IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 8, 2022, accepted April 24, 2022, date of publication April 26, 2022, date of current version May 3, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3170734

Single Machine Scheduling Problem With a Weight-Modifying-Activity to Minimize the Total Weighted Completion Time

YARONG CHEN[1,2], ZAILIN GUAN[1], YA-CHIH TSAI[3], AND FUH-DER CHOU[2]
[1]College of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
[2]College of Mechanical and Electrical Engineering, Wenzhou University, Wenzhou, Zhejiang 325035, China
[3]Department of Hotel Management, Vanung University, Taoyuan 32061, Taiwan

Corresponding author: Fuh-Der Chou (fdchou@tpts7.seed.net.tw)

This work was supported by the National Natural Science Foundation of China under Grant 51705370.

ABSTRACT The single-machine scheduling problem with a weight-modifying-activity (WMA) to minimize the total weighted completion time was initially addressed by Mosheiov and Oron in 2020, where the activity was an option, and once the activity was performed, the weights of the subsequent jobs become decreased. This problem has proven to be NP-hard. Following their study, we propose two mixed integer linear programming models (model_1 and model_2). Based on some optimality properties, a heuristic algorithm with swap and insert procedures is developed. The computation results indicate that model_2 can optimally solve problems of up to 40 jobs efficiently, while the average relative percentage of error and hit rate of the proposed heuristic is 0.0005% and 98.2%, respectively. The influence of parameters, such as the number of jobs, the adjusted coefficient for the job weight, and the time of the WMA, on the performance of the proposed methods, are also analyzed.

INDEX TERMS Heuristic algorithm, mixed integer linear programming model, single-machine scheduling, weight-modifying activity.

I. INTRODUCTION

Single-machine scheduling problems (SMSPs) have been a very active topic in the field of operations research thus far. SMSPs can be regarded as a special case of other complicated machine configurations, and some properties derived by SMSPs can provide a basis for developing heuristic algorithms to solve more complicated scheduling problems (Pindo [1]). Substantial research has been carried out for classical SMSPs, which assume that a machine is available for processing jobs at all times. However, this assumption is not the case for many production operations. Strusevich and Rustogi [2] claimed that a decision-maker may implement a certain rate-modifying activity (RMA) on processing machines, which improves the efficiency of the system. An enhanced maintenance activity model has increasingly attracted many researchers in the last two decades. Lee and Leon [3], motivated by electronic assembly lines, were the first to study single-machine problems with an RMA. In this assembly line, the placement stage of a surface-mount technology (SMT) line is treated as the bottleneck stage, and a scheduler needs to decide whether to stop the machine or continue processing it at a lower production rate when malfunctions occur. For the problem with each objective of makespan, total (weighted) completion time, and maximum lateness, they proposed polynomial and pseudopolynomial algorithms. Mosheiov and Oron [4] considered a due-date assignment problem with an RMA and showed that the problem was solvable in polynomial time (O(n^4)). Following the study of Mosheiov and Oron [4], Gordon and Tarasevich [5] restricted the modifying rate to be less than or equal to 1 (i.e., $0 < \delta_j \leq 1$) and showed that the complexity of finding an optimal solution was O($n^3 l$). Yin *et al.* [6] extended the work of Mosheiov and Oren [4] to involve batching consideration to minimize the sum of earliness, tardiness, holding, common due date assignment, and batch delivery costs, which showed that some special cases were solvable in polynomial time. Mosheiov and Oron [7] considered the same problem as Lee and Leon [3] and adopted the objective of minimizing the total late work, a concept that was defined by Sterna [8]. They

The associate editor coordinating the review of this manuscript and approving it for publication was Bong Jun David Choi.

affirmed that even the special case of the scheduling problem without an RMA is NP-hard and provided a pseudopolynomial dynamic programming algorithm for the problem. Zhu, Zheng, and Chu [9] considered multitasking scheduling problems with an RMA, where a human operator performs tasks. They developed efficient algorithms for the scheduling problem with the objective of makespan, total completion time, maximum lateness, and due-date assignment. Another emerging topic in the literature is the integration of scheduling deteriorating jobs and an RMA. Deteriorating jobs is the concept that the later a job starts, the longer it takes to process due to deterioration, and generally, the actual processing time of the jobs depends either on the start time of a job (time deterioration) or on its position in the sequence (positional deterioration). Lodree and Geiger [10] appeared to have been the first to introduce the integration of position-deteriorating jobs and an RMA (Zhao and Tang [11]). They considered the SMSP with the minimization of makespan and proved that the optimal policy allocates an RMA in the middle of the job sequence under certain conditions.

In all of the above papers, an RMA impacts the processing times of subsequent jobs. Mosheiov and Oron [12] recently proposed a different type of RMA, where an RMA impacted the cost of subsequent jobs rather than their processing time. Each job was given a weight reflecting its cost, and if a job was processed after an RMA, its weight (cost) was reduced. For SMSPs with a weight-modifying-activity (WMA), they developed a pseudopolynomial dynamic programming (DP) algorithm to minimize the total weighted completion time. Apart from an RMA or WMA, other research focused on the maintenance activity (MA) either from the perspective of the product quality [13]–[15] or from the predeterministic preventive maintenance plans given by the engineering requirements [16], [17]. In such models, MA is not optional. For additional references about the MA in scheduling, we refer the readers to the review papers by Ma *et al.* [18], Sanlaville and Schmidt [19], Schmidt [20].

