Computational and Applied Mathematics (2024) 43:386
https://doi.org/10.1007/s40314-024-02881-6

Single-machine group scheduling with general linear deterioration and truncated learning effects

Na Yin[1] · Ming Gao[2]

Received: 30 May 2024 / Revised: 19 July 2024 / Accepted: 27 July 2024 /
Published online: 29 August 2024

Abstract

In this paper we consider single-machine scheduling problems with group technology, where the group setup times are general linear functions of their starting times and the jobs in the same group have general truncated learning effects. The objective is to minimize the makespan and total completion time, respectively. We show that the makespan minimization remains polynomially solvable. For the total completion time minimization, optimal properties are presented and then we introduce some heuristic algorithms and a branch-and-bound algorithm.

Keywords Scheduling · Deteriorating job · Learning effect · Group technology · Single-machine

Mathematics Subject Classification 90B35 · 68M20

List of symbols

GT	Group technology
m	Number of groups ($m \geq 2$)
G_i	Group i, $i = 1, 2, \ldots, m$
n_i	Number of jobs belonging to G_i, $i = 1, 2, \ldots, m$
n	Total number of jobs, i.e., $n_1 + n_2 + \cdots + n_m = n$
J_{ij}	Job j in G_i, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n_i$
p_{ij}	Normal processing time of J_{ij}, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n_i$
$p_{i[j]}$	Normal processing time of $J_{i[j]}$ scheduled in the jth position of G_i, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n_i$
a_i	Normal setup time of G_i, $i = 1, 2, \ldots, m$
b_i	Setup deterioration rate of G_i, $i = 1, 2, \ldots, m$

✉ Na Yin
shenyinyin@126.com

✉ Ming Gao
gm@dufe.edu.cn

1 School of Science, Shenyang Aerospace University, Shenyang 110136, People's Republic of China

2 School of Management Science and Engineering, Dongbei University of Finance and Economics, Dalian 116025, People's Republic of China

t_i	Start setup time of G_i, $i = 1, 2, \ldots, m$
θ_i	Truncation parameter of G_i, $i = 1, 2, \ldots, m$
s_i^{AST}	Actual setup time of G_i, $i = 1, 2, \ldots, m$
p_{ijr}^{APT}	Actual processing time of job J_{ij} scheduled in the rth position of G_i, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n_i$
$\alpha_i\ (\alpha_{i1}, \alpha_{i2}, \alpha_{i3})$	Learning index of G_i, $i = 1, 2, \ldots, m$
$\beta_i\ (\beta_{i1}, \beta_{i2}, \beta_{i3})$	Learning index of G_i, $i = 1, 2, \ldots, m$
ϱ :	A job schedule of n jobs
$C_{ij} = C_{ij}(\pi)$	Completion time for job J_{ij} in π
$C_{\max}$	Makespan, i.e., $C_{\max} = \max\{C_{ij} \mid i = 1, 2, \ldots, m, j = 1, 2, \ldots, n_i\}$
$\sum\sum C_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n_i} C_{ij}$	Total completion time of all jobs

1 Introduction

In classical scheduling theory, it is assumed that the job processing times fixed and constant values. In practice, however, we often encounter settings in which job processing times may be subject to change due to the phenomenon of deterioration and/or learning. Extensive surveys of scheduling problems involving deteriorating jobs (time-dependent processing times) can be found in Alidaee and Womer (1999), Cheng et al. (2004), Yin et al. (2015) and Gawiejnowicz (2020a, b). More recently, Pei et al. (2019) considered the parallel-batching scheduling with step-deteriorating jobs. For the objective of maximising the total net revenue, they proposed some solution algorithms. Qian and Han (2022) (resp. Miao et al. 2023; Mao et al. 2023; Lu et al. 2024) studied the due-date (resp. due-window) assignment scheduling with delivery times and deterioration effects. Sun et al. (2023) considered the single-machine maintenance activity scheduling with deterioration effects. Wang et al. (2023b) and Wang et al. (2024d) addressed single-machine resource allocation scheduling with deterioration effects. Zhang et al. (2023) and Li et al. (2024b) studied the single-machine two-agent scheduling with resource allocations and deterioration effects. Lv and Wang (2024a) considered the no-idle flow shop scheduling with deterioration effects. Under common due date, they proved that some special cases are polynomially solvable. Lv et al. (2024) studied the single-machine scheduling with ready times and deterioration effects. For the total weighted completion time minimization, they proposed a branch-and-bound algorithm and some heuristic algorithms. Mao et al. (2024) considered the single-machine delivery times scheduling with general deterioration effects. They proved that the makespan minimization is polynomially solvable. Zhang et al. (2024) studied the single-machine scheduling problems with deteriorating jobs. For the slack due window, they proved that the minmax type problem is polynomially solvable.

In addition, extensive surveys of research related to scheduling with learning effects were presented by Biskup (2008) and Azzouz et al. (2018). More recently, Qian and Zhan (2021) and Wang et al. (2023a) studied the single-machine scheduling with delivery times and truncated learning effect. Under the due date assignment, they proved that some problems can be solved in polynomial time. Sun et al. (2021) (resp. Li et al. 2024c) considered the flow shop scheduling with learning effects (resp. truncated learning effects). For the total weighted completion time (resp. makespan) minimization, Sun et al. (2021) (resp. Li et al. 2024c) proposed some solution algorithms. Wang et al. (2021), Wang and Wang (2023) and Qian et al. (2024) considered the single-machine resource allocation scheduling with learning effects. Zhao (2022) (resp. Wang et al. 2024b) studied the single-machine scheduling problems with truncated learning effects and setup (resp. delivery) times. Ma et al. (2023) considered the single-machine online scheduling with learning effect. For the sum of completion times, they designed an optimal online algorithm. Lv and Wang (2024b) addressed the two-machine flow shop scheduling with release dates and truncated learning effects. For the total completion time minimization, they proposed a branch-and-bound algorithm and some heuristic algorithms.

On the other hand, the production efficiency can be increased by grouping various parts and products with similar designs and/or production processes, this phenomenon is known as group technology. Group technology that groups similar products into families helps increase the efficiency of operations and decrease the requirement of facilities (Potts and Van Wassenhove 1992; Kuo and Yang 2006; Wu and Lee 2008a; Wu et al. 2008b; Wang et al. 2008; Yan et al. 2008; Wang et al. 2009 and Lee and Wu 2009). He and Sun (2015) considered single-machine group scheduling problems with deterioration and learning effects. They showed that the makespan and some special cases of total completion time minimizations remain polynomially solvable. Huang (2019) and Liu et al. (2019) studied single-machine group scheduling with deteriorating jobs. For the primary (resp. secondary) criterion of minimizing the total weighted completion time (resp. maximum cost), they proved that the bicriterion problem is polynomially solvable. Under ready times and makespan minimization, Liu et al. (2019) proposed some solution algorithms. Miao (2019) studied parallel-batch makespan minimization scheduling with deteriorating jobs. Under group technology, she proved that two single-machine problems are polynomially solvable. Xu et al. (2021) investigated single-machine group scheduling with deteriorating jobs and nonperiodical maintenance. He et al. (2023) dealt with single-machine group scheduling with due-date assignment and resource allocation. To solve the generalized case of minimizing earliness-tardiness cost, they proposed a branch-and-bound and heuristic algorithms. Yan et al. (2023) and Qian (2023) considered single-machine group scheduling problems with learning effects and resource allocation. Li et al. (2024a) addressed single-machine group scheduling with convex resource allocation, learning effects and due date assignment. To solve the generalized case of minimizing the weighted sum of earliness, tardiness, common due date, resource consumption, and makespan, they proposed the heuristic and branch-and-bound algorithms. Liu and Wang (2023) and Wang and Liu (2024) studied single-machine group scheduling with resource allocation. Under the due date, they proved that some special cases of the problem are polynomially solvable.

Recently, Sun and Ma (2020) considered single-machine group scheduling with learning effects and deteriorating jobs. They proved that the makespan and the total completion time (under some special cases) problems remain polynomially solvable. This paper extends the results of Sun and Ma (2020), by focusing on group setup times are general linear functions of their starting times and the jobs in the same group have general truncated learning effects. The main contributions of this article are given as follows: i) Single-machine group scheduling with general linear deterioration and truncated learning effects is modeled and studied; ii) For the makespan minimization, we prove the problem is polynomially solvable; iii) For general case of the total completion time minimization, we propose some heuristics and a branch-and-bound algorithm. The remaining part of the paper is organized as follows. In the next section, a precise formulation is given. The problem of minimizing the makespan (resp. total completion time) is given in Sect. 3 (resp. Sect. 4). The last section contains conclusions.

2 Problem formulation

In this section, we first define the notation that is used throughout this paper (see the symbols before the Introduction), followed by the description of the problem.

There are n jobs grouped into m groups, and these n jobs are to be processed on a single machine. A setup time is required if the machine switches from one group to another and all setup times of groups for processing at time $t_0 \geq 0$. The machine can handle one job at a time and job preemption is not allowed. Let n_i be the number of jobs belonging to group G_i, thus, $n_1 + n_2 + \cdots + n_m = n$. Let J_{ij} denote the jth job in group G_i, $i = 1, 2, \ldots, m$; $j = 1, 2, \ldots, n_i$. As in Browne and Yechiali (1990), the actual setup time of G_i is:

$$s_i^{AST} = a_i + b_i t_i, i = 1, 2, \ldots, m, \tag{1}$$

where a_i (resp. b_i) is the normal setup time (resp. setup deterioration rate) of G_i and t_i denotes the start setup time of G_i. As in Zhao (2022), if job J_{ij} in group G_i is scheduled in the rth position, then the actual job processing time of it is

$$p_{ijr}^{APT} = p_{ij} \max \left\{ f_i \left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) \right) g_i(r), \theta_i \right\}, i = 1, 2, \ldots, m; r, j = 1, 2, \ldots, n_i, \tag{2}$$

where p_{ij} is the normal (basic) processing time of job J_{ij}, $\frac{df_i(x)}{dx} \leq 0$, $\frac{d^2 f_i(x)}{dx^2} \geq 0$, $\frac{d^2 h_i(x)}{dx^2} \leq 0$, $g_i(r)$ is a non-increasing function with $f_i(0) = 1$, $h_i(0) = 0$, $g_i(1) = 1$ and $0 \leq \theta_i \leq 1$ is a truncation parameter of group G_i, $\sum_{l=1}^{0} h_i(p_{i[l]}) := 0$. Note that Yan et al. (2008), Yang and Yang (2010) considered the following models: $p_{ijr}^{APT} = p_{ij} r^{\alpha_{i1}}$, $i = 1, 2, \ldots, m; r, j = 1, 2, \ldots, n_i$, and $p_{ijr}^{APT} = p_{ij} \left(1 + p_{i[1]} + p_{i[2]} + \cdots + p_{i[r-1]}\right)^{\alpha_{i2}}$, $i = 1, 2, \ldots, m; r, j = 1, 2, \ldots, n_i$, where $\alpha_{i1} \leq 0$ ($\alpha_{i2} \leq 0$) is a constant learning index of group G_i. As in Wang and Xia (2005) and Cheng et al. (2009), the job processing time models also are: $p_{ijr}^{APT} = p_{ij} \left(1 + \ln p_{i[1]} + \ln p_{i[2]} + \ldots + \ln p_{i[r-1]}\right)^{\alpha_{i3}}$, $p_{ijr}^{APT} = p_{ij} {\beta_{i1}}^{r-1}$, $p_{ijr}^{APT} = p_{ij} {\beta_{i2}}^{(p_{i[1]} + p_{i[2]} + \cdots + p_{i[r-1]})}$, $i = 1, 2, \ldots, m; r, j = 1, 2, \ldots, n_i$, and $p_{ijr}^{APT} = p_{ij} {\beta_{i3}}^{(\ln p_{i[1]} + \ln p_{i[2]} + \cdots + \ln p_{i[r-1]})}$, $i = 1, 2, \ldots, m; r, j = 1, 2, \ldots, n_i$, where $\alpha_{i3} \leq 0$, $0 < \beta_{i1}, \beta_{i2}, \beta_{i3} \leq 1$ is a constant learning index of group G_i.

