International Journal of Industrial Engineering Computations 14 (2023) 265–274

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Composite heuristics and water wave optimality algorithms for tri-criteria multiple job classes and customer order scheduling on a single machine

Lung-Yu Li[a], Win-Chin Lin[b], Danyu Bai[c], Xingong Zhang[d], Ameni Azzouz[e,f], Shuenn-Ren Cheng[g], Ya-Li Wu[b] and Chin-Chia Wu[b*]

[a]*Department of Computer Science and Information Engineering, Cheng Shiu University, Kaohsiung City 83347, Taiwan*
[b]*Department of Statistics, Feng Chia University, Taichung, 40724, Taiwan*
[c]*School of Maritime Economics & Management, Dalian Maritime University, Dalian, 116026, China*
[d]*College of Mathematics Science, Chongqing Normal University, Chongqing 401331, China*
[e]*SMART Lab., Institut supérieur de gestion de Tunis, Université de Tunis, Tunis, Tunisia*
[f]*Tunis Business School, University of Tunis, Tunisia*
[g]*Graduate Institute of Business Administration, Cheng Shiu University, Kaohsiung City 83347, Taiwan*

C H R O N I C L E

Article history:
Received May 10 2022
Accepted February 14 2023
Available online
February, 14 2023

Keywords:
Multiple job classes
Tri-criteria
Water wave optimality algorithm
Setup time

A B S T R A C T

Among the well-known scheduling problems, the customer order scheduling problem (COSP) has always been of great importance in manufacturing. To reflect the reality of COSPs as much as possible, this study considers that jobs from different orders are classified in various classes. This paper addresses a tri-criteria single-machine scheduling model with multiple job classes and customer orders on which the measurement minimizes a linear combination of the sum of the ranges of all orders, the tardiness of all orders, and the total completion times of all jobs. Due to the NP-hard complexity of the problem, a lower bound and a property are developed and utilized in a branch-and-bound for solving an exact solution. Afterward, four heuristics with three local improved searching methods each and a water wave optimality algorithm with four variants of wavelengths are proposed. The tested outputs report the performances of the proposed methods.

1. Introduction

Currently, the real manufacturing industry has been the subject of extensive study both from a theoretical and a practical point of view. Several challenges have been raised, such as increasing product varieties, higher product customization and tremendous competition. Hence, the necessity of optimizing these manufacturing systems has received considerable attention. Production scheduling is one of the oldest and most difficult problems in manufacturing systems. Among the well-known scheduling problems, the customer order scheduling problem (COSP) has always been of great importance in manufacturing. The main challenge of the COSP is to obtain the job sequences to satisfy the customer's demand; customers order many kinds of products, which are executed in only one machine (Della Croce *et al*., 2996).

The COSP mainly exists in make-to-order and make-to-assembly production systems where a single machine executes different types of products. Once the customer asks for two different products, the order will be produced, packed and shipped together. Julien and Magazine (1990) are pioneers who introduced the COSP into the scheduling field. Over the last decades, COSP topics have become an active research field. For the total order lead time criterion, Erel and Gosh (2007) address the COSP and show that the problem is NP-hard for the first time. Under parallel machines and dispatched in batches, Su *et al.* (2013) developed some heuristics to solve the COSP for the maximum lateness. Framinan and Perez-Gonzalez (2018) proposed two metaheuristics (Zheng, 2015; Zheng et al., 2019), created a constructive heuristic, and provided an MIP formulation for the total tardiness COSP model. Wu *et al.* (2019) also applied a branch-and-bound, some heuristics and metaheuristics to solve the COSP with learning considerations.

Regarding the real-life settings, a setup time is needed whenever the machine switches from one product to another. Job scheduling with setup costs or setup times in many manufacturing and service environments has received increasing attention in the research community. Allahverdi and Soroush (2008) emphasize that reducing setup costs or times may contribute to reliable services or products to be shipped in time. For more real applications of setup costs or times, we refer readers to five articles from the literature by Yang and Liao (1999), Cheng *et al*. (2000), Allahverdi *et al*. (1999, 2008), and Allahverdi

* Corresponding author
E-mail: cchwu@fcu.edu.tw (C.-C. Wu)
ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)
2023 Growing Science Ltd.
doi: 10.5267/j.ijiec.2023.2.002

(2015). Despite the importance of the setup time with job scheduling, the literature on the COSP with a setup time is very limited. To minimize the average customer order flowtime criterion, Hazir *et al.* (2008) address a comparative study between four different metaheuristics to solve this problem. Recently, de Athayde Prata *et al.* (2021a) proposed two mixed integer linear programming models and a fixed variable list algorithm for solving the COSP with a sequence-dependent setup time (SDST). Furthermore, they propose a discrete differential evolution algorithm for the same problem in de Athayde Prata *et al.* (2021b). To reflect the reality of the customer order scheduling problem as much as possible, this study considers that jobs from different orders are classified in various classes. Each order consists of at least one job from each of the job classes. A setup time is needed whenever the machine switches from a job in one class to a job in another class. The literature indicates Gupta *et al.* (1997), who investigated the bicriteria COSP with multiple classes and setup times for the first time in which the makespan is the first objective and the total carrying costs of customer orders are the second objective. The carrying (holding) cost of a customer order is computed by the difference in the time between the completion times of the first job and the last job in the customer order. Since that study, no other research papers have considered this problem, although it is relevant to the practical importance of this production environment.

In this paper, the purpose of our investigation is to minimize a linear combination of a tri-criteria single-machine scheduling model with multiple job classes and customer orders, where the first criterion is the sum of the holding costs of all orders, the second is the total tardiness of all orders, and the third is the total completion times of all jobs. Given the NP-hard nature of the problem, a lower bound and property are developed and utilized in a B&B method for finding an optimal schedule. Then, four heuristics, each along with three local improved searching methods, as well as a water wave optimality algorithm with four variants of wavelengths are introduced to search for near optimal solutions. Then, the obtained results are analyzed and reported.

The remaining organization of this work is described as follows. Section 2 presents the notations and problem definition. Section 3 reviews one dominance property and one lower bound in a B&B method for an exact schedule. Section 4 discusses composite heuristics and four wavelengths of water wave optimality algorithms. Section 5 is dedicated to a discussion on the parameter settings of the proposed WWO algorithm. Section 6 presents several computational simulations to determine the performances of the proposed methods. Conclusions and suggestions are provided in Section 7.

2. Notations and problem formulation

First, some notation used in this study is defined as follows:

$O_M = \{O_1, O_2, \ldots, O_m\}$ denotes a set of m (customer) orders;

$B = \{B_1, B_2, \ldots, B_K\}$ denotes a set of K types of job classes;

$\{s_1, s_2 \ldots, s_K\}$: denotes a set of K setup times for K job classes;

$\{p_1, p_2 \ldots, p_n\}$: denotes a set of job processing times for n jobs ($n = m \times K$);

σ, σ': denotes two full job schedules;

δ, δ': denotes two partial schedules;

$\{d_1, d_2 \ldots, d_m\}$: denotes a set of due dates of m orders;

$C_i(\sigma), C_j(\sigma)$: denote the completion times of job i and job j in σ;

$C_i(\sigma'), C_j(\sigma')$: denote the completion times of job i and job j in σ';

[ ]: denotes the position in a given schedule;

$C_u^f(\sigma), C_u^f(\sigma')$: denotes the completion time of the first job in O_u in $\sigma(\sigma')$, $u = 1, 2, \ldots, m$;

$C_u^l(\sigma), C_u^l(\sigma')$: denotes the completion time of the last job in O_u in $\sigma(\sigma')$, $u = 1, 2, \ldots, m$;

$HC_u(\sigma) = C_u^l(\sigma) - C_u^f(\sigma)$: denotes the holding cost of order O_j in σ, $u = 1, 2, \ldots, m$;

$C_{max}(\sigma) = \max\{C_1(\sigma), C_2(\sigma), \ldots, C_n(\sigma)\}$;

$\sum_{u=1}^{m} HC_u(\sigma)$: denotes the total holding cost for the m orders.

The considered problem can be stated as follows. Suppose that there are a set of n jobs that are grouped into a set of m orders, and each order includes K jobs that belong to K different classes of jobs. Jobs are ready at time zero and will be operated on a single machine. No preemption is allowed during job processing. Suppose that each job J_i has a processing time and must belong to a job class. Furthermore, an order consists of at least one job from each job class. During the processing period, if job class a is scheduled immediately following the previous job class b, $a \neq b$, then a setup time s_a is needed; otherwise, it does not need a setup time. The definition of holding cost $HC_u(\sigma)$ of order u indicates the range between the completion time of the first job in order u and the completion time of the last job from the same order. In this study, we address a multiple-class order scheduling problem to minimize a linear combination of the total holding cost $\sum_{u=1}^{m} HC_u(\sigma)$, total tardiness cost $\sum_{u=1}^{m} T_u(\sigma)$, and total completion times $\sum_{j=1}^{n} C_j(\sigma)$ of all given n $(n = m \times K)$ jobs. For simplification, the tri-criterion is termed $h(\sigma)$ for schedule σ; that is, $h(\sigma) = \alpha \sum_{u=1}^{m} HC_u(\sigma) + \beta \sum_{u=1}^{m} T_u(\sigma) + \gamma \sum_{j=1}^{n} C_j(\sigma)$.

