln a_{[2]} + \ldots + \ln a_{[k-1]})^{\beta},\\ \widetilde{C}_{2[k]} &\geq b_{[1]} + b_{[2]}(1+\ln b_{[1]})^{\beta} + \ldots + b_{[k]}(1+\ln b_{[1]} + \ln b_{[2]} + \ldots + \ln b_{[k-1]})^{\beta}.\end{aligned}$$

Thus

$$\widetilde{C}_{\max}(\phi^*) \geq \sum_{h=1}^{n} \frac{\widehat{D}_h}{2}(1+\ln \widehat{P}_{\max})^{\beta} = \frac{(1+\ln \widehat{P}_{\max})^{\beta}}{2}\sum_{h=1}^{n}\widehat{D}_h. \tag{6}$$

From Eqs (5) and (6), we have

$$\frac{\widetilde{C}_{\max}(\phi)}{\widetilde{C}_{\max}(\phi^*)} \leq \frac{2}{(1+\ln \widehat{P}_{\max})^{\beta}}.$$

We prove that the bound $\frac{2}{(1+\ln \widehat{P}_{\max})^{\beta}}$ is tight. Consider the following. Learning takes place by the 100% learning curve (i.e., $\beta = 0$), we have the bound $\frac{2}{(1+\ln \widehat{P}_{\max})^{\beta}} = 2$. There are two jobs J_1 and J_2, where $\widehat{a}_1 = B$ $(B > 1)$, $\widehat{b}_1 = \varepsilon$, $\widehat{a}_2 = \varepsilon$, and $\widehat{b}_2 = B$. Since $\widehat{D}_j = \widehat{a}_j + \widehat{b}_j = B + \varepsilon$ for jobs J_1 and J_2, the SPT sequence may arrange the jobs in any order. We assume that the SPT sequence is $\phi = (J_1, J_2)$, we have $\widetilde{C}_{\max}(\phi) = 2B + \varepsilon$. Obviously, the optimal sequence is $\phi^* = (J_2, J_1)$, it has a cost $\widetilde{C}_{\max}(\phi^*) = B + \varepsilon$. Hence, $\frac{\widetilde{C}_{\max}(\phi)}{\widetilde{C}_{\max}(\phi^*)} = \frac{2B+\varepsilon}{B+\varepsilon} = 2(\varepsilon \to 0)$; i.e., this bound is tight.

4. A branch-and-bound (BB) method

4.1. Lower bounds

Let $\phi = (\phi_{sp}, \phi_{up})$ be a sequence where ϕ_{sp} (respectively. ϕ_{up}) is the sequenced (respectively. unsequenced) part, and μ jobs have been determined in ϕ_{sp}. For M_1 of the $(\mu+1)$th position, we have

$$\widetilde{C}_{1[\mu+1]}(\phi) = \widetilde{C}_{1[\mu]}(\phi) + \max\left\{\widehat{b}_{[\mu]}\left(1+\sum_{h=1}^{\mu-1}\ln \widehat{b}_{[h]}\right)^{\beta}, \widehat{a}_{[\mu+1]}\left(1+\sum_{h=1}^{\mu}\ln \widehat{a}_{[h]}\right)^{\beta}\right\}. \tag{7}$$

Similarly

$$\begin{aligned}\widetilde{C}_{2[\mu+1]}(\phi) &= \widetilde{C}_{1[\mu+1]}(\phi) + \widehat{b}_{[\mu+1]}\left(1+\sum_{h=1}^{\mu}\ln \widehat{b}_{[h]}\right)^{\beta}\\ &= \widetilde{C}_{1[\mu]}(\phi) + \max\left\{\widehat{b}_{[\mu]}\left(1+\sum_{h=1}^{\mu-1}\ln \widehat{b}_{[h]}\right)^{\beta}, \widehat{a}_{[\mu+1]}\left(1+\sum_{h=1}^{\mu}\ln \widehat{a}_{[h]}\right)^{\beta}\right\}\\ &\quad + \widehat{b}_{[\mu+1]}\left(1+\sum_{h=1}^{\mu}\ln \widehat{b}_{[h]}\right)^{\beta},\end{aligned}$$

$$\begin{aligned}\widetilde{C}_{2[\mu+i]}(\phi) = \widetilde{C}_{1[\mu]}(\phi) + \sum_{g=\mu}^{\mu+i-1} \max\left\{\widehat{b}_{[g]}\left(1+\sum_{h=1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta}, \widehat{a}_{[g+1]}\left(1+\sum_{h=1}^{g}\ln\widehat{a}_{[h]}\right)^{\beta}\right\} \\ + \widehat{b}_{[\mu+i]}\left(1+\sum_{h=1}^{\mu+i-1}\ln\widehat{b}_{[h]}\right)^{\beta},\end{aligned} \tag{8}$$

where $1 \le i \le n-\mu$.

Hence

$$\begin{aligned}\widetilde{C}_{\max}(\phi, M_1) &= \widetilde{C}_{2[n]}(\phi) \\ &= \widetilde{C}_{1[\mu]}(\phi) + \sum_{g=\mu}^{n-1} \max\left\{\widehat{b}_{[g]}\left(1+\sum_{h=1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta}, \widehat{a}_{[g+1]}\left(1+\sum_{h=1}^{g}\ln\widehat{a}_{[h]}\right)^{\beta}\right\} \\ &\quad + \widehat{b}_{[n]}\left(1+\sum_{h=1}^{n-1}\ln\widehat{b}_{[h]}\right)^{\beta}.\end{aligned} \tag{9}$$

From Eq (9), we have

$$\begin{aligned}\widetilde{C}_{\max}(\phi, M_1) &\ge \widetilde{C}_{1[\mu]}(\phi) + \sum_{g=\mu}^{n-1}\widehat{b}_{[g]}\left(1+\sum_{h=1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta} + \widehat{b}_{[n]}\left(1+\sum_{h=1}^{n-1}\ln\widehat{b}_{[h]}\right)^{\beta} \\ &= \widetilde{C}_{1[\mu]}(\phi) + \sum_{g=\mu}^{n}\widehat{b}_{[g]}\left(1+\sum_{h=1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta} \\ &= \widetilde{C}_{1[\mu]}(\phi) + \widehat{b}_{[\mu]}\left(1+\sum_{h=1}^{\mu-1}\ln\widehat{b}_{[h]}\right)^{\beta} + \sum_{g=\mu+1}^{n}\widehat{b}_{g}\left(1+\sum_{h=1}^{\mu}\ln\widehat{b}_{[h]}+\sum_{h=\mu+1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta},\end{aligned} \tag{10}$$

where $\sum_{h=\mu+1}^{\mu}\ln\widehat{b}_{[h]} = 0$.

From Eq (10), the terms $\widetilde{C}_{1[\mu]}(\phi)$, $\widehat{b}_{[\mu]}\left(1+\sum_{h=1}^{\mu-1}\ln\widehat{b}_{[h]}\right)^{\beta}$, and $\sum_{h=1}^{\mu}\ln\widehat{b}_{[h]}$ are fixed constants, $\sum_{g=\mu+1}^{n}\widehat{b}_{g}\left(1+\sum_{h=1}^{\mu}\ln\widehat{b}_{[h]}+\sum_{h=\mu+1}^{g-1}\ln\widehat{b}_{[h]}\right)^{\beta}$ can be minimized by the SPT order of $\widehat{b}_j$ (Lemma 1). Therefore, for the machine M_1, the first lower bound is

$$\begin{aligned}LB_1(\widetilde{C}_{\max}, M_1) = \widetilde{C}_{1[\mu]}(\phi) + \widehat{b}_{[\mu]}\left(1+\sum_{h=1}^{\mu-1}\ln\widehat{b}_{[h]}\right)^{\beta} \\ + \sum_{g=\mu+1}^{n}\widehat{b}_{<g>}\left(1+\sum_{h=1}^{\mu}\ln\widehat{b}_{[h]}+\sum_{h=\mu+1}^{g-1}\ln\widehat{b}_{<h>}\right)^{\beta},\end{aligned} \tag{11}$$

where $\widehat{b}_{<\mu+1>} \le \widehat{b}_{<\mu+2>} \le \ldots \le \widehat{b}_{<n>}$.