For a given schedule ϱ, let $C_{ij}(\varrho)$ represent completion time of job J_{ij} in group G_i. The objective is to find a schedule that minimizes the makespan $C_{\max} = \max\{C_{ij} | i = 1, 2, \ldots, f; j = 1, 2, \ldots, n_i\}$ and total completion time $\sum\sum C_{ij} = \sum_{i=1}^{m} \sum_{j=1}^{n_i} C_{ij}$. By using the three-field notation, the problems can be denoted by

$$1 \left| s_i^{AST} = a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} \max \left\{ f_i \left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) \right) g_i(r), \theta_i \right\}, GT \right| C_{\max} \tag{3}$$

and

$$1 \left| s_i^{AST} = a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} \max \left\{ f_i \left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) \right) g_i(r), \theta_i \right\}, GT \right| \sum\sum C_{ij}, \tag{4}$$

where GT denotes the group technology. Sun and Ma (2020) considered the learning effect model $p_{ijr}^{APT} = p_{ij} F_i \left(\sum_{l=1}^{r-1} \chi_{il} p_{i[l]}, r\right)$, where $\chi_{i1} \leq \chi_{i2} \leq \cdots \leq \chi_{in_i}$,

$0 \leq F_i\left(\sum_{l=1}^{r-1} \chi_{il} p_{i[l]}, r\right) \leq 1$ is a non-increasing function on $\sum_{l=1}^{r-1} \chi_{il} p_{i[l]}$ and r, where $F_i(\sum_{l=1}^{0} \chi_{il} p_{i[l]}, 1) = 1$. Sun and Ma (2020) proved that $1\left|s_i^{AST} = a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} F_i\left(\sum_{l=1}^{r-1} \chi_{il} p_{i[l]}, r\right), GT\right| C_{\max}$ and a special case of $1\left|s_i^{AST} = a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} F_i\left(\sum_{l=1}^{r-1} \chi_{il} p_{i[l]}, r\right), GT\right| \sum\sum C_{ij}$ are polynomially solvable.

3 Makespan minimization

Lemma 1 (Niu et al. 2015) *$M(x) = \frac{f_i(V+h_i(x))g_i(r+1) - f_i(V)g_i(r)}{x}$ is a non-decreasing function on x, where $V > 0$, $\frac{df_i(x)}{dx} \leq 0$, $\frac{d^2 f_i(x)}{dx^2} \geq 0$, $\frac{d^2 h_i(x)}{dx^2} \leq 0$, $g_i(r)$ is a non-increasing function with $h_i(0) = 0$, $g_i(1) = 1$, and $x \geq 0$.*

Theorem 1 *For $1\left|s_i^{AST} = a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\}, GT\right| C_{\max}$, the optimal job sequence in each group is obtained by the nondecreasing order of p_{ij}, i.e.,*

$$p_{i\langle 1\rangle} \leq p_{i<2>} \leq \cdots \leq p_{i<n_i>}, i = 1, 2, \ldots, m.$$

Proof For the group G_i, let $\pi_i = [s_1, J_{ij}, J_{ik}, s_2]$, $\pi_i' = [s_1, J_{ik}, J_{ij}, s_2]$, where s_1 and s_2 are partial schedules, $p_{ij} \leq p_{ik}$, and there are $r-1$ jobs in S_1. Under π_i and π_i', we have

$$\begin{aligned} C_{ik}(\pi_i) = W + p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\} \\ + p_{ik} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ij})\right) g_i(r+1), \theta_i\right\}, \end{aligned} \tag{5}$$

and

$$\begin{aligned} C_{ij}(\pi_i') = W + p_{ik} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\} \\ + p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ik})\right) g_i(r+1), \theta_i\right\}, \end{aligned} \tag{6}$$

where W is the completion time of the last job in s_1.

From (5) and (6), we have

$$\begin{aligned} C_{ij}(\pi_i') - C_{ik}(\pi_i) = (p_{ik} - p_{ij}) \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\} \\ + p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ik})\right) g_i(r+1), \theta_i\right\} \\ - p_{ik} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ij})\right) g_i(r+1), \theta_i\right\}. \end{aligned} \tag{7}$$

Case 1): If $f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r) \leq \theta_i$, from (7), we have

$$C_{ij}(\pi_i') - C_{ik}(\pi_i) = (p_{ik} - p_{ij})\theta_i + p_{ij}\theta_i - p_{ik}\theta_i = 0. \tag{8}$$

Case 2): If $f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r) > \theta_i \geq f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ij})\right) g_i(r+1)$, from (7), we have

$$\begin{aligned} C_{ij}(\pi_i') - C_{ik}(\pi_i) &= (p_{ik} - p_{ij}) f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r) + p_{ij}\theta_i - p_{ik}\theta_i \\ &= (p_{ik} - p_{ij})\left[f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r) - \theta_i\right] \\ &\geq 0. \end{aligned} \tag{9}$$

Case 3): If $f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ij})\right) g_i(r+1) > \theta_i \geq f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ik})\right)$ $g_i(r+1)$, from (7), we have

$$\begin{aligned} & C_{ij}(\pi_i') - C_{ik}(\pi_i) \\ &= (p_{ik} - p_{ij}) f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r) \\ &\quad + p_{ij}\theta_i - p_{ik} f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ij})\right) g_i(r+1) \\ &\geq (p_{ik} - p_{ij}) f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r) + p_{ij} f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ik})\right) g_i(r+1) \\ &\quad - p_{ik} f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ij})\right) g_i(r+1) \\ &= p_{ij} p_{ik}\left[\frac{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ik})\right) g_i(r+1) - f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r)}{p_{ik}}\right. \\ &\quad \left. - \frac{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ij})\right) g_i(r+1) - f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r)}{p_{ij}}\right] \\ &= p_{ij} p_{ik}\left[\frac{f_i\left(V + h_i(p_{ik})\right) g_i(r+1) - f_i(V) g_i(r)}{p_{ik}}\right. \\ &\quad \left. - \frac{f_i\left(V + h_i(p_{ij})\right) g_i(r+1) - f_i(V) g_i(r)}{p_{ij}}\right], \end{aligned} \tag{10}$$

where $V = \sum_{l=1}^{r-1} h_i(p_{i[l]})$. From Lemma 1, if $p_{ij} \leq p_{ik}$, $C_{ij}(\pi_i') - C_{ik}(\pi_i) \geq 0$.

Case 4): If $f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ik})\right) g_i(r+1) > \theta_i$, from (7), we have

$$C_{ij}(\pi_i') - C_{ik}(\pi_i) = (p_{ik} - p_{ij}) f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r) \\ + p_{ij} f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ik})\right) g_i(r+1) \\ - p_{ik} f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]}) + h_i(p_{ij})\right) g_i(r+1).$$

Similar to Case 3), we have $C_j(\pi') - C_k(\pi) \geq 0$. □

Theorem 2 *For* $1\left|s_i^{AST} = a_i + b_i t_i,\ p_{ijr}^{APT} = p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\}, GT\right| C_{\max}$*, the optimal group schedule is arranged in non-decreasing order of*

$$\frac{a_i + \widetilde{P}_i}{b_i}, i = 1, 2, \ldots, m, \tag{11}$$

where

$$\widetilde{P}_i = \sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{f_i\left(\sum_{l=1}^{z-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i\right\}. \tag{12}$$

Proof From Theorem 1, the optimal job sequence in each group is obtained by the nondecreasing order of p_{ij}, i.e., $p_{i\langle 1\rangle} \leq p_{i\langle 2\rangle} \leq \cdots \leq p_{i\langle n_i\rangle}, i = 1, 2, \ldots, m$.. Let ϱ and ϱ' be two group schedules where $\varrho = [S_1, G_i, G_j, S_2]$, $\varrho' = [S_1, G_j, G_i, S_2]$ and S_1 and S_2 are partial sequences. Let t be the completion time of the last job in S_1, for ϱ, we have

$$C_{i\langle 1\rangle]}(\varrho) = t + a_i + b_i t + p_{i\langle 1\rangle} = a_i + t(1 + b_i) + p_{i\langle 1\rangle},$$
$$C_{i\langle 2\rangle}(\varrho) = C_{i\langle 1\rangle} + p_{i\langle 2\rangle} \max\left\{f_i\left(h_i(p_{i\langle 1\rangle})\right) g_i(2), \theta_i\right\} \\ = a_i + t(1 + b_i) + p_{i\langle 1\rangle} + p_{i\langle 2\rangle} \max\left\{f_i\left(h_i(p_{i\langle 1\rangle})\right) g_i(2), \theta_i\right\},$$
$$\ldots\ldots$$
$$C_{i\langle n_i\rangle}(\varrho) = a_i + t(1 + b_i) + \sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{f_i\left(\sum_{l=1}^{z-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i\right\}.$$

$$
\begin{aligned}
C_{j\langle 1\rangle}(\varrho) &= C_{i\langle n_i\rangle}(\varrho) + a_j + b_j C_{i\langle n_i\rangle}(\varrho) + p_{j\langle 1\rangle}\\
&= a_j + C_{i\langle n_i\rangle}(\varrho)(1+b_j) + p_{j\langle 1\rangle}\\
&= a_j + a_i(1+b_j) + t(1+b_i)(1+b_j)\\
&\quad + (1+b_j)\sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{ f_i\left(\sum_{l=1}^{z-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i \right\} + p_{j\langle 1\rangle},\\
C_{j\langle 2\rangle}(\varrho) &= a_j + a_i(1+b_j) + t(1+b_i)(1+b_j)\\
&\quad + (1+b_j)\sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{ f_i\left(\sum_{l=1}^{z-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i \right\}\\
&\quad + p_{j\langle 1\rangle} + p_{j\langle 2\rangle} \max\left\{ f_i\left(h_j(p_{j\langle 1\rangle})\right) g_j(2), \theta_j \right\},\\
&\cdots\cdots\\
C_{j\langle n_j\rangle}(\varrho) &= a_j + a_i(1+b_j) + t(1+b_i)(1+b_j)\\
&\quad + (1+b_j)\sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{ f_i\left(\sum_{l=1}^{h-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i \right\}\\
&\quad + \sum_{z=1}^{n_j} p_{j\langle z\rangle} \max\left\{ f_j\left(\sum_{l=1}^{z-1} h_j(p_{j\langle l\rangle})\right) g_j(z), \theta_j \right\}.
\end{aligned}
\tag{13}
$$

Similarly, for ϱ', we have

$$
\begin{aligned}
C_{i\langle n_i\rangle}(\varrho') = a_i + a_j(1+b_i) + t(1+b_j)(1+b_i)&\\
+(1+b_i)\sum_{z=1}^{n_j} p_{j\langle z\rangle} \max\left\{ f_j\left(\sum_{l=1}^{z-1} h_j(p_{j\langle l\rangle})\right) g_j(z), \theta_j \right\}&\\
+\sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{ f_i\left(\sum_{l=1}^{h-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i \right\}.&
\end{aligned}
\tag{14}
$$

From (13) and (14), we have

$$
\begin{aligned}
&C_{j\langle n_j\rangle}(\varrho) - C_{i\langle n_i\rangle}(\varrho')\\
&= a_i b_j + b_j \sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{ f_i\left(\sum_{l=1}^{h-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i \right\}\\
&\quad -a_j b_i - b_i \sum_{z=1}^{n_j} p_{j\langle z\rangle} \max\left\{ f_j\left(\sum_{l=1}^{h-1} h_j(p_{j\langle l\rangle})\right) g_j(z), \theta_j \right\}\\
&= b_i b_j \left(\frac{a_i + \widetilde{P}_i}{b_i} - \frac{a_j + \widetilde{P}_j}{b_j} \right)\\
&\le 0
\end{aligned}
$$

if and only if

$$
\frac{a_i + \widetilde{P}_i}{b_i} \le \frac{a_j + \widetilde{P}_j}{b_j},
$$

where $\widetilde{P}_i = \sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{f_i\left(\sum_{l=1}^{z-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i\right\}$ and $\widetilde{P}_j = \sum_{z=1}^{n_j} p_{j\langle z\rangle} \max\left\{f_j\left(\sum_{l=1}^{z-1} h_j(p_{j\langle l\rangle})\right) g_j(z), \theta_j\right\}$, this completes the proof. □

From Theorems 1-2, $1\left|s_i^{AST} = !a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\}, GT\right|C_{\max}$ can be solved by the following algorithm:

Algorithm 1.

Step 1. Jobs in each group are scheduled in non-decreasing order of p_{ij}, i.e., $p_{i\langle 1\rangle} \le p_{i\langle 2\rangle} \le \ldots \le p_{i\langle n_i\rangle}$, $i = 1, 2, \ldots, m$.
Step 2. Calculate $\rho(G_i) = \frac{a_i + \widetilde{P}_i}{b_i}$, $i = 1, 2, \ldots, m$, where $\widetilde{P}_i = \sum_{z=1}^{n_i} p_{i[z]} \max\left\{f_i\left(\sum_{l=1}^{z-1} h_i(p_{i[l]})\right) g_i(z), \theta_i\right\}$.
Step 3. Groups are scheduled in non-decreasing order of $\rho(G_i)$, i.e., $\rho(G_1) \le \rho(G_2) \le \ldots \le \rho(G_m)$.