The proposed problem is also NP-hard because for fixed $m = 1, \alpha = \beta = 0, \gamma = 1,$ (i.e., minimize $\sum_{j=1}^{n} C_j(\sigma)$ only), and each class has only one job and all processing times are equal; thus, it is an NP-hard problem (see Liaee & Emmons, 1997).

3. A lower bound and a property

To apply the B&B method to find the optimal schedule, it is necessary to determine a lower bound on a node of $\sigma = (\pi, \pi^c)$, where π is the scheduled part with n_π jobs and π^c is the set of n_{π^c} unscheduled jobs. Let t_π be the completion time of the last job J_L in π and $\sum_{u \in \pi} HC_u$ be the total holding cost of those completed orders in π. For the lower bound of the holding cost in π^c, we sort the processing times of n jobs by the smallest processing times first rule, i.e., $p_{(1)} \leq p_{(2)} \leq \cdots \leq p_{(n)}$ is a nondecreasing order of $\{p_1, p_2, \ldots, p_n\}$, and we record the frequency number f_i for each order in π^c, i=1, 2, …, q, and set $1 \leq f_1 \leq f_2 \ldots \leq f_q$ with $q < m$. Following this, we calculate the estimated completion times of jobs in π^c as $C_{[n_\pi+i]}(\sigma) = t_\pi + \sum_{j=1}^{i} p_{(j)}$, i=1, 2, …, n_{π^c}. It should be noted that $n_{\pi^c} < n$. Thus, we can estimate the holding cost for q orders as follows:

$$\sum_{u \in \pi^c} HC_u = \{C_{[n_\pi+f_1]}(\sigma) - C_{[n_\pi+1]}(\sigma) + C_{[n_\pi+f_1+f_2]}(\sigma) - C_{[n_\pi+f_1+1]}(\sigma) + \ldots + C_{[n_\pi+f_1+f_2+\cdots+f_q]}(\sigma) - C_{[n_\pi+f_1+f_2+\cdots+f_{q-1}+1]}(\sigma)\}$$
$$= \sum_{u=1}^{q} (C_{[n_\pi+\sum_{v=1}^{u} f_v]}(\sigma) - C_{[n_\pi+\sum_{v=0}^{u-1} f_v]}(\sigma)),$$

where it is assumed that $f_0 = 1$, the range of $C_{[n_\pi+f_1]}(\sigma) - C_{[n_\pi+1]}(\sigma)$ denotes the estimated holding cost of the remaindering order with f_1 jobs, the range of $C_{[n_\pi+f_1+f_2]}(\sigma) - C_{[n_\pi+f_1+1]}(\sigma)$ denotes the estimated holding cost of the remaindering order with f_2 jobs, …, and the range of $C_{[n_\pi+f_1+f_2+\cdots+f_q]}(\sigma) - C_{[n_\pi+f_1+f_2+\cdots+f_{q-1}+1]}(\sigma)$ denotes the estimated holding cost of the remaindering order with f_q jobs.

For the lower bound of the tardiness cost of those orders in π^c, we sort the due dates of q orders by the earliest due dates first rule, i.e., $d_{(1)} \leq d_{(2)} \leq \cdots \leq d_{(q)}$ is a nondecreasing order of $\{d_1, d_2, \ldots, d_q\}$, which can be determined as follows:

$$\sum_{u \in \pi^c} TC_u = \{\max\{C_{[n_\pi+f_1]}(\sigma) - d_{(1)}, 0\} + \max\{C_{[n_\pi+f_1+f_2]}(\sigma) - d_{(2)}, 0\} + \cdots + \max\{C_{[n_\pi+f_1+f_2+\cdots+f_q]}(\sigma) - d_{(q)}, 0\} = \sum_{u=1}^{q} (\max\{C_{[n_\pi+\sum_{v=1}^{u} f_v]}(\sigma) - d_{(i)}, 0\}).$$

For the lower bound of total completion times of those jobs in π^c, we assign the processing times of n_{π^c} jobs by the smallest processing times first rule, i.e., $p_{(1)} \leq p_{(2)} \leq \cdots \leq p_{(n_{\pi^c})}$ is a nondecreasing order of $\{p_i \in \pi^c\}$, and we have

$C_{(j)}(\sigma)=\sum_{i=1}^{j} t_\pi + p_{(i)}, j=1, 2, \ldots, n_{\pi^c}$,
According to the above formulas, we have the following:
$\sum_{j=1}^{n_{\pi^c}} C_{(j)}(\sigma)=n_{\pi^c} \times t_\pi + \sum_{j=1}^{n_{\pi^c}}(n_{\pi^c} - j + 1)p_{(j)}$
From the above, the lower bound can be obtained as follows:

$$\left\{\alpha \sum_{u=1,..,m} HC_u + \beta \sum_{u=1,..,m} TC_u + \gamma \sum_{i=1,..,n} C_i\right\} \geq \alpha \sum_{u\in\pi} HC_u + \beta \sum_{u\in\pi} TC_u + \gamma \sum_{j\in\pi} C_j$$
$$+\alpha \sum_{u=1}^{q} (C_{[n_\pi + \sum_{v=1}^{u} f_v]}(\sigma) - C_{[n_\pi + \sum_{v=0}^{u-1} f_v]}(\sigma))$$
$$+\beta \sum_{u=1}^{q} \max\{C_{[n_\pi + \sum_{v=1}^{u} f_v]}(\sigma) - d_{(u)}, 0\}$$
$$+\gamma(n_{\pi^c} \times t_\pi + \sum_{j=1}^{n_{\pi^c}}(n_{\pi^c} - j + 1)p_{(j)}).$$

Apart from that, a property is also proposed to increase the searching power of the branch-and-bound algorithm. Let $\sigma = (\pi, J_i, J_j, \pi^c)$ and $\sigma' = (\pi, J_j, J_i, \pi^c)$ present two full schedules in which job i is scheduled before job j in σ, while job j is scheduled before job i in σ′. Moreover, let job J_L denote the last job in π. Assuming that jobs J_L, J_i, J_j, are in the same class, and $J_i \in O_u$ and $J_j \in O_v$, $u \neq v$, the following property can be obtained.

Property 1: As jobs J_L, J_i, and J_j are in the same class, and $J_i \in O_u$, $J_j \in O_v$, $u \neq v$, if the first assigned job of O_u is scheduled after the first assigned job of O_v (in π), both jobs J_i and J_j are the last assigned jobs of O_u and O_v, $p_i < p_j$ and $d_u < d_v$, then σ dominates σ'.
Proof: Recall that the objective function of this study is $h(\sigma) = \alpha \sum_{w=1}^{m} HC_w(\sigma) + \beta \sum_{w=1}^{m} T_w(\sigma) + \gamma \sum_{j=1}^{n} C_j(\sigma)$.
It should be to be shown that $h(\sigma') > h(\sigma)$.
(1) Both jobs J_i and J_j belong to the same class, and no additional setup times are needed to make a pairwise exchange of these two jobs. Therefore, $\sum_{j=1}^{n} C_j(\sigma) = \sum_{j=1}^{n} C_j(\sigma')$.
(2) Since $C_u^f(\sigma') = C_u^f(\sigma)$, and $C_v^f(\sigma') = C_v^f(\sigma)$, the $\sum_{w=1}^{m} HC_w(\sigma') - \sum_{w=1}^{m} HC_u(\sigma) = [HC_v(\sigma') + HC_u(\sigma')] - [HC_u(\sigma) + HC_v(\sigma)] = [C_v^l(\sigma') - C_v^f(\sigma') + C_u^l(\sigma') - C_u^f(\sigma')] - [C_u^l(\sigma) - C_u^f(\sigma) + C_v^l(\sigma) - C_v^f(\sigma)] = [C_v^l(\sigma') - C_v^l(\sigma)] + [C_u^l(\sigma') - C_u^l(\sigma)] = p_j - (p_i + p_j) + (p_j + p_i - p_i) = p_j - p_i > 0$.
(3) The rest of this proof is to show that $\mathrm{TD} = \sum_{w=1}^{m} T_w(\sigma') - \sum_{w=1}^{m} T_w(\sigma) = [T_v(\sigma') + T_u(\sigma')] - [T_u(\sigma) + T_v(\sigma)] \geq 0$, where

$$T_u(\sigma) = max(t_\pi + p_i - d_u, 0), T_v(\sigma) = max(t_\pi + p_i + p_j - d_v, 0)$$
$$T_v(\sigma') = max(t_\pi + p_j - d_v, 0), T_u(\sigma') = max(t_\pi + p_j + p_i - d_u, 0).$$