Similarly, from Eq (9), we have

$$\begin{aligned}\widetilde{C}_{\max}(\phi, M_1) &\geq \widetilde{C}_{1[\mu]}(\phi) + \sum_{g=\mu}^{n-1} \widehat{a}_{[g+1]} \left(1 + \sum_{h=1}^{g} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{b}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta} \\ &= \widetilde{C}_{1[\mu]}(\phi) + \sum_{g=\mu}^{n-2} \widehat{a}_{[g+1]} \left(1 + \sum_{h=1}^{\mu} \ln \widehat{a}_{[h]} + \sum_{h=\mu+1}^{g} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{a}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{a}_{[h]}\right)^{\beta} \\ &\quad + \widehat{b}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta}. \end{aligned} \tag{12}$$

From Eq (12), $\widetilde{C}_{1[\mu]}(\phi)$ and $\sum_{h=1}^{\mu} \ln \widehat{a}_{[h]}$ are fixed constants, and

$$\sum_{g=\mu}^{n-2} \widehat{a}_{[g+1]} \left(1 + \sum_{h=1}^{\mu} \ln \widehat{a}_{[h]} + \sum_{h=\mu+1}^{g} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{a}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{b}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta}$$

can be minimized by the Algorithm 2. Therefore, for M_1, the second lower bound is:

$$\begin{aligned} LB_2(\widetilde{C}_{\max}, M_1) = \widetilde{C}_{1[\mu]}(\phi) &+ \sum_{g=\mu}^{n-2} \widehat{a}_{[g+1]} \left(1 + \sum_{h=1}^{\mu} \ln \widehat{a}_{[h]} + \sum_{h=\mu+1}^{g} \ln \widehat{a}_{[h]}\right)^{\beta} \\ &+ \widehat{a}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{b}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta}, \end{aligned} \tag{13}$$

where the value of
$\sum_{g=\mu}^{n-2} \widehat{a}_{[g+1]} \left(1 + \sum_{h=1}^{\mu} \ln \widehat{a}_{[h]} + \sum_{h=\mu+1}^{g} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{a}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{a}_{[h]}\right)^{\beta} + \widehat{b}_{[n]} \left(1 + \sum_{h=1}^{n-1} \ln \widehat{b}_{[h]}\right)^{\beta}$
can obtain by Algorithm 2.

In order to make the lower bound tighter, we choose the maximum value of Eqs (11) and (13) as a lower bound for

$$F2 \left| no-wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij} \left(1 + \sum_{h=1}^{r-1} \ln \widehat{P}_{i[h]}\right)^{\beta} \right| \widetilde{C}_{\max},$$

which is

$$LB(\widetilde{C}_{\max}) = \max\{LB_1(\widetilde{C}_{\max}, M_1), LB_2(\widetilde{C}_{\max}, M_1)\}. \tag{14}$$

4.2. Upper bound

In this subsection, we obtain an initial upper bound by the SPT and a heuristic algorithm (i.e., Algorithm 3) for the branch-and-bound algorithm. The time complexity of the heuristic algorithm is $O(n^3)$, and the steps of the heuristic algorithm are as follows.

Algorithm 3: A heuristic algorithm for $F2\left|no-wait, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$

Input: n, β, $\widehat{P}_{ij}$

Output: A job sequence ϕ_2 and its makespan $\widetilde{C}_{max}$

Let $\phi_1[1..n] \leftarrow [1,2,\ldots,n]$ and $\phi_2 \leftarrow []$;
Let $bestsub \leftarrow []$ and $bestsubfit \leftarrow INF$;
for $i = 1$ *to* n **do**
  for $j = 1$ *to* n **do**
    if $i \neq j$ *AND* $\widehat{a}_i + \max\left\{\widehat{b}_i, \widehat{a}_j\,(1+\widehat{a}_i)^{\beta}\right\} < bestsubfit$ **then**
      $bestsub \leftarrow [i, j]$;
      $bestsubfit \leftarrow \widehat{a}_i + \max\left\{\widehat{b}_i, \widehat{a}_j\,(1+\widehat{a}_i)^{\beta}\right\}$;
    end
  end
end
$\phi_2 \leftarrow bestsub$;
$\phi_1 \leftarrow \phi_1 \backslash bestsub$; \\ Remove all elements from array ϕ_1 that are present in the array $bestsub$.
for $i = 1$ *to* $n-2$ **do**
  Let $bestj$ and $bestjfit \leftarrow INF$;
  for $j = 1$ *to* $n-i-1$ **do**
    if $\max\left\{\widehat{b}_{\phi_2[1+i]}(1+\sum_{h=1}^{i}\ln_{\phi_2[h]})^{\beta}, \widehat{a}_{\phi_1[j]}\left(1+\sum_{h=1}^{1+i}\ln_{\phi_2[h]}\right)^{\beta}\right\} < bestjfit$ **then**
      $bestj \leftarrow j$;
      $bestjfit \leftarrow \max\left\{\widehat{b}_{\phi_2[1+i]}(1+\sum_{h=1}^{i}\ln_{\phi_2[h]})^{\beta}, \widehat{a}_{\phi_1[j]}\left(1+\sum_{h=1}^{1+i}\ln_{\phi_2[h]}\right)^{\beta}\right\}$;
    end
  end
  $\phi_2 \leftarrow add(\phi_2, \phi_1[bestj])$;
  $\phi_1 \leftarrow \phi_1 \backslash \phi_1[bestj]$;
end
return ϕ_2 *and its makespan* $\widetilde{C}_{max}(\phi_2)$.

In order to make the upper bound tighter, we choose the minimum upper bound of Algorithm 3 and SPT rule as initial upper bound (denoted by UB), which is:

$$UB(\widetilde{C}_{\max}) = \min\{UB_{A3}, UB_{SPT}\}, \tag{15}$$

where UB_{A3} is obtained by Algorithm 3 and UB_{SPT} is obtained by the SPT rule.

4.3. BB method

Based on the above upper bound and lower bound, the branch-and-bound (BB) algorithm (i.e., Algorithm 4) can be proposed to solve $F2\left|no-wait, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$.

Algorithm 4: *BB* for $F2\left|no-wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln \widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$

Input: n, β, $\widehat{P}_{ij}$
Output: The best job sequence and its makespan

Phase 1: Initialization
Sort jobs in non decreasing order of D_j to obtain a sequence ϕ_1 ;
Obtain a sequence ϕ_2 by Algorithm 3;
Compute $\widetilde{C}_{max}(\phi_1)$ and $\widetilde{C}_{max}(\phi_2)$ by Eq (2);
Let $UB \leftarrow \widetilde{C}_{max}(\phi_2)$ and $bestsequence \leftarrow \phi_2$;
if $\widetilde{C}_{max}(\phi_1) < \widetilde{C}_{max}(\phi_2)$ **then**
    $UB \leftarrow \widetilde{C}_{max}(\phi_1)$;
    $bestsequence \leftarrow \phi_1$;
end
Phase 2: Branching and bounding
Initialize the root node; \\ Node contains the sequenced part, the unsequended part, and its lower bound.
Generate the priority queue; \\ Priority rule: Nodes with more scheduled jobs come first. For nodes with an equal number of scheduled jobs, the one with a smaller lower bound is selected first.
Insert the root node into priority queue;
while *priority queue is not empty* **do**
    Let Nownode ← top(priority queue);
    pop(priority queue);
    if *the number of scheduled jobs of the Nownode* $= n$ **then**
        if *the lower bound of the Nownode* $< UB$ **then**
            $UB \leftarrow$ the lower bound of the Nownode;
            $bestsequence \leftarrow$ the sequenced part of the Nownode;
        end
        Continue;
    end
    if *the lower bound of the Nownode* $> UB$ **then**
        Continue;
    end
    Generate all branches of the Nownode and insert them into the priority queue;
end
return $bestsequence$ *and its makespan* UB.

5. Other heuristic algorithms

5.1. Nawaz-Enscore-Hamalgorithm

As in Nawaz et al. [54], we proposed a revised Nawaz-Enscore-Ham (NEH) algorithm (i.e., Algorithm 5).

Algorithm 5: NEH algorithm for $F2\left|no-wait, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$

Input: n, β, $\widehat{P}_{ij}$

Output: A job sequence ϕ_2 and its makespan $\widetilde{C}_{max}$

Phase 1: Initialization
Sort jobs in non decreasing order of D_j to obtain a sequence ϕ_1 ;
Let $\phi_2 \leftarrow \phi_1[1]$;
$\phi_1 \leftarrow \phi_1 \backslash \phi_1[1]$;
Phase 2: Insert improvement
for $i = 1$ *to* $n-1$ **do**
  Let $bestj$ and $bestjfit \leftarrow INF$;
  for $j = 1$ *to* $i+1$ **do**
    Insert $\phi_1[1]$ into the jth position of sequence ϕ_2 to generate sequence ϕ_3;
    Compute $\widetilde{C}_{max}(\phi_3)$ by Eq (2);
    if $\widetilde{C}_{max}(\phi_3) <$ *bestjfit* **then**
      $bestj \leftarrow j$;
      $bestjfit \leftarrow \widetilde{C}_{max}(\phi_3)$;
    end
  end
  Insert $\phi_1[1]$ into the $bestj$th position of sequence ϕ_2;
  $\phi_1 \leftarrow \phi_1 \backslash \phi_1[1]$;
end
return ϕ_2 *and its makespan* $\widetilde{C}_{max}(\phi_2)$.