Our study mainly follows the study of Mosheiov and Oron [12], who, to the best of our knowledge, were the first to study the SMSP with a WMA. They introduced a pseudopolynomial DP algorithm to minimize the total weighted completion time and proposed polynomial time solutions for some special cases. For the sake of completeness for this new research topic, we propose two mixed integer linear programming (MILP) models not only to systematically express the scheduling problem of interest but also to obtain optimal solutions. Additionally, the considered problem is also known to be NP-hard since it is similar to minimizing the total weighted completion time on two parallel identical machines, which has been shown to be NP-hard (Bruno *et al.* [21]). Therefore, we also develop a heuristic algorithm to efficiently obtain near-optimal solutions.

The paper is organized as follows: Section II contains the problem definition and the MILP models. In Section III, we propose some properties that are used for developing a heuristic algorithm that is described in Section IV. Computational results from the comparisons between the models and the heuristics are described in Section V. Finally, in Section VI, conclusions for future research are provided.

II. PROBLEM DEFINITION AND MILP PROGRAMMING MODELS

The problem under consideration is the SMSP with a weight-modifying-activity (WMA) to minimize the total weighted completion time. The main problem is the determination of when to implement a WMA and which jobs are allocated before and after the WMA to minimize the total weighted completion time. The two determinations are interrelated. The following assumptions are usually adopted.

1. All jobs arrive at a time, i.e., the release times of jobs are equal to 0.
2. The machine is available for jobs and a WMA in the planning horizon.
3. Preemption is not allowed.
4. The parameters of jobs including processing time, weights, and adjusted weight coefficient are deterministic and known in advance.
5. The time of a WMA is known in advance.

For the scheduling problem, two versions of MILP formulations are proposed, and the known and decision variables are given as follows:

A. KNOWN VARIABLES

n: The number of jobs

p_j : The processing time of jobs

w_j : The weight value of jobs

α : An adjustment coefficient for job weight, $0 < \alpha < 1$, which is given in advance.

T: WMA time

M: An extremely large positive integer, where M has defaulted to 32767.

B. DECISION VARIABLES

x_{ij} : A binary variable; $x_{ij} = 1$ if job i is processed before job j and 0 otherwise; $i \neq j$. For example, if job 5 is processed before job 1 in the sequence, $x_{51} = 1$.

y_i : A binary variable; $y_i = 1$ if job i is processed before WMA and 0 otherwise. For example, if job 5 is processed before a WMA in the sequence, $y_5 = 1$.

c_j : The completion time of job j is used for model_1

U_j : The actual weights of job j; $U_j = w_j$ if job j is processed before a WMA and $U_j = (\alpha \times w_j)$ if otherwise.

F_j : The actual completion time of job j used for model_2.

model_1

Objective function

$$Min \sum_{j=1}^{n} (U_j \times c_j) \tag{1}$$

Subject to

$$x_{ij} + x_{ji} = 1 \quad \forall i,j = 1,\ldots,(n+1);i<j \tag{2}$$

$$c_j \geq p_j \quad \forall j = 1,\ldots,(n+1) \tag{3}$$

$$c_j \geq c_i + p_j + (x_{ij} - 1) \times M \quad \forall i,j = 1,\ldots,(n+1);i \neq j \tag{4}$$

$$U_j \geq \alpha \times w_j \quad \forall j = 1,\ldots,n \tag{5}$$

$$U_j \geq w_j + (x_{j,n+1} - 1) \times M \quad \forall j = 1,\ldots,n \tag{6}$$

x_{ij} is a binary; c_j is a positive integer, and U_j is a positive real number (7)

In model_1, the WMA is regarded as another job, i.e., J_{n+1}, where the processing time and the weight of J_{n+1} are T and 1, respectively. The objective of model_1 is to minimize the total weighted completion time as a constraint (1). Constraint (2) ensures the precedence relationship for a pair of jobs (i, j), where job i must be either preceded by job j or succeeded by job j. Constraint (3) specifies that the completion time of job j is greater than or equal to the processing time of job j. Constraint (4) is used to calculate the completion time of each job and to ensure that no job can precede and succeeded in the same job. Constraints (5) and (6) are used to decide the actual weight of job j depending on whether the job is assigned before or after the WMA. Constraint (7) identifies the ranges for x_{ij}, c_j, and U_j, where $x_{ij} \in \{0,1\}$, $c_j \in Z^+$, and $U_j \in R^+$.

model_2

Objective function (7)

$$Min \sum_{j=1}^{n} (w_j \times F_j) \tag{8}$$

Subject to

$$F_1 \geq p_1 + (y_1 - 1) \times M \tag{9}$$

$$F_1 \geq \left[\left(\sum_{i=1}^{n} (y_i \times p_i) + T + p_1\right) \times \alpha\right] - (y_1 \times M) \tag{10}$$

$$F_j \geq \sum_{i=1}^{j-1} (y_i \times p_i) + p_j + (y_j - 1) \times M \quad \forall j = 2,\ldots,n \tag{11}$$

$$F_j \geq [(\sum_{i=1}^{n} (y_i \times p_i) + T + \sum_{i=1}^{j-1} (1 - y_i) \times p_i + p_j) \times \alpha] - (y_j \times M) \quad \forall j = 2,\ldots,n \tag{12}$$

y_i is a binary; F_j is a positive real number (13)

Before implementing model_2, jobs must be sorted by the WSPT rule. Model_2 is used to check jobs to determine if the job is located before or after the WMA and to calculate the actual completion time of the jobs to ensure that the objective of the total weighted completion time is minimized as a constraint (8). Constraints (9) and (10) are used to calculate the actual completion time for the first job in the sequence depending on whether the job is processed before or after the WMA. Constraints (11) and (12) specify the actual completion time of job j, excluding the first job. Constraint (13) identifies the ranges for y_i, and F_j, where $y_i \in \{0, 1\}$ and $F_j \in R^+$

The number of constraints, and the real, integer, and binary variables for the above two formulations are listed in Table 1 where n is the number of jobs.