Theorem 3 *The* $1\left|s_i^{AST} = a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\}, GT\right|C_{\max}$ *can be solved by Algorithm 1 in* $O(n\log n)$ *time.*

Proof Obviously, the optimal schedule in a certain group G_i can be obtained in $O(n_i \log n_i)$ and the optimal group schedule can be obtained in $O(m\log m)$. Obviously $\sum_{i=1}^{m} O(n_i \log n_i) \le O(n\log n)$, hence, the total time for Algorithm 1 is $O(n\log n)$. □

Example 1 There are $n = 10$ jobs, where $m = 3$, $n_1 = n_2 = 3$, $n_3 = 4$, $t_0 = 0$, $f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) = \left(1 + \sum_{l=1}^{r-1} \ln p_{i[l]}\right)^{\alpha_i}$ $(\alpha_i \le 0)$, $g_i(r) = {\beta_i}^{r-1}$ $(0 < \beta_i \le 1)$, and the other coefficients of all jobs are given in Table 1.

Solution:

Step 1. By Theorem 1, the optimal internal job sequences are:
$\pi_1^*: [J_{13} \to J_{11} \to J_{12}]$,
$\pi_2^*: [J_{23} \to J_{22} \to J_{21}]$,
$\pi_3^*: [J_{34} \to J_{32} \to J_{31} \to J_{33}]$.

Table 1 The values of Example 1

	G_1	G_2	G_3
a_i	4	6	5
b_i	6	10	2
α_i	−0.3	−0.32	−0.25
β_i	0.8	0.9	0.75
θ_i	0.5	0.6	0.5

	J_{11}	J_{12}	J_{13}	J_{21}	J_{22}	J_{23}	J_{31}	J_{32}	J_{33}	J_{34}
p_{ij}	15	20	13	19	15	4	18	16	21	8

Step 2. By Eqs. (11) and (12), we have
$\rho(G_1) = \frac{a_1+\widetilde{P_1}}{b_1} = 5.865885,$
$\rho(G_2) = \frac{a_2+\widetilde{P_2}}{b_2} = 3.162026,$
$\rho(G_3) = \frac{a_3+\widetilde{P_3}}{b_3} = 20.77932.$

Step 3. By Theorem 2, it follows that the optimal group schedule is $\varrho^* : [G_2 \to G_1 \to G_3]$.

4 Total completion time minimization

Theorem 4 *For* $1\left|s_i^{AST} = a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\},\right.$ $GT\left|\sum\sum C_{ij}\right.$, *the optimal job sequence in each group is obtained by the nondecreasing order of* p_{ij}, *i.e.,*

$$p_{i\langle 1\rangle} \le p_{i\langle 2\rangle} \le \cdots \le p_{i\langle n_i\rangle}, i = 1, 2, \ldots, m.$$

Proof Similar to Theorem 1. For the group G_i, from schedules $\pi_i = [s_1, J_{ij}, J_{ik}, s_2]$, $\pi_i' = [s_1, J_{ik}, J_{ij}, s_2]$ and $p_{ij} \le p_{ik}$, we have

$$\begin{aligned} C_{ij}(\pi_i) &= W + p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\} \\ &\le C_{ik}(\pi_i') = W + p_{ik} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\} \end{aligned}$$

and $C_{ik}(\pi_i) \le C_{ij}(\pi_i')$, hence $C_{ik}(\pi_i) + C_{ij}(\pi_i) \le C_{ik}(\pi_i') + C_{ij}(\pi_i')$, this completes the proof. □

From Mosheiov (1991), the problem $1\left|p_j^{APT} = 1 + \beta_j t_j\right|\sum C_j$ is an open problem. In order to determine the optimal group sequence, we have the following result:

Theorem 5 *For*

$$1\left|s_i^{AST} = a_i + b_i t_i, p_{ijr}^{APT} = p_{ij} \max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right) g_i(r), \theta_i\right\}, GT\right|\sum\sum C_{ij},$$

if $\frac{b_i}{n_i(1+b_i)} \le \frac{b_j}{n_j(1+b_j)} \Rightarrow \frac{a_i+\widetilde{P_i}}{n_i(1+b_i)} \le \frac{a_j+\widetilde{P_j}}{n_j(1+b_j)}$, *the optimal group schedule is arranged in non-decreasing order of* $\frac{b_i}{n_i(1+b_i)}$ *or* $\frac{a_i+\widetilde{P_i}}{n_i(1+b_i)}$, $i = 1, 2, \ldots, m$, *where*

$$\widetilde{P_i} = \sum_{z=1}^{n_i} p_{i\langle z\rangle} \max\left\{f_i\left(\sum_{l=1}^{z-1} h_i(p_{i\langle l\rangle})\right) g_i(z), \theta_i\right\}$$

and

$$\widetilde{P_j} = \sum_{z=1}^{n_j} p_{j\langle z\rangle} \max\left\{f_j\left(\sum_{l=1}^{z-1} h_j(p_{j\langle l\rangle})\right) g_j(z), \theta_j\right\}.$$

Proof From Theorem 2, for ϱ and ϱ', we have

$$\sum\sum C_{ij}(\varrho) - \sum\sum C_{ij}(\varrho') = tn_in_j(1+b_i)(1+b_j)\left(\frac{b_i}{n_i(1+b_i)} - \frac{b_j}{n_j(1+b_j)}\right) + n_in_j(1+b_i)(1+b_j)\left(\frac{a_i+\widetilde{P}_i}{n_i(1+b_i)} - \frac{a_j+\widetilde{P}_j}{n_j(1+b_j)}\right).$$

If $\frac{b_i}{n_i(1+b_i)} \le \frac{b_j}{n_j(1+b_j)}$ and $\frac{a_i+\widetilde{P}_i}{n_i(1+b_i)} \le \frac{a_j+\widetilde{P}_j}{n_j(1+b_j)}$, we have $\sum\sum C_{ij}(\varrho) \le \sum\sum C_{ij}(\varrho')$, this completes the proof. □

4.1 Branch-and-bound algorithm

From Theorems 4-5, to solve

$$1\left|s_i^{AST} = a_i + b_it_i,\, p_{ijr}^{APT} = p_{ij}\max\left\{f_i\left(\sum_{l=1}^{r-1} h_i(p_{i[l]})\right)g_i(r), \theta_i\right\}, GT\right|\sum\sum C_{ij},$$

an heuristic algorithm is proposed (i.e., an upper bound of branch-and-bound algorithm).

Algorithm 2.

Step 1. Jobs in each group are scheduled in non-decreasing order of p_{ij}, i.e., $p_{i\langle 1\rangle} \le p_{i\langle 2\rangle} \le \ldots \le p_{i\langle n_i\rangle}, i = 1, 2, \ldots, m$.
Step 2. Groups are scheduled in non-decreasing order of $\rho(G_i) = a_i, i = 1, 2, \ldots, m$.
Step 3. Groups are scheduled in non-decreasing order of $\rho(G_i) = b_i, i = 1, 2, \ldots, m$.
Step 4. Groups are scheduled in non-increasing order of $\rho(G_i) = n_i, i = 1, 2, \ldots, m$.
Step 5. Groups are scheduled in non-decreasing order of $\rho(G_i) = \frac{a_i}{n_i(1+b_i)}, i = 1, 2, \ldots, m$.
Step 6. Groups are scheduled in non-decreasing order of $\rho(G_i) = \frac{b_i}{n_i(1+b_i)}, i = 1, 2, \ldots, m$.
Step 7. Groups are scheduled in non-decreasing order of $\rho(G_i) = \frac{a_i + \sum_{h=1}^{n_i} p_{i[h]}\max\left\{f_i\left(\sum_{l=1}^{h-1} h_i(p_{i[l]})\right)g_i(h), \theta_i\right\}}{n_i(1+b_i)}, i = 1, 2, \ldots, m$.
Step 8. Choose the one with smaller $\sum\sum C_{ij}$ as the solution by Step 2-7.

Lemma 2 (Gawiejnowicz 2020a) *The term* $\sum_{k=1}^{m} x_k \prod_{l=k+1}^{m} y_l$ *can be minimized by the non-decreasing order of* $\frac{x_i}{y_i-1}$.

Lemma 3 *The term* $\sum_{k=1}^{m} x_k \prod_{l=k+1}^{m+1} y_l$ *can be minimized by the non-decreasing order of* $\frac{x_i}{y_i-1}$.

Proof For $\delta = [s_1, i, j, s_2]$ and $\delta' = [s_1, j, i, s_2]$, there is

$$\begin{aligned}\sum_{k=1}^{m} x_k \prod_{l=k+1}^{m+1} y_l(\delta) - \sum_{k=1}^{m} x_k \prod_{l=k+1}^{m+1} y_l(\delta') &= x_i y_j (y_{i+2}...y_{m+1}) + x_j (y_{i+2}...y_{m+1}) \\ &\quad -x_j y_i (y_{i+2}...y_{m+1}) - x_i (y_{i+2}...y_{m+1}) \\ &= (y_{i+2}...y_{m+1})[x_i(y_j - 1) - x_j(y_i - 1)] \\ &= (y_{i+2}...y_{m+1})(y_i - 1)(y_j - 1) \\ &\quad \times \left(\frac{x_i}{y_i - 1} - \frac{x_j}{y_j - 1}\right) \\ &\le 0,\end{aligned}$$

if and only if $\frac{x_i}{y_i-1} \le \frac{x_j}{y_j-1}$. □

Lemma 4 *The term $\sum_{k=1}^{m} x_k \prod_{l=1}^{k} y_l$ can be minimized by the non-decreasing order of $\frac{y_i-1}{x_i y_i}$.*

Proof For $\delta = [s_1, i, j, s_2]$ and $\delta' = [s_1, j, i, s_2]$, there is

$$\begin{aligned}\sum_{k=1}^{m} x_k \prod_{l=1}^{k} y_l(\delta) - \sum_{k=1}^{m} x_k \prod_{l=1}^{k} y_l(\delta') &= x_i y_1 y_2 ... y_i + x_j y_1 y_2 ... y_i y_j \\ &\quad -(x_j y_1 y_2 ... y_j + x_i y_1 y_2 ... y_j y_i) \\ &= y_1 y_2 ... y_{i-1}[x_j y_j (y_i - 1) - x_i y_i (y_j - 1)] \\ &= y_1 y_2 ... y_{i-1} x_i y_i x_j y_j \left(\frac{y_i - 1}{x_i y_i} - \frac{y_j - 1}{x_j y_j}\right) \\ &\le 0\end{aligned}$$

if and only if $\frac{y_i-1}{x_i y_i} \le \frac{y_j-1}{x_j y_j}$. □

From Theorem 4, the optimal job sequence within each group can be obtained, i.e., $p_{i\langle 1\rangle} \le p_{i\langle 2\rangle} \le \cdots \le p_{i\langle n_i\rangle}$, $i = 1, 2, \ldots, m$.. Let $\varrho = (\chi^{pp}, \chi^{uu})$ be a group schedule, where χ^{pp} (resp. χ^{uu}) is the scheduled (resp. unscheduled) part, and there are ψ groups in χ^{pp}, from the proof of Theorem 2, we have

$$\begin{aligned}&\sum_{i=1}^{\psi} \sum_{j=1}^{n_i} C_{i[j]}(\chi^{pp}) + \sum_{i=\psi+1}^{m} \sum_{j=1}^{n_i} C_{[i][j]}(\chi^{uu}) \\ &\quad = \sum_{i=1}^{\psi} \sum_{j=1}^{n_i} C_{i[j]}(\chi^{pp}) + \sum_{i=\psi+1}^{m} n_{[i]} \left(\sum_{k=\psi+1}^{i} a_{[k]} \prod_{l=k+1}^{i} (1 + b_{[l]})\right) \\ &\quad\quad + t \left(\sum_{k=\psi+1}^{m} n_{[k]} \prod_{l=\psi+1}^{k} (1 + b_{[l]})\right) \\ &\quad\quad + \sum_{i=\psi+1}^{m-1} n_{[i+1]} \left(\sum_{k=\psi+1}^{i} \widetilde{P_{[k]}} \prod_{l=k+1}^{i+1} (1 + b_{[l]})\right) \\ &\quad\quad + \sum_{i=\psi+1}^{m} \sum_{h=1}^{n_{[i]}} \left(n_{[i]} - h + 1\right) p_{[i]\langle h\rangle} \max \left\{ f_{[i]} \left(\sum_{l=1}^{h-1} h_{[i]}(p_{[i]\langle l\rangle})\right) g_{[i]}(h), \theta_{[i]} \right\},\end{aligned} \tag{15}$$

where $\prod_{l=k}^{k-1}(1+b_{[l]}) = 1$ and t is the completion time of the last job in χ^{pp}, i.e., $t = C_{\psi[n_\psi]}(\chi^{pp})$. From (15), $\sum_{i=1}^{\psi}\sum_{j=1}^{n_i} C_{i[j]}(\chi^{pp})$ and $t = C_{\psi[n_\psi]}(\chi^{pp})$ are constants, $\sum_{k=\psi+1}^{i} a_{[k]}\prod_{l=k+1}^{i}(1+b_{[l]})$ (resp. $\sum_{k=\psi+1}^{i}\widetilde{P_{[k]}}\prod_{l=k+1}^{i+1}(1+b_{[l]})$) can be minimized according to Lemma 2 (resp. Lemma 3) by non-decreasing of $\frac{a_i}{b_i}$ (resp. $\frac{\widetilde{P_i}}{b_i}$), and $\sum_{k=\psi+1}^{m} n_{[k]}\prod_{l=\psi+1}^{k}(1+b_{[l]})$ can be minimized by the non-decreasing order of $\frac{b_i}{n_i(1+b_i)}$ according to Lemma 4 ($i \in \chi^{uu}$).