The time order $t_\pi < t_\pi + p_i < t_\pi + p_j < t_\pi + p_i + p_j$ and the given condition $d_u < d_v$ determine the TD values of ten situations. Three of the ten situations are proven as follows, and the remaining proofs of others are similar.
Case (i): if $d_u < t_\pi + p_i < d_v < t_\pi + p_j$
$\mathrm{TD} = [(t_\pi + p_j - d_v) + (t_\pi + p_j + p_i - d_u)] - [(t_\pi + p_i - d_u) + (t_\pi + p_i + p_j - d_v)] = p_j - p_i > 0$.
Case (ii): if $t_\pi + p_i < d_u < t_\pi + p_j < d_v < t_\pi + p_i + p_j$
$\mathrm{TD} = [0 + (t_\pi + p_j + p_i - d_u)] - [0 + (t_\pi + p_i + p_j - d_v)] = d_v - d_u > 0$.
Case (iii): if $d_u < t_\pi + p_i$ and $d_v > t_\pi + p_i + p_j$
$\mathrm{TD} = [0 + (t_\pi + p_j + p_i - d_u)] - [(t_\pi + p_i - d_u) + 0] = p_j > 0$.
Combining (1), (2), and (3), $h(\sigma') > h(\sigma)$ follows. □

4. Composite heuristics and four wavelengths of water wave optimality algorithms

In this section, based on the problem features of the jobs class and customer order of jobs, we develop four problem-based heuristics and then improve each by applying three local searching schemes to refine the quality of approximate solutions. Following this, we adopt four variants of wavelengths of the water wave optimality algorithm to solve this problem as well. The details of the proposed algorithms are discussed as follows.
For the first heuristic method, in light of the concept of Gupta *et al.* (1997), we assign m orders according to the smallest value first principle on $\{OP_1, OP_2, \ldots, OP_m\}$, where $OP_u = \sum_{j\in O_u} p_j$, u=1, 2, …, m. In what follows, we assign the jobs in each order according to the largest value first principle to obtain a complete schedule. We record it as the OSPT_LPT (OSL in brief). For the second heuristic, we assign m orders according to the smallest values first principle on $\{CP_1, CP_2, \ldots, CP_K\}$, where $CP_a = \sum_{j\in B_a} p_j + s_a$, a=1, 2, …, K. Next, we assort the jobs in the first class according to the largest processing time first to yield the sequence and then assort all the jobs according to the smallest processing times first in each class for the remaining (m-1) classes to obtain a full job schedule. We call this method CSPT_LPTSPT (CSL in brief). For the third and fourth heuristics, we assign m orders first according to the earlier due dates of $\{d_1, d_2, \ldots, d_m\}$ in both heuristics. Next, we assign the jobs in each order according to the largest processing times first to obtain a complete schedule for the third method, while we assign the jobs in each order according to the smallest processing times first to obtain a complete schedule for the fourth method. We record them as OEDD_LPT (OEL in brief) for the third method and OEDD_SPT (OES in brief) for the fourth heuristic. To refine the quality of the solutions, we improve each of the proposed OSL, CSL, OEL and OES by three local searching methods (Della Croce *et al*., 1996), including extraction and forward-shifted reinsertion, pairwise interchange, and extraction and backward-shifted reinsertion. In total, 12 composite heuristics are developed and termed OSL_p, OSL_f, OSL_b, CSL_p, CSL_f, CSL_b, OEL_p, OEL_f, OEL_b, OES_p, OES_f, and OES_b.

In addition, to solve the proposed problem, we apply four variants of a water wave optimization algorithm (WWOA), where each variant has its own wavelength. In WWOA design, a wavelength presents a full job schedule. Two important factors, diversification and the intensification of searching metaheuristics, are considered when performing a WWOA (Zheng *et al*., 2019). That is, diversification involves exploring low-adaptability solutions in a large space; however, intensification involves exploiting highly adaptable solutions in a small space.

Adopting the following wavelength formulations (2), (3), (4) and (5) in Zheng *et al*. (2019) and considering the tri-criteria objective, $h(\sigma) = \alpha \sum_{u=1}^{m} HC_u(\sigma) + \beta \sum_{u=1}^{m} T_u(\sigma) + \gamma \sum_{j=1}^{n} C_j(\sigma)$, four wavelength formulas are summarized as follows:

$$\lambda_{\sigma_i} = \lambda_{\sigma_i} \times \alpha^{-\frac{h(\sigma_i) - h_{min} + \epsilon}{h_{max} - h_{min} + \epsilon}}, \quad (1)$$
$$\lambda_{\sigma_i} = \lambda_{max} \times \frac{\sum_{i=1}^{isize} h(\sigma_i) - h(\sigma_i)}{\sum_{i=1}^{isize} h(\sigma_i)}, \quad (2)$$
$$\lambda_{\sigma_i} = \lambda_{min} + (\lambda_{max} - \lambda_{min}) \times \frac{h_{max} - h(\sigma_i) + \epsilon}{h_{max} - h_{min} + \epsilon}, \quad (3)$$

and

$$\lambda_{\sigma_i} = \lambda_{min} \times b^{\frac{h_{max}-h(\sigma_i)+\epsilon}{h_{max}-h_{min}+\epsilon}}, \tag{4}$$

where $h_{min}(h_{max}) = min\ (max)\ \{h(\sigma_i), \sigma_i \in \Omega, i = 1, 2, \dots, isize\}$, Ω presents a set of initial generated solutions in the WWOA, α denotes the wavelength reduction coefficient, $b = \lambda_{max}/\lambda_{min}$, and *isize* records the number of waves in Ω. λ_{max} and λ_{min} denote the maximum and minimum allowable wavelengths, respectively, and ϵ is an extremely small number with $0 < \epsilon < 1$. In particular, $\lambda_{max} = \theta \times n$, $\lambda_{min} = 1$, α=1.0026, and $0 < \theta < 1$ is a controllable number in the study according to the setting in Zheng (2015), Zheng *et al*. (2019), and Zhao *et al*. (2018).

We run four WWOAs by adopting the swapping propagation operation to interchange the p^{th} position job and the q^{th} position job in σ_i *u* times, where integers p and q ($p < q$) are selected randomly from a discrete uniform distribution over 1 and *n* (or U(1, *n*)), and integer *u* is randomly generated from U(1, λ_{σ_i}). For simplification, the WWOA algorithms with wavelength formulations (1) to (4) are named WWOA1, WWOA2, WWOA3, and WWOA4, respectively. Two more parameters, *iteration* and *Nb,* are recorded as the total number of cycles required to perform the WWOAs and the improved number after the end of running the propagation operations of the WWOA. The procedures of WWOA are discussed as follows.

<u>The details of the proposed WWOA</u>:

Input *iteration* θ, *Nb*, and *isize*;
Set α=1.0026, $b = \lambda_{max}/\lambda_{min}$, $\lambda_{min} = 1, \epsilon = 10^{-8}$;
Generate a group of initial waves $\{\sigma_i$, *i*=1, …, *isize*$\}$ and compute each objective function value $h(\sigma_i)$, *i*=1, …, *isize*;
Find the best schedule σ^* with $h(\sigma^*) = min_{i\in\Omega}\{h(\sigma_i)\}$;
j=1;
Do while {*j* <= *iteration*}
  Find λ_{σ_i} by Formulas (1) or (2) or (3) or (4), $i \in \Omega$;
  i=1
  Do while { *i* <= *isize*}
    for each σ_i;
    Select a $u \in$ U[1, λ_{σ_i}] and apply a swapping propagation to σ_i after *u* times to obtain a new σ'_i;
    Determine if $h(\sigma'_i) < h(\sigma_i)$, replace σ_i by σ'_i and $h(\sigma_i)$ by $h(\sigma'_i)$;
    Determine if $h(\sigma_i) < h(\sigma^*)$, improve σ_i by a local search (e.g., pairwise interchange) method up to *Nb* times to find the final best one (say σ_i''), and replace σ^* by σ_i'';
    i=*i*+1
  End while;
j=*j*+1 and update the population;
End while;
Output the final solution σ^* and $h(\sigma^*)$.

5. Tuning the parameters in the WWOA

In addition to problem-specified heuristics, to acquire (near-)optimal job sequences for an NP-hard scheduling problem, researchers often use metaheuristics, such as, for example, genetic algorithms or simulated annealing. However, the water wave optimality algorithm is utilized in this study to produce high-quality (low value of $h(\sigma)$) job sequences. Parameter adjustment is an essential step before the intensive use of WWOA to obtain satisfactory job sequences. For parameter tests, the number of jobs is set at *n*=12 (100) for small (large) size problems. The processing times and setup times are generated randomly from a uniform distribution over integers 1 to 100 and over 1 to 20, respectively. The number of each test problem instance is 100. The average of error percentages (AEP) is AEP= $100[(A_k - B_k)/B_k][\%]$, where A_k (and B_k) is obtained from each algorithm (and the B&B method) and recorded for small-size problems. While there is no optimal solution and its corresponding objective value is available, the mean of objective values (MOV) is recorded for large-size problems.