III. PROPERTIES

For the considered problem in this research, the decision-maker needs to evaluate the trade-off between the production loss due to the WMA and the benefit of cost reduction of subsequent jobs (Mosheiov and Oron [12]). A schedule is used when either (i) the WMA starts at the beginning time or does not schedule at all; or (ii) a schedule is treated as two buckets, which contain a set of jobs scheduled prior to the WMA and the other set of jobs scheduled after the WMA, as shown in Fig. 1, where $J_{[k]}$ denotes the job is placed in position k of the sequence.

TABLE 1. The number of constraints and real, integer, and binary variables in model_1 and model_2

	Constraints	Variables		
		Real	Integer	Binary
model_1	$(3n^2 + 9n + 6)/2$	n	n+1	$(n + 1)^2$
model_2	$2n + 1$	n	---	n

Property 1: For the first case, there are two candidates for optimality: either the WMA is performed at time zero when the time (T) of the WMA is less than or equal to the smallest ratio of p_j/w_j, or the WMA is not performed at all when the time (T) of the WMA is greater than or equal to the largest ratio of p_j/w_j. The optimal solution required to compute $\alpha(\mathrm{T}\sum_{j=1}^{n} w_{[j]} + \sum_{j=1}^{n} w_{[j]}c_{[j]}$ or $\sum_{j=1}^{n} w_{[j]}c_{[j]}$ is, where the jobs are sorted in increasing order of $p_{[j]}/w_{[j]}$.

Property 2: For the second case, the WMA is scheduled between two consecutive jobs [12].

Proof: We assume that the WMA is delayed for Δ unit time to perform when the (k-1)*th* job is completed or the k*th* job is delayed for Δ unit time to be processed when the WMA is finished (e.g. Fig. 2). It is obvious that the schedule will benefit from performing the WMA or processing the k*th* jobs without any delay, i.e., the objective value will decrease.

Property 3: For the second case (e.g. Fig. 1), there is an optimal schedule where the jobs in each bucket will be sorted based on the WSPT rule (i.e., the jobs are sorted in increasing order of $p_{[j]}/w_{[j]}$ when the jobs are processed before the WMA or sorted in increasing order of $p_{[j]}/w_{[j]}$ when the jobs are processed after the WMA).

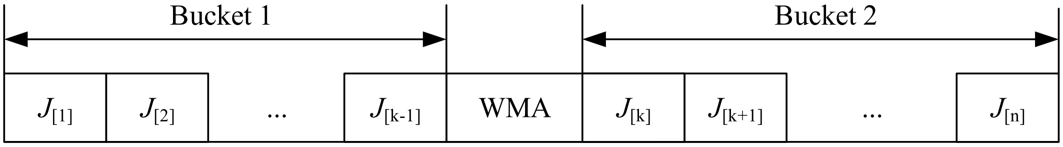

FIGURE 1. A schedule is separated into two sets of jobs by the WMA.

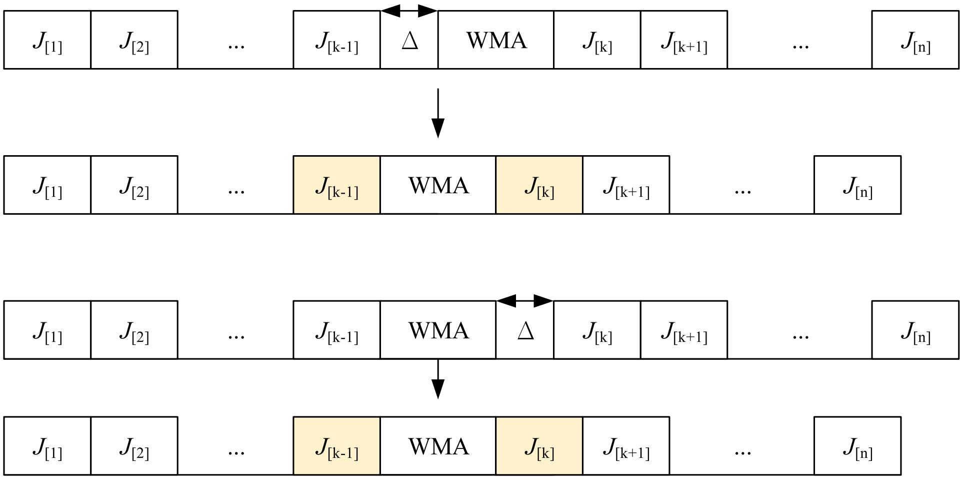

FIGURE 2. The WMA is scheduled between two consecutive jobs without delay.

