Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A robust single-machine scheduling problem with scenario-dependent processing times and release dates

Chin-Chia Wu[a], Juin-Han Chen[b*], Win-Chin Lin[a], Xingong Zhang[c], Tao Ren[d], Zong-Lin Wu[a] and Yu-Hsiang Chung[e]

[a]*Department of Statistics, Feng Chia University, Taichung, 40724, Taiwan*
[b]*Department of Industrial Engineering and Management, National Quemoy University, Kinmen County 892, Taiwan*
[c]*College of Mathematics Science, Chongqing Normal University, Chongqing 401331, China*
[d]*Software College, Northeastern University, Shenyang, 110819, China*
[e]*Department of Industrial Engineering and Management, National Chin-Yi University of Technology, Taichung 411030, Taiwan*

C H R O N I C L E

Article history:
Received September 16 2024
Received in Revised Format October 26 2024
Accepted November 23 2024
Available online November 23 2024

Keywords:
Scheduling
Scenario-dependent
Iterated greedy population-based algorithm
Total completion time

A B S T R A C T

Many uncertainties arise during the manufacturing process, such as changes in the working environment, traffic transportation delays, machine breakdowns, and worker performance instabilities. These factors can cause job processing times and ready times to change. In this study, we address a scheduling model for a single machine where both job release dates and processing times are scenario dependent. The objective is to minimize the total completion time across the worst-case scenarios. Even without the uncertainty factor, this problem is NP-hard. To solve it, we derive several properties and a lower bound used in a branch-and-bound method to find an optimal solution. We propose nine heuristics based on a linear combination of scenario-dependent processing times and release times for approximate solutions. Additionally, we offer an iterated greedy population-based algorithm that efficiently solves this problem by taking advantage of the diversity of solutions. We evaluate the performance of the proposed nine heuristics and the iterated greedy population-based algorithm.

1. Introduction

It is common in scheduling models to assume that job parameters like processing times or release dates are fixed integers. However, in real-life productions, several factors, such as changes in the working environment, traffic transportation delays, machine breakdowns, and worker performance instabilities, can affect job processing times or release dates. In such situations, job processing times or release dates cannot be assumed to be fixed numbers. They are often collected based on past historical statistical data. However, two critical situations can be faced in such cases: one where the data variance is huge or the data primary distribution needs to be corrected. Additionally, the worst-case performance of the system is usually more important than the average performance. Kouvelis and Yu (1996) and Yang and Yu (2002) suggest a robust approach to defend against the worst case in such scenarios.

In literature, it has been shown that the robust version of the single machine scheduling problem to minimize the sum of completion time's criterion is NP-complete, even for highly restricted cases. Yang and Yu (2002) proposed a dynamic programming algorithm and two polynomial-time heuristics to solve this problem. Aloulou and Croce (2008) analyzed computational complexity results for several single-machine scheduling problems with uncertain job characteristics, adopting the absolute robustness criterion. In another study, de Farias et al. (2010) researched various multiple-scenario single-machine scheduling models, where the criterion is the maximum of the total weighted completion time. They provided some inequalities and branch-and-cut techniques to solve their proposed models. Aissi et al. (2011) studied a single-machine problem where the processing times of the jobs are known, but the due dates are still being determined. They adopted the best worst-case performance to minimize the number of late jobs. Mastrolilli et al. (2013) developed a polynomial-time algorithm

* Corresponding author
E-mail jhchen@nqu.edu.tw (J.-H. Chen)
ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)
2025 Growing Science Ltd.
doi: 10.5267/j.ijiec.2024.11.002

based on dynamic programming to solve a multiple-scenario single-machine scheduling problem where the objective is to minimize the weighted sum of completion times. Gilenson et al. (2018) provided a 2-approximation algorithm to solve the bi-scenario sum of completion times problem. They also proved that this algorithm was asymptotically tight.

Gilenson et al. (2019) tackled problems related to scheduling in single and dual-machine flow-shop scenarios. The objective was to maximize the weighted number of jobs completed exactly on their due date. On the other hand, Hermelin et al. (2020) studied problems related to scheduling in single-machine, multi-scenario scenarios. The criteria for these problems included the total weighted completion time, the weighted number of tardy jobs, and the weighted number of jobs completed exactly on their due date. If you want to learn more about scheduling models using robust approaches with random variables, fuzzy numbers, or scenarios, Sotskov and Werner's work (2014) is recommended. Similarly, for complete discrete optimization with various representations of uncertainty and concepts, readers may refer to Kasperski and Zielinski's (2016) book, which includes a chapter reviewing recent results. In semiconductor manufacturing, new and modern machines often work alongside old and less efficient ones that are kept in operation due to their high replacement cost. Dessouky (1998) emphasized the importance of identifying a schedule in which each job cannot be started before its release time and must not be completed after its due date. Several traditional single-machine scheduling studies have been conducted, with varying release dates and a focus on minimizing the total completion time. These studies include works by Chekuri et al. (1997), Hochbaum and Shmoys (1987), Sevastianov and Woeginger (1998), Alon et al. (1998), and Schuurnman and Woeginger (1999). Readers may refer to Chen et al. (1998) for a general overview of approximation techniques.

In this study, we focus on a scheduling problem for a single machine where both the job processing times and release dates depend on the scenario. Our objective is to minimize the total completion time. Even without considering uncertainty, this problem is known to be NP-hard. We propose a branch-and-bound method using several lemmas and a lower bound to find an optimal solution. Additionally, we suggest three types of heuristics considering both processing times and release dates and explore three heuristics to each kind, totaling nine heuristics. Furthermore, we introduce an iterated greedy population-based algorithm that utilizes solution diversity to solve the problem for small and large job instances.

The following is a summary of the remaining sections of this paper. Section 2 describes the problem. Section 3 proposes a lower bound and four properties to be used in a branch-and-bound method. Section 4 presents nine heuristics and an iterated greedy population-based (IGPB) algorithm. In Section 5, we conduct computational experiments to evaluate the performance of the nine heuristics and the IGPB algorithm. Finally, the last section contains the conclusions and suggestions for future studies.

2. Problem formulation

The problem can be stated formally as follows: we have a set J = {J1, J2, …, Jn} of n independent, non-preemptive jobs that need to be processed on a single machine. Each job has its release date and cannot be started before that date. Some uncertainties can significantly affect the production process in many practical production environments. For example, machines can break down, working environments can change, worker performance can be unstable, tool quality can vary, and other complex external factors can be involved. Due to these uncertainties, it is reasonable to consider cases where job processing times and other job-related properties (such as due dates) are random, or where the machine(s) is (are) subject to unexpected breakdowns, or both. To address the uncertainties in the situation, we will adopt the approach proposed by Kouvelis and Yu in 1996. In addition, we will consider the two-scenario natural flexible manufacturing systems discussed in the recent study by Wu et al. (2021). We will assume that there are two distinct scenarios for the job parameters, one for job processing times and another for job release dates. Namely, $p_j^{(s)}$ be the processing time of job J_j under scenario s=1, 2. Moreover, let $r_j^{(s)}$ the release date of job J_j under scenario s=1, 2. The measurement criterion total completion time of the jobs across the worst possible scenarios is considered in this study. Adopting the worst-case performance, the goal of this study is to find an appropriate job sequence σ^* such that σ^*=arg $min_{\sigma\in\Omega}\{max_{s=1,2}\sum_{j=1}^{n} C_j^s(\sigma)\}$, where Ω denotes all possible permutations sequences of jobs in J, $C_j^s(\sigma)$ is the completion time of job J_j in the sequence σ for scenario s. For simplification, we let $RCT(\sigma)=max_{s=1,2}\sum_{j=1}^{n} C_j^s(\sigma)$.

3. Properties and a lower bound

The problem of scheduling jobs on a single machine with different release dates while minimizing the total completion time has been proven to be strongly NP-hard (Lenstra et al. 1977; Yin et al. 2012; Bouamama et al., 2012; Wu et al. 2013). Similarly, the robust version of the same problem, without release dates, is also NP-hard (Yang and Yu, 2002). To solve this challenging problem, we will identify four properties and a lower bound that can be used in a branch-and-bound technique to find the optimal solution. Let $\sigma_1 = (\delta, i, j, \delta')$ and $\sigma_2 = (\delta, j, i, \delta')$ be two permutations which are identical except for the order in which the two adjacent jobs i and j are processed in which δ and δ', respectively, are partial sequences. To show that σ_1 is no worse than σ_2, the following condition suffices: $\max\{C_i^{(1)}(\sigma_1) + C_j^{(1)}(\sigma_1), C_i^{(2)}(\sigma_1) + C_j^{(2)}(\sigma_1)\} < \max\{C_j^{(1)}(\sigma_2) + C_i^{(1)}(\sigma_2), C_j^{(2)}(\sigma_2) + C_i^{(2)}(\sigma_2)\}$.

It is noted that we only prove property 1 here and skip other proofs since they follow the same idea. Let $t^{(s)}$ be the completion time of the last job in δ in the $\sigma_1(\sigma_2)$ with respective to scenario *s*.

Property 1: Consider two adjacent jobs *J_i* and *J_j* with $t^{(s)} < r_i^{(s)} < r_j^{(s)}$ and $r_i^{(s)} + p_i^{(s)} < r_j^{(s)} + p_j^{(s)}$ *s*=1, 2, then there is an optimal sequence in which job J_j follows after job J_i.

Proof: According to the definition of the completion time of a job, one has the following equations. For *s*=1, 2, and σ_1, σ_2 defined above,
$C_i^{(s)}(\sigma_1) = \sum_{j\in\delta} C_j^{(s)}(\sigma_1) + r_i^{(s)} + p_i^{(s)}$,
$C_j^{(s)}(\sigma_1) = \sum_{j\in\delta} C_j^{(s)}(\sigma_1) + max\left\{r_i^{(s)} + p_i^{(s)}, r_j^{(s)}\right\} + p_j^{(s)}$,
$C_j^{(s)}(\sigma_2) = \sum_{j\in\delta} C_j^{(s)}(\sigma_2) + r_j^{(s)} + p_j^{(s)}$,
$C_i^{(s)}(\sigma_2) = \sum_{j\in\delta} C_j^{(s)}(\sigma_2) + max\left\{r_j^{(s)} + p_j^{(s)}, r_i^{(s)}\right\} + p_i^{(s)} = \sum_{j\in\delta} C_j^{(s)}(\sigma_2) + r_j^{(s)} + p_j^{(s)} + p_i^{(s)}$, the last equation is obtained by applying the given condition $r_i^{(s)} < r_j^{(s)}$.