5.1 Tuning parameters for the WWOA for a small-sized problem

To tune the parameters of the WWOA, *n* was fixed at 12 for small-size problems. In the exploration of the parameter *isize*, after several trials, other parameters were fixed as *lambda*=0.5×12, *iteration*=5×12, and *Nb* =5. The test values of the *isize* are from 5 to 30 by increasing one per time. In Figure 1(a), we can see that as *isize* increases, the number of AEPs decreases, and when *isize* is greater than approximately 26, the value of AEP is relatively stable, so we set *isize* to 30. When adjusting the lambda parameter, other parameters were fixed as follows: *isize*=30, *iteration*=5×12, and *Nb* =5. The test values of *lambda* ranged from 0.1×12 to 0.9×12 with an interval of 0.1×12. In Figure 1(b), it can be observed that as *lambda* increases, AEP gradually converges downward, and when *lambda* is greater than 0.8×12, the value of AEP gradually stabilizes, so lambda is set to 0.8×12. In the calibration of the parameter *iteration*, other parameters were fixed as *isize*=30, *lambda*=0.8×12, and *Nb* =5. The test values of *iteration* ranged from 5×12 to 25×12 by increasing one per iteration. In Fig. 1(c), we can see that as the *iteration* increases, the AEP gradually converges downward, and when the iteration is greater than approximately 22×12, the value of the AEP is relatively stable, so we set the *iteration* to 22×12. In the alignment of the *Nb* parameter, the other parameters are fixed at *isize*=30, *lambda*=0.8×12, and *iteration*=22×12. The test values of *Nb* ranged from 1 to 10, with an interval of 1. Fig. 1(d) shows that when *Nb* is greater than approximately 7, AEP generally converges, so *Nb* is set to 9. As obtained from the parameter exploration, the parameters for the small size job are decided as (*isize*, *lambda*, *iteration*, *Nb*) and (30, 0.8×12=9.6, 22×12=264, 9), respectively, for later simulation computation experiments.

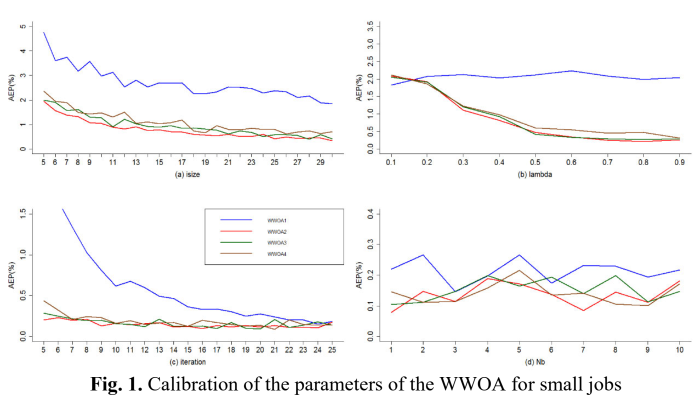

Fig. 1. Calibration of the parameters of the WWOA for small jobs

5.2 Tuning parameters for the WWOA for a large-sized problem

To tune the WWOA parameters, n was fixed at 100 for large problems. In the exploration of the parameter *isize*, other parameters were fixed as *lambda*=0.1×100, *iteration*=0.5×100, and *Nb* =1. The test values of *isize* ranged from 5 to 95, with a spacing of 5. In Fig. 2(a), we can see that as *isize* increases, the MOV (mean of objective values) converges downward, and when *isize* is greater than approximately 65, the value of MOV gradually stabilizes, so the selected value of *isize* is 65. In the adjustment of the lambda parameter, other parameters were fixed as *follows: isize*=65, *iteration*=0.5×100, and *Nb* =1. The test values of *lambda* ranged from 0.1×100 to 0.9×100, with an interval of 0.1×100. It is known that in Figure 2(b), as *lambda* increases, MOV converges downward, so lambda is set to 0.9×100. In the adjustment of the parameter *iteration*, other parameters were fixed as *isize*=65, *lambda*=0.9×100, and *Nb* =1. The test values of *iteration* ranged from 0.5×100 to 20×100 by incensement of 0.5×100 per trail. In Figure 2(c), we can see that as the *iteration* increases, the MOV converges downward, and when the *iteration* is greater than approximately 8×100, the value of the MOV gradually stabilizes, so the value of *iteration* is set at 8×100. In the adjustment of the parameter *Nb*, other parameters were fixed as *follows: isize*=65, *lambda*=0.9×100, and *iteration*=8×100. The test values of *Nb* ranged from 1 to 10 by an increment of one per trial. In Fig. 2(d), we can see that the MOV is stable for the test range, and *Nb* is finally selected to be 6. As obtained from the parameter exploration for the large size job, the parameters for the large size job are decided as (*isize*, *lambda*, *iteration*, *Nb*) and (65, 0.9×100=90, 8×100=800, 6), respectively, for subsequent computation experiments.

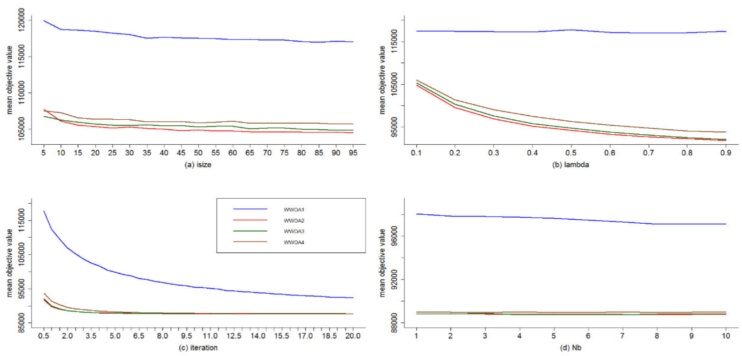

Fig. 2. Calibration of the parameters of the WWOA for a large job

6. Simulation report and statistical analysis

This section reports the results of carrying out two simulation sets to evaluate the performance of the proposed B&B method, 12 heuristics and four WWWO algorithms. All programs were coded in Fortran (Compaq Visual version 6.6) and executed on a PC with a 3.00-GHz Intel(R) Core(TM) i7-9700 CPU and 32.0-GB RAM in Windows 10. The test instances were generated by problem properties, i.e., the combinations of parameters, such as setup time, number of job classes (K) and number of orders (m). Regarding the set of simulations for a small number of jobs, the job size was set to n (=K*m) = 8, 10 and 12. We generated the job processing times from a uniform distribution U(1, 100). The due dates of the jobs following Fisher (1976) were generated from a uniform distribution $T \times U(1-\tau-R/2, 1+\tau+R/2)$, in which we let $T=\sum_{j=1}^{n} p_j$ be the sum of the processing times of the n jobs, and τ and R represent the tardiness factor and range of the due dates, respectively. Furthermore, the six cases of (τ, R) were (0.50, 0.25), (0.5, 0.50), (0.25, 0.25), (0.25, 0.50), (0.25, 0.75), and (0.25, 0.75). There are three different types of weights (α, β, γ), which were set to (1:1:1), (1:2:3), and (3:2:1). Regarding the setup times, there were two types: one followed U(1, 10), and the other followed U(1, 20). In addition, for each n, the combination of the number of orders and the number of classes (mo*mk) was set differently. There were 2, 2, and 4 combinations for n=8, 10, 12, respectively. A collection of 100 tested instances was generated randomly for each case of the aforementioned parameters. Consequently, 28,800 (=$\sum_n(\tau * R * \text{weight} * (\text{mo} * \text{mk}) * \text{setup} * 100)$= (2*3*3*2*2 + 2*3*3*2*2 + 2*3*3*4*2) *100) problem instances were tested. In addition, as the number of searched nodes exceeded 10^9, the B&B method jumped to the next set of data. Pertaining to the set of simulations for large numbers of jobs, the same parameter combination was used as the small number of jobs, except for the numbers of jobs n (were set at 50 and 80) and the combination of the number of orders and number of classes ((mo*mk), 4 (8) combinations for n=50 (80), respectively). A collection of 100 problem instances was generated randomly for each combination. There were 43,200 (=$\sum_n(\tau * R * \text{weight} * (\text{mo} * \text{mk}) * \text{setup} * 100)$ = (2*3*3*4*2 + 2*3*3*8*2) *100) problem instances that were tested for a large number of jobs.

6.1 Small-size simulation results

In this section, the criterion average error percentage (abbr. AEP) assesses the power of searching (near) optimal solutions for the proposed 12 heuristics and the four WWOAs (WWOA1, WWOA2, WWOA3, WWOA4). The AEP is defined as $100[(A_k-B_k)/B_k][\%]$, where A_k is received from each heuristic/algorithm and B_k is received from the B&B method. Regarding the obtained results of the B&B method, Table 1 delivers its performance. The mean of the nodes increased as *the number of* jobs increased, as displayed in Column 3 of Table 1. In addition, most generated test instances obtained the solution within 10^9 nodes. The average CPU times (in seconds) presented in Table 1 increased trivially as the number of jobs increased.