Let $n_{\min} = \min\{n_{[\psi+1]}, n_{[\psi+2]}, ..., n_{[m]}\}$, hence, we have the following lower bound:

$$
\begin{aligned}
LB = & \sum_{i=1}^{\psi}\sum_{j=1}^{n_i} C_{ij}(\chi^{pp}) + \sum_{i=\psi+1}^{m} n_{\min}\left(\sum_{k=\psi+1}^{i} a_{\langle k\rangle}\prod_{l=k+1}^{i}(1+b_{\langle l\rangle})\right) \\
& + t\left(\sum_{k=\psi+1}^{m} n_{\langle\langle k\rangle\rangle}\prod_{l=\psi+1}^{k}(1+b_{\langle\langle l\rangle\rangle})\right) \\
& + \sum_{i=\psi+1}^{m-1} n_{\min}\left(\sum_{k=\psi+1}^{i}\widetilde{P_{(k)}}\prod_{l=k+1}^{i+1}(1+b_{(l)})\right) \\
& + \sum_{i=\psi+1}^{m}\sum_{h=1}^{n_{[i]}}\left(n_{[i]}-h+1\right)p_{[i]\langle h\rangle}\max\left\{f_{[i]}\left(\sum_{l=1}^{h-1} h_{[i]}(p_{[i]\langle l\rangle})\right)g_{[i]}(h), \theta_{[i]}\right\},
\end{aligned}
\tag{16}
$$

where $\frac{a_{\langle\psi+1\rangle}}{b_{\langle\psi+1\rangle}} \le \frac{a_{\langle\psi+2\rangle}}{b_{\langle\psi+2\rangle}} \le \cdots \le \frac{a_{\langle m\rangle}}{b_{\langle m\rangle}}$, $\frac{b_{\langle\langle\psi+1\rangle\rangle}}{n_{\langle\langle\psi+1\rangle\rangle}}(1+b_{\langle\langle\psi+1\rangle\rangle}) \le \frac{b_{\langle\langle\psi+2\rangle\rangle}}{n_{\langle\langle\psi+2\rangle\rangle}}(1+b_{\langle\langle\psi+2\rangle\rangle}) \le \cdots \le \frac{b_{\langle\langle m\rangle\rangle}}{n_{\langle\langle m\rangle\rangle}}(1+b_{\langle\langle m\rangle\rangle})$, and $\frac{\widetilde{P_{(\psi+1)}}}{b_{(\psi+1)}} \le \frac{\widetilde{P_{(\psi+2)}}}{b_{(\psi+2)}} \le \cdots \le \frac{\widetilde{P_{(m)}}}{b_{(m)}}$ (where $\frac{a_{\langle i\rangle}}{b_{\langle i\rangle}}$, $\frac{b_{\langle\langle i\rangle\rangle}}{n_{\langle\langle i\rangle\rangle}(1+b_{\langle\langle i\rangle\rangle})}$, and $\frac{\widetilde{P_{(i)}}}{b_{(i)}}$ do not necessarily correspond to the same job, $i = \psi+1, \psi+2, ..., m$). For $G_{[i]}$, $p_{[i]\langle 1\rangle} \le p_{[i]\langle 2\rangle} \le \cdots \le p_{[i]<n_{[i]}>}$, and the sequence of $p_{(i)\langle j\rangle}$ involved in $P_{(i)}$ is the same as that of $G_{[i]}$, that is, $p_{(i)\langle 1\rangle} \le p_{(i)\langle 2\rangle} \le \cdots \le p_{(i)\langle n_{(i)}\rangle}$, where $j = 1, 2, ..., n_{(i)}$.

From the upper bound (i.e., Algorithm 2) and lower bound (see Eq. (16)), a branch-and-bound algorithm (B&B) is established as follows:

Algorithm 3. (B&B)

Step 1). Use Algorithm 2 to obtain an initial group sequence.
Step 2). Calculate the lower bound (see equation (16)) for the node (group). If the lower bound for an unfathomed partial group schedule is larger than or equal to the objective value of the initial solution (see equation (15)), eliminate the node and all the nodes following it in the branch. Calculate the objective value of the completed group schedule (see equation (15)). If it is less than the initial solution, replace it as the new solution; otherwise, eliminate it.
Step 3). Continue until all nodes have been explored.

4.2 Other algorithms

As in Nawaz et al. (1983) and Paredes-Astudillo et al. (2024), the following heuristic algorithm (HA) can be proposed.

Algorithm 4: HA

Step 1). Let ϱ^A be the group sequence obtained from Algorithm 2.
Step 2). Set $\lambda = 2$. Select the first two groups from the sorted list and select the better of the two possible sequences. Do not change the relative positions of these two jobs with respect to each other in the remaining steps of the algorithm. Set $\lambda = 3$.
Step 3). Pick the job in the λth position of the list generated in Step 1) and find the best group sequence by placing it at all possible λ positions in the partial sequence found in the previous step, without changing the relative positions to each other of the already assigned groups. The number of enumerations at this step equals λ.
Step 4). If $\lambda = m$, STOP; otherwise, set $\lambda = \lambda + 1$ and go to Step 3).

As in He et al. (2023), a tabu search (TS) algorithm is a method for $\sum\sum C_{ij}$ minimization. The initial group sequence of the TS algorithm is decided by Algorithm 2, and the maximum number of iterations for the TS algorithm is 1000 m.

Algorithm 5: TS

Step 1). Let the tabu list be empty and the iteration number be zero.
Step 2). Set the initial group sequence of the TS algorithm, calculate its objective cost (by equation (15)), and set the current group sequence as the best solution ϱ^{A*}.
Step 3). Search the associated neighborhood of the current group sequence and resolve if there is a group sequence ϱ^{A**} with the smallest objective cost in the associated neighborhood and it is not in the tabu list.
Step 4). If ϱ^{A**} is better than ϱ^{A*}, then let $\varrho^{A*} = \varrho^{A**}$. Update the tabu list and the iteration number.
Step 5). If there is not a group sequence in the associated neighborhood but it is not in the tabu list or the maximum number of iterations is reached (i.e., 1000 m), then output the final group sequence. Otherwise, update the tabu list and go to Step 3).

As in Li et al. (2024a, b), simulated annealing (SA) is also a method for $\sum\sum C_{ij}$ minimization.

Algorithm 6: SA

Step 1). Set the internal job sequence by the Second step of Algorithm 2.
Step 2). Use the pairwise interchange (PI) neighborhood generation method.
Step 3). Calculate the objective value of the original schedule ϱ^A.
Step 4). Calculate the objective value of the new schedule ϱ^{A*}. If the ϱ^{A*} is less than ϱ^A, it is accepted. Nevertheless, if the ϱ^{A*} is higher, it might still be accepted with a decreasing probability as the process proceeds. This acceptance probability is determined by the following exponential distribution function: $P(accept) = exp(-\alpha \times \triangle TC)$, where α is a parameter and $\triangle TC$ is the change in the objective function. In addition, the method is used to change α in the kth iteration as follows: $\alpha = \frac{k}{\delta}$, where δ is a constant. After preliminary trials, $\delta = 1$ is used.
Step 5). If ϱ^{A*} increases, the new sequence is accepted when $P(accept) > \beta$, where β is randomly sampled from the uniform distribution.
Step 6). The schedule is stable after $1000m$ iterations.

4.3 Number study

The heuristic algorithms (i.e., HA, TS, and SA) and the B&B algorithm were programmed in C++ (carried out on CPU Interl core i5-8250U 1.4GHz PC with 8.00GB RAM), where $n =$ 100, 200, 300, 400; $m=$ 8, 9, 10, 11; $p_{ijr}^{APT} = p_{ij} \max\left\{\left(1 + \sum_{l=1}^{r-1} \ln p_{i[l]}\right)^{\alpha_i} {\beta_i}^{r-1}, \theta_i\right\}$ ($\alpha_i \leq 0, 0 < \beta_i \leq 1$) and $n_i \geq 1$. The parameters setting is given as follows:

1) Real number: $\alpha_i \in [-0.5, -0.1]$; $\beta_i \in [0.4, 0.9]$; $\theta_i \in [0.3, 0.6]$;

2) Integer number: $a_i \in [3, 50]$ and $p_{ij} \in [3, 50]$; $a_i \in [51, 100]$ and $p_{ij} \in [51, 100]$; $a_i \in [3, 100]$ and $p_{ij} \in [3, 100]$;

3) $b_i \in [0.01, 0.05]$; $b_i \in [0.05, 0.15]$; $b_i \in [0.15, 0.3]$.

For simulation accuracy, each random instance was conducted 20 times. The error of algorithm H is calculated as

$$\frac{\sum\sum C_{ij}(H)}{\sum\sum C_{ij}^*}, \tag{17}$$

where $H \in \{HA, TS, SA\}$, $\sum\sum C_{ij}(H)$ (resp. $\sum\sum C_{ij}^*$) is the objective value (see (15)) generated by algorithm H (resp. B&B). In addition, running time (i.e., millisecond (ms)) of HA, TS, SA and B&B is defined. From Tables 2, 3 and 4, the maximum CPU time of B&B is 2,815,317 ms (i.e., $n \times m = 400 \times 11$). For the CPU time of B&B, $b_i \in [0.01, 0.05]$ needs more time than $b_i \in [0.05, 0.15]$, and $b_i \in [0.05, 0.15]$ needs more time than $b_i \in [0.15, 0.3]$. From Tables 5, 6 and 7, the maximum error of SA is less than HA and TS for $n \times m \leq 400 \times 11$ and the results of $b_i \in [0.01, 0.05]$ is more accurate than $b_i \in [0.05, 0.15]$ and $b_i \in [0.15, 0.3]$.

To further compare the algorithms TS, SA and B&B, the parameters $n = 300$, $m = 10$, $b_i \in [0.01, 0.05]$, $b_i \in [0.05, 0.15]$, and $b_i \in [0.15, 0.3]$ were given. The algorithms TS, SA and B&B were individually executed on the same instance. For each combination, 20 random instances were performed. It is also assumed that the CPU time for TS, SA are all obtained from B&B, that is, all three algorithms have the same CPU time. In order to calculate the error $\frac{\sum\sum C_{ij}(H)}{\sum\sum C_{ij}^*}$ ($H \in \{TS, SA\}$), the total completion time values of TS and SA are obtained and compared with the values obtained by B&B. From the data in Tables 8, 9 and 10, it can be seen that SA is more accurate that TS, especially for $n = 300$, $m = 10$, $b_i \in [0.01, 0.05]$ (i.e., there were 16 instances $\frac{\sum\sum C_{ij}(SA)}{\sum\sum C_{ij}^*} = 1$).