We claim that $[C_i^{(s)}(\sigma_1) + C_j^{(s)}(\sigma_1)] < [C_j^{(s)}(\sigma_2) + C_i^{(s)}(\sigma_1)]$, for s=1, 2, then the result, $max_{s=1,2}\left\{C_i^{(s)}(\sigma_1) + C_j^{(s)}(\sigma_1)\right\} < max_{s=1,2}\left\{C_j^{(s)}(\sigma_2) + C_i^{(s)}(\sigma_2)\right\}$ follows.
$\left[C_j^{(s)}(\sigma_2) + C_i^{(s)}(\sigma_2)\right] - \left[C_i^{(s)}(\sigma_1) + C_j^{(s)}(\sigma_1)\right] = \left[2r_j^{(s)} + 2p_j^{(s)} + p_i^{(s)}\right] - \left[r_i^{(s)} + p_i^{(s)} - max\left\{r_i^{(s)} + p_i^{(s)}, r_j^{(s)}\right\} - p_j^{(s)}\right] =$
$$\begin{cases} \left(r_j^{(s)} - r_i^{(s)}\right) + \left[\left(r_j^{(s)} + p_j^{(s)}\right) - \left(r_i^{(s)} + p_i^{(s)}\right)\right] > 0, & if\ max\left\{r_i^{(s)} + p_i^{(s)}, r_j^{(s)}\right\} = r_i^{(s)} + p_i^{(s)} \\ \left(r_j^{(s)} - r_i^{(s)}\right) + p_j^{(s)} > 0, & if\ max\left\{r_i^{(s)} + p_i^{(s)}, r_j^{(s)}\right\} = r_j^{(s)} \end{cases}$$
This completes the claim. □

Property 2: Consider two adjacent jobs *J_i* and *J_j* with $r_i^{(s)} < t^{(s)} < r_j^{(s)}$ and $p_i^{(s)} < p_j^{(s)}$, *s*=1, 2, then there is an optimal sequence in which job J_j follows after job J_i.

Property 3: Consider two adjacent jobs J_i and J_j with max $\{r_j^{(s)}, t^{(s)}\}$ > max $\{r_i^{(s)}, t^{(s)}\}$, $r_j^{(s)}$ < max $\{r_i^{(s)}, t^{(s)}\} + p_i^{(s)}$, and 2max $\{r_i^{(s)}, t^{(s)}\} + p_i^{(s)}$ < 2max $\{r_j^{(s)}, t^{(s)}\} + p_j^{(s)}$, *s*=1, 2, then there is an optimal sequence in which job J_j follows after job J_i.

Property 4: Consider two adjacent jobs J_i and J_j with $r_j^{(s)}$ > max $\{r_i^{(s)}, t^{(s)}\} + p_i^{(s)}$, *s*=1, 2, then there is an optimal sequence in which job J_j follows after job J_i.

A lower bound is used to determine if a partial node is cut to speed up branch-and-bound. This method has been used by researchers such as Lin and Wu (2006), Smith (1956), French (1982), Yang and Yu (2002), Pinedo (2008), Cheng et al. (2017), Lin et al. (2019), and Wang et al. (2023, 2024). A lower bound for a partial schedule can be derived as follows:

$$\text{LB} = \min\Big\{\sum_{i=1}^{k} C_{[i]}^{(1)} + \min\{n_1 t^{(1)} + \sum_{q=1}^{n_1}(n_1 - q + 1)p_q^{(1)}, \sum_{q=1}^{n_1}\min\{r_{(q)}^{(1)}, r_{(q)}^{(2)}\} + \sum_{q=1}^{n_1}\min\{p_{(q)}^{(1)}, p_{(q)}^{(2)}\}\Big\}, \sum_{i=1}^{k} C_{[i]}^{(2)} + \min\{n_1 t^{(2)} + \sum_{q=1}^{n_1}(n_1 - q + 1)p_q^{(2)}\}, \sum_{q=1}^{n_1}\min\{r_{(q)}^{(1)}, r_{(q)}^{(2)}\} + \sum_{q=1}^{n_1}\min\{p_{(q)}^{(1)}, p_{(q)}^{(2)}\}\},$$

where [ ] denotes the position in a given schedule, $n_1 = n - k$, $t^{(s)} = C_{[k]}^{(s)}$, $r_{(1)}^{(s)}, r_{(2)}^{(s)}, \cdots, r_{(n)}^{(s)}$ and $p_{(1)}^{(s)} \le p_{(2)}^{(s)} \le \cdots \le p_{(n)}^{(s)}$ denote the non-decreasing values of the release dates $r_1^{(s)}, r_2^{(s)}, \cdots, r_n^{(s)}$ and processing times $p_1^{(s)}, p_2^{(s)}, \cdots, p_n^{(s)}$ for s=1, 2, respectively. Furthermore, the term $\sum_{q=1}^{n_1}(n_1 - q + 1)p_{(q)}^{(s)}$ can be minimized for *s*=1, 2 if the sequences $\{(n_1 - q + 1),\ q = 1,2,..,n_1\}$ and $\{p_{(q)}^{(s)}, q = 1,2,..,n_1\}$ are ordered oppositely, by Hardy *et al.* (1967).

4. Nine heuristics and an iterated greedy population-based algorithm

To find near-optimal robust job sequences, we utilize nine mixed heuristics based on a linear combination of scenario-dependent processing times and release dates for different possible scenarios.

The details of HA1~HA3:

Step 0: Input α = 0.25, 0.5, and 0.75.
Step 1: Calculate Mrpt(j) = α · max $\{r_j^{(1)}, r_j^{(2)}\} + (1 - \alpha)$max $\{p_j^{(1)}, p_j^{(2)}\}$, *j*=1, 2, …, *n*.
Step 2: Find the sequence in non-decreasing order of {Mrpt(*j*), *j*=1,2, …, *n*} for each α = 0.25, 0.5, and 0.75, say *S₁*, *S₂*, *S₃*.
Step 3: Improve *S₁*, *S₂*, *S₃* by a pairwise interchange method, say HAa0.25, HAa0.50, and HAa0.75.

The details of HA4~HA6:
Step 0: Input β= 0.25, 0.5, and 0.75.

Step 1: Calculate mrpt(j) = $\beta \cdot \min\{r_j^{(1)}, r_j^{(2)}\} + (1-\beta)\min\{p_j^{(1)}, p_j^{(2)}\}$, j=1, 2, …, n.
Step 2: Find the sequence in non-decreasing order of {mrpt(j), j=1, 2, …, n}, for each β = 0.25, 0.5, and 0.75, say S_4, S_5, S_6.
Step 3: Improve S_4, S_5, S_6 by a pairwise interchange method, say HAb0.25, HAb0.50, and HAb0.75.

The details of HA$_7$~HA$_9$:
Step 0: Input γ= 0.25, 0.5, and 0.75.
Step 1: Calculate averpt(j)=$\gamma \cdot (r_j^{(1)} + r_j^{(2)})/2 + (1-\gamma)(p_j^{(1)} + p_j^{(2)})/2$, j=1,2, …, n.
Step 2: Find the sequence in non-decreasing order of {averpt(j), j=1, 2, …, n}, for each γ= 0.25, 0.5, and 0.75, say S_7, S_8, S_9.
Step 3: Improve S_7, S_8, S_9 by a pairwise interchange method, say HAg0.25, HAg0.50, and HAg0.75.

In this paper, we present the iterated greedy population-based (IGPB) algorithm as an extension of nine heuristics. The algorithm begins by generating a set of randomly scheduled candidate sequences, which we call the population. From this population, we select a candidate sequence and perform several cycles that alternate between destruction and construction stages until a given condition is met. During the destruction stage, we randomly select d jobs from a given job sequence S and divide them into two subsequences: S_d with d jobs and S_r with the remaining n-d jobs. During the construction stage, we take the first-position job in Sd and move it to one of the (n-d+1) positions in S_r, forming (n-d+1) subsequences. We then choose the best subsequence S* with the smallest total completion time of jobs across the worst two possible scenarios among the (n-d+1) subsequences. In the following steps, we modify the current best subsequence S* with Sr, which consists of the remaining (n-d+1) jobs. Then, we update Sd with (d-1) jobs and repeat this process until there are no more jobs left in Sd, following the Nawaz-Enscore-Ham (NEH) method (Nawaz et al. 1983). Similarly, Ruiz and Stützle (2007, 2008) proposed a similar approach where a probability is used to determine whether a new complete job sequence Snew should be accepted or not. The probability is calculated as max{0, exp(RCT(S) - RCT(S_{new}))/TT)}, where RCT(S_{new}) represents the objective function's value (which is the total completion time of the jobs across the worst possible scenarios) for S_{new}, $TT = T \times max_{s=1,2}\{\sum_{j=1}^{n}\left(t_j^{(s)} + r_j^{(s)}\right)/(n \times 2 \times 100)\}$, and $0 < T < 1$ is a temperature control variable. Let psize, iter_no, and d_no denote the number of randomly generated populations, the maximum number of iterations of IGPB algorithm, and the number of jobs in S_d.

Iterated Greedy Population-Based (IGPB) algorithm:

```
Begin:
   Input psize, iter_no, d_no, and TT;
   Generate a population of psize sequences, (say $S_1, S_2, \dots, S_{psize}$);
   Compute $RCT(S_1)$, $RCT(S_2), \dots, RCT(S_{psize})$.
   Keep the best schedule $S^{**}$ with the smallest value among $RCT(S_1)$, $RCT(S_2), \dots, RCT(S_{psize})$;
   For each $S_i$, i=1 to psize
      Set $S = S_i$ and RCT(S)
         For j=1 to iter_no
               Partition $S$ into subsequences $S_r$ and $S_d$;
               For each job in $S_d$, k=1 to d
                     Move each job in $S_d$ to insert in all possible positions in $S_r$ by the NEH method to find a best subsequence;
         End for
               Find a final full best $S^*$ and $RCT(S^*)$;
         Acceptance rule:
         If $RCT(S^*) < RCT(S)$, then
               Replace $S$ by $S^*$;
               If $RCT(S) < RCT(S^{**})$, then
                  Replace $S^{**}$ by $S$;
               End if;
         Else
            Generate a random number r
               If r ≤ exp(RCT(S)- RCT($S^*$))/TT), then
               Replace $S$ by $S^*$;
         End if
      End if
     End for
   End for
   Output $S^{**}$ and $RCT(S^{**})$
End Begin.
```

5. Tuning parameters of the IGPB algorithm

To achieve better solutions or reduce runtime in the IGPB algorithm, one must adjust the values of its parameters. These parameters are the temperature control variable T, the number of repetitions iter_no, the number of a group of population psize, and the number of jobs to be removed d_no. The number of jobs to be removed were set to 10 and 200 for small and large-sized problem instances respectively. It is essential to fine-tune these parameters before conducting intensive computational experiments.