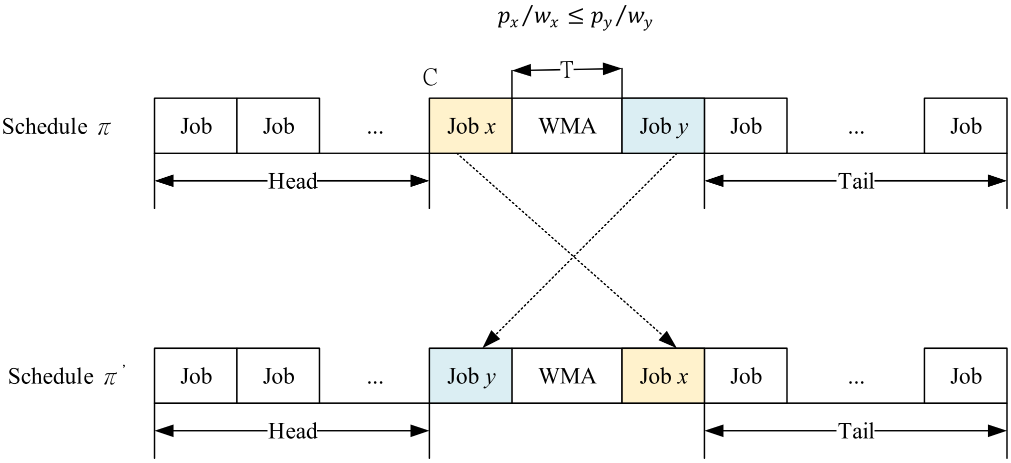

FIGURE 3. The new schedule π' is obtained by interchanging jobs x and y.

Proof: The jobs in two buckets are similar to those that are processed on two parallel machines. The jobs in each machine are given and can be treated as a $1||\sum w_j c_j$ problem. Obviously, the optimal solution is obtained by the WSPT rule for each $1||\sum w_j c_j$ problem [22].

Property 3 implies that the jobs are sorted by the WSPT rule as long as the jobs assigned to each bucket are known and fixed; thus, the allocation of jobs to each bucket is the main key decision.

Property 4: A schedule satisfying property 3 (i.e., jobs are sequenced according to the WSPT rule) is not guaranteed to be an optimal schedule.

Proof: We prove property 4 by contradiction. We assume that Property 4 is incorrect and an optimal schedule π has intersecting jobs x and y. If a new schedule π' is obtained by interchanging jobs x and y, as shown in Fig. 3, and $\mathrm{TWC}(\pi') \leq \mathrm{TWC}(\pi)$, the property is proven.

Based on Fig. 3, we can calculate the total weighted completion time for two schedules as follows:

$$\mathrm{TWC}(\pi) = TWC_{Head} + (C + p_x)\, w_x + \left(C + p_x + T + p_y\right) \cdot w_y \cdot \alpha + TWC_{Tail}$$

$$\mathrm{TWC}\left(\pi'\right) = TWC_{Head} + \left(C + p_y\right) w_y$$

$$
\begin{aligned}
&+ \left(C + p_y + T + p_x\right) \cdot w_x \cdot \alpha + TWC_{Tail}\\
\theta &= TWC\left(\pi\right) - \mathrm{TWC}\left(\pi'\right)\\
&= \left[\left(C + p_x\right) w_x - \left(C + p_y\right) w_y\right]\\
&\quad + \left[\alpha \left(C + p_x + T + p_y\right) \left(w_y - w_x\right)\right]
\end{aligned}
$$

Schedule π in Fig. 3 with two intersecting jobs x and y exhibits the condition of $p_x w_y \leq p_y w_x$ due to the WSPT rule. Based on the law of trichotomy, there are nine possible cases that satisfy the WSPT policy for jobs x and y, as shown in Table 2.

TABLE 2. Nine possible cases satisfied the WSPT policy forθ.

Cases	The relationship for p_x, p_y, w_x, w_y	Is WSPT policy satisfied; $(p_x w_y \leq p_y w_x)$?
1	$p_x < p_y$ and $w_x < w_y$	Partially
2	$p_x < p_y$ and $w_x = w_y$	Completely
3	$p_x < p_y$ and $w_x > w_y$	Completely
4	$p_x = p_y$ and $w_x < w_y$	No
5	$p_x = p_y$ and $w_x = w_y$	Completely
6	$p_x = p_y$ and $w_x > w_y$	Completely
7	$p_x > p_y$ and $w_x < w_y$	No
8	$p_x > p_y$ and $w_x = w_y$	No
9	$p_x > p_y$ and $w_x > w_y$	Partially

Case 1: The condition of $0 < \alpha < 1$ is known, and $p_x w_y \leq p_y w_x$ where $p_x < p_y$ and $w_x < w_y$. Then, $\theta = [(C + p_x)w_x - (C + p_y)w_y] + [\alpha(C + p_x + T + p_y)(w_y - w_x)]$, the value of θ is not sure to be greater than or equal to zero because $[(C + p_x)w_x - (C + p_y)w_y] < 0$ and $[\alpha(C + p_x + T + p_y)(w_y - w_x)] > 0$.

Case 2: The condition of $0 < \alpha < 1$ is known, and $p_x w_y \leq p_y w_x$, where $p_x < p_y$ and $w_x = w_y$. Then, $\theta = [(C + p_x)w_x - (C + p_y)w_y] + [\alpha(C + p_x + T + p_y)(w_y - w_x)] < 0$ due to $[(C + p_x)w_x - (C + p_y)w_y] < 0$, and $[\alpha(C + p_x + T + p_y)(w_y - w_x)] = 0$

Thus, we can conclude that job x is processed before job y when the condition of $p_x < p_y$ and $w_x = w_y$ is satisfied.