5.2. Simulated annealing

In this subsection, we use the simulated annealing (SA) algorithm (see Osman and Potts [55], Zhang et al. [56], and Sun et al. [57]) (i.e., Algorithm 6) to solve

$$F2\left|no-wait, \widehat{P}_{ij[r]}=\widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln\widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}.$$

Algorithm 6: *SA* algorithm for $F2\left|no-wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln \widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$

Input: n, β, $\widehat{P}_{ij}$

Output: A job sequence ϕ and its makespan $\widetilde{C}_{max}$

Phase 1: Initialization

Initialize some parameters (maximum number of iterations Γ_1, initial temperature Δ_1, final temperature Δ_2, and cooling factor σ);

Sort jobs in non decreasing order of D_j to obtain a sequence ϕ;

Compute $\widetilde{C}_{max}(\phi)$ by Eq (2);

Let $\lambda \leftarrow 0$, $bestsequence \leftarrow \phi$, $bestfit \leftarrow \widetilde{C}_{max}(\phi)$;

Phase 2: Search

while $\Delta_1 \geq \Delta_2$ **do**

  Swap any two jobs in the sequence ϕ to generate the sequence ϕ';

  if $\widetilde{C}_{max}(\phi') < \widetilde{C}_{max}(\phi)$ **then**

    $\phi \leftarrow \phi'$;

    if $\widetilde{C}_{max}(\phi') < bestfit$ **then**

      $bestsequence \leftarrow \phi'$;

      $bestfit \leftarrow \widetilde{C}_{max}(\phi')$;

    end

  end

  else

    if $rand() < exp((C_{\max}(\phi) - C_{\max}(\phi'))/\Delta_1)$ **then**

      $\phi \leftarrow \phi'$;

    end

  end

  $\lambda \leftarrow \lambda + 1$;

  if $\lambda = \Gamma_1$ **then**

    $\Delta_1 \leftarrow \sigma * \Delta_1$;

    $\lambda \leftarrow 0$;

  end

end

return ϕ *and its makespan* $\widetilde{C}_{max}(\phi)$.

5.3. Genetic algorithm

In this subsection, a genetic algorithm (GA) (i.e., Algorithm 7) is proposed to solve

$$F2\left|no-wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln \widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}.$$

Algorithm 7: GA for $F2\left|no-wait, \widehat{P}_{ij[r]} = \widehat{P}_{ij}\left(1+\sum_{h=1}^{r-1}\ln \widehat{P}_{i[h]}\right)^{\beta}\right|\widetilde{C}_{\max}$

Input: n, β, $\widehat{P}_{ij}$

Output: A job sequence and its makespan $\widetilde{C}_{max}$

Phase 1: Initialization

Initialize some parameters (maximum number of iterations Γ_2, population size ξ_1, crossover probability θ_1, and mutation probability θ_2);

Sort jobs in non decreasing order of D_j to obtain a sequence ϕ_1^0;

Obtain a sequence ϕ_2^0 by Algorithm 3;

Randomly generate $\xi_1 - 2$ sequences $\phi_3^0, \phi_4^0, \ldots, \phi_{\xi_1}^0$;

These initial sequences $\phi_1^0, \phi_2^0, \ldots, \phi_{\xi_1}^0$ form a initial population Z_0;

Sort sequences in the population Z_0 by increasing the value of $\widetilde{C}_{\max}$;

Let $\lambda \leftarrow 0$;

Phase 2: Search

for $i = 1$ *to* Γ_2 **do**

  Initialize an empty population Z_i;

  Copy the first and second sequences of population Z_{i-1} into population Z_i;

  while *number of sequences in population* $Z_i \neq \xi_1$ **do**

    Select randomly two sequences ϕ' and ϕ'' of population Z_{i-1} by roulette wheel selection;

    \\ Sequence k of population Z_{i-1} is selected with probability of $\frac{\sum_{h=1}^{\xi_1}\widetilde{C}_{\max}(\phi_h^{i-1})-\widetilde{C}_{\max}(\phi_k^{i-1})}{(\xi_1-1)*\sum_{h=1}^{\xi_1}\widetilde{C}_{\max}(\phi_h^{i-1})}$.

    if *rand()* $< \theta_1$ **then**

      $(\phi', \phi'') \leftarrow$ subtour exchange crossover (ϕ', ϕ'');

    end

    else

      Continue;

    end

    if *rand()* $< \theta_2$ **then**

      $\phi' \leftarrow$ exchange mutation (ϕ');

    end

    if *rand()* $< \theta_2$ **then**

      $\phi'' \leftarrow$ exchange mutation (ϕ'');

    end

    Insert ϕ' and ϕ'' to population Z_i;

  end

  Sort sequences in the population Z_i by increasing the value of $\widetilde{C}_{\max}$;

end

return *the first sequence* $\phi_1^{\Gamma_2}$ *of population* Z_{Γ_2} *and its makespan* $\widetilde{C}_{max}(\phi_1^{\Gamma_2})$.

6. Numerical test

We tested Algorithm 4 (i.e., BB), Algorithm 5 (i.e., NEH), Algorithm 6 (i.e., SA), and Algorithm 7 (i.e., GA) in C++ Visual studio 2026. All algorithms were executed on a laptop with an AMD R9-7945H central processing unit (CPU) (2.50 GHz, 16 cores, 32 threads), a 2-TB solid state drive (SSD) and 32-GB random-access memory (RAM). The CPU time is defined as the running time of the BB, NEH, SA, and GA, and the time unit is milli seconds (ms).

In addition, similar to Li et al. [33], the test parameters are randomly generated as follows:

(1) $\widehat{P}_{ij} \in U(1, 100]$;

(2) $\beta = -0.15, -0.2, -0.25, -0.3, -0.35, -0.4$;

(3) Small-sized instances: $n = 10, 11, 12, 13, 14$. Large-sized instances: $n = 100, 200, 300, 400, 500$.

The parameters for the algorithms and models used in the instances are presented in the following. In SA, the maximum number of iterations $\Gamma_1 = 10$, the initial temperature $\Delta_1 = \sum_{i=1}^{2} \sum_{j=1}^{n} \widehat{P}_{ij}/(5 * 2 * n)$, the final temperature $\Delta_2 = 1$, and the cooling factor $\sigma = 0.96$ (see Osman and Potts [55]). In GA, the maximum number of iterations $\Gamma_2 = 1000$, the population size $\xi_1 = 60$, the crossover probability $\theta_1 = 0.8$, and the mutation probability $\theta_2 = 0.15$ (see Wang [58], Arik [59], and Umam et al. [60]). For each combination of n and β, 20 instances were generated and solved, where the maximum and mean value of the number of searched nodes, CPU time, error, and relative performance ratio are calculated. For the SA algorithm and GA, each instance is run independently 10 times, and the average values of CPU time, error, and relative performance ratio are adopted as the final result of the instance.

For the small-scale instances, the error formula of heuristic algorithms is as follows

$$\frac{\widetilde{C}_{\max}(\phi)}{\widetilde{C}_{\max}(\phi^*)},$$

where ϕ is a sequence obtained by NEH, SA, and GA and ϕ^* is a optimal sequence obtained by the BB. The corresponding results are summarized in Table 1, Table 2, Figure 2, and Figure 3.

Table 1. Number of nodes searched by the BB and CPU time (ms) of algorithms for small-scale instances.