5.1 Tuning parameters for small-size jobs problem

To optimize the parameters for small-sized jobs (n=10), the processing times, denoted as $p_j^{(1)}$ and $p_j^{(2)}$, were randomly generated from uniform distributions of integers over (50, 100) and (150, 200), respectively. Similarly, the release dates $r_j^{(1)}$ and $r_j^{(2)}$ were generated from uniform distributions of integers over (0, (100-50)*n*0.25), and (0, (200-150)*n*0.25), respectively. A total of 100 problem instances were considered, and the error percentage (EP) was recorded. EP represents the error percentage of the objective function (total completion time of jobs across worst-case scenarios) relative to an optimal value obtained from the branch-and-bound method.

For tuning the population size (*psize*), the number of iterations (*iter_no*) was fixed at 90, with parameters *d_no* at 1, and T at 0.1. The test range for *psize* was from 2 to 10, with increments of 1. The maximum error percentage (max_EP) was depicted in Fig. 1 (row 1, column 1). Observing Fig. 1, it was evident that max_EP decreased with increasing *psize*, reaching a minimum at *psize* = 5. Consequently, *psize* was selected as 5.

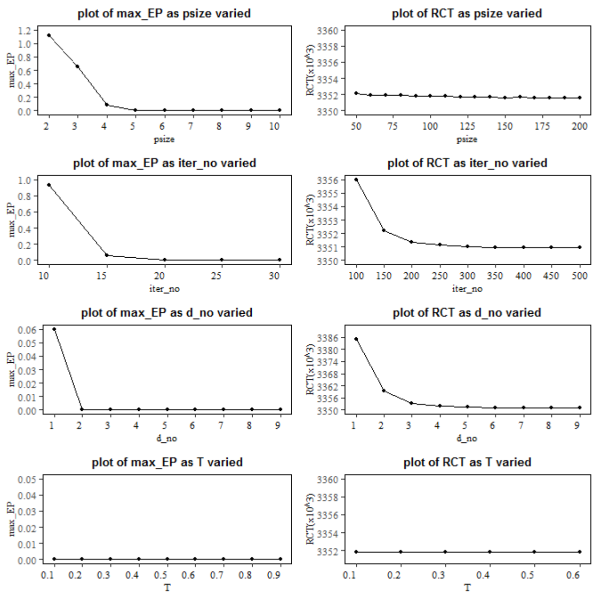

Fig. 1 exploring parameters of IGPB for small *n* and large *n*

Subsequently, for tuning the *iter_no*, *psize* was fixed at 5, with *d_no* at 1, and T at 0.1. The test range for *iter_no* was from 5 to 30, with increments of 5. The max_EP was illustrated in Fig. 1 (row 2, column 1). Fig. 1 revealed a significant decrease in max_EP as *iter_no* increased, eventually stabilizing. The optimal *iter_no* was determined to be 15.

To fine-tune the number of jobs to be removed (*d_no*), psize was fixed at 5, *iter_no* at 15, and *T* at 0.1. The test range for *d_no* was from 1 to 9, with increments of 1. The max_EP was displayed in Fig. 1 (row 3, column 1). Notably, max_EP reached 0 when *d_no* was greater than or equal to 2, indicating an appropriate fit of *d_no* at 2. Finally, for calibrating the temperature control factor (*T*), psize, *d_no*, and *iter_no* were fixed at 5, 2, and 15, respectively. The test range for *T* was from 0.1 to 0.9, with increments of 0.1. The max_EP was visualized in Fig. 1 (row 4, column 1). Fig. 1 demonstrated the relative stability of max_EP for all tested *T* values, and *T* was chosen to be 0.1. Based on the optimization results, the selected parameters for small-sized jobs were (*psize*, *iter_no*, *d_no*, *T*) = (5, 15, 2, 0.1).

5.2 Tuning parameters for large-size jobs problem

To optimize parameters in extensive job settings, we configured the number of jobs, denoted as 'n,' to be 200. The processing times and ready times were generated following the approach used for smaller jobs. We established 100 problem instances. Due to the absence of an optimal value, the objective function values (representing the total completion time of jobs under the worst-case scenarios) were computed for each generated instance. The subsequent RCT illustrates the average objective function values derived from testing various parameter combinations. In order to fine-tune the parameter '*psize*,' we set '*iter_no*' to 200, '*d_no*' to 4, and '*T*' to 0.1. The test range for 'psize' spanned from 100 to 200, with increments of 10. The RCT is presented in Figure 1 (row 1, column 2). Analysis of Fig. 1 indicates a slight oscillation in the RCT throughout the test range. The oscillation range, or error, of the RCT is pre-determined to remain within 3% of the lowest objective function value, observed at '*psize*=150.' Consequently, '*psize*=100' emerges as the appropriate setting. To optimize the 'iter_no' parameter, the repetition count, we maintained a fixed setting with 'psize' at 100, 'd_no' at 5, and 'T' at 0.1. The test span for 'iter_no' ranged from 100 to 500, incremented by 50 each time. Fig. 1 (row 2, column 2) displays the RCT. Examining Figure 1 reveals that, with an increase in 'iter_no,' the RCT values stabilize, particularly beyond the threshold of 200. Given the criterion of controlling errors within a predetermined 3%, 'iter_no' was ultimately selected at 200. To fine-tune the 'd_no' parameter, representing the number of jobs to be removed, we kept 'psize' fixed at 100, 'iter_no' at 200, and 'T' at 0.1. The test range for 'd_no' varied from 1 to 9, with increments of 1. The RCT in Figure 1 (row 3, column 2) illustrates that as 'd_no' increases, RCT values decrease. However, RCT stabilizes for 'd_no' values greater than 4. Consequently, 'd_no' was determined to be appropriately set at 4. Subsequently, for testing the 'T' parameter, with 'psize' fixed at 100, 'iter_no' at 200, and 'd_no' at 4, the value of 'T' was incremented by 0.1 units within the range of 0.1 to 0.6. Figure 1 (row 4, column 2) presents the RCT, indicating that errors in RCT values remained within 3% for 'T' from 0.1 to 0.6. Therefore, 'T' was set at 0.1. The algorithm demonstrated indifference to the value of 'T' due to the use of "one factor at a time" experiments for parameter calibration, where 'T' was tested last, irrespective of small or large job sizes. Ultimately, based on the tuning results, the adjusted parameter values (psize, iter_no, d_no, T) for addressing large-size job problems were determined as (100, 200, 4, 0.1).

6. Computational experiments and result analysis

Johnson (2001) highlighted three prevailing approaches—worst-case analysis, average-case analysis, and experimental analysis—that are commonly employed to evaluate and differentiate algorithms. In favor of the experimental analysis approach, we carried out numerous computational experiments to scrutinize the computational efficiency of the proposed heuristics and the Iterated Greedy Population-Based (IGPB) algorithm. For a comprehensive exploration of the theoretical analysis of algorithms, readers are encouraged to consult Johnson (2001) for more detailed insights.

Table 1
Types of uniform distributions for test instances

Type	p_j^1	p_j^2	r_j^1	r_j^2	Name of type
			U(0~(100-50)×n×0.25)	U(0~(200-150)×n×0.25)	Type1-1
	U(50,100)	U(150,200)	U(0~(100-50)×n×0.25)	U(0~(200-150)×n×0.50)	Type1-2
			U(0~(100-50)×n×0.25)	U(0~(200-150)×n×0.75)	Type1-3
			U(0~(110-60)×n×0.25)	U(0~(210-160)×n×0.25)	Type1-4
1	U(60,110)	U(160,210)	U(0~(110-60)×n×0.25)	U(0~(210-160)×n×0.50)	Type1-5
			U(0~(110-60)×n×0.25)	U(0~(210-160)×n×0.75)	Type1-6
			U(0~(120-70)×n×0.25)	U(0~(220-170)×n×0.25)	Type1-7
	U(70,120)	U(170,220)	U(0~(120-70)×n×0.25)	U(0~(220-170)×n×0.50)	Type1-8
			U(0~(120-70)×n×0.25)	U(0~(220-170)×n×0.75)	Type1-9
			U(0~(100-50)×n×0.50)	U(0~(150-100)×n×0.25)	Type2-1
	U(50,100)	U(100,150)	U(0~(100-50)×n×0.50)	U(0~(150-100)×n×0.50)	Type2-2
			U(0~(100-50)×n×0.50)	U(0~(150-100)×n×0.75)	Type2-3
			U(0~(90-60)×n×0.50)	U(0~(140-110)×n×0.25)	Type2-4
2	U(60,90)	U(110,140)	U(0~(90-60)×n×0.50)	U(0~(140-110)×n×0.50)	Type2-5
			U(0~(90-60)×n×0.50)	U(0~(140-110)×n×0.75)	Type2-6
			U(0~(80-70)×n×0.50)	U(0~(130-120)×n×0.25)	Type2-7
	U(70,80)	U(120,130)	U(0~(80-70)×n×0.50)	U(0~(130-120)×n×0.50)	Type2-8
			U(0~(80-70)×n×0.50)	U(0~(130-120)×n×0.75)	Type2-9
			U(0~(100-50)×n×0.75)	U(0~(160-110)×n×0.25)	Type3-1
	U(50,100)	U(110,160)	U(0~(100-50)×n×0.75)	U(0~(160-110)×n×0.50)	Type3-2
			U(0~(100-50)×n×0.75)	U(0~(160-110)×n×0.75)	Type3-3
			U(0~(90-60)×n×0.75)	U(0~(150-120)×n×0.25)	Type3-4
3	U(60,90)	U(120,150)	U(0~(90-60)×n×0.75)	U(0~(150-120)×n×0.50)	Type3-5
			U(0~(90-60)×n×0.75)	U(0~(150-120)×n×0.75)	Type3-6
			U(0~(80-70)×n×0.75)	U(0~(140-130)×n×0.25)	Type3-7
	U(70,80)	U(130,140)	U(0~(80-70)×n×0.75)	U(0~(140-130)×n×0.50)	Type3-8
			U(0~(80-70)×n×0.75)	U(0~(140-130)×n×0.75)	Type3-9