Table 1
Performance of the branch-and-bound method

n	τ	node	CPU time	total
8	0.25	13345.3	0.05	1800
	0.5	13667.6	0.06	1800
10	0.25	857868.3	3.74	1800
	0.5	897994.7	3.97	1800
12	0.25	68567401.8	411.50	3600
	0.5	74437881.8	444.20	3600
	R			
8	0.25	13683.2	0.06	1200
	0.5	13325.4	0.05	1200
	0.75	13510.8	0.06	1200
10	0.25	886928.2	3.86	1200
	0.5	882504.5	3.92	1200
	0.75	864361.8	3.80	1200
12	0.25	72602054.6	436.80	2400
	0.5	71277073.4	424.50	2400
	0.75	70628797.4	422.30	2400
	weight	node	CPU time	total

8	1:1:1	13913.2	0.06	1200
	1:2:3	8449.8	0.04	1200
	3:2:1	18156.3	0.07	1200
10	1:1:1	861976.0	3.92	1200
	1:2:3	394033.8	2.12	1200
	3:2:1	1377784.7	5.54	1200
12	1:1:1	67107408.3	422.40	2400
	1:2:3	22539483.9	170.80	2400
	3:2:1	124861033.3	690.30	2400
	mo×mk			
8	2×4	17125.5	0.06	1800
	4×2	9887.4	0.05	1800
10	2×5	1236287.3	4.90	1800
	5×2	519575.7	2.81	1800
12	2×6	128840598.9	660.80	1800
	3×4	72676883.8	445.60	1800
	4×3	46571209.8	323.90	1800
	6×2	37921874.8	281.20	1800
	setup			
8	U(1,10)	13446.6	0.05	1800
	U(1,20)	13566.3	0.06	1800
10	U(1,10)	871116.0	3.89	1800
	U(1,20)	884747.0	3.83	1800
12	U(1,10)	71899465.7	432.70	3600
	U(1,20)	71105817.9	423.00	3600
	mean	35974180.4	214.91	

Table 2 summarizes the CPU time and number of node results of the B&B for each of the Factors *n, τ*, *R*, weight, and setup. Moreover, for the summaries of (mo*mk), see Table 1. The mean nodes and mean CPU times grow dramatically as the number of jobs (*n*) increases, which is one property of NP-hard problems.

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

n	node	CPU	*R*	node	CPU
8	13506.44	0.06	0.25	36526180.13	219.36
10	877931.49	3.86	0.5	35862494.15	213.25
12	71502641.79	427.87	0.75	35533866.85	212.13
mean	35974180.38	214.91	mean	35974180.38	214.91
τ			setup		
0.25	34501504.30	206.72	U(1,10)	36170873.49	217.36
0.5	37446856.46	223.11	U(1,20)	35777487.26	212.47
mean	35974180.38	214.91	mean	35974180.38	214.91
weight					
1:1:1	33772676.44	212.22			
1:2:3	11370362.82	85.95			
3:2:1	62779501.88	346.57			
mean	35974180.38	214.91			

The performances in terms of the AEPs of the proposed 12 heuristics and four WWO algorithms are presented in Table 3. Overall, the means of AEP are 0.16, 0.13, 0.13, and 0.13 for WWOA1, WWOA 2, WWOA3, and WWOA 4, respectively, and the group of heuristics that has the largest AEP is (OES_b, OES_f, OES_p) with AEP values of (20.01, 19.22, 18.01) for a small-size number of job cases (*n*=8, 10, 12). The WWO algorithms perform better than all of the other 12 problem-based heuristics.

Table 3
AEP of 12 heuristics and 4 WWO algorithms for small *n*=8, 10, 12

n	*τ*	CSL_b	CSL_f	CSL_p	OEL_b	OEL_f	OEL_p	OES_b	OES_f	OES_p	OSL_b	OSL_f	OSL_p	WWOA1	WWOA2	WWOA3	WWOA4
8	0.25	19.80	13.31	14.71	17.54	16.22	15.33	13.60	13.00	12.62	6.17	6.07	5.35	0.05	0.04	0.04	0.04
	0.5	18.97	12.88	14.21	17.35	16.07	15.22	13.45	12.82	12.53	5.94	5.87	5.19	0.04	0.03	0.03	0.04
10	0.25	21.94	15.64	17.18	20.79	19.82	18.49	15.16	14.72	14.30	8.39	8.36	7.27	0.15	0.13	0.13	0.12
	0.5	21.49	15.32	16.73	20.14	19.24	17.93	14.67	14.30	13.88	7.96	7.97	6.92	0.14	0.13	0.13	0.13
12	0.25	24.06	19.66	19.40	21.32	20.86	19.52	15.41	15.46	14.94	9.34	9.43	8.36	0.25	0.18	0.17	0.18
	0.5	23.54	19.25	18.97	20.81	20.36	19.05	15.10	15.17	14.63	8.92	9.00	8.00	0.22	0.17	0.16	0.17
	R																
8	0.25	19.52	13.21	14.56	17.83	16.55	15.72	13.84	13.25	12.92	6.13	6.07	5.37	0.05	0.03	0.03	0.04
	0.5	19.27	13.12	14.36	17.30	16.01	15.09	13.42	12.76	12.42	6.03	5.91	5.21	0.04	0.03	0.04	0.04
	0.75	19.36	12.96	14.45	17.20	15.87	15.01	13.33	12.70	12.38	6.01	5.93	5.23	0.04	0.04	0.04	0.05
10	0.25	21.74	15.70	17.10	20.29	19.40	18.05	14.78	14.42	14.01	8.08	8.01	7.01	0.14	0.13	0.13	0.12
	0.5	21.82	15.43	16.96	20.72	19.72	18.43	15.12	14.64	14.21	8.22	8.21	7.11	0.15	0.12	0.12	0.12
	0.75	21.59	15.30	16.81	20.38	19.48	18.16	14.83	14.48	14.04	8.22	8.27	7.17	0.15	0.13	0.13	0.14
12	0.25	23.84	19.52	19.23	21.21	20.72	19.40	15.39	15.43	14.92	9.06	9.15	8.13	0.24	0.17	0.17	0.18
	0.5	23.81	19.47	19.21	21.07	20.60	19.29	15.22	15.27	14.73	9.19	9.25	8.22	0.24	0.18	0.17	0.18
	0.75	23.74	19.38	19.12	20.92	20.49	19.16	15.16	15.25	14.70	9.13	9.25	8.19	0.22	0.16	0.17	0.17
	weight																
8	1:1:1	16.98	10.00	11.90	17.68	15.37	15.00	13.27	11.73	11.82	5.54	4.92	4.48	0.02	0.01	0.02	0.02
	1:2:3	11.72	6.06	8.18	22.26	21.27	19.00	13.53	13.00	12.15	10.63	11.18	9.68	0.00	0.00	0.00	0.00
	3:2:1	29.44	23.23	23.29	12.39	11.80	11.82	13.79	13.99	13.75	1.99	1.80	1.65	0.10	0.09	0.09	0.11