		Number of nodes searched by BB		BB		NEH		SA		GA	
n	β	Mean	Max	Mean	Max	Mean	Max	Mean	Max	Mean	Max
10	-0.15	11516.20	147645	1123.35	13422	0.00	0	0.70	1	2123.50	2329
	-0.2	2800.75	7431	250.65	703	0.00	0	0.50	1	1795.40	2192
	-0.25	3400.55	26867	270.40	2094	0.75	15	0.45	1	1624.25	1676
	-0.3	2240.95	6749	182.90	578	0.75	15	0.50	1	1605.05	1659
	-0.35	1843.20	11748	165.40	890	0.00	0	0.50	1	1603.50	1656
	-0.4	1744.15	11698	143.85	907	0.80	16	0.70	1	1582.55	1634
11	-0.15	29012.70	240218	2549.15	23125	0.00	0	0.65	1	1727.35	2182
	-0.2	13096.50	60728	1001.60	6719	0.80	16	0.35	1	1264.40	2153
	-0.25	23938.50	176965	1396.05	10016	0.00	0	0.40	1	1102.85	1125
	-0.3	16778.10	190799	998.55	11250	0.00	0	0.55	1	1115.85	1251
	-0.35	3406.85	28293	232.80	1781	0.00	0	0.60	1	1372.15	1526
	-0.4	6674.80	37503	485.25	2703	0.00	0	0.70	1	1379.85	1792
12	-0.15	27419.50	104697	2159.30	8344	0.00	0	0.70	1	1393.95	1476
	-0.2	43188.70	180912	3238.40	13469	0.75	15	0.40	1	1388.90	1514
	-0.25	56103.60	746074	4004.00	44313	0.00	0	0.70	1	1488.50	2207
	-0.3	15124.20	129029	1132.00	9532	0.00	0	0.65	1	1384.75	1475
	-0.35	48424.10	371619	3564.80	26829	0.00	0	0.50	1	1380.40	1471
	-0.4	8184.60	28761	676.60	2250	0.75	15	0.55	1	1545.80	2215
13	-0.15	533210.00	6038349	32171.10	332984	0.00	0	0.60	1	1380.00	1959
	-0.2	88869.60	393402	5557.00	24531	0.75	15	0.60	1	1132.95	1148
	-0.25	134124.00	1110586	8255.55	66906	0.80	16	0.55	1	1127.35	1142
	-0.3	33566.70	269258	2141.30	16688	0.00	0	0.45	1	1132.50	1146
	-0.35	27929.40	98258	1786.65	6515	0.00	0	0.70	1	1128.65	1148
	-0.4	56363.30	618972	3564.15	38765	0.00	0	0.75	1	1129.85	1151
14	-0.15	2093830.00	25082713	271917.00	3271750	0.80	16	1.15	3	2274.90	2453
	-0.2	1117230.00	18526460	118006.00	1911062	0.00	0	1.15	3	2025.80	2423
	-0.25	484132.00	3069941	32053.90	203063	0.00	0	0.60	1	1116.25	1134
	-0.3	202003.00	1273922	13416.50	83719	0.00	0	0.50	1	1113.25	1128
	-0.35	219143.00	1372558	14499.40	91625	0.75	15	0.45	1	1108.25	1121
	-0.4	52046.60	229074	3583.50	16078	0.00	0	0.75	1	1108.30	1126

Table 2. Error of algorithms for small-scale instances.

		$\frac{\widetilde{C}_{\max}(NEH)}{\widetilde{C}_{\max}(BB)}$		$\frac{\widetilde{C}_{\max}(SA)}{\widetilde{C}_{\max}(BB)}$		$\frac{\widetilde{C}_{\max}(GA)}{\widetilde{C}_{\max}(BB)}$	
n	β	Mean	Max	Mean	Max	Mean	Max
10	-0.15	1.01569	1.06864	1.00969	1.02720	1.00119	1.00541
	-0.2	1.01206	1.05986	1.01155	1.02475	1.00125	1.01137
	-0.25	1.01711	1.07489	1.01401	1.02906	1.00143	1.00625
	-0.3	1.01021	1.03101	1.01059	1.02797	1.00111	1.00565
	-0.35	1.01431	1.05952	1.01123	1.02486	1.00109	1.00529
	-0.4	1.01315	1.04636	1.01042	1.02436	1.00112	1.00539
11	-0.15	1.01420	1.04471	1.01507	1.03109	1.00323	1.01198
	-0.2	1.01016	1.02906	1.01421	1.02993	1.00309	1.01438
	-0.25	1.02080	1.04621	1.01517	1.02750	1.00300	1.01034
	-0.3	1.01830	1.09062	1.01738	1.03218	1.00207	1.00748
	-0.35	1.02052	1.07294	1.01505	1.03290	1.00213	1.00910
	-0.4	1.02728	1.07902	1.01848	1.03683	1.00220	1.00660
12	-0.15	1.02139	1.06731	1.02483	1.14058	1.00392	1.02650
	-0.2	1.01946	1.05493	1.01810	1.03535	1.00339	1.01215
	-0.25	1.01532	1.05430	1.02118	1.04991	1.00320	1.01147
	-0.3	1.01989	1.05974	1.02704	1.06135	1.00425	1.01659
	-0.35	1.02927	1.06383	1.02032	1.04178	1.00440	1.01267
	-0.4	1.02018	1.05674	1.02400	1.04751	1.00377	1.00883
13	-0.15	1.02200	1.07736	1.02091	1.03719	1.00654	1.01668
	-0.2	1.01580	1.06453	1.01865	1.03515	1.00486	1.01304
	-0.25	1.02217	1.05276	1.02295	1.03490	1.00504	1.01234
	-0.3	1.01953	1.04323	1.02572	1.05273	1.00736	1.02807
	-0.35	1.02348	1.07362	1.02364	1.04644	1.00454	1.01059
	-0.4	1.02590	1.08348	1.02739	1.06795	1.00641	1.01332
14	-0.15	1.01919	1.05575	1.01930	1.03594	1.00578	1.01843
	-0.2	1.02216	1.08155	1.02181	1.03866	1.00750	1.01875
	-0.25	1.02159	1.04337	1.02375	1.04412	1.00572	1.01160
	-0.3	1.02826	1.11703	1.02629	1.04621	1.00819	1.02184
	-0.35	1.02849	1.07206	1.02861	1.06066	1.00710	1.01745
	-0.4	1.02891	1.08183	1.03459	1.05752	1.00761	1.01627

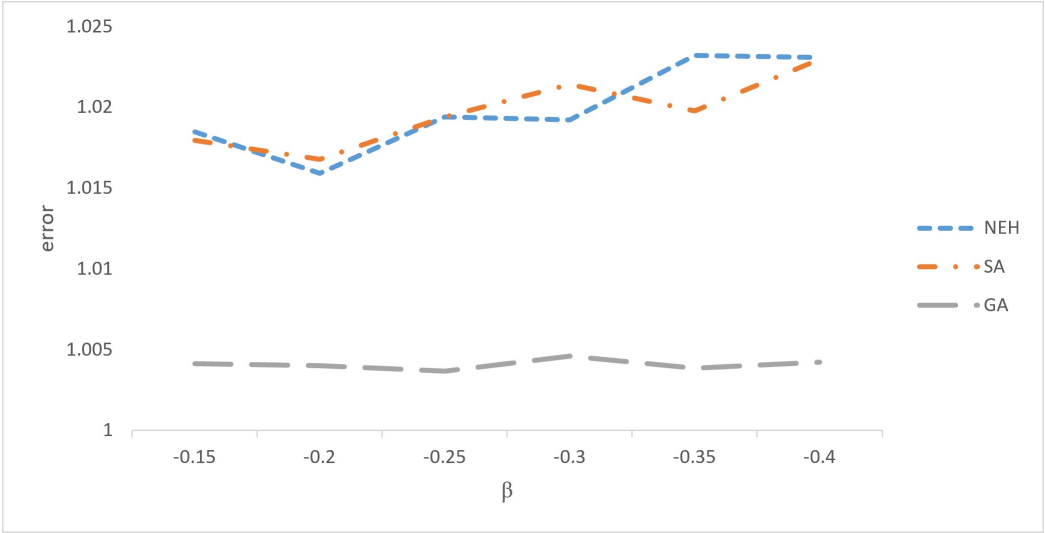

Figure 2. The effect of different learning factors on the mean error of various algorithms for small-scale instances.

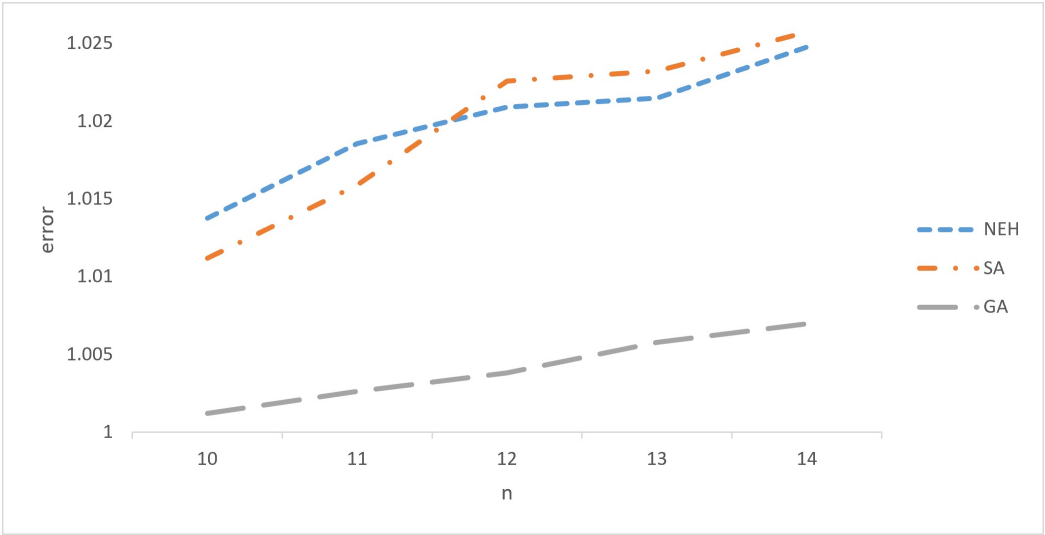

Figure 3. The effect of different scales on the mean error of various algorithms for small-scale instances.