[

Table 1 delineates the three categories of test data created for processing times and release dates. To accentuate the dual scenarios, the processing times, denoted as $p_j^{(1)}$ and $p_j^{(2)}$, were randomly generated from integers within the intervals Unif(a, b) and Unif(c, d), respectively, following the approach outlined by Kouvelis et al. (2000). Simultaneously, the release dates $r_j^{(1)}$ and $r_j^{(2)}$ were generated from integers within the ranges Unif(0, (b-a)*n*θ_1), and Unif(0, (c-d)*n*θ_2), respectively, based on the methodology proposed by Reever (1995), where θ_1, θ_2 take values of 0.25, 0.5, and 0.75. For every combination of ($p_j^{(1)}$, $p_j^{(2)}$), there were three corresponding sets of ($r_j^{(1)}$, $r_j^{(2)}$). This resulted in a total of 27 possible combinations for ($p_j^{(1)}$, $p_j^{(2)}$) and ($r_j^{(1)}$, $r_j^{(2)}$). Each combination was used to generate 100 distinct problem instances, forming a comprehensive test bank. Furthermore, if the number of explored nodes surpasses 10^8, the branch-and-bound method will be prematurely terminated, advancing to the next set of instances. The performance evaluation of the branch-and-bound method, nine local heuristics, and the Iterated Greedy Population-Based (IGPB) algorithm involved experiments conducted for job sizes, with n set at 8 and 10 for a smaller number of jobs, and n at 100 and 200 for a larger number of jobs. A total of 5400 problem instances were generated for each job size category. Implementation-wise, the nine heuristics and the IGPB algorithm were coded in Fortran (Compaq Visual) and executed on a system equipped with a 3.60GHz Intel(R) Core™ i7-4790 processor and 16GB RAM, running on Windows 7 (64 bits. We present the outcomes derived from meticulously designed computational experiments aimed at assessing the efficacy of the branch-and-bound method, nine local heuristics, and the Iterated Greedy Population-Based (IGPB) algorithm. Tables 2-6 and Fig. 2, along with Tables 4-7 and Fig. 3, succinctly encapsulate the experimental findings for small and large job sizes, respectively. Let $O_i's$ denote the optimal values attained by executing the branch-and-bound method on the test instances designed for small-sized jobs. In gauging the performances of the nine heuristics and the IGPB, we employed the Average Error Percentage (AEP). The AEP is defined as the mean of 100 times the expression$100[(H_i - O_i)/O_i]\%$, for each heuristic or the IGPB algorithm, where H_i is the value obtained by each method.

Table 2
The performance of the branch-and-bound method

		node		CPU_time	
N	Name of type	mean	max	mean	max
8	Type1-1	28961.49	28964	0.09	0.09
	Type1-2	28961.55	28968	0.09	0.11
	Type1-3	28795.27	28964	0.09	0.14
	Type1-4	28517.33	28962	0.09	0.11
	Type1-5	28961.9	28968	0.09	0.11
	Type1-6	28962.06	28972	0.09	0.11
	Type1-7	28914.54	28972	0.09	0.11
	Type1-8	28965.25	28984	0.09	0.11
	Type1-9	28965.83	29008	0.09	0.11
	Type2-1	28965.19	28996	0.09	0.11
	Type2-2	28961.37	28964	0.09	0.11
	Type2-3	28956.27	28966	0.09	0.16
	Type2-4	28771.36	28964	0.09	0.11
	Type2-5	28425.46	28962	0.09	0.11
	Type2-6	28961.74	28966	0.09	0.11
	Type2-7	28962.25	28972	0.09	0.11
	Type2-8	28909.62	29008	0.09	0.11
	Type2-9	28963.99	28984	0.09	0.11
	Type3-1	28965.24	29080	0.09	0.11
	Type3-2	28965.21	28984	0.09	0.11
	Type3-3	28868.46	28966	0.09	0.11
	Type3-4	28961.43	28966	0.1	0.11
	Type3-5	28960.72	28966	0.09	0.11
	Type3-6	28921.84	28966	0.09	0.11
	Type3-7	28961.52	28966	0.09	0.11
	Type3-8	28961.44	28966	0.09	0.11
	Type3-9	28952.71	28966	0.09	0.09
10	Type1-1	2606502	2606506	16.39	18.99
	Type1-2	2605416	2606508	15	15.65
	Type1-3	2566366	2606503	14.79	15.32
	Type1-4	2549317	2606506	14.72	15.35
	Type1-5	2606503	2606512	14.97	15.57
	Type1-6	2604837	2606512	15.01	16.65
	Type1-7	2589525	2606512	15.18	16.86
	Type1-8	2606515	2606740	15.21	16.96
	Type1-9	2606516	2606788	15.09	16.77
	Type2-1	2606513	2606644	15.13	16.65
	Type2-2	2606284	2606508	15.07	17.1
	Type2-3	2598972	2606512	16.36	19.13
	Type2-4	2572055	2606504	14.75	16.69
	Type2-5	2555212	2606502	14.7	16.38
	Type2-6	2606503	2606548	14.88	17.24
	Type2-7	2603245	2606524	14.77	17.64
	Type2-8	2584276	2606512	12.34	14.35
	Type2-9	2606513	2606596	12.45	14.1
	Type3-1	2606516	2606680	12.43	14.27
	Type3-2	2606514	2606740	12.41	12.75
	Type3-3	2592673	2606504	16.32	18.94
	Type3-4	2606502	2606524	16.36	18.77
	Type3-5	2603895	2606512	15.83	19.11
	Type3-6	2595714	2606512	14.96	15.41
	Type3-7	2606502	2606506	15.02	15.44
	Type3-8	2604800	2606508	15.03	15.4
	Type3-9	2600095	2606506	14.96	15.34

Table 2 illustrates the effectiveness of the branch-and-bound method, showcasing its capability to successfully solve all test instances within the constraint of 10^8 nodes. Notably, the computational CPU times, encompassing both average and maximum execution times (in seconds), exhibited a significant escalation as the job count, denoted by 'n,' increased (refer to columns 5 and 6 in Table 2). Correspondingly, with the growth of 'n,' there was a noticeable increase in both the mean and maximum nodes (columns 3 and 4 in Table 2). Concerning the performance evaluation of the proposed nine heuristics and the Iterated Greedy Population-Based (IGPB) algorithm for small-sized jobs, their Average Error Percentages (AEPs) are presented in Table 3 and depicted in Fig. 2. The AEPs for the HAb0.25, HAb0.50, and HAb0.75 heuristics demonstrated an increase as 'n' advanced from 8 to 10. In contrast, AEPs remained relatively consistent for the groups of heuristics HAb*, Hag*, and the IGPB for both 'n' values of 8 and 10. The mean AEPs for HAa* (HAa0.25, HAa0.50, HAa0.75) were (0.1022, 0.1157, 0.1243), AEPs for HAb* (HAb0.25, HAb0.50, HAb0.75) were (0.2959, 0.2791, 0.2876), AEPs for HAg* (HAg0.25, HAg0.50, HAg0.75) were (0.0552, 0.0583, 0.0593), and AEP for IGPB was 0.0000, all pertaining to small-sized jobs. Figure 2 visually presents boxplots of AEP for the nine heuristics and the IGPB. Notably, as the CPU times were all under 0.1 second, they are omitted from discussion here.