Case 3: The condition of $0 < \alpha < 1$ is known, and $p_x w_y \leq p_y w_x$, where $p_x < p_y$ and $w_x > w_y$. Then $\theta = [(C + p_x)w_x - (C + p_y)w_y] + [\alpha(C + p_x + T + p_y)(w_y - w_x)]$, where θ will either be greater than 0 or less than 0 due to $[\alpha(C + p_x + T + p_y)(w_y - w_x)] < 0$.

Case 5: The condition of $0 < \alpha < 1$ is known, and $p_x w_y \leq p_y w_x$, where $p_x = p_y$ and $w_x = w_y$. Than,

$$
\begin{aligned}
\theta = &\left[\left(C + p_x\right) w_x - \left(C + p_y\right) w_y\right]\\
&+ \left[\alpha \left(C + p_x + T + p_y\right) \left(w_y - w_x\right)\right] = 0
\end{aligned}
$$

Case 6: The condition of $0 < \alpha < 1$ is known, and $p_x w_y \leq p_y w_x$, where $p_x = p_y$ and $w_x > w_y$. Then, $\theta = [(C + p_x)w_x - (C + p_y)w_y] + [\alpha(C + p_x + T + p_y)(w_y - w_x)]$, where $[\alpha(C + p_x + T + p_y)(w_y - w_x)] < 0$, but $[(C + p_x)w_x - (C + p_y)w_y] > 0$. Thus, θ is not guaranteed to be less than 0.

Case 9: The condition of $0 < \alpha < 1$ is known, and $p_x w_y \leq p_y w_x$, where $p_x > p_y$ and $w_x > w_y$. Then $\theta = [(C + p_x)w_x - (C + p_y)w_y] + [\alpha(C + p_x + T + p_y)(w_y - w_x)]$, where $[\alpha(C + p_x + T + p_y)(w_y - w_x)] < 0$, but $[(C + p_x)w_x - (C + p_y)w_y] > 0$. Thus, θ is not guaranteed to be greater than or less than 0.

According to the above analyses, the θ value (i.e., $\mathrm{TWC}\left(\pi\right) - \mathrm{TWC}\left(\pi'\right)$) is not guaranteed to be less than or equal to zero, which implies that schedule π is not guaranteed to be optimal, property 4 is proved.

A five-job instance where $p_1 = 19$, $p_2 = 24$, $p_3 = 12$, $p_4 = 11$, $p_5 = 18$, $w_1 = 3$, $w_2 = 1$, $w_3 = 8$, $w_4 = 13$, $w_5 = 13$, $\alpha = 0.5$, and T = 30 is used to illustrate property 4. A schedule π is obtained as shown in Fig. 4, where jobs are sorted by the WSPT rule. However, schedule π is not optimal because we can obtain a better schedule, π_{new}, by swapping jobs 5 and 3, whose optimal solution is 980.5, as shown in Fig. 4.

IV. A FAST HEURISTIC ALGORITHM

In this section, we propose a heuristic algorithm that makes use of the results obtained in Section III. Based on properties 1 and 3, the WSPT rule is a necessary condition for solution optimality; thus, in the proposed heuristic algorithm, the jobs are first sorted in the nondecreasing ratio of p_j / w_j, and then we determine the location of the WMA to obtain an initial complete schedule. After obtaining an initial schedule, we apply the swap and insert procedure to improve the solution quality by changing the job allocation because the decision regarding the allocation of jobs to the buckets (i.e., buckets are similar to machines) is critical for the considered problem (Bruno *et al.* [21]) and the result of property 4 mentioned above. Before describing the proposed heuristic algorithm, particular notations for this heuristic algorithm are listed as follows.

J_j	The job j
$J_{[j]}$	The job allocated at the j*th* position in the sequence.
$C_{[j]}$	The completion time of the job at the j*th* position in the sequence.
$w_{[j]}$	The weight of the job at at the j*th* position in the sequence.
δ	The position of the WMA in the sequence, where $0 \leq \delta \leq n$, δ is an integer.
π	The sequence of jobs
π^k	The set of π, where WMA is performed after the k*th* position.
TWC(π^k)	The total weighted completion time of π^k

The steps of the proposed heuristic algorithm are described below.

Step 0 The job information is input.

Step 1 Jobs are sorted in nondescending order of p_j / w_j to obtain a sequence π of jobs; $\pi = \{J_{[1]}, J_{[2]}, \ldots, J_{[n]}$

Step 2 The following variables are set: $k = 0$, $\delta = 0$, $\pi_{best} = \pi = \{J_{[1]}, J_{[2]}, \ldots, J_{[n]}\}$, and $\mathrm{TWC}\left(\pi_{best}^{\delta}\right) = \infty$.

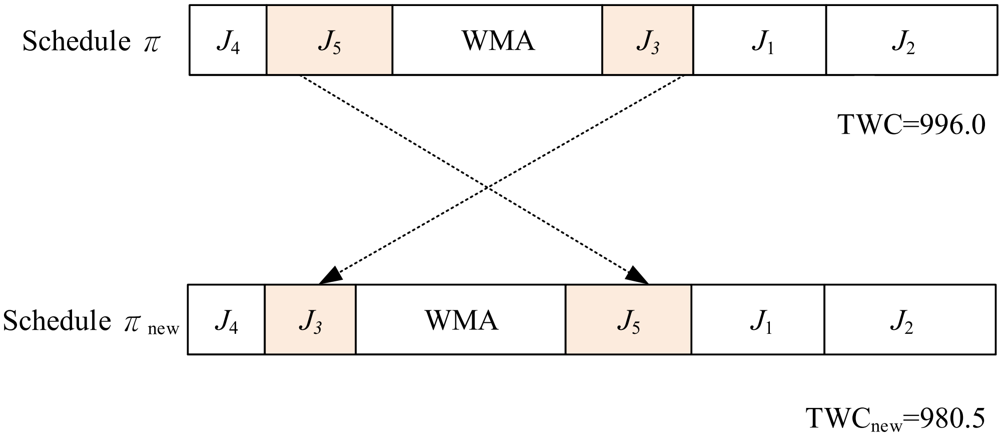

FIGURE 4. Swapping two jobs to obtain a better schedule.