Table 2 CPU time for $a_i \in [3, 50]$ and $p_{ij} \in [3, 50]$ (ms)

$n \times m$	b_i	CPU of HA		CPU of TS		CPU of SA		CPU of B&B	
		Avg	Max	Avg	Max	Avg	Max	Avg	Max
100×8	$b_i \in [0.01, 0.05]$	13.9	29	2621.9	3497	82	107	490.2	1092
	$b_i \in [0.05, 0.15]$	12.6	16	2870.3	5185	81.3	103	329.5	521
	$b_i \in [0.15, 0.3]$	12.9	17	2652.4	2982	81.1	97	223	338
100×9	$b_i \in [0.01, 0.05]$	14.7	17	3284.5	3565	73.1	84	1828.7	3217
	$b_i \in [0.05, 0.15]$	14.1	17	3350.9	4339	77.7	105	1109.4	2758
	$b_i \in [0.15, 0.3]$	13.6	18	3405.7	4041	79.1	95	534	943
100×10	$b_i \in [0.01, 0.05]$	14.4	18	4202.5	5189	73.4	94	12,527.7	20,101
	$b_i \in [0.05, 0.15]$	14.1	17	4758.7	7029	82.8	128	5961.8	10,750
	$b_i \in [0.15, 0.3]$	14.2	18	4421.1	5096	77.6	88	1912	5384
100×11	$b_i \in [0.01, 0.05]$	16.6	20	5560.7	6124	77	90	31,633.2	85,237
	$b_i \in [0.05, 0.15]$	15.8	22	6237.1	7045	87.3	103	35,789.8	63,334
	$b_i \in [0.15, 0.3]$	15.6	21	5591.7	6025	78.5	104	7067.9	11,044
200×8	$b_i \in [0.01, 0.05]$	24.3	27	8853.7	10,261	303.5	361	2232.2	4012
	$b_i \in [0.05, 0.15]$	23.8	30	8587.7	13,699	290.5	467	1556	3089
	$b_i \in [0.15, 0.3]$	22.2	24	7631.3	10,240	258.1	355	920.9	1503
200×9	$b_i \in [0.01, 0.05]$	26.8	37	10,147	15,503	270	413	10,568.3	15,446
	$b_i \in [0.05, 0.15]$	23.7	27	9427.9	10,836	240.7	282	4364.9	10,778
	$b_i \in [0.15, 0.3]$	25.3	29	9704.8	11,851	252.4	311	3259.2	5355
200×10	$b_i \in [0.01, 0.05]$	27.5	41	12,993.5	258,24	266.8	564	50,961	107,493
	$b_i \in [0.05, 0.15]$	29.1	40	13,709.4	21,859	277.3	469	27,469.3	45,352
	$b_i \in [0.15, 0.3]$	27	32	12,554.6	1585	257.2	332	11,937.4	27,018
200×11	$b_i \in [0.01, 0.05]$	31.1	41	16,268.3	20,445	262.9	361	347,334.6	537,588
	$b_i \in [0.05, 0.15]$	28.7	39	14,972.3	17,425	244.7	307	153,143.7	387,782
	$b_i \in [0.15, 0.3]$	30.5	43	16,988.8	24,221	272.1	354	46171.7	92,975

Table 2 continued

$n \times m$	b_i	CPU of HA		CPU of TS		CPU of SA		CPU of B&B	
		Avg	Max	Avg	Max	Avg	Max	Avg	Max
300×8	$b_i \in [0.01, 0.05]$	38.2	47	17,205.5	24,269	599.4	856	4303.6	10,104
	$b_i \in [0.05, 0.15]$	36.7	46	15,157.6	16,831	527.8	606	3167.7	5657
	$b_i \in [0.15, 0.3]$	40.7	45	17,299.6	21,577	603.4	771	2155.2	3283
300×9	$b_i \in [0.01, 0.05]$	40.7	51	20,212.6	27,748	545.7	750	23,274.9	47,334
	$b_i \in [0.05, 0.15]$	44.5	53	22,082	26,012	599.6	716	15,568.6	24,271
	$b_i \in [0.15, 0.3]$	41.5	57	21270.4	33,379	574.7	904	6646.9	9101
300×10	$b_i \in [0.01, 0.05]$	46.8	57	25,374	32,603	541.1	701	138,374.6	268,386
	$b_i \in [0.05, 0.15]$	44.6	51	25,174.6	27,114	542.3	592	55,987.7	82,886
	$b_i \in [0.15, 0.3]$	288.8	2439	27,664.8	32,788	599.3	711	28,103.8	66,533
300×11	$b_i \in [0.01, 0.05]$	50.4	82	32,444.1	65,240	567	1176	844,306.7	1,321,981
	$b_i \in [0.05, 0.15]$	56.7	67	37,329.3	47,416	651.4	844	364,236.2	612,584
	$b_i \in [0.15, 0.3]$	53.6	58	33,733.4	41,029	588.9	726	106,476	205,758
400×8	$b_i \in [0.01, 0.05]$	53.6	66	26,992.1	36,071	947.2	1260	7625.1	11,050
	$b_i \in [0.05, 0.15]$	62.2	75	32,103.8	41,033	1135	1455	6834.3	9643
	$b_i \in [0.15, 0.3]$	62.4	76	32,823.4	47,973	1156.3	1701	4324.1	7756
400×9	$b_i \in [0.01, 0.05]$	58.9	68	33,493.8	40,571	912.3	1103	42,995.9	64,666
	$b_i \in [0.05, 0.15]$	62.4	78	35,923	46,638	981.8	1323	26,464.2	40,410
	$b_i \in [0.15, 0.3]$	63.4	76	36,044.4	48,144	984.5	1323	14,354.7	24,071
400×10	$b_i \in [0.01, 0.05]$	68.88	87	44,441.3	63,646	965.778	1383	229,335	496,400
	$b_i \in [0.05, 0.15]$	68.6	82	43,043.1	59,238	931.2	1291	121,197.2	262,078
	$b_i \in [0.15, 0.3]$	68.6	85	42,674.9	49,253	929	1058	52,286.8	81,724
400×11	$b_i \in [0.01, 0.05]$	79	93	55,351.8	74,898	971	1334	1,703,654.5	2,815,317
	$b_i \in [0.05, 0.15]$	72.8	79	50,063.6	58,231	880.6	1024	465,220.6	838,654
	$b_i \in [0.15, 0.3]$	81.2	97	57,413.6	63,091	1018.3	1114	210,970	315,621

Table 3 CPU time for $a_i \in [51, 100]$ and $p_{ij} \in [51, 100]$ (ms)

$n \times m$	b_i	CPU of HA		CPU of TS		CPU of SA		CPU of B&B	
		Avg	Max	Avg	Max	Avg	Max	Avg	Max
100×8	$b_i \in [0.01, 0.05]$	13	18	2582.4	2856	81.7	118	487.8	729
	$b_i \in [0.05, 0.15]$	12.2	16	2322.3	2649	68.3	79	365.2	477
	$b_i \in [0.15, 0.3]$	13.6	17	2602.6	2938	78.9	94	203.7	252
100×9	$b_i \in [0.01, 0.05]$	13.1	19	3218.5	3596	72.3	88	1519	2335
	$b_i \in [0.05, 0.15]$	13.3	16	3168.1	3811	71.2	85	1250.1	1973
	$b_i \in [0.15, 0.3]$	13.8	19	3380.2	3770	77.8	89	670.1	845
100×10	$b_i \in [0.01, 0.05]$	15.5	21	5681	139,20	83.5	109	15,608.9	43,951
	$b_i \in [0.05, 0.15]$	15.2	22	4355.3	5155	78.1	93	4980.3	10,404
	$b_i \in [0.15, 0.3]$	16	19	5026.4	6229	89.5	108	3024.6	4654
100×11	$b_i \in [0.01, 0.05]$	16.5	25	5641.4	6279	79.4	94	44,337	74,124
	$b_i \in [0.05, 0.15]$	16	22	5675.7	6353	77.3	90	25,438.6	48,795
	$b_i \in [0.15, 0.3]$	15.6	21	5611.4	6083	76	85	8795.9	12,136
200×8	$b_i \in [0.01, 0.05]$	22.5	25	7359.5	9459	248.5	325	826.5	1113
	$b_i \in [0.05, 0.15]$	37.9	42	17,007.5	21,974	593.5	772	3335.9	5383
	$b_i \in [0.15, 0.3]$	22.1	26	7760	10,339	262.2	335	890.5	1446
200×9	$b_i \in [0.01, 0.05]$	25.3	31	10,080.8	12,142	264.9	316	10,103	19,447
	$b_i \in [0.05, 0.15]$	25.7	31	9929.1	14,990	260	400	5524.8	10,629
	$b_i \in [0.15, 0.3]$	24.6	39	9026.2	11,144	234	290	2460.1	4136
200×10	$b_i \in [0.01, 0.05]$	29.4	36	14,080.5	19,470	287.8	415	70,481.3	115,096
	$b_i \in [0.05, 0.15]$	27.3	31	12,244.7	15,710	240.1	283	26,462.4	54,836
	$b_i \in [0.15, 0.3]$	29.1	39	13,117.2	17,012	268.9	354	12,648.6	21,222
200×11	$b_i \in [0.01, 0.05]$	31	40	16,132.4	19,786	265.5	331	319,445	485087
	$b_i \in [0.05, 0.15]$	31.7	45	17,217.6	19,901	278	331	141,557.3	276,368
	$b_i \in [0.15, 0.3]$	31	37	16,637.3	19,579	278.9	348	46,191.5	115,817

Table 3 continued

$n \times m$	b_i	CPU of HA		CPU of TS		CPU of SA		CPU of B&B	
		Avg	Max	Avg	Max	Avg	Max	Avg	Max
300×8	$b_i \in [0.01, 0.05]$	36.5	44	14,959.1	16,976	522.5	591	4083.6	6821
	$b_i \in [0.05, 0.15]$	39.2	51	15,947	22,194	550	764	3090.7	4189
	$b_i \in [0.15, 0.3]$	37.9	51	16,923.4	20,428	590.1	704	2626.6	3607
300×9	$b_i \in [0.01, 0.05]$	40.3	46	19,575.2	25,463	531.3	694	20,680.9	40,142
	$b_i \in [0.05, 0.15]$	47.1	54	24,303.6	33,384	656.7	900	24,144.4	38,137
	$b_i \in [0.15, 0.3]$	42.6	60	21,556.6	35,979	580.7	984	8584.3	20,132
300×10	$b_i \in [0.01, 0.05]$	43.9	52	24,051.8	26,491	517	580	132,859.1	228,098
	$b_i \in [0.05, 0.15]$	45.3	60	24,719.2	32,677	529.9	706	92,817.8	196,719
	$b_i \in [0.15, 0.3]$	45.4	52	26,078.5	32,922	558.2	690	30,683.7	69,731
300×11	$b_i \in [0.01, 0.05]$	54.5	70	34,380.6	46,158	596.8	815	1,073,264.2	1,896,520
	$b_i \in [0.05, 0.15]$	51.1	61	30,796.1	35,228	535.1	623	233,685	308,567
	$b_i \in [0.15, 0.3]$	53.1	61	33,604.2	40,466	583.6	718	118,402.7	403,293
400×8	$b_i \in [0.01, 0.05]$	58.66	68	30,715.55	35,988	1071.77	1279	6958.44	8215
	$b_i \in [0.05, 0.15]$	57.222	66	29,011.33	35,049	1023.77	1226	5410.11	9720
	$b_i \in [0.15, 0.3]$	54.2	67	26,541.1	37,782	912.5	1334	3042.5	4854
400×9	$b_i \in [0.01, 0.05]$	60.7	76	33,834.4	484,39	922.5	1337	37,927.1	64,279
	$b_i \in [0.05, 0.15]$	64.5	90	36,529.7	54,684	998.8	1502	28,087.7	48,918
	$b_i \in [0.15, 0.3]$	65.3	100	32,863.9	38,094	948.6	1475	11,629.3	21,071
400×10	$b_i \in [0.01, 0.05]$	70.3	88	43,795.6	53,749	957.8	1152	266,807.5	528,487
	$b_i \in [0.05, 0.15]$	69.8	88	44,247.7	60,302	961.1	1342	130,981.4	221,241
	$b_i \in [0.15, 0.3]$	69.4	89	43,792.7	65,310	947.5	1427	54,457.3	143,389
400×11	$b_i \in [0.01, 0.05]$	74.8	93	51,511	62,342	913.1	1109	1,280,255.9	2,280,752
	$b_i \in [0.05, 0.15]$	73.8	86	51,789.4	62,791	914.4	1119	622,038.5	1,146,756
	$b_i \in [0.15, 0.3]$	81.5	142	57,438.2	109,956	1019.2	1994	247,214.9	590,915

Table 4 CPU time for $a_i \in [3, 100]$ and $p_{ij} \in [3, 100]$ (ms)

$n \times m$	b_i	CPU of HA		CPU of TS		CPU of SA		CPU of B&B	
		Avg	Max	Avg	Max	Avg	Max	Avg	Max
100×8	$b_i \in [0.01, 0.05]$	12.4	15	2438.1	3200	72.2	88	399.6	743
	$b_i \in [0.05, 0.15]$	12.9	16	2589.9	3041	79.1	99	261.4	484
	$b_i \in [0.15, 0.3]$	12.1	16	2434.3	2845	75.3	92	160.7	302
100×9	$b_i \in [0.01, 0.05]$	13.3	17	3155.5	3610	70.3	85	1513	2934
	$b_i \in [0.05, 0.15]$	13.6	19	3175.9	3379	72.1	80	986.6	1702
	$b_i \in [0.15, 0.3]$	13.3	16	3057.9	3265	68.8	79	595.5	895
100×10	$b_i \in [0.01, 0.05]$	14.3	20	4367.3	4778	74.9	88	7509.5	16172
	$b_i \in [0.05, 0.15]$	15.9	26	4739.1	5548	85.7	105	4275.4	8644
	$b_i \in [0.15, 0.3]$	14.5	20	4408.4	5542	81.9	113	2368	4553
100×11	$b_i \in [0.01, 0.05]$	16.7	23	5658	7401	78.1	107	61,647.6	168,407
	$b_i \in [0.05, 0.15]$	17	22	6143.9	7333	82.8	102	18,556.1	38,504
	$b_i \in [0.15, 0.3]$	23.2	28	6214.6	7777	89.4	138	9467.3	26,086
200×8	$b_i \in [0.01, 0.05]$	24.3	30	9340.5	14,788	313.7	520	1940.1	3285
	$b_i \in [0.05, 0.15]$	28.7	36	9657.9	12,304	333.9	414	1806.4	3136
	$b_i \in [0.15, 0.3]$	25.8	35	9150.9	13,338	311.5	461	1184.5	2030
200×9	$b_i \in [0.01, 0.05]$	26.3	32	9915.4	13,101	256.4	344	10,807.2	19,809
	$b_i \in [0.05, 0.15]$	38.2	52	10,719.4	13,355	371.9	1266	6647	10,726
	$b_i \in [0.15, 0.3]$	26.8	31	10,365.4	11,633	272.7	309	2439.7	4014
200×10	$b_i \in [0.01, 0.05]$	29.6	34	15,066.4	21,451	290.8	418	59,189.8	154,225
	$b_i \in [0.05, 0.15]$	15.9	26	4739.1	5548	85.7	105	4275.4	8644
	$b_i \in [0.15, 0.3]$	29.6	36	14,637.2	17,299	303.5	361	14,258.1	28,303
200×11	$b_i \in [0.01, 0.05]$	30.2	38	15,820	18,220	256.8	297	282,992.1	785,377
	$b_i \in [0.05, 0.15]$	30.5	34	15,372.9	18,338	252.8	304	110,502.1	247,065
	$b_i \in [0.15, 0.3]$	31.9	43	16,261.3	20,489	272.1	344	56,807.7	123,472