Table 3
The AEP for nine heuristics and IGPB algorithm

n	Type	HAa0.25		HAa0.50		HAa0.75		HAb0.25		HAb0.50		HAb0.75		HAg0.25		HAg0.50		HAg0.75		IGPB	
		AEP		AEP		AEP		AEP		AEP		AEP		AEP		AEP		AEP		AEP	
		mean	max	mean	max	mean	max	mean	max	mean	max	mean	max	mean	max	mean	max	mean	max	mean	max
8	1-1	0.03	1.64	0.03	0.98	0.07	2.03	0.08	1.64	0.06	2.66	0.08	2.66	0.06	2.19	0.04	1.78	0.04	1.64	0.00	0.00
	1-2	0.02	0.94	0.04	1.27	0.05	1.27	0.05	1.27	0.04	1.74	0.05	1.74	0.05	1.27	0.04	1.27	0.04	1.27	0.00	0.00
	1-3	0.00	0.31	0.03	1.31	0.05	1.70	0.57	13.73	0.37	8.76	0.37	8.76	0.05	4.84	0.02	1.31	0.02	1.31	0.00	0.00
	1-4	0.02	1.43	0.02	1.43	0.08	4.17	0.10	2.71	0.08	2.16	0.07	1.79	0.08	3.20	0.03	1.79	0.03	1.79	0.00	0.00
	1-5	0.02	1.20	0.03	1.20	0.04	1.20	0.06	3.83	0.01	0.56	0.00	0.06	0.02	1.20	0.03	1.20	0.03	1.20	0.00	0.00
	1-6	0.01	0.30	0.01	0.30	0.01	0.30	0.48	7.74	0.43	7.08	0.46	7.08	0.07	7.02	0.03	0.94	0.02	0.83	0.00	0.00
	1-7	0.10	3.70	0.08	3.70	0.10	3.70	0.04	0.92	0.06	1.52	0.07	1.52	0.04	1.80	0.04	0.83	0.08	1.52	0.00	0.00
	1-8	0.03	1.73	0.04	1.73	0.04	1.73	0.08	1.32	0.07	1.6	0.07	1.60	0.04	1.73	0.06	1.73	0.06	1.73	0.00	0.00
	1-9	0.01	0.61	0.01	0.61	0.04	1.35	0.72	9.18	0.34	7.46	0.49	9.82	0.01	0.61	0.02	1.25	0.02	1.25	0.00	0.00
	2-1	0.16	4.53	0.19	3.09	0.2	3.09	0.11	2.84	0.12	2.84	0.11	3.36	0.12	1.92	0.07	1.92	0.07	2.36	0.00	0.00
	2-2	0.20	4.72	0.19	4.72	0.33	5.73	0.29	4.00	0.41	7.02	0.51	7.02	0.10	2.25	0.15	3.18	0.14	2.54	0.00	0.00
	2-3	0.18	7.38	0.06	2.21	0.06	2.21	1.18	18.19	0.72	18.19	0.88	18.19	0.10	7.38	0.03	1.73	0.05	1.73	0.00	0.00
	2-4	0.12	3.82	0.21	3.82	0.22	3.82	0.03	0.94	0.03	0.94	0.07	1.71	0.02	1.27	0.06	3.37	0.03	1.33	0.00	0.00
	2-5	0.17	5.40	0.12	5.40	0.14	4.82	0.06	2.83	0.03	1.13	0.03	1.13	0.05	2.83	0.05	2.83	0.03	1.14	0.00	0.00
	2-6	0.12	7.68	0.03	0.86	0.03	0.86	0.13	3.51	0.10	3.37	0.09	4.21	0.03	2.16	0.02	0.86	0.02	0.86	0.00	0.00
	2-7	0.03	1.53	0.08	1.66	0.09	1.66	0.01	0.47	0.02	0.79	0.02	0.79	0.01	0.58	0.01	0.25	0.01	0.25	0.00	0.00
	2-8	0.05	1.87	0.04	1.51	0.06	1.51	0.00	0.11	0.00	0.13	0.01	0.47	0.04	1.87	0.01	0.47	0.01	0.47	0.00	0.00
	2-9	0.02	0.88	0.01	0.88	0.01	0.88	0.04	2.04	0.01	0.42	0.01	0.42	0.01	0.88	0.01	0.42	0.01	0.42	0.00	0.00
	3-1	0.18	6.21	0.26	6.21	0.21	6.21	0.19	3.12	0.16	3.12	0.14	2.13	0.05	1.10	0.11	2.13	0.12	2.13	0.00	0.00
	3-2	0.35	11.97	0.53	20.2	0.53	20.2	0.27	7.97	0.30	7.97	0.33	7.97	0.16	4.31	0.15	4.31	0.24	7.97	0.00	0.00
	3-3	0.55	15.53	0.45	15.53	0.37	11.26	0.76	9.86	0.60	9.86	0.50	9.86	0.26	6.44	0.16	6.44	0.10	3.46	0.00	0.00
	3-4	0.13	2.37	0.13	2.37	0.13	2.37	0.05	1.69	0.05	1.69	0.07	1.69	0.04	1.10	0.05	1.29	0.04	1.29	0.00	0.00
	3-5	0.15	2.41	0.17	3.22	0.16	3.22	0.03	0.91	0.03	0.91	0.04	0.99	0.03	0.99	0.03	0.99	0.02	0.74	0.00	0.00
	3-6	0.15	4.60	0.24	6.59	0.12	4.60	0.07	1.79	0.06	1.79	0.03	1.01	0.02	0.89	0.04	1.27	0.03	1.27	0.00	0.00
	3-7	0.04	0.99	0.06	1.63	0.07	1.63	0.03	1.17	0.02	0.50	0.02	0.52	0.02	0.76	0.04	1.46	0.04	1.46	0.00	0.00
	3-8	0.03	1.02	0.05	1.02	0.06	1.02	0.03	0.73	0.02	0.35	0.04	0.9	0.03	1.02	0.04	1.02	0.03	0.80	0.00	0.00
	3-9	0.05	4.57	0.07	4.57	0.07	4.15	0.00	0.24	0.01	0.35	0.01	0.35	0.01	0.35	0.00	0.35	0.00	0.35	0.00	0.00
mean		0.11		0.12		0.12		0.20		0.15		0.17		0.06		0.05		0.05		0.00	
10	1-1	0.02	1.83	0.05	1.88	0.09	1.88	0.03	0.91	0.08	1.67	0.13	1.67	0.06	1.83	0.05	1.16	0.09	1.67	0.00	0.00
	1-2	0.02	0.71	0.06	2.62	0.06	2.62	0.29	3.92	0.26	4.62	0.26	4.62	0.03	1.76	0.06	2.62	0.07	2.62	0.00	0.00
	1-3	0.04	0.93	0.04	0.93	0.04	0.93	1.18	14.9	1.31	13.57	1.27	13.57	0.03	0.93	0.04	0.93	0.04	0.93	0.00	0.00
	1-4	0.03	1.06	0.05	0.96	0.08	1.49	0.04	0.98	0.05	0.96	0.06	0.96	0.03	1.06	0.04	0.96	0.04	0.96	0.00	0.00
	1-5	0.02	0.78	0.02	0.78	0.05	1.35	0.19	5.69	0.21	5.69	0.12	2.88	0.03	1.35	0.04	1.35	0.05	1.35	0.00	0.00
	1-6	0.01	0.54	0.02	0.84	0.02	0.84	0.75	7.36	0.9	9.09	0.8	10.85	0.01	0.54	0.03	1.13	0.02	0.84	0.00	0.00
	1-7	0.04	1.05	0.05	1.27	0.06	1.07	0.03	1.05	0.04	1.27	0.04	0.87	0.02	0.68	0.04	1.27	0.06	1.55	0.00	0.00
	1-8	0.03	1.36	0.02	0.98	0.04	0.98	0.21	3.61	0.16	3.61	0.16	3.61	0.01	0.23	0.03	0.98	0.04	0.98	0.00	0.00
	1-9	0.03	1.07	0.05	1.13	0.04	1.07	0.79	11.3	0.7	7.24	0.7	9.38	0.03	1.07	0.04	1.13	0.05	1.13	0.00	0.00
	2-1	0.17	4.59	0.18	4.59	0.21	4.59	0.07	1.06	0.1	1.12	0.13	1.96	0.08	1.65	0.1	1.72	0.1	1.31	0.00	0.00
	2-2	0.35	7.99	0.26	7.99	0.28	12.92	0.84	13.22	1.19	15.1	1.25	15.1	0.24	7.99	0.13	3.36	0.18	5.89	0.00	0.00
	2-3	0.09	4.2	0.13	4.2	0.12	4.2	2.36	18.8	2.13	22.8	2.38	22.8	0.27	6.58	0.06	1.56	0.07	1.56	0.00	0.00
	2-4	0.05	1.2	0.17	3.63	0.22	3.86	0.03	0.84	0.06	1.3	0.09	1.92	0.05	2.26	0.04	0.97	0.1	3.06	0.00	0.00
	2-5	0.13	4.21	0.12	2.04	0.1	2.04	0.16	4.09	0.18	4.09	0.13	3.25	0.04	1.73	0.04	0.75	0.04	0.75	0.00	0.00
	2-6	0.03	1.16	0.03	1.16	0.02	1.16	0.49	7.01	0.54	8.92	0.58	8.92	0.02	1.16	0.02	1.16	0.02	1.16	0.00	0.00
	2-7	0.02	0.47	0.05	1.22	0.06	1.22	0.01	0.31	0.02	0.41	0.04	0.74	0.01	0.76	0.02	0.89	0.03	0.89	0.00	0.00
	2-8	0.01	0.13	0.01	0.44	0.02	0.56	0.01	0.34	0.01	0.34	0.01	0.34	0	0.12	0.01	0.12	0.01	0.12	0.00	0.00
	2-9	0	0.23	0	0.23	0	0.23	0	0.23	0	0.23	0	0.23	0	0.23	0	0.23	0	0.23	0.00	0.00
	3-1	0.17	3.34	0.24	4.71	0.23	4.83	0.23	2.62	0.19	2.41	0.12	2.01	0.06	1.3	0.08	1.37	0.12	2.13	0.00	0.00
	3-2	0.49	7.17	0.43	6.82	0.36	6.82	0.8	11.47	0.79	11.47	0.85	11.47	0.2	7.17	0.3	7.17	0.28	4.85	0.00	0.00
	3-3	0.38	8.62	0.37	8.62	0.4	8.62	1.75	19.84	1.67	19.84	1.42	19.84	0.29	7.38	0.33	14.51	0.33	14.51	0.00	0.00
	3-4	0.15	2.42	0.23	3.63	0.23	3.63	0.03	0.78	0.05	1	0.04	1	0.04	2.1	0.04	1.65	0.04	1.65	0.00	0.00
	3-5	0.16	3.92	0.2	4.57	0.22	4.57	0.01	0.43	0.04	1.26	0.05	1.26	0.03	0.64	0.03	0.64	0.02	0.64	0.00	0.00
	3-6	0.07	3.1	0.19	10.12	0.26	10.41	0.2	4.86	0.19	4.86	0.26	6.73	0.01	0.34	0.03	1.84	0.03	1.84	0.00	0.00
	3-7	0.04	0.83	0.05	0.92	0.05	0.92	0.01	0.24	0.03	0.41	0.03	0.41	0.02	0.47	0.02	0.33	0.02	0.33	0.00	0.00
	3-8	0.02	0.57	0.03	1.24	0.04	1.24	0.01	0.19	0.01	0.37	0.03	0.45	0.01	0.35	0.01	0.6	0.01	0.6	0.00	0.00
	3-9	0.03	1.04	0.02	0.67	0.07	3.54	0	0.04	0.01	0.41	0.01	0.41	0.01	0.41	0.01	0.41	0.01	0.41	0.00	0.00
mean		0.09		0.11		0.12		0.39		0.40		0.41		0.06		0.06		0.07		0.00	
Total mean		0.10		0.12		0.12		0.30		0.28		0.29		0.06		0.06		0.06		0.00	

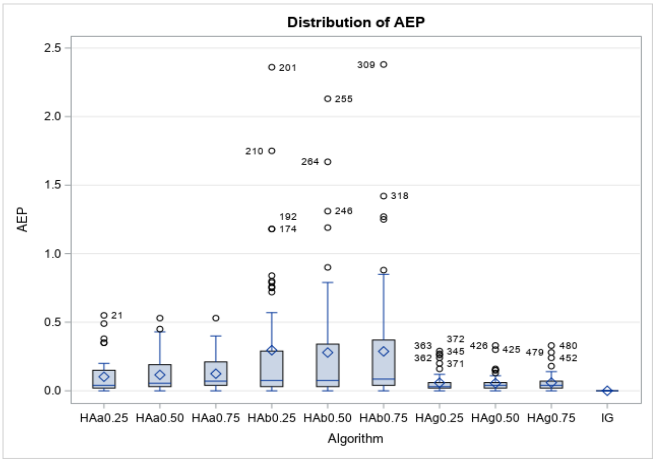

Fig. 2 boxplot for Distribution of AEP

To assess the statistical significance of the performances exhibited by the nine heuristics and the IGPB algorithm, we conducted an analysis of variance (ANOVA) on the Average Error Percentage (AEP). As indicated in Table 4 (columns 2 and 3), the p-values resulting from four commonly used normality tests were all below 0.01, which is smaller than the conventional significance level α=0.05. This suggests that the normality assumption does not hold for the observed AEP data.