From Table 1, we can see that the CPU time does not increase significantly with the increase in the number of jobs (n), except for the BB. Although the CPU time and number of searched nodes of BB increase significantly with the increase in the number of jobs (n), but the ratio of the number of searched nodes to the number of total nodes (i.e., $\frac{Num_{SN}}{Num_{TN}}$) decreases. For example, the maximum value of the maximum number of searched nodes is 147645 and the ratio $\frac{Num_{SN}}{Num_{TN}} = \frac{147645}{9864101} \approx 1.50\%$ when $n = 10$, but the ratio $\frac{Num_{SN}}{Num_{TN}} = \frac{25082713}{236975164805} \approx 0.01\%$ when $n = 14$. It suggests that the BB has good lower bounds. For the NEH algorithm, it can find a solution within microsecond-level time in the most cases which is better than other algorithms. In contrast, the SA algorithm generally obtains a solution within 1 ms, and its CPU time is second only to the NEH algorithm. The GA generally requires more than 1000 ms to solve each instance, which is slower than the other heuristic algorithms. From Table 2, for the $\widetilde{C}_{\max}$ minimization, the NEH algorithm and SA algorithm have good accuracy for different instances. The GA can find a good approximate solution in the shorter CPU time than the BB in many instances. The mean error of GA is generally less than 1.008, while the other algorithms are usually greater than 1.01. From Figure 2, the vertical axis denotes the average value of the mean errors of the algorithm across five scales (10, 11, 12, 13, and 14) at each learning factor. The GA outperforms the SA algorithm and the NEH algorithm and is less sensitive to the learning factor. For the SA algorithm and the NEH algorithm, the mean error increases as the learning factor decreases and the mean error of two algorithms are relatively close. From Figure 3, the vertical axis denotes the average value of the mean errors of the algorithm across six learning factor (-0.15, -0.2, -0.25, -0.3, -0.35, and -0.4) at each scale. Overall, the GA outperforms the SA algorithm and the NEH algorithm. The mean error of all algorithms rises with an increase in the problem size. This may be attributed to the explosive expansion of the solution space as the problem grows, which prevents the algorithms from conducting sufficient searches. In the small-scale problem, the GA has good performance in terms of the error and CPU time.

For the large-scale instances, the SPT is taken as the benchmark to compare the relative performance ratio of each algorithm. A smaller relative performance ratio closer to 0 indicates that the solution obtained by the algorithm outperforms the SPT's solution. The relative performance ratio formula of the solution generated by heuristic algorithms is as follows:

$$\frac{\widetilde{C}_{\max}(\phi)}{\widetilde{C}_{\max}(\phi^{SPT})},$$

where ϕ is a sequence obtained by the NEH, SA, and GA, and ϕ^{SPT} is a sequence obtained by the SPT's order of $\widehat{D}_j$. The corresponding results are summarized in Table 3, Table 4, Figure 4, and Figure 5. From Table 3, we can see that the CPU time increases rapidly with the increase of the number of jobs (n), especially the NEH algorithm and GA. The SA algorithm can find a solution in short time. In Table 4, as the problem size increases, it becomes more evident that the SA algorithm fails to find solutions superior to the SPT's solution, and its relative performance ratio gradually approaches 1. The results of the NEH algorithm are more accurate and stable than the SA and GA algorithms. From Figure 4, the vertical axis denotes the average value of the mean relative performance ratios of the algorithm across five scales (100, 200, 300, 400, and 500) at each learning factor. The NEH algorithm outperforms the SA algorithm and the GA. The NEH algorithm and the GA are less sensitive to the learning factor. In contrast, the mean relative performance ratio of the SA algorithm increases as the learning factor decreases. From Figure 5, the vertical axis denotes the average value of the mean errors of the algorithm

across six learning factors (-0.15, -0.2, -0.25, -0.3, -0.35, and -0.4) at each scale. Overall, the NEH algorithm outperforms the SA algorithm and the GA and its mean relative performance ratio decreases as the problem size increases. By contrast, the mean relative performance ratios of the SA algorithm and the GA rise as the problem size increases. In the large-scale problem, the NEH has good performance in terms of relative performance ratio and CPU time.

Table 3. CPU time (ms) of algorithms for large-scale instances.

		NEH		SA		GA	
n	β	Mean	Max	Mean	Max	Mean	Max
100	-0.15	259.35	359	41.70	57	7248.50	8456
	-0.2	271.85	375	41.05	57	7038.70	7937
	-0.25	304.00	391	49.45	60	8449.25	9200
	-0.3	254.75	312	40.20	45	6916.10	7181
	-0.35	275.80	328	44.30	59	7795.70	8765
	-0.4	305.30	359	48.35	57	8388.35	8606
200	-0.15	4314.10	4547	187.15	226	26629.80	27226
	-0.2	4336.70	4469	182.20	195	26624.20	27593
	-0.25	4362.60	4797	182.40	195	26660.30	28164
	-0.3	4387.45	4703	180.85	203	26866.60	28437
	-0.35	4374.35	4703	183.95	210	26818.90	28148
	-0.4	4331.15	4454	182.15	203	26691.70	27656
300	-0.15	21886.80	23985	404.05	461	58093.40	60164
	-0.2	18301.50	22031	344.65	445	48557.00	58164
	-0.25	16618.10	16687	312.05	320	44208.30	44383
	-0.3	16681.20	16906	312.00	320	44334.90	44828
	-0.35	16678.80	16843	313.30	343	44335.90	44671
	-0.4	16673.30	16781	312.80	320	44361.20	44570
400	-0.15	59454.60	65218	623.90	765	87598.10	96750
	-0.2	57117.10	59422	596.25	633	84817.20	91031
	-0.25	54878.20	66016	572.70	664	80262.20	85164
	-0.3	54006.40	54422	566.85	586	80266.70	86781
	-0.35	54879.10	64657	583.25	828	81163.10	86695
	-0.4	54163.20	55281	574.70	601	80994.30	86555
500	-0.15	145199.00	154953	973.50	1039	130790.00	140844
	-0.2	145904.00	148984	984.45	1015	131348.00	133054
	-0.25	146359.00	148391	988.85	1039	131668.00	132930
	-0.3	146819.00	148828	997.45	1015	131810.00	133930
	-0.35	146926.00	149375	995.90	1015	132034.00	133054
	-0.4	145058.00	157719	995.75	1172	137338.00	147742

Table 4. Relative performance ratio of algorithms for large-scale instances.

		$\frac{\widetilde{C}_{\max}(NEH)}{\widetilde{C}_{\max}(SPT)}$		$\frac{\widetilde{C}_{\max}(SA)}{\widetilde{C}_{\max}(SPT)}$		$\frac{\widetilde{C}_{\max}(GA)}{\widetilde{C}_{\max}(SPT)}$	
n	β	Mean	Max	Mean	Max	Mean	Max
100	-0.15	0.876127	0.899004	0.935122	0.951818	0.912634	0.925244
	-0.2	0.868217	0.898930	0.940429	0.962572	0.907646	0.921038
	-0.25	0.870367	0.894107	0.951623	0.977198	0.908274	0.928690
	-0.3	0.870403	0.900689	0.961285	0.997752	0.906771	0.938975
	-0.35	0.869608	0.910453	0.975925	0.999403	0.907753	0.939587
	-0.4	0.874456	0.903281	0.986623	0.998894	0.908154	0.931643
200	-0.15	0.866335	0.889517	0.953178	0.972333	0.933048	0.951192
	-0.2	0.865904	0.885376	0.965521	0.982577	0.932488	0.945216
	-0.25	0.859924	0.880771	0.968290	0.991347	0.927592	0.942305
	-0.3	0.861144	0.880401	0.982770	1.000000	0.929203	0.942729
	-0.35	0.864041	0.880474	0.997875	1.000000	0.930922	0.940390
	-0.4	0.861513	0.884776	0.998598	1.000000	0.931816	0.957141
300	-0.15	0.862091	0.880403	0.964683	0.984010	0.937545	0.948937
	-0.2	0.861734	0.876130	0.971885	0.983378	0.938988	0.947137
	-0.25	0.861403	0.875379	0.984518	0.999268	0.939229	0.950593
	-0.3	0.859315	0.871677	0.994717	1.000000	0.938318	0.945684
	-0.35	0.856758	0.873428	0.998922	1.000000	0.936146	0.948118
	-0.4	0.858429	0.871378	0.999243	1.000000	0.938760	0.949068
400	-0.15	0.857343	0.867456	0.967621	0.975379	0.944235	0.953760
	-0.2	0.859339	0.873203	0.979877	0.988811	0.946040	0.957657
	-0.25	0.859110	0.875624	0.992010	0.999570	0.945219	0.955414
	-0.3	0.853316	0.879112	0.996982	1.000000	0.943367	0.959876
	-0.35	0.854755	0.865837	0.999523	1.000000	0.942088	0.952624
	-0.4	0.850440	0.865810	0.999318	1.000000	0.942274	0.952504
500	-0.15	0.854211	0.869834	0.973013	0.984393	0.959531	0.966814
	-0.2	0.856910	0.866429	0.983734	0.994344	0.960979	0.968965
	-0.25	0.855871	0.870409	0.994421	1.000000	0.960822	0.970888
	-0.3	0.856887	0.869956	0.999234	1.000000	0.962346	0.974190
	-0.35	0.851839	0.859704	0.999499	1.000000	0.958882	0.969182
	-0.4	0.853722	0.864706	0.999641	1.000000	0.957004	0.966899