10	1:1:1	18.96	12.09	14.22	20.73	18.97	18.07	14.60	13.44	13.37	7.94	7.34	6.50	0.07	0.04	0.05	0.05
	1:2:3	13.50	8.07	10.20	25.67	25.28	22.39	15.44	15.41	14.34	13.32	14.08	12.08	0.01	0.01	0.01	0.01
	3:2:1	32.69	26.27	26.44	14.99	14.35	14.17	14.70	14.69	14.56	3.26	3.07	2.70	0.36	0.33	0.33	0.32
12	1:1:1	19.83	15.06	15.45	20.98	19.83	18.98	14.65	14.00	13.85	8.63	8.23	7.47	0.14	0.09	0.09	0.10
	1:2:3	14.05	9.63	10.81	27.48	27.77	24.78	17.19	17.74	16.55	14.94	15.89	13.76	0.04	0.02	0.02	0.02
	3:2:1	37.51	33.68	31.30	14.73	14.22	14.09	13.92	14.20	13.96	3.82	3.53	3.32	0.52	0.41	0.40	0.42
	mo*mk																
8	2*4	11.25	11.31	8.60	16.20	15.40	14.45	8.03	8.63	8.14	9.65	9.18	8.07	0.00	0.00	0.00	0.00
	4*2	27.51	14.88	20.32	18.69	16.89	16.10	19.03	17.18	17.01	2.46	2.76	2.47	0.08	0.07	0.07	0.08
10	2*5	11.23	10.58	8.64	19.62	19.12	17.45	8.21	8.80	8.35	13.34	13.06	11.36	0.01	0.01	0.01	0.01
	5*2	32.20	20.38	25.27	21.30	19.95	18.98	21.61	20.22	19.83	3.01	3.27	2.83	0.28	0.24	0.25	0.24
12	2*6	10.93	9.93	8.51	22.72	22.40	20.37	8.48	9.07	8.61	16.91	16.73	14.75	0.04	0.02	0.02	0.02
	3*4	20.64	18.61	16.69	18.97	18.84	17.58	12.43	13.02	12.47	9.55	9.67	8.75	0.07	0.04	0.04	0.04
	4*3	27.52	24.25	21.95	19.19	18.89	17.80	16.34	16.56	15.97	6.47	6.64	5.96	0.20	0.13	0.12	0.14
	6*2	36.10	25.04	29.60	23.39	22.30	21.39	23.76	22.61	22.09	3.58	3.83	3.26	0.63	0.51	0.49	0.50
	setup																
8	U(1,10)	20.55	14.14	14.84	17.26	15.83	15.03	13.35	12.58	12.48	5.67	5.52	4.79	0.03	0.02	0.03	0.03
	U(1,20)	18.21	12.05	14.07	17.63	16.46	15.52	13.71	13.23	12.67	6.44	6.42	5.75	0.05	0.04	0.04	0.05
1	U(1,10)	23.02	16.76	17.47	20.04	19.01	17.80	14.34	13.88	13.70	7.73	7.68	6.63	0.12	0.11	0.11	0.11
0	U(1,20)	20.41	14.19	16.44	20.88	20.05	18.62	15.48	15.14	14.48	8.62	8.66	7.57	0.17	0.14	0.14	0.14
1	U(1,10)	25.33	21.10	20.08	20.59	20.00	18.78	14.70	14.64	14.35	8.36	8.41	7.46	0.19	0.14	0.13	0.15
2	U(1,20)	22.27	17.81	18.29	21.54	21.21	19.79	15.81	15.99	15.22	9.89	10.02	8.90	0.28	0.21	0.21	0.21
	mean	22.17	16.87	17.45	20.01	19.22	18.01	14.74	14.51	14.06	8.12	8.14	7.18	0.16	0.13	0.13	0.13

Table 4 summarizes the results in terms of AEP for levels of the Factors *n*, *τ*, *R*, weight, and setup; for (mo×mk), see Table 3. It can be observed that the AEP of all algorithms increases slightly as *n* increases from 8 to 12. Furthermore, Fig. 3 displays the violin plots of AEP (distributions) for the 12 heuristics and four WWO algorithms. Regarding the CPU times, 16 heuristics/algorithms cost less than 1 second.

Table 4
Summary of AEP of 12 heuristics and 4 WWO algorithms for small *n*

n	CSL_b	CSL_f	CSL_p	OEL_b	OEL_f	OEL_p	OES_b	OES_f	OES_p	OSL_b	OSL_f	OSL_p	WWOA1	WWOA2	WWOA3	WWOA4
8	19.38	13.09	14.46	17.44	16.15	15.27	13.53	12.91	12.58	6.06	5.97	5.27	0.04	0.03	0.04	0.04
10	21.72	15.48	16.95	20.46	19.53	18.21	14.91	14.51	14.09	8.17	8.17	7.10	0.14	0.13	0.13	0.13
12	23.80	19.46	19.19	21.07	20.61	19.28	15.25	15.31	14.79	9.13	9.21	8.18	0.23	0.17	0.17	0.18
mean	22.17	16.87	17.45	20.01	19.22	18.01	14.74	14.51	14.06	8.12	8.14	7.18	0.16	0.13	0.13	0.13
tao																
0.25	22.46	17.07	17.67	20.24	19.44	18.21	14.89	14.66	14.20	8.31	8.33	7.33	0.17	0.13	0.13	0.13
0.5	21.89	16.68	17.22	19.77	19.01	17.81	14.58	14.36	13.92	7.93	7.96	7.03	0.16	0.12	0.12	0.13
mean	22.17	16.87	17.45	20.01	19.22	18.01	14.74	14.51	14.06	8.12	8.14	7.18	0.16	0.13	0.13	0.13
R																
0.25	22.24	16.98	17.53	20.14	19.35	18.14	14.85	14.63	14.20	8.08	8.09	7.16	0.17	0.13	0.12	0.13
0.5	22.18	16.87	17.43	20.04	19.24	18.02	14.74	14.49	14.02	8.16	8.16	7.19	0.17	0.13	0.13	0.13
0.75	22.11	16.75	17.38	19.86	19.08	17.87	14.62	14.42	13.96	8.12	8.17	7.19	0.16	0.12	0.13	0.13
mean	22.17	16.87	17.45	20.01	19.22	18.01	14.74	14.51	14.06	8.12	8.14	7.18	0.16	0.13	0.13	0.13
weight																
1:1:1	18.90	13.05	14.25	20.09	18.50	17.76	14.29	13.29	13.22	7.68	7.18	6.48	0.09	0.06	0.06	0.06
1:2:3	13.33	8.35	10.00	25.72	25.52	22.74	15.84	15.97	14.90	13.46	14.26	12.32	0.02	0.01	0.01	0.01
3:2:1	34.29	29.21	28.08	14.21	13.65	13.54	14.08	14.27	14.06	3.22	2.98	2.75	0.38	0.31	0.30	0.32
mean	22.17	16.87	17.45	20.01	19.22	18.01	14.74	14.51	14.06	8.12	8.14	7.18	0.16	0.13	0.13	0.13
setup																
(1,10)	23.56	18.28	18.12	19.62	18.71	17.60	14.27	13.94	13.72	7.53	7.51	6.58	0.13	0.10	0.10	0.11
(1,20)	20.79	15.47	16.77	20.40	19.73	18.43	15.20	15.09	14.40	8.71	8.78	7.78	0.19	0.15	0.15	0.15
mean	22.17	16.87	17.45	20.01	19.22	18.01	14.74	14.51	14.06	8.12	8.14	7.18	0.16	0.13	0.13	0.13

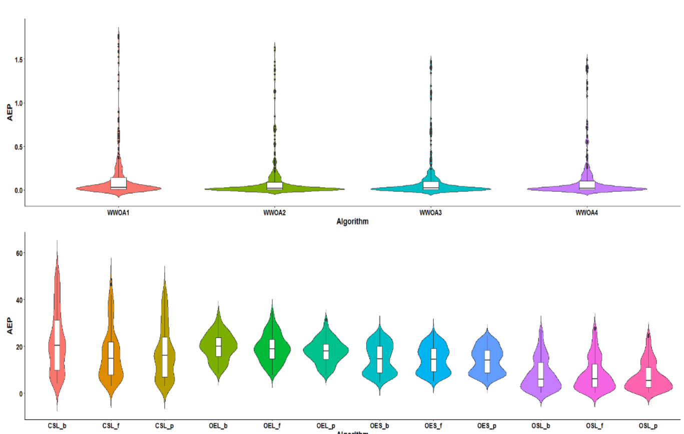

Fig. 3. Violin plots of the AEP distribution for 12 heuristics and 4 WWO algorithms

Statistical evidence to verify whether the differences among the 12 heuristics and four WWOA algorithms are important is discussed in the following. First, an analysis of a variance (ANOVA) method and a linear model on AEPs were executed in SAS (version 9.4). However, the normality hypothesis was violated according to the Kolmogorov-Smirnov normality test (with a p-value that is smaller than 0.01, and the value of the D statistic is 0.0706). Therefore, on the significance level of α=0.05, the Freidman test (with p-value <0.0001 and with value of the chi-square statistic equal to 3006.5 and degrees of freedom 15) was executed and affirmed that AEPs are from different distributions based on the AEP ranks obtained on the 288 (n×τ×R×weight×(mo×mk)×setup) blocks of test instances.

The WNMT (or Wilcoxon–Nemenyi–McDonald–Thompson) was utilized to determine 120 pairwise differences among the 12 heuristics and four WWOAs (see Holland *et al*., 2014). Table 5 (Column 2) reveals the obtained results of the AEP's rank across the 288 blocks for the 12 heuristics and four WWO algorithms. The rank sums of WWOA1 to WWOA4 are 997.0, 661.0, 678.0, and 700.0, respectively. There is no statistically significant difference among the four WWO algorithms at a significance level of 0.05, but the WWOA group is considerably better than any of the 12 heuristics. WWOA2 has the smallest AEP and rank sum for a small number of jobs.

Table 5

Rank sums of AEP and RPD for 12 heuristics and 4 WWO algorithms to make multiple comparisons.

Algorithm	Small n (288 blocks) rank-sum	Large n (432 blocks) rank-sum	Algorithm	Small n (288 blocks) rank-sum	Large n (432 blocks) rank-sum
CSL_b	3715.0	4966.0	OES_p	2751.5	3785.0
CSL_f	2960.0	4997.0	OSL_b	2271.5	4840.0
CSL_p	3015.0	3857.0	OSL_f	2363.0	4084.0
OEL_b	4010.0	5370.0	OSL_p	1943.0	3091.0
OEL_f	3692.0	5683.0	WWOA1	997.0	2324.0
OEL_p	3374.0	4432.0	WWOA1	661.0	697.0
OES_b	3059.0	4363.0	WWOA1	678.0	721.0
OES_f	2978.0	4229.0	WWOA1	700.0	1313.0

1.As |pairwise rank-sum difference| > 383.0*(small n), 469.1*(large n) indicates that the compared heuristic/algorithm is significant at 0.05.
2.The approximate values marked in * are calculated from a formula in Holland et al. (2014), Page 316.