Table 4
Normality Tests for small *n* and large *n*

	Small_n		Large_n	
Method of Normality Test	Statistic	P value	Statistic	P value
Shapiro-Wilk normality test	0.7597	<0.0001	0.9282	<0.0001
Kolmogorov-Smirnov test	0.1672	<0.0100	0.0958	<0.0100
Cramer-von Mises normality test	3.0924	<0.0050	1.0893	<0.0050
Anderson-Darling normality test	20.6541	<0.0050	7.2662	<0.0050

Hence, a non-parametric statistical approach was employed to scrutinize the distinctions among the nine heuristics and the IGPB. Utilizing the ranks of observed AEPs, the Kruskal–Wallis test was deployed to assess the null hypothesis positing that the populations of AEPs originated from the same population. As evident in column 2 of Table 5, the results confirm significant differences between the proposed nine heuristics and the IGPB, with a p-value less than 0.001 (below the threshold of α=0.05).

Table 5
Kruskal-Wallis Test

	Kruskal-Wallis Test	
	Small *n*	Large *n*
Chi-square	171.8537	237.4639
DF	9	9
Pr>Chi-square	<.0001	<.0001

In conducting pairwise comparisons among all proposed nine heuristics and the IGPB, we utilized the Dwass–Steel–Critchlow–Fligner (DSCF) procedure by executing PROC NPAR1WAY on SAS 9.4. For a more comprehensive understanding of the procedure, interested readers can consult Holland et al., 2014 or the SAS manual. Table 6 validates that the mean ranks of AEP can be categorized into distinct performance groups at a significance level of α=0.05. As evident in columns 3 and 4 of Table 6, the IGPB (with an AEP of 0.0000) demonstrated the best performance, while the six heuristics, namely HAa* and HAb*, were assigned to the worst performance group.

Table 6
The DSCF pairwise comparison procedure

	Small *n*		Large *n*	
Algorithm	Statistic	P-value	Statistic	P-value
HAa0.25 vs. HAa0.50	1.7612	0.965	0.4007	1
HAa0.25 vs. HAa0.75	2.9823	0.5211	0.3963	1
HAa0.25 vs. HAb0.25	2.497	0.757	9.1269	<.0001
HAa0.25 vs. HAb0.50	2.5524	0.7324	9.0921	<.0001
HAa0.25 vs. HAb0.75	2.8479	0.5892	9.0704	<.0001
HAa0.25 vs. HAg0.25	2.5039	0.754	1.4142	0.9923
HAa0.25 vs. HAg0.50	2.0118	0.9205	1.1617	0.9983
HAa0.25 vs. HAg0.75	2.0582	0.9094	1.0274	0.9994

Table 6
The DSCF pairwise comparison procedure (Continued)

	Small *n*		Large *n*	
Algorithm	Statistic	P-value	Statistic	P-value
HAa0.25 vs. IGPB	13.1611	<.0001	9.3654	<.0001
HAa0.50 vs. HAa0.75	1.5976	0.9818	0.0131	1
HAa0.50 vs. HAb0.25	1.3454	0.9947	9.2226	<.0001
HAa0.50 vs. HAb0.50	1.3842	0.9935	9.2138	<.0001
HAa0.50 vs. HAb0.75	1.9192	0.94	9.1878	<.0001
HAa0.50 vs. HAg0.25	4.1499	0.0962	1.7144	0.9706
HAa0.50 vs. HAg0.50	4.0616	0.1134	1.4271	0.9918
HAa0.50 vs. HAg0.75	3.7602	0.1907	1.2669	0.9966
HAa0.50 vs. IGPB	13.3525	<.0001	9.2587	<.0001
HAa0.75 vs. HAb0.25	0.1392	1	9.2269	<.0001
HAa0.75 vs. HAb0.50	0.2306	1	9.2313	<.0001
HAa0.75 vs. HAb0.75	0.8529	0.9999	9.1923	<.0001
HAa0.75 vs. HAg0.25	5.9836	0.001	1.7013	0.9721
HAa0.75 vs. HAg0.50	6.1931	0.0005	1.4315	0.9916
HAa0.75 vs. HAg0.75	5.5758	0.0032	1.2712	0.9965
HAa0.75 vs. IGPB	13.3518	<.0001	9.2587	<.0001
HAb0.25 vs. HAb0.50	0.0348	1	0.0087	1
HAb0.25 vs. HAb0.75	0.5571	1	0.0217	1
HAb0.25 vs. HAg0.25	4.3156	0.0695	8.2216	<.0001
HAb0.25 vs. HAg0.50	4.0781	0.11	8.3344	<.0001
HAb0.25 vs. HAg0.75	3.9416	0.1407	8.3875	<.0001
HAb0.25 vs. IGPB	12.7651	<.0001	12.7767	<.0001
HAb0.50 vs. HAb0.75	0.4524	1	0.0261	1
HAb0.50 vs. HAg0.25	4.6401	0.0348	8.2389	<.0001
HAb0.50 vs. HAg0.50	4.4811	0.0493	8.3518	<.0001
HAb0.50 vs. HAg0.75	4.2193	0.0841	8.3701	<.0001
HAb0.50 vs. IGPB	13.1538	<.0001	12.733	<.0001
HAb0.75 vs. HAg0.25	5.3576	0.0059	8.2433	<.0001
HAb0.75 vs. HAg0.50	5.2115	0.0087	8.3389	<.0001
HAb0.75 vs. HAg0.75	4.9644	0.0162	8.3919	<.0001
HAb0.75 vs. IGPB	13.1539	<.0001	12.7066	<.0001
HAg0.25 vs. HAg0.50	0.666	1	0.3653	1
HAg0.25 vs. HAg0.75	0.7612	0.9999	0.5003	1
HAg0.25 vs. IGPB	13.1657	<.0001	10.0674	<.0001
HAg0.50 vs. HAg0.75	0.0833	1	0.2175	1
HAg0.50 vs. IGPB	13.1751	<.0001	10.0583	<.0001
HAg0.75 vs. IGPB	13.1646	<.0001	9.7041	<.0001

For the assessments conducted on large-sized scenarios with job counts set at n = 100 and 200, a series of 100 random problem instances were generated for each combination of$(p_j^{(1)}, p_j^{(2)})$ and $(r_j^{(1)}, r_j^{(2)})$. In total, 5400 instances were generated. Due to the unavailability of exact objective function values for optimal solutions in the large-size job instances, the mean Relative Percentage Deviation (RPD) was reported. The RPD is calculated as $\text{RPD} = 100[(H_i - Best_i)/Best_i]\%)$, where H_i represents the objective function value obtained by each heuristic or the IGPB algorithm, and $Best_i$ is the smallest value among those found by the nine heuristics and the IGPB. Table 7 presents the average RPD for the nine heuristics and the IGPB. Notably, the IGPB consistently yielded the lowest RPD, regardless of the job count (n). Fig. 3 visually portrays boxplots of RPD for the nine heuristics and the IGPB. The mean RPD values for (HAa0.25, HAa0.50, HAa0.75) were (0.7348, 0.7113, 0.7126), for (HAb0.25, HAb0.50, HAb0.75) were (5.1361, 5.1359, 5.1357), for (HAg0.25, HAg0.50, HAg0.75) were (1.0530, 1.0228, 1.0156), and the RPD for the IGPB was 0.0133.