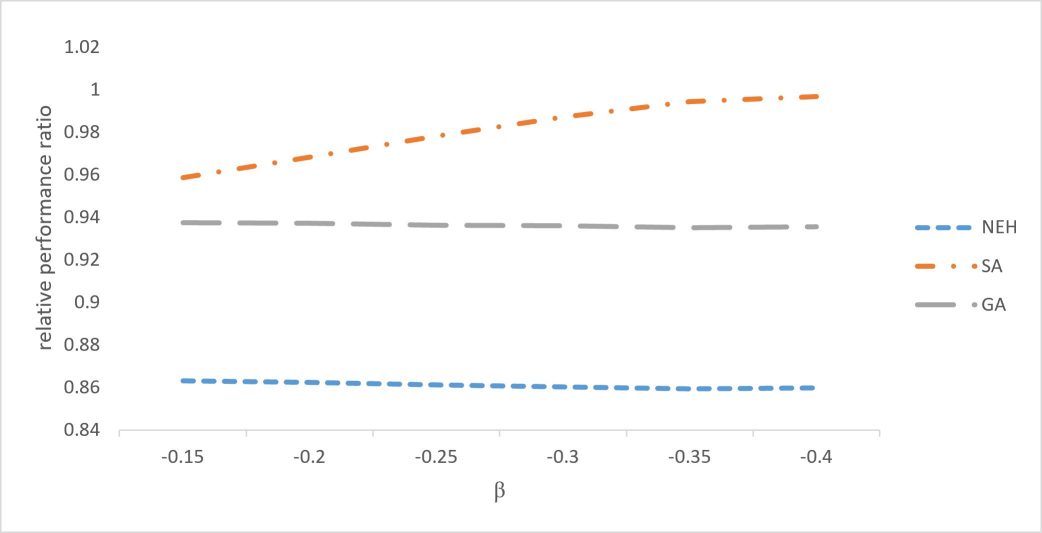

Figure 4. The effect of different learning factors on the mean error of various algorithms for large-scale instances.

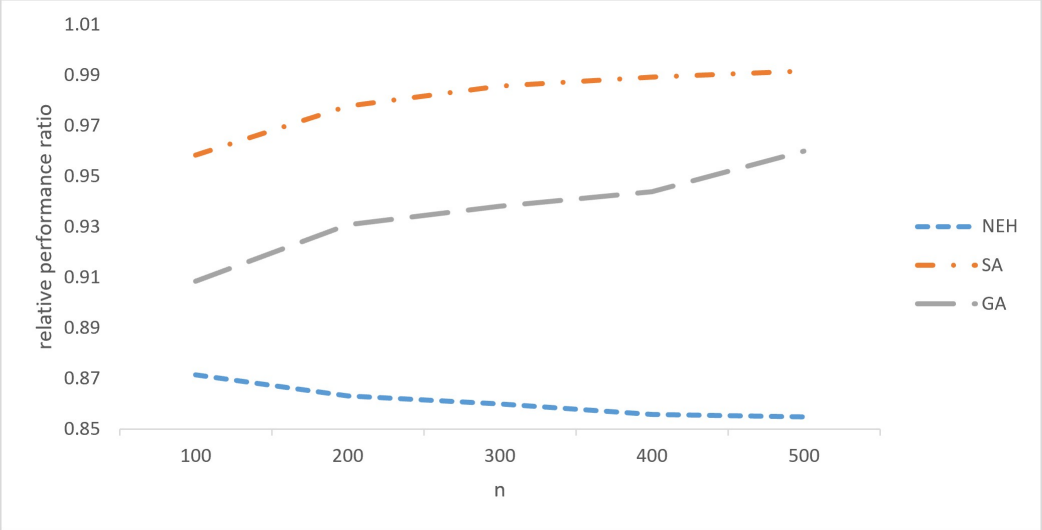

Figure 5. The effect of different scales on the mean error of various algorithms for large-scale instances.

7. Conclusion

We investigated the no-wait flow shop problem with the LE based on the sum of the logarithms of processing times. For the makespan minimization, we proposed polynomial algorithms to solve two special dominating machine constraints and two special fixed processing times on the machines. We also proposed the branch-and-bound algorithm, some heuristic and metaheuristic algorithms to solve the general problem. The experimental results demonstrated that the GA algorithm can solve small sized problems well and the NEH algorithm can solve large problems well. However, whether a polynomial-time algorithm exists for this problem remains unknown. Although the two-machine no-wait flow shop problem is common in the cold rolling production process, the obtained conclusions may not be extended to flow shop problems with more than two machines. Moreover, the generality of the adopted LE model needs to be further investigated. Despite these limitations, the development of these special cases, the branch-and-bound algorithm, and various heuristic and meta-heuristic algorithms in this paper also result in some directions for future research. First, it would be helpful to develop tighter lower bounds to enhance the effectiveness of the branch-and-bound algorithm. Second, these special cases can be helpful to develop other special polynomial solvable cases. Finally, these special cases and proposed algorithms can be adapted to solve other problems, for example, no-wait flow shop problems under group technology (see Yin and Gao [61], Lv and Wang [62], Yin et al. [63], Zhang and Wang [64], and Zhang and Yin [65]), the no-wait flow shop problems with deterioration effects (see Mao et al. [66], Sun et al. [67], Kong et al. [68], Kong et al. [69], Choi et al. [70], and Zhang et al. [71]), or the no-wait flow shop problems with resource allocations (see Qian and Guo [72], Wang et al. [73], Wang and Liu [74], Kovalev et al. [75], and Lu and Li [76]).

Author contributions

Jia-Ming Gu constructed the model, performed the theoretical derivations, designed the algorithms and conducted the numerical experiments; Jia-Ming Gu wrote the original draft; Lin Lin and Ji-Bo Wang presented the model and methodology, reviewed and edited the manuscript.

Use of generative AI tools declaration

The authors declare they have used artificial intelligence (AI) tools in the creation of this article. AI tools used: Doubao. How were the AI tools used: The AI tool assisted in linguistic polishing and tone refinement of the authors' original content. It was not used to generate research data or scientific conclusions, and all modifications were finalized by the authors. Where in the article is the information located: Throughout the manuscript, primarily for linguistic polishing and formatting consistency.

Acknowledgments

The authors acknowledge the helpful comments from the editor and anonymous reviewers. This research was funded by the Natural Science Foundation of Liaoning Province (2026-BS-0304) and the Fundamental Research Funds for the Universities of Liaoning Province (LJ212610143035).

Conflict of interest

The authors declare that there is no conflict of interest.