Step 3 The value of TWC $\left(\pi^k\right)$ is calculated based on the following formula

$$\mathrm{TWC}\left(\pi^k\right) = \begin{cases} \alpha \sum_{j=k+1}^{n} w_{[j]}C_{[j]} & k=0 \\ \sum_{j=1}^{k} w_{[j]}C_{[j]} + \alpha \sum_{j=k+1}^{n} w_{[j]}C_{[j]} & 0<k<n \\ \sum_{j=1}^{k} w_{[j]}C_{[j]} & k=n \end{cases}$$

Step 4 If TWC $\left(\pi^k\right) < $ TWC $\left(\pi^{\delta}_{best}\right)$, $\delta = k$, TWC $\left(\pi^{\delta}_{best}\right) = $ TWC $\left(\pi^k\right)$.

Step 5 $k = k{+}1$, if $k \leq n$, go to ***Step*** 3.

Step 6 If $\delta > 0$ and $\delta < n$, go to ***Step*** 7; otherwise go to ***Step*** 9.

Step 7 Jobs $J_{[\delta]}$ and $J_{[\delta+1]}$ are swapped (i.e., the WMA is between the jobs $J_{[\delta]}$ and $J_{[\delta+1]}$) and a new sequence of jobs is obtained; $\pi_{new} = J_{[1]}, J_{[2]}, \ldots J_{[\delta+1]}, J_{[\delta]}, \ldots, J_{[n]}$ and the value of TWC(π^{δ}_{new}) is calculated.

Step 8 If TWC(π^{δ}_{new}) $<$ TWC(π^{δ}_{best}), $\pi_{best} = \pi_{new}$, TWC(π^{δ}_{best}) $= TWC(\pi^{\delta}_{new})$

Step 9 If $\delta < (n{-}1)$, let $k = \delta + 2$, $\delta' = \delta + 1$, and go to ***Step*** 10; otherwise go to ***Step*** 13.

Step 10 Job $J_{[k]}$ is inserted behind the job $J_{[\delta]}$ to form a new sequence of π_{new}; that is, $\pi_{new} = J_{[1]}, J_{[2]}, \ldots J_{[\delta]}, J_{[k]}J_{[\delta+1]}, \ldots, J_{[k-1]}, J_{[k+1]}, \ldots, J_{[n]}$ and the value of TWC($\pi^{\delta'}_{new}$) is calculated.

Step 11 If TWC($\pi^{\delta'}_{new}$) $<$ TWC(π^{δ}_{best}), $\delta = \delta'$, $\pi_{best} = \pi_{new}$, and TWC(π^{δ}_{best}) $=$ TWC($\pi^{\delta'}_{new}$).

Step 12 Let $k = k{+}1$. If $k \leq n$, go to ***Step*** 10.

Step 13 The result of π_{best}, δ, and TWC(π^{δ}_{best}) are output.

In this algorithm, Step 1 is used to form a WSPT-based sequence of jobs, and the computational complexity is $O\left(n \cdot logn\right)$. Steps 2 to 5 are used to determine when to perform the WMA based on the given WSPT-based sequence of jobs, and the complexity of Steps 2 to 5 is $O\left(n^2\right)$. Steps 6 to 8 (called the SWAP procedure) are used to examine the possibility of improving solution quality based on the analysis of property 4. Steps 9 to 12 (called the INSERT procedure) are based on the insertion of a job after WMA into the previous position of the WMA. Notably, the improvement procedures, including swap and insertion, must obey the WSPT rule, and the computational complexities for the SWAP and INSERT procedures are $O\left(n\right)$ and $O\left(n^2\right)$, respectively. In total, the computational complexity of the proposed heuristic algorithm is $O\left(n^2\right)$.

The following example is provided to illustrate the proposed heuristic algorithm. The job information is listed in Table 3. The adjusted weight coefficient (α) is 0.7, and the time (T) of a WMA is 40. First, jobs are arranged in non-decreasing order of p_j/w_j to form a sequence list, which is $\{J_5, J_9, J_8, J_{10}, J_7, J_3, J_2, J_4, J_6, J_1\}$, and then the WMA insertion location is determined by Steps 2 to 5 of the proposed heuristic algorithm. An initial schedule π is obtained, as shown in Fig. 5, where TWC(π) $= 3893.7$. Based on the given schedule π, the swap and insertion procedures are used to improve the solution quality by changing the jobs in the buckets, which results in the total weighted completion time of the initial schedule π being improved to be 3883.3 by the insertion procedure, as shown in Fig. 5. It is worth noting that the final result is also an optimal solution. The swap procedure is illustrated in Fig. 4 for another example mentioned above.

TABLE 3. A 10-job example for illustrating the proposed heuristic algorithm.