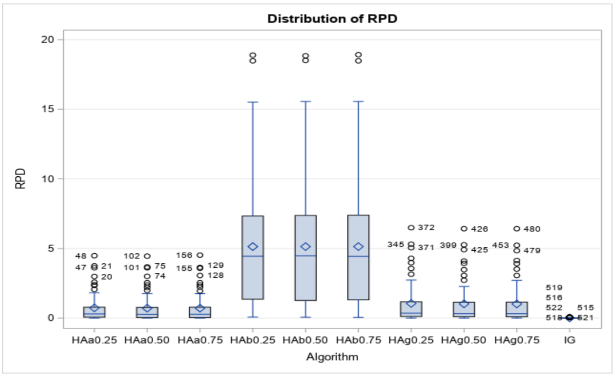

Fig. 3. boxplot for distribution of RPD

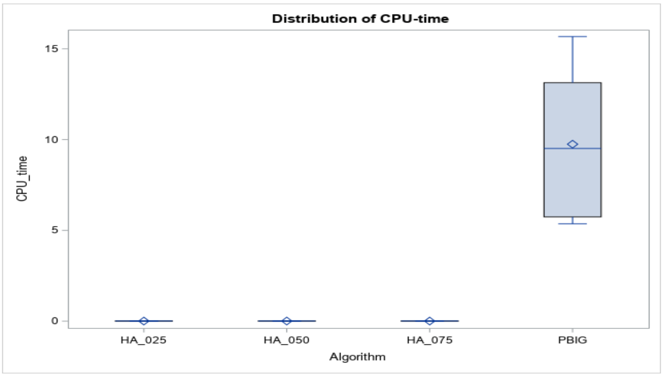

Fig. 4. CPU time of heuristics and the algorithm for large *n*

Table 7
RPD for nine heuristics and IGPB algorithm

n	Type	HAa0.25		HAa0.50		HAa0.75		HAb0.25		HAb0.50		HAb0.75		HAg0.25		HAg0.50		HAg0.75		IGPB	
		RPD		RPD		RPD		RPD		RPD		RPD		RPD		RPD		RPD		RPD	
		mean	max	mean	max	mean	max	mean	max	mean	max	mean	max	mean	max	mean	max	mean	max	mean	max
100	1-1	0.08	0.86	0.06	0.89	0.06	0.89	2.24	4.95	2.47	5.01	2.47	5.46	0.14	2.22	0.10	0.97	0.09	1.03	0.00	0.01
	1-2	0.17	1.68	0.14	1.68	0.14	1.68	5.38	9.73	5.44	9.74	5.47	9.74	0.29	2.11	0.23	1.98	0.24	1.98	0.01	0.06
	1-3	0.35	2.98	0.28	2.98	0.30	2.98	7.98	14.13	7.95	14.13	7.85	14.13	0.52	3.82	0.50	3.82	0.49	3.82	0.04	0.17
	1-4	0.05	0.58	0.03	0.58	0.03	0.50	2.14	5.40	2.13	5.40	2.14	5.40	0.09	1.43	0.07	1.36	0.05	0.83	0.00	0.00
	1-5	0.19	2.26	0.15	2.26	0.14	2.89	4.81	9.96	4.80	9.96	4.85	9.62	0.29	4.40	0.28	4.40	0.27	4.40	0.01	0.10
	1-6	0.35	3.08	0.33	3.08	0.35	3.08	7.33	11.58	7.25	11.58	7.30	11.58	0.48	3.08	0.48	3.08	0.46	3.08	0.04	0.20
	1-7	0.06	0.83	0.04	1.21	0.04	1.21	1.67	4.54	1.70	4.54	1.71	4.33	0.11	1.30	0.11	1.90	0.11	1.90	0.00	0.01
	1-8	0.20	1.79	0.13	1.79	0.12	1.19	4.18	8.56	4.24	8.56	4.18	8.56	0.26	1.98	0.24	2.42	0.22	2.42	0.01	0.04
	1-9	0.32	2.73	0.29	2.73	0.28	2.73	6.28	11.53	6.22	11.52	6.26	11.52	0.29	3.02	0.26	2.94	0.25	2.94	0.03	0.14
	2-1	0.90	4.33	0.83	4.38	0.86	4.32	3.82	9.24	3.73	8.81	3.62	8.81	1.18	4.53	1.11	4.53	1.12	4.53	0.00	0.03
	2-2	1.45	5.16	1.36	4.31	1.38	4.31	10.19	15.78	10.23	15.78	10.22	15.78	2.28	7.73	2.18	7.73	2.15	7.32	0.01	0.19
	2-3	2.35	7.69	2.20	7.78	2.22	7.78	15.51	23.04	15.56	23.04	15.56	23.04	3.56	9.40	3.48	11.52	3.52	11.52	0.03	0.43
	2-4	0.13	1.21	0.12	1.21	0.11	1.21	1.36	4.22	1.27	4.22	1.31	4.15	0.20	2.33	0.16	2.33	0.16	2.33	0.00	0.00
	2-5	0.37	2.41	0.35	2.41	0.34	2.41	3.96	8.82	3.94	8.82	3.98	8.82	0.57	5.01	0.54	5.01	0.55	5.01	0.00	0.05
	2-6	0.72	3.78	0.67	3.78	0.64	3.78	6.40	10.23	6.41	10.23	6.43	10.15	0.84	4.94	0.80	4.94	0.76	4.94	0.01	0.12
	2-7	0.00	0.03	0.00	0.03	0.00	0.03	0.10	0.91	0.11	0.95	0.11	0.95	0.00	0.02	0.00	0.02	0.00	0.02	0.00	0.00
	2-8	0.00	0.03	0.00	0.03	0.00	0.03	0.44	1.73	0.43	1.68	0.41	1.68	0.01	0.24	0.00	0.24	0.00	0.24	0.00	0.00
	2-9	0.00	0.02	0.00	0.18	0.00	0.18	0.94	3.10	0.89	3.10	0.86	3.10	0.00	0.43	0.01	0.33	0.00	0.28	0.00	0.00
	3-1	2.06	5.29	1.99	5.04	1.95	5.04	3.99	9.01	4.12	9.65	4.01	9.53	2.29	6.04	2.27	6.15	2.16	6.35	0.00	0.02
	3-2	2.99	7.67	3.00	6.46	3.06	6.41	9.13	13.62	9.22	14.57	9.15	14.57	4.29	9.48	4.10	9.24	4.14	9.28	0.00	0.29
	3-3	3.75	8.99	3.68	8.71	3.64	8.75	14.87	25.28	14.85	23.85	14.94	23.85	5.31	12.13	5.26	12.13	5.23	12.13	0.00	0.00
	3-4	0.21	2.57	0.19	2.28	0.18	2.28	0.85	4.66	0.88	4.15	0.86	4.15	0.26	2.28	0.29	2.57	0.29	2.57	0.00	0.00
	3-5	0.38	1.64	0.39	1.98	0.39	1.98	3.04	6.84	2.96	7.04	2.91	7.04	0.64	2.76	0.56	2.69	0.60	2.69	0.00	0.03
	3-6	0.74	2.83	0.72	2.83	0.74	2.83	5.63	10.89	5.41	12.28	5.39	12.28	1.16	4.53	1.14	4.53	1.15	4.59	0.01	0.07
	3-7	0.00	0.05	0.00	0.05	0.00	0.05	0.09	1.05	0.07	0.85	0.07	0.85	0.00	0.04	0.00	0.04	0.00	0.04	0.00	0.00
	3-8	0.00	0.03	0.00	0.03	0.00	0.03	0.09	0.90	0.08	0.90	0.07	0.78	0.00	0.05	0.00	0.05	0.00	0.05	0.00	0.00
	3-9	0.01	0.61	0.01	0.61	0.01	0.61	0.64	2.07	0.63	2.07	0.62	2.07	0.03	0.99	0.04	1.00	0.04	1.00	0.00	0.00
mean		0.66		0.63		0.63		4.56		4.55		4.55		0.93		0.9		0.89		0.01	
200	1-1	0.15	0.83	0.13	0.83	0.13	0.83	3.10	5.57	3.07	5.57	3.09	5.57	0.21	1.19	0.18	1.19	0.16	0.83	0.01	0.08
	1-2	0.35	1.66	0.32	1.66	0.31	1.66	6.18	10.38	6.20	10.38	6.20	10.39	0.40	2.44	0.34	2.44	0.33	2.44	0.05	0.28
	1-3	0.45	2.21	0.42	1.88	0.42	1.88	8.76	12.03	8.83	12.60	8.84	12.60	0.61	3.65	0.62	3.65	0.60	3.65	0.08	0.32
	1-4	0.14	1.47	0.11	0.97	0.11	0.97	2.60	5.34	2.66	5.34	2.71	5.01	0.18	1.59	0.15	1.59	0.16	1.59	0.02	0.07
	1-5	0.26	1.72	0.24	1.33	0.24	1.33	5.66	10.00	5.66	9.06	5.59	9.06	0.35	2.35	0.29	2.35	0.28	1.92	0.06	0.24
	1-6	0.42	2.11	0.41	2.04	0.42	2.04	8.02	12.51	8.02	11.74	8.05	11.74	0.69	3.48	0.66	3.48	0.65	3.48	0.08	0.33
	1-7	0.13	1.15	0.10	1.15	0.09	1.15	2.31	5.14	2.34	4.64	2.35	4.64	0.13	1.33	0.11	1.21	0.10	1.21	0.02	0.05
	1-8	0.27	2.24	0.24	1.37	0.23	1.23	5.17	8.75	5.15	8.87	5.12	8.87	0.32	2.23	0.29	2.24	0.27	2.24	0.05	0.18
	1-9	0.39	2.30	0.36	2.27	0.36	2.27	7.30	11.56	7.37	10.51	7.39	11.46	0.56	2.62	0.51	2.62	0.51	2.62	0.08	0.28
	2-1	1.34	3.47	1.32	4.57	1.32	4.58	5.26	8.77	5.10	8.77	5.19	9.25	1.68	4.92	1.62	4.90	1.61	4.85	0.00	0.10
	2-2	1.82	4.35	1.76	4.35	1.75	4.35	13.03	18.45	12.95	18.45	12.95	18.45	2.73	7.35	2.72	6.96	2.71	6.96	0.01	0.34
	2-3	2.39	6.37	2.37	6.37	2.37	6.37	18.89	26.45	18.84	27.58	18.92	27.58	3.98	10.80	3.93	9.73	3.91	9.73	0.00	0.36
	2-4	0.18	1.09	0.19	1.09	0.19	1.09	1.68	3.87	1.69	3.91	1.71	3.87	0.29	2.11	0.28	2.11	0.27	2.11	0.01	0.05
	2-5	0.48	2.23	0.48	2.23	0.49	2.23	5.12	9.91	5.17	9.91	5.16	9.91	0.77	2.73	0.76	2.73	0.75	2.67	0.01	0.14
	2-6	0.78	3.73	0.77	3.73	0.78	3.73	8.01	11.15	8.05	11.46	8.08	11.46	1.21	6.14	1.20	6.14	1.21	6.14	0.02	0.22
	2-7	0.00	0.03	0.00	0.03	0.00	0.03	0.12	0.79	0.12	0.79	0.12	0.79	0.00	0.14	0.00	0.14	0.00	0.14	0.00	0.00
	2-8	0.00	0.03	0.00	0.07	0.00	0.07	0.52	1.46	0.51	1.42	0.50	1.42	0.01	0.12	0.01	0.11	0.01	0.11	0.00	0.01
	2-9	0.03	0.67	0.02	0.67	0.02	0.67	0.90	2.44	0.92	2.44	0.90	2.44	0.03	0.67	0.03	0.67	0.03	0.67	0.00	0.02
	3-1	2.58	5.14	2.54	5.10	2.54	5.12	6.52	11.31	6.53	11.72	6.54	11.74	3.15	9.36	3.06	9.34	3.06	9.37	0.00	0.00
	3-2	3.61	7.25	3.61	6.79	3.58	6.81	12.02	15.54	12.00	15.54	11.99	15.54	5.05	9.98	4.92	9.97	4.84	8.79	0.00	0.00
	3-3	4.48	9.41	4.45	9.33	4.52	9.33	18.48	25.10	18.51	25.10	18.48	25.10	6.49	11.86	6.42	12.35	6.42	12.35	0.00	0.27
	3-4	0.26	1.48	0.25	1.48	0.26	1.48	1.41	3.81	1.40	3.81	1.44	3.81	0.35	1.93	0.34	1.49	0.35	1.49	0.00	0.03
	3-5	0.52	2.15	0.52	1.92	0.52	1.92	4.70	7.52	4.70	7.55	4.68	7.62	0.96	2.81	0.95	2.86	0.96	2.86	0.01	0.09
	3-6	0.80	2.39	0.82	2.40	0.83	2.40	7.59	11.39	7.64	11.39	7.66	11.39	1.58	4.34	1.59	4.34	1.57	4.34	0.01	0.23
	3-7	0.00	0.04	0.00	0.04	0.00	0.04	0.07	0.57	0.06	0.57	0.05	0.50	0.00	0.11	0.00	0.11	0.00	0.11	0.00	0.00
	3-8	0.00	0.19	0.00	0.19	0.00	0.19	0.10	0.76	0.11	0.77	0.11	0.77	0.01	0.19	0.01	0.19	0.00	0.19	0.00	0.00
	3-9	0.02	0.27	0.02	0.27	0.02	0.27	0.77	2.41	0.75	2.41	0.76	2.41	0.03	0.59	0.03	0.56	0.03	0.56	0.00	0.01
	mean	0.81		0.79		0.80		5.71		5.72		5.73		1.18		1.15		1.14		0.02	
Total mean		0.74		0.71		0.72		5.14		5.14		5.14		1.06		1.03		1.02		0.02	