References

1. T. Wu, R. Q. Chen, Heuristic algorithm of scheduling model for cold coils annealing, *2006 6th World Congress on Intelligent Control and Automation*, (2006), 7263–7266. https://doi.org/10.1109/WCICA.2006.1713398
2. Y. Lu, F. Teng, Z. Feng, Scheduling jobs with truncated exponential sum-of-logarithm-processing-times based and position-based learning effects, *Asia-Pac J. Oper. Res.*, **32** (2015), 1–17. https://doi.org/10.1142/S0217595915500268
3. X. Zhang, Y. Wang, S. Bai, Single-machine group scheduling problems with deteriorating and learning effect, *Int. J. Syst. Sci.*, **47** (2016), 2402–2410. https://doi.org/10.1080/00207721.2014.998739
4. Z. Jiang, F. Chen, X. Zhang, Single-machine scheduling with times-based and job-dependent learning effect, *J. Oper. Res. Soc.*, **68** (2017), 809–815. https://doi.org/10.1057/jors.2016.40
5. N. Yin, Single machine due window assignment resource allocation scheduling with job-dependent learning effect, *J. Appl. Math. Comput.*, **56** (2018), 715–725. https://doi.org/10.1007/s12190-017-1091-6
6. A. Azzouz, M. Ennigrou, L. Ben Said, Scheduling problems under learning effects: classification and cartography, *Int. J. Prod. Res.*, **56** (2018), 1642–1661. https://doi.org/10.1080/00207543.2017.1355576
7. C. Wu, W. Lin, X. Zhang, I. Chung, T. Yang, K. Lai, Tardiness minimisation for a customer order scheduling problem with sum-of-processing-time-based learning effect, *J. Oper. Res. Soc.*, **70** (2019), 487–501. https://doi.org/10.1080/01605682.2018.1447249
8. J. Wang, M. Gao, J. Wang, L. Liu, H. He, Scheduling with a position-weighted learning effect and job release dates, *Eng. Optim.*, **52** (2020), 1475–1493. https://doi.org/10.1080/0305215X.2019.1664498
9. S. Zhao, Scheduling jobs with general truncated learning effects including proportional setup times, *Comput. Appl. Math.*, **41** (2022), 146. https://doi.org/10.1007/s40314-022-01851-0
10. R. Ma, L. Zhang, Y. Zhang, A best possible algorithm for an online scheduling problem with position-based learning effect, *J. Ind. Manag. Optim.*, **18** (2022), 3989–4010. https://doi.org/10.3934/jimo.2021144
11. Y. Jiang, Z. Zhang, X. Song, Y. Yin, Seru scheduling problems with multiple due-windows assignment and learning effect, *J. Syst. Sci. Syst. Eng.*, **31** (2022), 480–511. https://doi.org/10.1007/s11518-022-5534-8
12. D. Deliktas, Self-adaptive memetic algorithms for multi-objective single machine learning-effect scheduling problems with release times, *Flex. Serv. Manuf. J.*, **34** (2022), 748–784. https://doi.org/10.1007/s10696-021-09434-7
13. N. Ren, J. Wang, E. Wang, Research on delivery times scheduling with truncated learning effects, *Comput. Appl. Math.*, **42** (2023), 243. https://doi.org/10.1007/s40314-023-02379-7
14. Y. Luo, Stochastic single machine scheduling with time-dependent deterioration or position-dependent learning effect, *Optim. Lett.*, **17** (2023), 1811–1832. https://doi.org/10.1007/s11590-022-01955-w

15. J. Wang, D. Lv, J. Xu, P. Ji, F. Li, Bicriterion scheduling with truncated learning effects and convex controllable processing times, *Int. Trans. Oper. Res.*, **28** (2021), 1573–1593. https://doi.org/10.1111/itor.12888

16. B. Mor, D. Mor, N. Shani, D. Shapira, Single machine scheduling with generalized due-dates, learning effect, and job-rejection, *J. Appl. Math. Comput.*, **70** (2024), 6063–6083. https://doi.org/10.1007/s12190-024-02198-x

17. W. Liu, X. Wang, L. Li, W. Dai, Due-window assignment scheduling with job-rejection, truncated learning effects and setup times, *J. Ind. Manag. Optim.*, **20** (2024), 313–324. https://doi.org/10.3934/jimo.2023079

18. J. Gao, J. Zou, Y. Zhang, Single-machine scheduling with a deteriorating maintenance activity and DeJong's learning effect, *Int. J. Found. Comput. Sci.*, **36** (2025), 635–648. https://doi.org/10.1142/S0129054122460030

19. Y. Zhang, X. Sun, T. Liu, J. Wang, X. Geng, Single-machine scheduling simultaneous consideration of resource allocations and exponential time-dependent learning effects, *J. Oper. Res. Soc.*, **76** (2025), 528–540. https://doi.org/10.1080/01605682.2024.2371527

20. B. Cheng, Y. Wang, M. Zhou, X. Zhu, Integrated scheduling of batch production and intermediate delivery with batch-position-based learning effect, *Asia-Pac. J. Oper. Res.*, **42** (2025), 2550003. https://doi.org/10.1142/S0217595925500034

21. L. Wang, Due-window assignment scheduling problems with position-dependent weights, truncated learning effects and past-sequence-dependent setup-times, *Symmetry*, **18** (2026), 396. https://doi.org/10.3390/sym18030396

22. H. Song, J. Yang, J. Wang, Study on multiple due-windows assignment scheduling with learning and deteriorating effects, *Asia-Pac. J. Oper. Res.*, (2026), 2650001. https://doi.org/10.1142/S0217595926500016

23. B. Cheng, W. Bao, J. Gao, Optimising batch operations under energy constraint with truncated position-based learning effect and ageing effect, *J. Oper. Res. Soc.*, (2026), 1–16. https://doi.org/10.1080/01605682.2026.2621244

24. D. Shabtay, D. Oron, Proportionate flow-shop scheduling with rejection, *J. Oper. Res. Soc.*, **67** (2016), 752–769. https://doi.org/10.1057/jors.2015.95

25. N. Zhao, S. Ye, K. Li, S. Chen, Effective iterated greedy algorithm for flow-shop scheduling problems with time lags, *Chin. J. Mech. Eng.*, **30** (2017), 652–662. https://doi.org/10.1007/s10033-017-0108-2

26. S. Sheikh, G. M. Komaki, V. Kayvanfar, E. Teymourian, Multi-stage assembly flow shop with setup time and release time, *Oper. Res. Perspect.*, **6** (2019), 100111. https://doi.org/10.1016/j.orp.2019.100111

27. M. Al-Behadili, D. Ouelhadj, D. Jones, Multi-objective biased randomised iterated greedy for robust permutation flow shop scheduling problem under disturbances, *J. Oper. Res. Soc.*, **71** (2020), 1847–1859. https://doi.org/10.1080/01605682.2019.1630330

28. J. Wang, D. Lv, C. Wan, Proportionate flow shop scheduling with job-dependent due windows and position-dependent weights, *Asia-Pac. J. Oper. Res.*, **42** (2025), 2450011. https://doi.org/10.1142/S0217595924500118

29. R. Jin, W. Tong, Y. Xu, J. Dong, Two-stage flexible flow shop scheduling in GPU systems, *J. Oper. Res. Soc.*, **77** (2026), 793–804. https://doi.org/10.1080/01605682.2025.2504435

30. C. Wu, J. Chen, W. Lin, K. Lai, D. Bai, S. Lai, A two-stage three-machine assembly scheduling flowshop problem with both two-agent and learning phenomenon, *Comput. Ind. Eng.*, **130** (2019), 485–499. https://doi.org/10.1016/j.cie.2019.02.047

31. X. Sun, X. Geng, F. Liu, Flow shop scheduling with general position weighted learning effects to minimise total weighted completion time, *J. Oper. Res. Soc.*, **72** (2021), 2674–2689. https://doi.org/10.1080/01605682.2020.1806746

32. D. Lv, J. Wang, Considering the peak power consumption problem with learning and deterioration effect in flow shop scheduling, *Comput. Ind. Eng.*, **197** (2024), 110599. https://doi.org/10.1016/j.cie.2024.110599

33. M. Li, D. Lv, L. Zhang, J. Wang, Permutation flow shop scheduling with makespan objective and truncated learning effects, *J. Appl. Math. Comput.*, **70** (2024), 2907–2939. https://doi.org/10.1007/s12190-024-02080-w

34. Y. A. Paredes-Astudillo, V. Botta-Genoulaz, J. R. Montoya-Torres, Impact of learning effect modelling in flowshop scheduling with makespan minimisation based on the Nawaz-Enscore-Ham algorithm, *Int. J. Prod. Res.*, **62** (2024), 1999–2014. https://doi.org/10.1080/00207543.2023.2204967

35. D. Lv, J. Wang, Research on two-machine flow shop scheduling problem with release dates and truncated learning effects, *Eng. Optim.*, **57** (2025), 1828–1848. https://doi.org/10.1080/0305215X.2024.2372633

36. K. Zhang, J. Wang, Y. Lu, On flowshop scheduling with the DeJong's learning effect, *Asia-Pac. J. Oper. Res.*, (2026). https://doi.org/10.1142/S0217595926500296

37. M. S. Nagano, H. H. Miyata, Review and classification of constructive heuristics mechanisms for no-wait flow shop problem, *Int. J. Adv. Manuf. Technol.*, **86** (2016), 2161–2174. https://doi.org/10.1007/s00170-015-8209-5

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

39. D. M. Utama, S. Z. Umamy, C. N. Al-Imron, No-Wait Flow Shop scheduling problem: a systematic literature review and bibliometric analysis, *RAIRO Oper. Res.*, **58** (2024), 1281–1313. https://doi.org/10.1051/ro/2024008

40. H. Samarghandi, M. Behroozi, On the exact solution of the no-wait flow shop problem with due date constraints, *Comput. Oper. Res.*, **81** (2017), 141–159. https://doi.org/10.1016/j.cor.2016.12.013

41. H. Ye, W. Li, A. Abedini, B. Nault, An effective and efficient heuristic for no-wait flow shop production to minimize total completion time, *Comput. Ind. Eng.*, **108** (2017), 57–69. https://doi.org/10.1016/j.cie.2017.04.002