Table 4 continued

$n \times m$	b_i	CPU of HA		CPU of TS		CPU of SA		CPU of B&B	
		Avg	Max	Avg	Max	Avg	Max	Avg	Max
300×8	$b_i \in [0.01, 0.05]$	36.9	45	16,049.6	24,243	559.1	863	3780.7	7072
	$b_i \in [0.05, 0.15]$	39.4	47	18,804.6	34,505	603.4	811	3889.8	6147
	$b_i \in [0.15, 0.3]$	37.9	51	16,672.6	24,845	579	861	1627.7	2814
300×9	$b_i \in [0.01, 0.05]$	46.6	73	22,554.7	29,647	609.3	805	26,705.7	42,666
	$b_i \in [0.05, 0.15]$	40.2	48	19,269.1	24,366	521.1	667	11,605.3	27,181
	$b_i \in [0.15, 0.3]$	37.8	41	17,780.9	20,867	477.4	556	5428.1	9944
300×10	$b_i \in [0.01, 0.05]$	44	48	23,368.2	26,824	497.5	575	108,693.3	20,008
	$b_i \in [0.05, 0.15]$	46	57	26,420.8	36,872	561.8	789	61,377	104,706
	$b_i \in [0.15, 0.3]$	47.7	65	26,236.9	33,303	564.7	704	29,147.1	55,446
300×11	$b_i \in [0.01, 0.05]$	49.9	59	30,082.5	37,313	520.7	647	847,295.6	1,838,027
	$b_i \in [0.05, 0.15]$	48.8	58	30,575.4	39,188	530.6	673	339,681.6	732,423
	$b_i \in [0.15, 0.3]$	52.8	78	33,815.2	49,498	587.5	864	119,156.6	174,276
400×8	$b_i \in [0.01, 0.05]$	54.1	60	27,047.4	32,571	954.7	1159	7623.7	10717
	$b_i \in [0.05, 0.15]$	57.1	75	29,043.5	51,362	994.9	1562	5315.6	10,884
	$b_i \in [0.15, 0.3]$	57.4	82	28,723	42,912	1015	1526	3779.3	7716
400×9	$b_i \in [0.01, 0.05]$	69.6	84	40,551.5	50,569	1087.8	1388	59,243.6	111,898
	$b_i \in [0.05, 0.15]$	63.9	78	36,634.7	42,653	993.6	1155	27,425.8	45,730
	$b_i \in [0.15, 0.3]$	79	192	37,266.9	51,645	1901.6	9762	15,600.7	25,798
400×10	$b_i \in [0.01, 0.05]$	72	82	47171.6	59,377	1033.2	1303	286,142.7	462,692
	$b_i \in [0.05, 0.15]$	77.3	98	50,475.2	73,363	1099.1	1560	150,222.3	349,776
	$b_i \in [0.15, 0.3]$	74.1	85	47,763.8	55,179	1041.9	1215	53,537.5	103,896
400×11	$b_i \in [0.01, 0.05]$	74.9	85	48790.6	53,361	850	922	1,238,078.6	1,946,396
	$b_i \in [0.05, 0.15]$	78.4	92	54,904.9	67,376	969.3	1214	629,933.5	1,180,752
	$b_i \in [0.15, 0.3]$	79.2	99	54,742.4	70,217	967.2	1254	200,425.9	399,793

Table 5 Error bound for $a_i \in [3, 50]$ and $p_{ij} \in [3, 50]$

$n \times m$	b_i	$\frac{\sum\sum C_{ij}(HA)}{\sum\sum C^*_{ij}}$		$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C^*_{ij}}$		$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C^*_{ij}}$	
		Avg	Max	Avg	Max	Avg	Max
100×8	$b_i \in [0.01, 0.05]$	1.0008	1.0027	1.014	1.0416	1	1
	$b_i \in [0.05, 0.15]$	1.0028	1.0093	1.0281	1.0643	1.0011	1.0093
	$b_i \in [0.15, 0.3]$	1.0168	1.0676	1.0652	1.1571	1.0027	1.0217
100×9	$b_i \in [0.01, 0.05]$	1.0004	1.0023	1.0405	1.0743	1.0002	1.0023
	$b_i \in [0.05, 0.15]$	1.0012	1.0076	1.0432	1.0619	1	1
	$b_i \in [0.15, 0.3]$	1.0094	1.0288	1.1114	1.1886	1.0012	1.0083
100×10	$b_i \in [0.01, 0.05]$	1.0003	1.0023	1.029	1.0713	1.0001	1.001
	$b_i \in [0.05, 0.15]$	1.0044	1.0109	1.0667	1.1036	1	1
	$b_i \in [0.15, 0.3]$	1.0096	1.0351	1.1218	1.2301	1	1
100×11	$b_i \in [0.01, 0.05]$	1.0004	1.0016	1.0461	1.0763	1	1
	$b_i \in [0.05, 0.15]$	1.01	1.0257	1.0649	1.1163	1	1
	$b_i \in [0.15, 0.3]$	1.0199	1.0864	1.11585	1.1776	1.001	1.0106
200×8	$b_i \in [0.01, 0.05]$	1.0002	1.0028	1.0158	1.0354	1	1
	$b_i \in [0.05, 0.15]$	1.0061	1.0216	1.0335	1.0745	1.0033	1.0207
	$b_i \in [0.15, 0.3]$	1.02	1.0606	1.0664	1.1262	1.0138	1.0497
200×9	$b_i \in [0.01, 0.05]$	1.0005	1.0035	1.026	1.0487	1.0002	1.0021
	$b_i \in [0.05, 0.15]$	1.0027	1.0166	1.0402	1.0634	1.0024	1.0166
	$b_i \in [0.15, 0.3]$	1.0104	1.0288	1.0886	1.1711	1.0019	1.0108
200×10	$b_i \in [0.01, 0.05]$	1.0033	1.0154	1.0363	1.0607	1	1
	$b_i \in [0.05, 0.15]$	1.0046	1.0197	1.0643	1.1122	1.0006	1.0065
	$b_i \in [0.15, 0.3]$	1.017	1.0531	1.1375	1.2105	1.0015	1.0072
200×11	$b_i \in [0.01, 0.05]$	1.003	1.0121	1.0326	1.0584	1.0002	1.0015
	$b_i \in [0.05, 0.15]$	1.0151	1.0678	1.0688	1.1336	1.0004	1.004
	$b_i \in [0.15, 0.3]$	1.0262	1.0545	1.15	1.245	1.0066	1.034
300×8	$b_i \in [0.01, 0.05]$	1.0009	1.0044	1.0153	1.0269	1.0008	1.0044
	$b_i \in [0.05, 0.15]$	1.0052	1.0196	1.0316	1.0557	1.0009	1.0094
	$b_i \in [0.15, 0.3]$	1.0122	1.043	1.0545	1.1167	1.0076	1.0284
300×9	$b_i \in [0.01, 0.05]$	1.0016	1.0057	1.0228	1.046	1.0016	1.0057
	$b_i \in [0.05, 0.15]$	1.0082	1.0365	1.0431	1.0685	1.0014	1.0104
	$b_i \in [0.15, 0.3]$	1.0145	1.0545	1.083	1.1423	1.0024	1.0209
300×10	$b_i \in [0.01, 0.05]$	1.0018	1.0068	1.0254	1.0512	1.0005	1.0036
	$b_i \in [0.05, 0.15]$	1.0084	1.0253	1.0636	1.0967	1.0047	1.0236
	$b_i \in [0.15, 0.3]$	1.0256	1.0809	1.12	1.1879	1.0066	1.0572
300×11	$b_i \in [0.01, 0.05]$	1.0029	1.0118	1.0319	1.0409	1.0013	1.0118
	$b_i \in [0.05, 0.15]$	1.0094	1.0322	1.0799	1.1105	1.0032	1.0232
	$b_i \in [0.15, 0.3]$	1.0335	1.0851	1.1347	1.2083	1.0071	1.0552

Table 5 continued

$n \times m$	b_i	$\frac{\sum\sum C_{ij}(HA)}{\sum\sum C^*_{ij}}$ Avg	Max	$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C^*_{ij}}$ Avg	Max	$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C^*_{ij}}$ Avg	Max
400×8	$b_i \in [0.01, 0.05]$	1.0007	1.0029	1.0272	1.0523	1.0002	1.0029
	$b_i \in [0.05, 0.15]$	1.0047	1.0191	1.0307	1.0596	1.0032	1.0186
	$b_i \in [0.15, 0.3]$	1.0191	1.0542	1.0579	1.0962	1.0043	1.0333
400×9	$b_i \in [0.01, 0.05]$	1.0002	1.0011	1.0213	1.0448	1	1
	$b_i \in [0.05, 0.15]$	1.0012	1.0039	1.0421	1.0673	1	1
	$b_i \in [0.15, 0.3]$	1.0166	1.0537	1.0788	1.1452	1.003	1.0146
400×10	$b_i \in [0.01, 0.05]$	1.0041	1.0172	1.0299	1.0544	1.0001	1.0014
	$b_i \in [0.05, 0.15]$	1.0113	1.0364	1.0585	1.0882	1.0065	1.0241
	$b_i \in [0.15, 0.3]$	1.0278	1.0654	1.1086	1.1697	1.005	1.0319
400×11	$b_i \in [0.01, 0.05]$	1.0024	1.0062	1.034	1.0583	1.0014	1.0062
	$b_i \in [0.05, 0.15]$	1.009	1.029	1.0675	1.1184	1.0031	1.0225
	$b_i \in [0.15, 0.3]$	1.0289	1.0695	1.1339	1.2166	1.0103	1.0329

Table 6 Error bound for $a_i \in [51, 100]$ and $p_{ij} \in [51, 100]$

$n \times m$	b_i	$\frac{\sum\sum C_{ij}(HA)}{\sum\sum C^*_{ij}}$ Avg	Max	$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C^*_{ij}}$ Avg	Max	$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C^*_{ij}}$ Avg	Max
100×8	$b_i \in [0.01, 0.05]$	1.0002	1.0016	1.0172	1.0528	1	1
	$b_i \in [0.05, 0.15]$	1.0018	1.0132	1.0236	1.0545	1.0005	1.003
	$b_i \in [0.15, 0.3]$	1.0042	1.0154	1.0585	1.0894	1.0007	1.007
100×9	$b_i \in [0.01, 0.05]$	1.0002	1.0014	1.0232	1.0461	1.0001	1.0011
	$b_i \in [0.05, 0.15]$	1.0018	1.0137	1.0435	1.1028	1	1
	$b_i \in [0.15, 0.3]$	1.0153	1.0799	1.0749	1.1567	1.0052	1.0375
100×10	$b_i \in [0.01, 0.05]$	1.0005	1.0025	1.0319	1.0563	1.0001	1.0008
	$b_i \in [0.05, 0.15]$	1.0031	1.0098	1.0483	1.1069	1.0005	1.0057
	$b_i \in [0.15, 0.3]$	1.0083	1.0371	1.0986	1.1739	1.00130	1.0051
100×11	$b_i \in [0.01, 0.05]$	1.0006	1.0026	1.0411	1.0567	1	1
	$b_i \in [0.05, 0.15]$	1.0035	1.0114	1.0555	1.1032	1.0011	1.0114
	$b_i \in [0.15, 0.3]$	1.0057	1.0272	1.1157	1.2215	1.0001	1.0003
200×8	$b_i \in [0.01, 0.05]$	1.0155	1.0506	1.0677	1.143	1.0064	1.032
	$b_i \in [0.05, 0.15]$	1.0046	1.0199	1.0409	1.0703	1.0031	1.0136
	$b_i \in [0.15, 0.3]$	1.0089	1.0232	1.0528	1.1422	1.0039	1.0161
200×9	$b_i \in [0.01, 0.05]$	1.0007	1.0029	1.0163	1.0259	1.0002	1.0023
	$b_i \in [0.05, 0.15]$	1.0078	1.0361	1.0366	1.0681	1.001	1.0091
	$b_i \in [0.15, 0.3]$	1.0185	1.0567	1.0994	1.1514	1.0024	1.0207
200×10	$b_i \in [0.01, 0.05]$	1.0017	1.005	1.0188	1.036	1.0005	1.005
	$b_i \in [0.05, 0.15]$	1.0051	1.0206	1.0555	1.0735	1.0019	1.0195
	$b_i \in [0.15, 0.3]$	1.0092	1.0579	1.0804	1.157	1.0007	1.0055