6.2 Large size simulation results

With regard to large-size problems, a collection of one hundred tested instances were randomly generated for each case of parameters (n, τ, R, weight, mo×mk, and setup); in total, 43,200 problem instances were tested. The number of jobs was fixed at n = 50 and 80. Due to the complexity of this problem, especially for the large size instances, the criterion (or relative percent deviation RPD, defined as RPD $=100[(A_k - B^*)/B^*][\%]$) is applied to assess the relative behavior of the 12 heuristics and four WWOAs, where A_k is received from each algorithm and B^* is the best founded value among A_ks, all the proposed 12 heuristics, and four WWOAs. Table 6 exhibits the average RPDs for the four WWOAs and 12 heuristics. Overall, in Table 6, WWOA1-WWOA4 produced RPD values 8.06, 0.37, 0.37 and 0.51, respectively, while the 12 heuristics produced RPD values from 22.60 to 33.42. Table 7 summarizes the average RPD for different levels of the Factors n, τ, R, weight, and setup; for (mo×mk), see Table 6.

Table 6

RPD of 12 heuristics and 4 WWO algorithms for large n=50, 80

n	τ	CSL_b	CSL_f	CSL_p	OEL_b	OEL_f	OEL_p	OES_b	OES_f	OES_p	OSL_b	OSL_f	OSL_p	WWOA1	WWOA2	WWOA3	WWOA4
50	0.25	31.03	29.75	24.01	33.06	32.82	26.42	27.63	22.56	21.45	27.14	21.67	19.80	6.69	0.41	0.42	0.53
	0.5	30.50	29.41	23.74	32.57	32.91	26.12	27.19	22.35	21.19	26.91	21.51	19.60	6.60	0.40	0.42	0.53
80	0.25	30.96	30.10	24.99	33.84	33.64	27.50	29.03	24.22	23.11	27.93	23.19	21.57	8.79	0.34	0.34	0.48
	0.5	30.84	29.80	24.75	33.60	33.19	27.29	28.83	24.07	22.93	27.71	23.02	21.39	8.75	0.37	0.37	0.52
	R																
50	0.25	30.76	29.52	23.81	32.85	32.41	26.29	27.42	22.37	21.33	26.94	21.48	19.68	6.62	0.39	0.41	0.53
	0.5	30.68	29.52	23.85	32.73	33.01	26.16	27.33	22.43	21.20	27.04	21.65	19.66	6.67	0.43	0.41	0.54
	0.75	30.85	29.70	23.97	32.86	33.17	26.37	27.48	22.55	21.43	27.10	21.64	19.76	6.65	0.40	0.43	0.53
80	0.25	30.91	29.90	24.81	33.69	33.20	27.38	28.90	24.14	22.99	27.80	23.09	21.45	8.75	0.33	0.34	0.48
	0.5	30.77	29.98	24.90	33.81	33.59	27.48	29.03	24.22	23.11	27.85	23.13	21.53	8.82	0.38	0.38	0.54
	0.75	31.01	29.98	24.91	33.65	33.45	27.33	28.86	24.09	22.94	27.80	23.10	21.46	8.73	0.35	0.34	0.48
	weight																
50	1:1:1	26.35	27.63	21.82	30.93	33.63	26.94	25.33	22.50	21.57	27.64	21.70	20.22	5.89	0.22	0.23	0.33
	1:2:3	20.93	29.69	22.66	32.47	41.75	34.88	26.08	29.60	29.24	35.75	29.66	28.02	4.65	0.14	0.15	0.22
	3:2:1	45.02	31.42	27.14	35.05	23.21	16.99	30.82	15.26	13.16	17.70	13.42	10.87	9.39	0.85	0.88	1.04
80	1:1:1	26.45	28.08	22.90	31.78	33.90	28.17	26.84	24.50	23.47	28.57	23.48	22.17	7.79	0.20	0.20	0.32
	1:2:3	21.50	30.87	24.76	34.33	44.08	37.29	28.72	33.11	32.43	37.70	32.59	30.96	6.38	0.14	0.15	0.22
	3:2:1	44.74	30.90	26.95	35.06	22.27	16.73	31.23	14.83	13.15	17.18	13.24	11.31	12.13	0.71	0.71	0.95
	mo×mk																
50	2×25	6.91	25.83	9.82	26.78	45.32	43.41	11.94	15.33	29.52	44.29	28.56	41.01	3.05	0.08	0.08	0.10
	5×10	20.79	23.11	17.72	26.50	29.67	25.06	21.26	19.83	20.51	25.45	20.73	20.78	6.77	0.28	0.29	0.46
	10×5	38.83	29.19	27.33	33.52	25.36	18.10	31.59	22.13	16.63	18.74	17.36	11.49	8.89	0.32	0.33	0.52
	25×2	56.54	40.19	40.64	44.46	31.11	18.52	44.85	32.52	18.63	19.64	19.71	5.51	7.87	0.94	0.96	1.04
80	2×40	6.57	28.99	11.08	29.67	51.74	50.19	13.25	18.03	34.48	50.78	33.10	48.23	3.79	0.07	0.07	0.09
	4×20	14.23	24.78	16.49	27.88	38.73	35.54	19.97	23.20	28.28	35.89	28.15	32.54	6.53	0.13	0.14	0.23
	5×16	17.01	23.94	17.55	27.51	34.89	31.19	21.40	23.06	25.66	31.38	25.57	27.63	7.24	0.19	0.19	0.30
	8×10	25.08	24.56	20.96	28.75	28.89	23.87	25.28	22.46	20.89	24.16	21.15	19.23	9.19	0.30	0.30	0.52
	10×8	30.48	26.37	23.70	30.67	27.34	21.61	28.09	22.66	19.45	21.98	19.84	16.33	10.30	0.30	0.31	0.56
	16×5	43.80	32.01	30.78	36.98	26.25	18.73	35.74	24.19	17.80	19.06	18.18	11.53	11.54	0.32	0.32	0.53
	20×4	49.42	35.32	34.57	40.19	26.59	18.01	39.42	25.40	17.47	18.43	17.93	9.75	11.60	0.38	0.38	0.56
	40×2	60.57	43.65	43.84	48.12	32.90	20.02	48.31	34.17	20.09	20.88	20.94	6.60	9.97	1.12	1.11	1.21
	setup																
50	U(1,10)	32.78	32.27	20.62	31.05	31.32	18.78	30.83	20.75	19.01	32.05	20.94	18.16	6.18	0.44	0.45	0.54
	U(1,20)	28.75	26.89	27.13	34.58	34.40	33.77	23.99	24.15	23.63	22.01	22.24	21.24	7.11	0.37	0.38	0.52
80	U(1,10)	32.96	31.99	21.87	32.01	31.37	19.77	31.64	21.91	19.99	31.74	22.11	19.46	7.86	0.36	0.36	0.47
	U(1,20)	28.83	27.92	27.87	35.43	35.46	35.02	26.23	26.38	26.04	23.90	24.10	23.50	9.68	0.34	0.35	0.53
	mean	30.85	29.83	24.54	33.42	33.23	27.02	28.42	23.58	22.45	27.56	22.60	20.89	8.06	0.37	0.37	0.51