42. C. Koulamas, G. J. Kyparisis, The no-wait flow shop with rejection, *Int. J. Prod. Res.*, **59** (2021), 1852–1859. https://doi.org/10.1080/00207543.2020.1727042

43. Y. Li, C. Zheng, X. Pei, Multi-search mechanism-improved differential evolution algorithm for the no-wait flow shop scheduling problem, *Appl. Soft Comput.*, **175** (2025), 113094. https://doi.org/10.1016/j.asoc.2025.113094

44. Y. Liu, Z. Feng, Two-machine no-wait flowshop scheduling with learning effect and convex resource-dependent processing times, *Comput. Ind. Eng.*, **75** (2014), 170–175. https://doi.org/10.1016/j.cie.2014.06.017

45. F. Gao, M. Liu, J. Wang, Y. Lu, No-wait two-machine permutation flow shop scheduling problem with learning effect, common due date and controllable job processing times, *Int. J. Prod. Res.*, **56** (2018), 2361–2369. https://doi.org/10.1080/00207543.2017.1371353

46. W. Liu, C. Jiang, Flow shop resource allocation scheduling with due date assignment, learning effect and position-dependent weights, *Asia-Pac. J. Oper. Res.*, **37** (2020), 2050014. https://doi.org/10.1142/S0217595920500141

47. D. Lv, J. Wang, Study on resource-dependent no-wait flow shop scheduling with different due-window assignment and learning effects, *Asia-Pac. J. Oper. Res.*, **38** (2021), 2150008. https://doi.org/10.1142/S0217595921500081

48. Y. Sun, D. Lv, X. Huang, Properties for due window assignment scheduling on a two-machine no-wait proportionate flow shop with learning effects and resource allocation, *J. Oper. Res. Soc.*, **77** (2026), 599–615. https://doi.org/10.1080/01605682.2025.2492817

49. J. Gu, J. Wang, Minimizing total completion time in no-wait flow shop scheduling with learning effects and group technology, *J. Oper. Res. Soc.*, (2026), 1–22. https://doi.org/10.1080/01605682.2026.2721567

50. T. C. E. Cheng, P. Lai, C. Wu, W. Lee, Single-machine scheduling with sum-of-logarithm-processing-times-based learning considerations, *Inf. Sci.*, **197** (2009), 3127–3135. https://doi.org/10.1016/j.ins.2009.05.002

51. D. Biskup, A state-of-the-art review on scheduling with learning effects, *Eur. J. Oper. Res.*, **188** (2008), 315–329. https://doi.org/10.1016/j.ejor.2007.05.040

52. P. C. Gilmore, R. E. Gomory, Sequencing a one state-variable machine: A solvable case of the traveling salesman problem, *Oper. Res.*, **12** (1964), 655–679. https://doi.org/10.1287/opre.12.5.655

53. S. S. Reddi, C. V. Ramamoorthy, On the flow-shop sequencing problem with no wait in process, *J. Oper. Res. Soc.*, **23** (1972), 323–331. https://doi.org/10.1057/jors.1972.52

54. M. Nawaz, E. E. Enscore Jr, I. Ham, A heuristic algorithm for the m-machine,n-job flow-shop sequencing problem, *Omega*, **11** (1983), 91–95. https://doi.org/10.1016/0305-0483(83)90088-9

55. I. H. Osman, C. N. Potts, Simulated annealing for permutation flow-shop scheduling, *Omega*, **17** (1989), 551–557. https://doi.org/10.1016/0305-0483(89)90059-5

56. L. Zhang, M. Li, L. Lin, Study on controllable processing time and minmax group scheduling with common due-window assignment, *Symmetry*, **18** (2026), 358. https://doi.org/10.3390/sym18020358

57. Z. Sun, D. Lv, P. Ji, J. Wang, Minimizing total completion time scheduling problem with ready times and linear deterioration functions of processing times, *J. Appl. Math. Comput.*, **72** (2026), 42. https://doi.org/10.1007/s12190-025-02694-8

58. Z. Wang, Optimal scheduling of flow shop based on genetic algorithm, *J. Adv. Manuf. Syst.*, **21** (2022), 111–123. https://doi.org/10.1142/S021968672150044X

59. O. A. Arik, Genetic algorithm application for permutation flow shop scheduling problems, *Gazi Univ. J. Sci.*, **35** (2022), 92–111. https://doi.org/10.35378/gujs.682388

60. M. S. Umam, M. Mustafid, S. Suryono, A hybrid genetic algorithm and tabu search for minimizing makespan in flow shop scheduling problem, *J. King Saud Univ. Comput. Inf. Sci.*, **34** (2022), 7459–7467. https://doi.org/10.1016/j.jksuci.2021.08.025

61. N. Yin, M. Gao, Single-machine group scheduling with general linear deterioration and truncated learning effects, *Comput. Appl. Math.*, **43** (2024), 386. https://doi.org/10.1007/s40314-024-02881-6

62. D. Lv, J. Wang, Single-machine group technology scheduling with resource allocation and slack due window assignment including minmax criterion, *J. Oper. Res. Soc.*, **76** (2025), 1696–1712. https://doi.org/10.1080/01605682.2024.2430351

63. N. Yin, H. He, Y. Zhao, Y. Chang, N. Wang, Integrating group setup time deterioration effects and job processing time learning effects with group technology in single-machine green scheduling, *Axioms*, **14** (2025), 480. https://doi.org/10.3390/axioms14070480

64. L. Zhang, J. Wang, Resource allocation and minmax scheduling under group technology and different due-window assignments, *Axioms*, **14** (2025), 827. https://doi.org/10.3390/axioms14110827

65. L. Zhang, N. Yin, Study on single-machine group scheduling with convex resource allocations and different due-date assignments, *Bull. Malays. Math. Sci. Soc.*, **49** (2026), 33. https://doi.org/10.1007/s40840-025-02031-z

66. R. Mao, Y. Wang, D. Lv, J. Wang, Y. Lu, Delivery times scheduling with deterioration effects in due window assignment environments, *Mathematics*, **11** (2023), 3983. https://doi.org/10.3390/math11183983

67. X. Sun, T. Liu, X. Geng, Y. Hu, J. Xu, Optimization of scheduling problems with deterioration effects and an optional maintenance activity, *J. Sched.*, **26** (2023), 251–266. https://doi.org/10.1007/s10951-022-00756-4

68. F. Kong, C. Miao, Y. Huo, J. Song, Y. Zhang, Single-machine scheduling with step-deteriorating jobs and rejection, *J. Oper. Res. Soc. China*, **12** (2024), 1088–1102. https://doi.org/10.1007/s40305-023-00481-5

69. Q. Kong, C. Wei, J. Wang, Minmax delivery completion time scheduling with delivery times and deteriorating jobs, *Comput. Appl. Math.*, **45** (2026), 276. https://doi.org/10.1007/s40314-026-03684-7

70. B. Choi, E. Kim, J. Lee, Scheduling step-deteriorating jobs on a single machine with multiple critical dates, *J. Oper. Res. Soc.*, **76** (2025), 1553–1563. https://doi.org/10.1080/01605682.2024.2419979

71. L. Zhang, J. Wang, P. Ji, Group parallel-batching scheduling with position-dependent learning and time-dependent deterioration effects, *Comput. Appl. Math.*, **46** (2027), 45. https://doi.org/10.1007/s40314-026-03899-8

72. J. Qian, Z. Guo, Common due window assignment and single machine scheduling with delivery time, resource allocation, and job-dependent learning effect, *J. Appl. Math. Comput.*, **70** (2024), 4441–4471. https://doi.org/10.1007/s12190-024-02090-8

73. J. Wang, Y. Wang, C. Wan, D. Lv, L. Zhang, Controllable processing time scheduling with total weighted completion time objective and deteriorating jobs, *Asia-Pac. J. Oper. Res.*, **41** (2024), 2350026. https://doi.org/10.1142/S0217595923500264

74. X. Wang, W. Liu, Single machine group scheduling jobs with resource allocations subject to unrestricted due date assignments, *J. Appl. Math. Comput.*, **70** (2024), 6283–6308. https://doi.org/10.1007/s12190-024-02216-y

75. S. Kovalev, I. Chalamon, A. Becuwe, Single machine scheduling with resource constraints: Equivalence to two-machine flow-shop scheduling for regular objectives, *J. Oper. Res. Soc.*, **75** (2024), 1343–1346. https://doi.org/10.1080/01605682.2023.2244529

76. Y. Lu, M. Li, A unified analysis for single-machine scheduling with resource allocations, learning and deteriorating effects simultaneously, *Comput. Appl. Math.*, **45** (2026), 388. https://doi.org/10.1007/s40314-026-03784-4

AIMS Press