Table 6 continued

$n \times m$	b_i	$\frac{\sum\sum C_{ij}(HA)}{\sum\sum C_{ij}^*}$		$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C_{ij}^*}$		$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C_{ij}^*}$	
		Avg	Max	Avg	Max	Avg	Max
200×11	$b_i \in [0.01, 0.05]$	1.0004	1.0013	1.0296	1.0505	1	1
	$b_i \in [0.05, 0.15]$	1.0046	1.0111	1.0589	1.0881	1.0012	1.007
	$b_i \in [0.15, 0.3]$	1.0152	1.0539	1.1205	1.2821	1.0007	1.0045
300×8	$b_i \in [0.01, 0.05]$	1.0001	1.0008	1.01332	1.0203	1	1
	$b_i \in [0.05, 0.15]$	1.0065	1.0226	1.0382	1.0816	1.0057	1.0226
	$b_i \in [0.15, 0.3]$	1.01697	1.0827	1.0595	1.1118	1.0065	1.0202
300×9	$b_i \in [0.01, 0.05]$	1.0013	1.0079	1.023	1.062	1.0006	1.0068
	$b_i \in [0.05, 0.15]$	1.0096	1.054	1.0347	1.0655	1.003	1.017
	$b_i \in [0.15, 0.3]$	1.0224	1.1071	1.0507	1.1234	1.0015	1.0154
300×10	$b_i \in [0.01, 0.05]$	1.0018	1.0092	1.032	1.0681	1.0004	1.0037
	$b_i \in [0.05, 0.15]$	1.0122	1.0419	1.0413	1.0784	1.0061	1.0322
	$b_i \in [0.15, 0.3]$	1.0188	1.0688	1.1032	1.1939	1.0041	1.035
300×11	$b_i \in [0.01, 0.05]$	1.0018	1.0073	1.0258	1.0412	1	1
	$b_i \in [0.05, 0.15]$	1.0009	1.0046	1.0605	1.1033	1	1
	$b_i \in [0.15, 0.3]$	1.0329	1.0898	1.1138	1.1435	1.005	1.0273
400×8	$b_i \in [0.01, 0.05]$	1.0006	1.0035	1.0173	1.0342	1	1
	$b_i \in [0.05, 0.15]$	1.0081	1.0211	1.0439	1.1106	1.0027	1.0086
	$b_i \in [0.15, 0.3]$	1.021	1.0676	1.0727	1.1273	1.0071	1.0268
400×9	$b_i \in [0.01, 0.05]$	1.0015	1.0132	1.022	1.0383	1.0007	1.007
	$b_i \in [0.05, 0.15]$	1.0052	1.026	1.0405	1.1184	1.0011	1.008
	$b_i \in [0.15, 0.3]$	1.0261	1.0653	1.0801	1.1293	1.0055	1.0284
400×10	$b_i \in [0.01, 0.05]$	1.0031	1.0107	1.0232	1.0401	1.0017	1.0091
	$b_i \in [0.05, 0.15]$	1.0098	1.0328	1.0567	1.1222	1.0013	1.0064
	$b_i \in [0.15, 0.3]$	1.027	1.0829	1.0881	1.1333	1.0071	1.0236
400×11	$b_i \in [0.01, 0.05]$	1.0034	1.0076	1.04	1.0604	1.0006	1.0065
	$b_i \in [0.05, 0.15]$	1.0082	1.0234	1.0786	1.116	1.0043	1.0153
	$b_i \in [0.15, 0.3]$	1.025	1.0779	1.1205	1.2146	1.0105	1.0607

Table 7 Error bound for $a_i \in [3, 100]$ and $p_{ij} \in [3, 100]$

$n \times m$	b_i	$\frac{\sum\sum C_{ij}(HA)}{\sum\sum C_{ij}^*}$		$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C_{ij}^*}$		$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C_{ij}^*}$	
		Avg	Max	Avg	Max	Avg	Max
100×8	$b_i \in [0.01, 0.05]$	1.0001	1.001	1.022	1.0598	1.0001	1.001
	$b_i \in [0.05, 0.15]$	1.0023	1.0124	1.0343	1.0608	1.0007	1.0075
	$b_i \in [0.15, 0.3]$	1.0037	1.0239	1.0907	1.1733	1.003	1.0239
100×9	$b_i \in [0.01, 0.05]$	1.0007	1.0035	1.0343	1.0698	1.0002	1.0022
	$b_i \in [0.05, 0.15]$	1.002	1.017	1.0413	1.0896	1.0002	1.0021
	$b_i \in [0.15, 0.3]$	1.0059	1.027	1.0988	1.153	1.0002	1.0017

Table 7 continued

$n \times m$	b_i	$\frac{\sum\sum C_{ij}(HA)}{\sum\sum C_{ij}^*}$		$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C_{ij}^*}$		$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C_{ij}^*}$	
		Avg	Max	Avg	Max	Avg	Max
100×10	$b_i \in [0.01, 0.05]$	1.0002	1.0015	1.0431	1.0872	1	1
	$b_i \in [0.05, 0.15]$	1.0022	1.0082	1.0582	1.1444	1	1
	$b_i \in [0.15, 0.3]$	1.0163	1.0495	1.1132	1.1664	1.0022	1.0116
100×11	$b_i \in [0.01, 0.05]$	1.001	1.0039	1.04	1.07278	1.0001	1.0002
	$b_i \in [0.05, 0.15]$	1.003	1.0096	1.1053	1.1827	1.0005	1.0051
	$b_i \in [0.15, 0.3]$	1.0146	1.044	1.1353	1.186	1.0004	1.0048
200×8	$b_i \in [0.01, 0.05]$	1.0008	1.0032	1.0172	1.0367	1.0001	1.0016
	$b_i \in [0.05, 0.15]$	1.0055	1.0266	1.0352	1.1088	1.0031	1.0266
	$b_i \in [0.15, 0.3]$	1.01385	1.042	1.0775	1.1472	1.005	1.0249
200×9	$b_i \in [0.01, 0.05]$	1.0012	1.0057	1.0312	1.0607	1.0004	1.002
	$b_i \in [0.05, 0.15]$	1.0037	1.0172	1.0568	1.0897	1.0003	1.0031
	$b_i \in [0.15, 0.3]$	1.008	1.0243	1.0843	1.1395	1.0006	1.0043
200×10	$b_i \in [0.01, 0.05]$	1.002	1.0064	1.0333	1.0525	1.0003	1.003
	$b_i \in [0.05, 0.15]$	1.0022	1.0082	1.0582	1.1444	1	1
	$b_i \in [0.15, 0.3]$	1.0064	1.0216	1.1084	1.1817	1.0033	1.0216
200×11	$b_i \in [0.01, 0.05]$	1.0013	1.0034	1.0325	1.0579	1	1
	$b_i \in [0.05, 0.15]$	1.0097	1.0363	1.0815	1.1225	1.0045	1.0131
	$b_i \in [0.15, 0.3]$	1.0386	1.1701	1.1112	1.1967	1.0042	1.0183
300×8	$b_i \in [0.01, 0.05]$	1.0005	1.0027	1.0221	1.0482	1.0002	1.0027
	$b_i \in [0.05, 0.15]$	1.0057	1.0282	1.037	1.094	1.0011	1.0071
	$b_i \in [0.15, 0.3]$	1.0182	1.081	1.0559	1.11	1.0053	1.0262
300×9	$b_i \in [0.01, 0.05]$	1.0017	1.0101	1.02	1.0343	1.0008	1.0069
	$b_i \in [0.05, 0.15]$	1.007	1.0262	1.05	1.0897	1.0023	1.0203
	$b_i \in [0.15, 0.3]$	1.0165	1.0423	1.0721	1.1077	1.0041	1.0174
300×10	$b_i \in [0.01, 0.05]$	1.0014	1.0073	1.0312	1.061	1.0008	1.0073
	$b_i \in [0.05, 0.15]$	1.0127	1.0706	1.0604	1.08	1.0003	1.003
	$b_i \in [0.15, 0.3]$	1.0215	1.065	1.095	1.1401	1.0074	1.0348
300×11	$b_i \in [0.01, 0.05]$	1.0025	1.0086	1.0363	1.092	1.0004	1.0045
	$b_i \in [0.05, 0.15]$	1.0169	1.0676	1.0637	1.1037	1.0011	1.0081
	$b_i \in [0.15, 0.3]$	1.032	1.1143	1.1419	1.2263	1.0051	1.0254
400×8	$b_i \in [0.01, 0.05]$	1.0034	1.0129	1.0185	1.0392	1.0001	1.0004
	$b_i \in [0.05, 0.15]$	1.0056	1.0223	1.031	1.0468	1.0003	1.0034
	$b_i \in [0.15, 0.3]$	1.0131	1.0506	1.0642	1.1351	1.0055	1.0237
400×9	$b_i \in [0.01, 0.05]$	1.0028	1.0129	1.0243	1.045	1.0014	1.0099
	$b_i \in [0.05, 0.15]$	1.0086	1.0407	1.0523	1.0712	1.0025	1.0132
	$b_i \in [0.15, 0.3]$	1.0104	1.0296	1.0824	1.1604	1.0043	1.0204

Table 7 continued

$n \times m$	b_i	$\frac{\sum\sum C_{ij}(HA)}{\sum\sum C_{ij}^*}$ Avg	Max	$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C_{ij}^*}$ Avg	Max	$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C_{ij}^*}$ Avg	Max
400×10	$b_i \in [0.01, 0.05]$	1.0038	1.0108	1.0229	1.0426	1.0002	1.0015
	$b_i \in [0.05, 0.15]$	1.0077	1.0361	1.0516	1.0879	1.0019	1.0195
	$b_i \in [0.15, 0.3]$	1.0362	1.1628	1.1113	1.195	1.0088	1.0396
400×11	$b_i \in [0.01, 0.05]$	1.0014	1.0101	1.0349	1.0636	1	1
	$b_i \in [0.05, 0.15]$	1.0076	1.0284	1.0708	1.1032	1.0022	1.0092
	$b_i \in [0.15, 0.3]$	1.0391	1.0728	1.1307	1.1687	1.0083	1.0323

Table 8 Results of Algorithms of $n = 300, m = 10$, $b_i \in [0.01, 0.05]$

Instance	CPU time (ms)	$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C_{ij}^*}$	$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C_{ij}^*}$
1	107,339	1.1441	1
2	164,432	1.1252	1
3	81,965	1.154	1.0027
4	167,851	1.0581	1
5	372,789	1.0618	1
6	60,311	1.0222	1
7	113,158	1.0356	1
8	182,675	1.0235	1
9	74,764	1.0546	1
10	163,167	1.0913	1
11	128,852	1.0301	1.0041
12	82,988	1.0825	1.005
13	190,792	1.0878	1
14	96,842	1.071	1
15	52,382	1.0507	1
16	86,546	1.0725	1
17	110,334	1.0092	1
18	96,888	1.0374	1
19	165,049	1.0556	1
20	114,608	1.0745	1.0004

Table 9 Results of algorithms of $n = 300, m = 10, b_i \in [0.05, 0.15]$

Instance	CPU time (ms)	$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C_{ij}^*}$	$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C_{ij}^*}$
1	64,590	1.2581	1
2	71,967	1.0807	1
3	67,577	1.169	1
4	52,815	1.19	1
5	65,175	1.0878	1.0045
6	102,639	1.2263	1.0071
7	95,399	1.189	1
8	85,117	1.1119	1.0027
9	27,313	1.1512	1
10	179,983	1.107	1.009
11	130,690	1.2035	1.0021
12	74,422	1.0647	1
13	61,280	1.1668	1
14	49,197	1.086	1.0028
15	132,698	1.0535	1
16	151,276	1.1472	1.0196
17	92,060	1.1496	1
18	84,679	1.0919	1.0092
19	73,203	1.1702	1.0021
20	57,456	1.0882	1

Table 10 Results of Algorithms of $n = 300$, $m = 10$, $b_i \in [0.15, 0.3]$

Instance	CPU time (ms)	$\frac{\sum\sum C_{ij}(TS)}{\sum\sum C^*_{ij}}$	$\frac{\sum\sum C_{ij}(SA)}{\sum\sum C^*_{ij}}$
1	17,176	1.1494	1.0016
2	23,535	1.2702	1
3	32,165	1.2369	1
4	15,839	1.2738	1
5	24,537	1.1608	1.0084
6	115,686	1.2718	1.0008
7	29,347	1.3257	1
8	55,542	1.1239	1.0039
9	21,313	1.2513	1.0017
10	21,294	1.1118	1.0019
11	10,969	1.1155	1.0212
12	62,743	1.201	1.0197
13	36,119	1.3594	1.011
14	18,132	1.176	1.0049
15	12,125	1.1377	1
16	24,867	1.1102	1.0032
17	31,678	1.3194	1.0225
18	30,478	1.1246	1
19	27,935	1.2001	1.0054
20	21,227	1.2653	1

5 Conclusions

We discussed the single-machine group scheduling where group setup times are increasing functions of their starting times and the jobs in the same group have general truncated learning effects. We showed that the makespan minimization problem remains polynomially solvable. For the total completion time minimization, we proposed heuristic algorithms and a branch-and-bound algorithm. Experimental study showed that the B&B algorithm can solve instances of 400×11 jobs in less than 2,815,317 ms, and the algorithms of SA are more accurate than HA and TS. In future research, we expect to explore more general group models with deteriorating jobs and/or learning effect, extend the problems to due date (window) assignments (Geng et al. 2023; Wang et al. 2024a; Sun et al. 2024), or consider flow shop scheduling (Yu et al. 2023; Wang et al. 2024c) with group technology.