	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}
p_j	14	21	14	8	11	15	16	11	8	13
w_j	2	8	8	3	12	3	12	9	8	10

V. COMPUTATIONAL EXPERIMENTS

To examine the performance of the proposed methods, we conduct a numerical experiment. To the best of our knowledge, there are no benchmark instances of our problem to

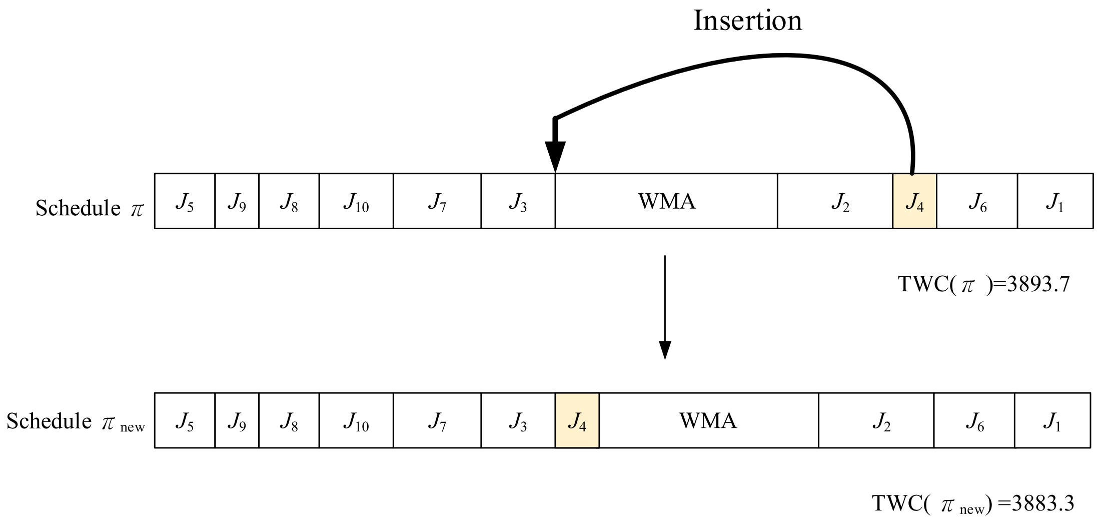

FIGURE 5. An example illustration for the proposed heuristic algorithm.

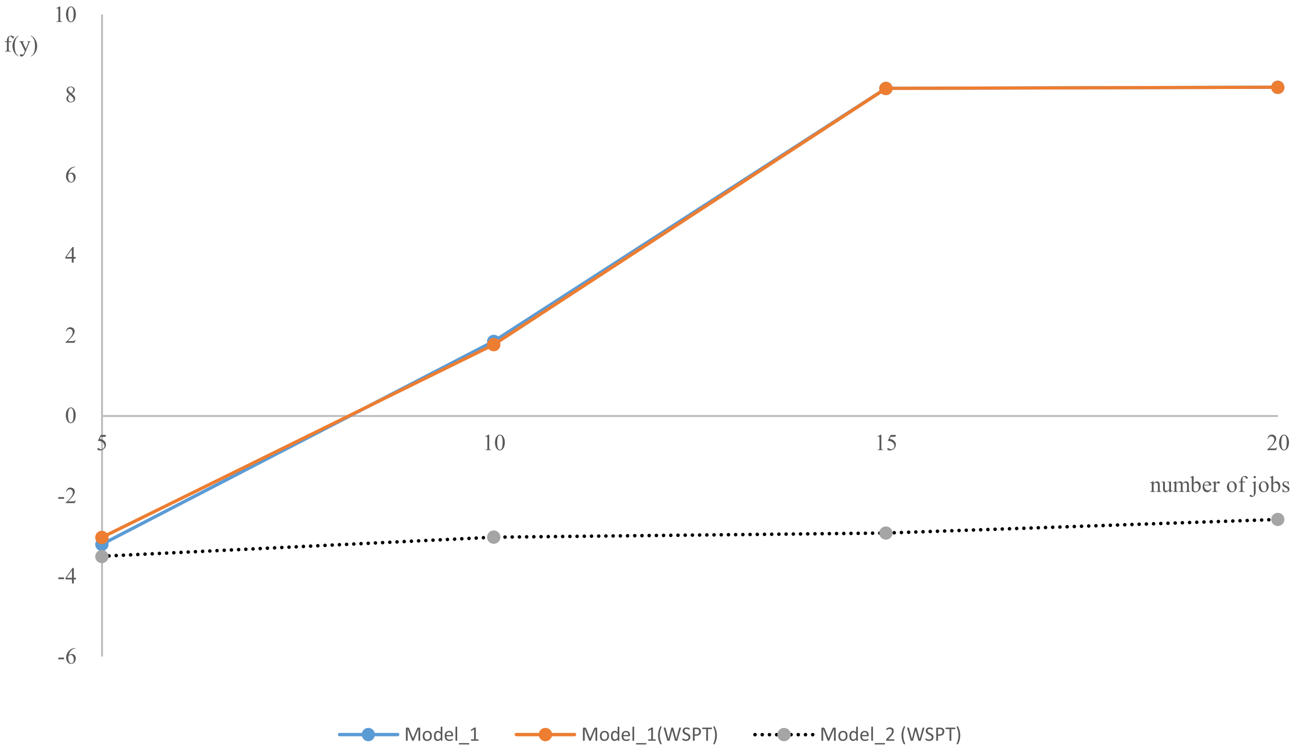

FIGURE 6. The natural logarithm function of the average computational time of the proposed models for small jobs.

date. Thus, we generate testbeds based on the parameters shown in Table 4.