Another analysis of variance (ANOVA) was conducted to assess the normality assumption of the Relative Percentage Deviation (RPD) observations. Confirming the absence of normality, Table 4 (columns 4 and 5) demonstrated that the p-values, obtained from four normality tests, were all below 0.01. Following this, relying on the ranks of RPDs, Table 5 (column 3) revealed that the Kruskal–Wallis test significantly confirmed that "the RPD samples did not come from the same distribution, given a p-value of less than 0.001". Subsequently, the Dwass–Steel–Critchlow–Fligner (DSCF) procedure was employed to discern pairwise differences between the nine heuristics and the IGPB algorithm. Columns 4 and 5 of Table 6 reported that the IGPB consistently held the best performance position, whereas the three heuristics (HAb*) were relegated to the worst performance group for large-size jobs. Additionally, Fig. 3 boxplots illustrated that the RPD observations for the IGPB exhibited less dispersion than those for the nine heuristics. This implies that the IGPB is not only accurate but also stable compared to the nine heuristics when addressing the challenges posed by large-sized job problems. Furthermore, regarding computational time, Fig. 4 displayed boxplots of CPU times (in seconds) for HAa* heuristics and the IGPB algorithm.

7. Conclusions

In this investigation, we addressed a scheduling problem characterized by scenario-dependent variations in job processing times and release dates. For the optimization of robust schedules in scenarios involving small-sized jobs, we introduced four distinctive properties and a lower bound, seamlessly integrated into a branch-and-bound methodology. Additionally, nine local heuristics, leveraging various weights of scenario-dependent parameters, were proposed. To tackle the scheduling challenge presented by large-sized instances, we designed an iterated greedy population-based (IGPB) algorithm. The performance of all proposed algorithms was meticulously assessed and compared using statistical methodologies. Despite the IGPB algorithm requiring more CPU time for robust job sequence identification, it demonstrated superior optimality and reliability compared to its counterparts.

This study delves into the intricacies of a scheduling problem where processing times and release dates fluctuate under distinct scenarios. While we addressed scenario-dependent factors, there exist other uncertain variables in single-machine scheduling problems, such as rush orders, alterations in due dates or order quantities, order cancellations, or a stringent lower bound based on scenarios and robust properties. Future research avenues could explore these factors. Additionally, a comparative analysis between the proposed branch-and-bound algorithm and integer programming-based approaches could be undertaken. Furthermore, incorporating the fuzzy concept into the model could enhance its accuracy and accommodate uncertainties more effectively.

The statement of data

The corresponding author will provide the data sets upon request.

Acknowledgements

The authors would like to thank the editors and reviewers for their valuable suggestions and positive comments. This article was supported in part by the Ministry of Science and Technology of Taiwan, NSTC 112-2221-E-035-060-MY2, and in part by the National Natural Science Foundation of China (11991022, 11571321), the Joint Fund for Innovation and Development of the Chongqing Natural Science Foundation, and Key Research Fund Projects of Chongqing Graduate Education and Teaching Reform (yjg182019).

References

Aissi, H., Aloulou, M.A., & Kovalyov, M. Y. (2011). Minimizing the number of late jobs on a single machine under due date uncertainty, *Journal of Scheduling*, *14*(4), 351-360.

Alon, N., Azar, N.Y., Weginger, G.J., & Yadid, T. (1998). Approximation schemes for scheduling on parallel machines, *Journal of scheduling*, *1*, 55-66.

Aloulou, M.A., & Della Croce, F. (2008). Complexity of single machine scheduling problems under scenario-based uncertainty, *Operations Research Letters*, *36*(3), 338-342.

Bouamama, S., Blum, C., & Boukerram, A. (2012). A population-based iterated greedy algorithm for the minimum weight vertex cover problem. *Applied Soft Computing*, *12*(6), 1632-1639.

Chekuri, C., Motwani, R., Natarajan, B., & Stein, C. (1997). Approximation Techniques for average completion time scheduling, *Proceedings of the annual ACM-SIAM symposium on discrete algorithm* (SODA), pp 609-617.

Chen, B., Potts, C.N., & Weginger, J.G. (1998). A review of machine scheduling, complexity and approximability, *Handbook of combinatorial optimization*, D-Z Du and P. Paradalos (eds.), pp 21-169, Kluwer Academic Press, Boston.

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

Dessouky, M.M. (1998). Scheduling identical jobs with unequal ready times on uniform parallel machines to minimize the maxmun total lateness, *Computer & Industrial Engineering*, *34*(4), 793-806.

de Farias, I. R., Zhao, H., & Zhao, M. (2010). A family of inequalities valid for the robust single machine scheduling polyhedron. *Computers and Operations Research*, *37*(9), 1610-1614.

French, S. (1982). *Sequencing and Scheduling, An Introduction to the Mathematics of the Job Shop*. Ellis Horwood Limited.

Gilenson, M., Naseraldin, H., & Yedidsion, L. (2018). An approximation scheme for the bi-scenario sum of completion times trade-off problem, *Journal of Scheduling*, *22*(3), 289-304.

Gilenson, M., & Shabtay, D. (2021). Multi-scenario scheduling to maximise the weighted number of just-in-time jobs. *Journal of the Operational Research Society*, *72*(8), 1762-1779.

Hardy, G.H., Littlewood, J. E., & Polya, G. (1967). *Inequalities* (p. 261). London, Cambridge University Press.

Hermelin, D., Manoussakis, G., Pinedo, M., Shabtay, D., & Yedidsion, L. (2020). Parameterized multi-scenario single-machine scheduling problems, *Algorithmica*, *82*, 2644-2667.

Hochbaum, D.S., & Shmoys, D.B. (1987). Using dual approximation algorithms for scheduling problems, theoretical and practical results, *Journal of the ACM*, *34*, 144-162.

Hollander, M. D., Wolfe, A., & Chicken, E. (2014). *Nonparametric Statistical Methods*, third edition, John Wiley & Sons, Inc., Hoboken, New Jersey.

Johnson, D. (2001). A theoretician's guide to the experimental analysis of algorithms. Conference, Data Structures, Near

Neighbor Searches, and Methodology, Fifth and Sixth DIMACS Implementation Challenges.

Kasperski, A., & Zieliński, P. (2016). Robust discrete optimization under discrete and interval uncertainty, A survey. In *Robustness analysis in decision aiding, optimization, and analytics* (pp.113-143), Springer, Cham.

Kouvelis, P., & Yu, G. (1996). *Robust Discrete Optimization and It Application* (Vol.14). Springer Science & Business Media.

Kouvelis, P., Daniels, R. L., & Vairaktarakis, G. (2000). Robust scheduling of a two-machine flow shop with uncertain processing times. *Iie Transactions*, *32*(5), 421-432.

Lenstra, J.K., Rinnooy Kan, A.H.G., & Brucker, P. (1977). Complexity of machine scheduling problems, *Annals of Discrete Mathematics*, *1*, 343-362.

Lin, W.-C., Xu, J., Bai, D., Chung, I-H., Liu, S.-C., & Wu, C.-C (2019). Artificial bee colony algorithms for the order scheduling with release dates, *Soft Computing*, *23*(18), 8677-8688.

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

Mastrolilli, M., Mutsanas, N., & Svensson, O. (2013). Single machine scheduling with scenarios. *Theoretical Computer Science*, *477*, 57-66.

Nawaz, M., Enscore Jr, E.E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem, *Omega*, *11*(1), 91-95.

Pinedo, M. (2008). *Scheduling, theory, algorithms and systems*. NJ, Prentice-Hall, Upper Saddle River. Third version.

Reever, C. (1995). Heuristics for scheduling a single machine subject to unequal job release times*, European Journal of Operational Research, 80*, 397-403.

Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem, *European Journal of Operational Research*, *177*(3), 2033-2049.

Ruiz, R., & Stützle, T. (2008). An Iterated Greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives, *European Journal of Operational Research*, *187*(3),1143-1159.

Schuurman, P., & Woeginger, G.J. (1999). Polynomial time approximation algorithms for machine scheduling, ten open problems, *Journal of scheduling*, *2,* 203-214.

Sevastianov, S.V., & Woeginger, G.J. (1998). Makespan minimization in open shops, a polynomial time approximation scheme, *Mathematical Programming*, *82*, 191-198.

Smith, W.E. (1956). Various optimizers for single stage production, *Naval Research Logistics Quarterly*, *3*(1), 56-66.

Sotskov, I. N., & Werner, F. (2014). *Sequencing and scheduling with inaccurate data*. Hauppauge, NY, Nova Science Publishers.

Wang, J. B., Lv, D. Y., Wang, S. Y., & Jiang, C. (2023). Resource allocation scheduling with deteriorating jobs and position-dependent workloads. *Journal of Industrial and Management Optimization*, *19*(3), 1658-1669.

Wang, F., & Wu, B. (2024). The k-Sombor Index of Trees. *Asia-Pacific Journal of Operational Research*, *41*(1). DOI, 10.1142/S0217595923500264.

Wu, C.-C., Wu, W.-H., Chen, J.-C., Yin, Y., & Wu, W.-H. (2013). A study of the single-machine two-agent scheduling problem with release times, *Applied Soft Computing*, *13*, 998-1006.

Wu, C.-C., Gupta, J.N.D., Cheng, S.R., Lin, B.M.T., Yip, S.H., & Lin, W.C. (2021). Robust scheduling of a two-stage assembly shop with scenario-dependent processing times. *International Journal of Production Research*, *59*(17), 5372-5387.

Yang, J., & Yu, G. (2002). On the robust single machine scheduling problem, *Journal of Combinatorial Optimization*, *6*(1), 17-33.

Yin, Y., Wu, W.-H., Cheng, S.-R., **&** Wu C.-C. (2012). An investigation on a two-agent single-machine scheduling problem with unequal release dates. *Computers & Operations Research*, *39*, 3062-3073.