Table 7
Summary of AEP of 12 heuristics and 4 WWO algorithms for large *n*

n	CSL_b	CSL_f	CSL_p	OEL_b	OEL_f	OEL_p	OES_b	OES_f	OES_p	OSL_b	OSL_f	OSL_p	WWOA1	WWOA2	WWOA3	WWOA4
50	30.77	29.58	23.88	32.82	32.86	26.27	27.41	22.45	21.32	27.03	21.59	19.70	6.65	0.41	0.42	0.53
80	30.90	29.95	24.87	33.72	33.42	27.40	28.93	24.15	23.02	27.82	23.11	21.48	8.77	0.35	0.35	0.50
τ																
0.25	30.98	29.99	24.67	33.58	33.37	27.14	28.56	23.67	22.55	27.67	22.68	20.98	8.09	0.36	0.36	0.50
0.5	30.72	29.67	24.41	33.26	33.09	26.90	28.28	23.49	22.35	27.44	22.52	20.79	8.03	0.38	0.38	0.52
R																
0.25	30.86	29.77	24.48	33.41	32.94	27.02	28.41	23.55	22.44	27.52	22.56	20.86	8.04	0.35	0.36	0.50
0.5	30.74	29.83	24.55	33.45	33.40	27.04	28.46	23.62	22.48	27.58	22.63	20.91	8.10	0.40	0.39	0.54
0.75	30.96	29.89	24.59	33.39	33.36	27.01	28.40	23.57	22.44	27.57	22.61	20.89	8.04	0.36	0.37	0.50
weight																
1:1:1	26.41	27.93	22.54	31.49	33.81	27.76	26.34	23.83	22.84	28.26	22.89	21.52	7.16	0.21	0.21	0.33
1:2:3	21.31	30.48	24.06	33.71	43.30	36.49	27.84	31.94	31.36	37.05	31.61	29.98	5.81	0.14	0.15	0.22
3:2:1	44.83	31.07	27.01	35.05	22.58	16.82	31.10	14.97	13.15	17.35	13.30	11.16	11.22	0.76	0.76	0.98
Setup																
U(1,10)	32.90	32.08	21.45	31.69	31.36	19.44	31.37	21.52	19.67	31.84	21.72	19.03	7.30	0.39	0.39	0.49
U(1,20)	28.80	27.57	27.63	35.15	35.10	34.60	25.48	25.64	25.24	23.27	23.48	22.74	8.82	0.35	0.36	0.53
mean	30.85	29.83	24.54	33.42	33.23	27.02	28.42	23.58	22.45	27.56	22.60	20.89	8.06	0.37	0.37	0.51

To verify whether the differences among the 12 heuristics and four WWOA algorithms are statistically significant, first, an analysis of a variance (ANOVA) method on RPDs was executed. However, the Kolmogorov-Smirnov normality test confirmed that the validity of the normality hypothesis was also violated for the linear model based on the fact that its p-value was smaller than 0.01 and the value of the D statistic was 0.0891. Therefore, the Freidman test was executed based on ranks of RPD on the 432 ($n \times \tau \times R \times$ weight×(mo×mk)×setup) test instances. We observed that the p-value < 0.0001 (with a value of the chi-square statistic of 3940.2 and degrees of freedom of 15). The obtained results confirm that RPD does not come from the same distribution at the level of significance of 0.05.

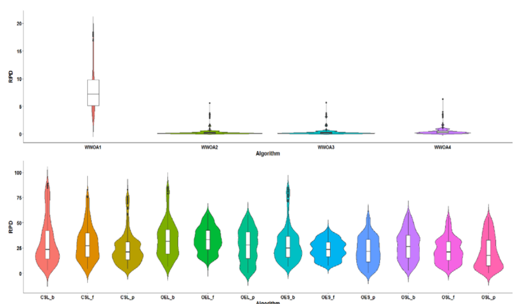

Fig. 4. Boxplots of the RPD distribution for 12 heuristics and 4 WWO algorithms

Furthermore, to make 120 pairwise differences among the 12 heuristics and four WWO algorithms, the WNMT procedure was employed. Table 5 (Column 3) reveals sums of the RPD ranks across the 432 blocks for the 12 heuristics and four WWO algorithms. The rank sums of WWOA1 to WWOA4 are 2324.0, 697.0, 721.0, and 1313.0, respectively. As presented in Table 5 (Column 3), WWOA2 and WWOA3 are positioned in the best obtained results group, while OEL_b and OEL_f, with rank sums of 5370.0 and 5683.0, are situated in the worst obtained results group. Moreover, any one of the WWOAs is pairwise considerably better than any of the 12 heuristics. WWOA2 has the smallest RPD and rank sum for a large number of jobs. In addition, Figure 4 shows the violin plots (distributions) of RPD for the 12 heuristics and four WWOAs.

7. Conclusion

The customer order scheduling problem is used to satisfy the demand of customers who order many kinds of products. From a time aspect, it is necessary to ship finished products to customers at an early date. However, processing various jobs in succession on one machine requires a setup time if the job classes are different. In this study, we discuss the customer order scheduling problem with jobs belonging to different classes. To balance the costs incurred by holding the finished jobs (products), tardiness, and total production completion time, the aim is to find an optimal schedule of jobs for an objective function that is a linear combination of holding cost, total tardiness, and total completion time. The branch-and-bound method with a dominance property is employed to produce solutions for a small number of jobs; it performs well up to n=12 jobs, which consist of different combinations of the number of job classes and number of customers. For a relatively large number of jobs, say n=50, 80, four problem-based heuristics, each along with three local improved searching methods, and a water wave optimality algorithm with four variants of wavelengths, are proposed for finding near-optimal solutions. The water wave optimality with the second wavelength formula (WWOA2) performed best among all 16 heuristics or algorithms regardless of the number of jobs. For future research work on this topic, we can consider multiple machine manufacturing processes with multiple job classes and customer orders. Moreover, multiple machines may be arranged in parallel or in flow-shop environments. The corresponding author will provide the Fortran programming codes as well as relative datasets upon request.

Acknowledgments

We would like to thank the editors and referees for their positive comments and useful suggestions. This paper was supported in part by the Ministry of Science and Technology of Taiwan, MOST 110-2221-E-035-082- MY2, and in part by the National Natural Science Foundation of China, grant number 72271048.

References

Allahverdi, A., & Soroush, H. M. (2008). The significance of reducing setup times/setup costs. *European Journal of Operational Research*, *187*(3), 978-984.

Allahverdi, A., Ng, C. T., Cheng, T. E., & Kovalyov, M. Y. (2008). A survey of scheduling problems with setup times or costs. *European journal of operational research*, *187*(3), 985-1032.

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

Allahverdi, A., Gupta, J. N., & Aldowaisan, T. (1999). A review of scheduling research involving setup considerations. *Omega*, *27*(2), 219-239.
Cheng, T.C.E., Gupta, J.N.D., & Wang, G. (2000). A review of flowshop scheduling research with setup times. *Production and Operations Management, 9*(3), 262–282.
de Athayde Prata, B., Rodrigues, C. D., & Framinan, J. M. (2021a). Customer order scheduling problem to minimize makespan with sequence-dependent setup times. *Computers & Industrial Engineering, 151*, 106962.
de Athayde Prata, B., Rodrigues, C. D., & Framinan, J. M. (2021b). A differential evolution algorithm for the customer order scheduling problem with sequence-dependent setup times. *Expert Systems with Applications,* 116097.
Della Croce, F., Narayan, V., & Tadei, R. (1996). The two-machine total completion time flow shop problem. *European Journal of Operational Research, 90*, 227–237.
Erel, E., & Ghosh, J. B. (2007). Customer order scheduling on a single machine with family setup times: Complexity and algorithms. *Applied Mathematics and Computation, 185*, 11–18.
Fisher, M. L. (1976). A dual algorithm for the one-machine scheduling problem. *Mathematical Programming, 11*(1), 229–251.
Framinan, J. M., & Perez-Gonzalez, P. (2018). Order scheduling with tardiness objective: Improved approximate solutions. *European Journal of Operational Research, 266*(3), 840–850.
Gupta, J. N., Ho, J. C., & van der Veen, J. A. (1997). Single machine hierarchical scheduling with customer orders and multiple job classes. *Annals of Operations Research, 70*, 127–143.
Hazır, Ö., Günalay, Y., & Erel, E. (2008). Customer order scheduling problem: a comparative metaheuristics study. *The International Journal of Advanced Manufacturing Technology, 37*(5-6), 589–598.
Holland, M. D., Wolfe, A., & Chicken, E. (2014). *Nonparametric Statistical Methods, 3rd edition*. Wiley, Hoboken.
Julien, F. M., & Magazine, M. J. (1990). Scheduling customer orders: An alternative production scheduling approach. *Journal of Manufacturing and Operations Management, 3*, 177–199.
Liaee, M. M., & Emmons, H. (1997). Scheduling families of jobs with setup times. *International Journal of Production Economics, 51*(3), 165–176.
Su, L. H., Chen, P. S., & Chen, S. Y. (2013). Scheduling on parallel machines to minimize maximum lateness for the customer order problem. *International journal of systems science, 44*(5), 926–936.
Wu, C. C., Lin, W. C., Zhang, X., Chung, I. H., Yang, T. H., & Lai, K. (2019). Tardiness minimisation for a customer order scheduling problem with sum-of-processing-time-based learning effect. *Journal of the Operational Research Society, 70*(3), 487–501.
Yang, W. H., & Liao, C.J. (1999). Survey of scheduling research involving setup times. *International Journal of Systems Science, 30*(2), 143–155.
Zhao, F., Liu, H., Zhang, Y., Mac, W., & Zhang, C. (2018) A discrete Water Wave Optimization algorithm for no-wait flow shop scheduling problem. *Expert Systems with Applications, 91*, 347–363.
Zheng, Y. J. (2015). Water wave optimization: A new nature-inspired metaheuristic. *Computers and Operations Research, 55*(1), 1–11.
Zheng, Y. J., Lu, X. Q., Du, Y. C., Xue, Y., & Sheng, W. G. (2019). Water wave optimization for combinatorial optimization: Design strategies and applications. *Applied Soft Computing*, *83*, 105611.