Funding This work was supported by the Natural Science Foundation of Liaoning Province, China (2024-MS-175) and the Basic Scientific Research Project of Liaoning Provincial Department of Education (JYTZD2023050).

Data availability The data used to support the findings of this paper are available from the corresponding author upon request.

Declarations

Conflict of interest The authors declare that they have no Conflict of interest.

References

Alidaee B, Womer NK (1999) Scheduling with time dependent processing processing times: review and extensions. J Oper Res Soc 50:711–720
Azzouz A, Ennigrou M, Said LB (2018) Scheduling problems under learning effects: classification and cartography. Int J Prod Res 56:1642–1661
Biskup D (2008) A state-of-the-art review on scheduling with learning effects. Eur J Oper Res 188:315–329
Browne S, Yechiali U (1990) Scheduling deteriorating jobs on a single processor. Oper Res 38:495–498
Cheng TCE, Ding Q, Lin BMT (2004) A concise survey of scheduling with time-dependent processing times. Eur J Oper Res 152:1–13
Cheng TCE, Lai P-J, Wu C-C, Lee W-C (2009) Single-machine scheduling with sum-of-logarithm-processing-times-based learning considerations. Inf Sci 179:3127–3135
Gawiejnowicz S (2020a) Models and algorithms of time-dependent scheduling. Springer, Berlin
Gawiejnowicz S (2020b) A review of four decades of time-dependent scheduling: main results, new topics, and open problems. J Sched 23:3–47
Geng X-N, Sun X, Wang J-Y, Pan L (2023) Scheduling on proportionate flow shop with job rejection and common due date assignment. Comput Ind Eng 181:109317
He Y, Sun L (2015) One-machine scheduling problems with deteriorating jobs and position-dependent learning effects under group technology considerations. Int J Syst Sci 46(7):1319–1326
He H, Zhao Y, Ma X, Lv Z, Wang J-B (2023) Branch-and-bound and heuristic algorithms for group scheduling with due-date assignment and resource allocation. Mathematics 11(23):4745
Huang X (2019) Bicriterion scheduling with group technology and deterioration effect. J Appl Math Comput 60:455–464
Kuo W-H, Yang D-L (2006) Single-machine group scheduling with a time-dependent learning effect. Comput Oper Res 33:2099–2112
Lee W-C, Wu C-C (2009) A note on single-machine group scheduling problems with position-based learning effect. Appl Math Model 33:2159–2163
Li M-H, Lv D-Y, Lu Y-Y, Wang J-B (2024a) Scheduling with group technology, resource allocation, and learning effect simultaneously. Mathematics 12(7):1029
Li M-H, Lv D-Y, Lv Z-G, Zhang L-H, Wang J-B (2024b) A two-agent resource allocation scheduling problem with slack due-date assignment and general deterioration function. Comput Appl Math 43(4):229

Li M-H, Lv D-Y, Zhang L-H, Wang J-B (2024c) Permutation flow shop scheduling with makespan objective and truncated learning effects. J Appl Math Comput. https://doi.org/10.1007/s12190-024-02080-w
Liu WG, Wang XY (2023) Group technology scheduling with due-date assignment and controllable processing times. Processes 11:1271
Liu F, Yang J, Lu Y-Y (2019) Solution algorithms for single-machine group scheduling with ready times and deteriorating jobs. Eng Optim 51(5):862–874
Lu Y-Y, Zhang S, Tao J-Y (2024) Earliness-tardiness scheduling with delivery times and deteriorating jobs. Asia Pac J Oper Res. https://doi.org/10.1142/S021759592450009X
Lv D-Y, Wang J-B (2024a) No-idle flow shop scheduling with deteriorating jobs and common due date under dominating machines. Asia Pac J Oper Res. https://doi.org/10.1142/S0217595924500039
Lv D-Y, Wang J-B (2024b) Research on two-machine flow shop scheduling problem with release dates and truncated learning effects. Eng Optim. https://doi.org/10.1080/0305215X.2024.2372633
Lv Z-G, Zhang L-H, Wang X-Y, Wang J-B (2024) Single machine scheduling proportionally deteriorating jobs with ready times subject to the total weighted completion time minimization. Mathematics 12:610
Ma R, Guo SA, Zhang XY (2023) An optimal online algorithm for single-processor scheduling problem with learning effect. Theor Comput Sci 928:1–12
Mao R-R, Wang Y-C, Lv D-Y, Wang J-B, Lu Y-Y (2023) Delivery times scheduling with deterioration effects in due window assignment environments. Mathematics 11:3983
Mao R-R, Lv D-Y, Ren N, Wang J-B (2024) Supply chain scheduling with deteriorating jobs and delivery times. J Appl Math Comput 70(3):2285–2312
Miao CX (2019) Parallel-batch scheduling with deterioration and group technology. IEEE Access 7:119082–119086
Miao CX, Song J, Zhang Y (2023) Single-machine time-dependent scheduling with proportional and delivery times. Asia Pac J Oper Res 40(4):2240015
Mosheiov G (1991) V-shaped policies for scheduling deteriorating jobs. Oper Res 39:979–991
Nawaz M, Enscore EE, Ham I (1983) A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega 11:91–95
Niu Y-P, Wan L, Wang J-B (2015) A note on scheduling jobs with extended sum-of-processing-times-based and position-based learning effect. Asia Pac J Oper Res 32(2):1550001
Paredes-Astudillo YA, Botta-Genoulaz V, Montoya-Torres JR (2024) Impact of learning effect modelling in flowshop scheduling with makespan minimisation based on the Nawaz–Enscore–Ham algorithm. Int J Prod Res 62(6):1999–2014
Pei J, Wang X, Fan W, Pardalos PM, Liu X (2019) Scheduling step-deteriorating jobs on bounded parallel-batching machines to maximise the total net revenue. J Oper Res Soc 70(10):1830–1847
Potts CN, Van Wassenhove LN (1992) Integrating scheduling with batching and lot-sizing: a review of algorithms and complexity. J Oper Res Soc 43:395–406
Qian J (2023) Note on single machine common flow allowance group scheduling with learning effect and resource allocation. Comput Ind Eng 186:109750
Qian J, Han HY (2022) The due date assignment scheduling problem with the deteriorating jobs and delivery time. J Appl Math Comput 68:2173–2186
Qian J, Zhan Y (2021) The due date assignment scheduling problem with delivery times and truncated sum-of-processing-times-based learning effect. Mathematics 9(23):3085
Qian J, Chang G, Zhang X (2024) Single-machine common due-window assignment and scheduling with position-dependent weights, delivery time, learning effect and resource allocations. J Appl Math Comput 70(3):1965–1994
Sun L, Ma W (2020) Research into group scheduling problems with deterioration and general learning effect. Oper Res Manag Sci 29(3):125–127 (**(in Chinese)**)
Sun X, Geng X-N, Liu F (2021) Flow shop scheduling with general position weighted learning effects to minimise total weighted completion time. J Oper Res Soc 72(12):2674–2689
Sun X, Liu T, Geng X-N, Hu Y, Xu J-X (2023) Optimization of scheduling problems with deterioration effects and an optional maintenance activity. J Sched 26:251–266
Sun X, Geng X-N, Wang J-Y, Pan L (2024) Slack due window assignment scheduling in the single-machine with controllable processing times. J Ind Manag Optim 20(1):15–35
Wang XY, Liu WG (2024) Optimal different due-dates assignment scheduling with group technology and resource allocation. Mathematics 12(3):436
Wang Y-C, Wang J-B (2023) Study on convex resource allocation scheduling with a time-dependent learning effect. Mathematics 11:3179
Wang J-B, Xia Z-Q (2005) Flow shop scheduling with a learning effect. J Oper Res Soc 56:1325–1330
Wang J-B, Lin L, Shan F (2008) Single machine group scheduling problems with deteriorating jobs. Int J Adv Manuf Technol 39:808–812

Wang J-B, Gao W-J, Wang L-Y, Wang D (2009) Single machine group scheduling with general linear deterioration to minimize the makespan. Int J Adv Manuf Technol 43:146–150
Wang J-B, Lv D-Y, Xu J, Ji P, Li F (2021) Bicriterion scheduling with truncated learning effects and convex controllable processing times. Int Trans Oper Res 28(3):1573–1593
Wang S-H, Lv D-Y, Wang J-B (2023a) Research on position-dependent weights scheduling with delivery times and truncated sum-of-processing-times-based learning effect. J Ind Manag Optim 19(4):2824–2837
Wang J-B, Lv D-Y, Wang S-Y, Jiang C (2023b) Resource allocation scheduling with deteriorating jobs and position-dependent workloads. J Ind Manag Optim 19(3):1658–1669
Wang J-B, Bao H, Wan C (2024a) Research on multiple slack due-date assignments scheduling with position-dependent weights. Asia Pac J Oper Res. https://doi.org/10.1142/S0217595923500392
Wang X-Y, Lv D-Y, Ji P, Yin N, Wang J-B, Jin Q (2024b) Single machine scheduling problems with truncated learning effects and exponential past-sequence-dependent delivery times. Comput Appl Math 43(4):194
Wang J-B, Lv D-Y, Wan C (2024c) Proportionate flow shop scheduling with job-dependent due windows and position-dependent weights. Asia Pac J Oper Res. https://doi.org/10.1142/S0217595924500118
Wang J-B, Wang Y-C, Wan C, Lv D-Y, Zhang L (2024d) Controllable processing time scheduling with total weighted completion time objective and deteriorating jobs. Asia Pac J Oper Res 41(3):2350026
Wu C-C, Lee W-C (2008) Single-machine group-scheduling problems with deteriorating setup times and job-processing times. Int J Prod Econ 115:128–133
Wu C-C, Shiau Y-R, Lee W-C (2008) Single-machine group scheduling problems with deterioration consideration. Comput Oper Res 35:1652–1659
Xu H, Li X, Ruiz R, Zhu H (2021) Group scheduling with nonperiodical maintenance and deteriorating effects. IEEE Trans Syst Man Cybern Syst 51(5):2860–2872
Yan Y, Wang D-Z, Wang D-W, Wang H-F (2008) Single-machine group scheduling problems with deterioration and learning effects. IEEE Proceedings of the 7th world congress on intelligent control and automation. pp 4933–4936 Chongqing, China. https://doi.org/10.1109/WCICA.2008.4593725.
Yan J-X, Ren N, Bei H-B, Bao H, Wang J-B (2023) Study on resource allocation scheduling problem with learning factors and group technology. J Ind Manag Optim 19(5):3419–3435
Yang S-J, Yang D-L (2010) Single-machine group scheduling problems under the effects of deterioration and learning. Comput Ind Eng 58:754–758
Yin Y, Wu W-H, Cheng TCE, Wu C-C (2015) Single-machine scheduling with time-dependent and position-dependent deteriorating jobs. Int J Comput Integr Manuf 28(7):781–790
Yu Y, Pan Q, Pang X, Tang X (2023) An attribution feature-based memetic algorithm for hybrid flowshop scheduling problem with operation skipping. IEEE Trans Autom Sci Eng. https://doi.org/10.1109/TASE.2023.3346446
Zhang L-H, Lv D-Y, Wang J-B (2023) Two-agent slack due-date assignment scheduling with resource allocations and deteriorating jobs. Mathematics 11(12):2737
Zhang L-H, Geng X-N, Xue J, Wang J-B (2024) Single machine slack due window assignment and deteriorating jobs. J Ind Manag Optim 20:1593–1614
Zhao S (2022) Scheduling jobs with general truncated learning effects including proportional setup times. Comput Appl Math 41(4):146