There are 256 different combinations of $n, \boldsymbol{p_j}, \boldsymbol{w_j}, \boldsymbol{\alpha}$, and T, and for each combination, ten different instances are randomly generated, which results in 2560 instances of varying scales. For our proposed models, all the instances are solved using IBM ILOG CPLEX 12.7.1 within a maximum elapsed CPU time, where the CPU time is set to 3600 seconds in this paper. The proposed heuristic algorithm is coded in C++. All the experiments in this paper are carried out on a PC with an Intel Xeon E-2124 3.4 GHz CPU with 32 GB of RAM.

TABLE 4. Parameters of instance.

Number of jobs	n	{5, 10, 15, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 300, 400, 500}
Processing time of jobs	p_j	U[5, 25]
Weight of jobs	w_j	U[1, 15]
Adjust coefficient of weight	α	0.9, 0.7, 0.5, 0.3
WMA time	T	40, 30, 20, 10

TABLE 5. The number of optimal solutions obtained by model_2(WSPT) within 3600 seconds.

	T	40				30				20				10			
n	α	0.9	0.7	0.5	0.3	0.9	0.7	0.5	0.3	0.9	0.7	0.5	0.3	0.9	0.7	0.5	0.3
5		**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**
10		**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**
15		**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**
20		**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**
30		**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**
40		**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**
50		**5**	**10**	**10**	**10**	**7**	**10**	**10**	**10**	**5**	**10**	**10**	**10**	**9**	**10**	**10**	**10**
60		0	**4**	**10**	**10**	0	**2**	**10**	**10**	0	**9**	**10**	**10**	0	**10**	**10**	**10**
70		0	0	**8**	**10**	0	0	**9**	**10**	0	0	**10**	**10**	0	**3**	**10**	**10**
80		0	0	0	**10**	0	0	0	**10**	0	0	**3**	**10**	0	0	**6**	**10**
90		0	0	0	**10**	0	0	0	**10**	0	0	0	**10**	0	0	0	**10**
100		0	0	0	**3**	0	0	0	**8**	0	0	0	**10**	0	0	0	**7**
200		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
300		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
500		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Remark: The number indicates the total number of instances when the upper and lower bounds obtained by model_2(WSPT) are equal within a limit of 3600 seconds under different parameters.

A. A COMPARISON OF THE PROPOSED MLIP MODELS

In this section, we compare the proposed models in terms of average computational effort. To understand the influence of the jobs sorted based on the WSPT rule before implementing models, model_1(WSPT) and model_2(WSPT) refer to the jobs are sorted based on the WSPT rule in advance while executing the models by the ILOG CPLEX solver. Fig. 6 shows the natural logarithm function of the average computational time of each model for small jobs, i.e., $f(\mathrm{y}) = ln(y)$, where y is the average computational time. It can be seen that mode1_2(WSPT) is substantially more efficient than model_1 and model_1(WSPT). It can also be observed that the operation of presorting jobs does not obviously impact the computational time consumed by model_1. It should be noted that the precondition of model_2 is that the jobs have to be sorted by the WSPT rule mentioned in Section II, and the proposed models are terminated within 3600 seconds.

Since the computational time by model_1 and model_1 (WSPT) increases dramatically as the number of jobs increases, and when the number of jobs is greater than 20, model_1 and model_1(WSPT) cannot obtain optimal solutions within the limit of 3600 seconds. Thus, Fig. 7 only shows the average computational time of model_2(WSPT) under a different α and the number of jobs. When the number of jobs is less than 50, model_2(WSPT) can solve a total of 960 instances, and it is surprising that the average computational time of model_2(WSPT) for all instances with 40 jobs is only 17.042 seconds. In Table 5, when the number of jobs is greater than or equal to 50 and $\alpha = 0.9$, model_2(WSPT) cannot be guaranteed to obtain optimal solutions within the limit of 3600 seconds; additionally, it can also be seen that the smaller α is, the less influence on the performance of model_2(WSPT). This is an expected result because the smaller α is, the larger the influence on the cost of subsequent jobs, which indicates that it is better for performing the WMA as soon as possible. Thus, the complexity of assigning jobs into g positions before the WMA (i.e. 2^g) will be smaller as α is smaller. Model_2(WSPT) can solve all instances of 90 jobs when $\alpha = 0.3$ (also seen in Table 4). However, it still has inherent NP-hard problems; thus, model_2(WSPT) cannot solve large problems within a reasonable computational time.

B. PERFORMANCE OF THE HEURISTIC ALGORITHM (HA)

In this section, we examine the effect of improvement procedures, including SWAP and INSERT. The proposed heuristic algorithm is separated into three subheuristic algorithms: H1, H1+SWAP, and H1+SWAP+INSERT. The subheuristic Algorithm H1 aims to produce an initial feasible solution using steps 0 to 5, H1+SWAP indicates that the initial feasible solution obtained by H1 is improved by the SWAP procedure, and H1+SWAP+INSERT indicates that the initial solution obtained by H1 is improved by the SWAP and INSERT procedures, which is also HA that is mentioned in Section IV.

For this comparison, we use the quality of the solution and the number of hits as the performance index. The quality of the solution is measured with the average relative percentage of error (ARPE) index over the number of instances, which is calculated as follows:

$$\mathrm{ARPE} = \frac{\sum_{i=1}^{k} (hs - ms)\big/ ms}{k} \times 100\%$$

where hs is the feasible solution obtained by the abovementioned algorithms, ms is either the optimum or the lower bounds obtained by model_2(WSPT) within the limit of

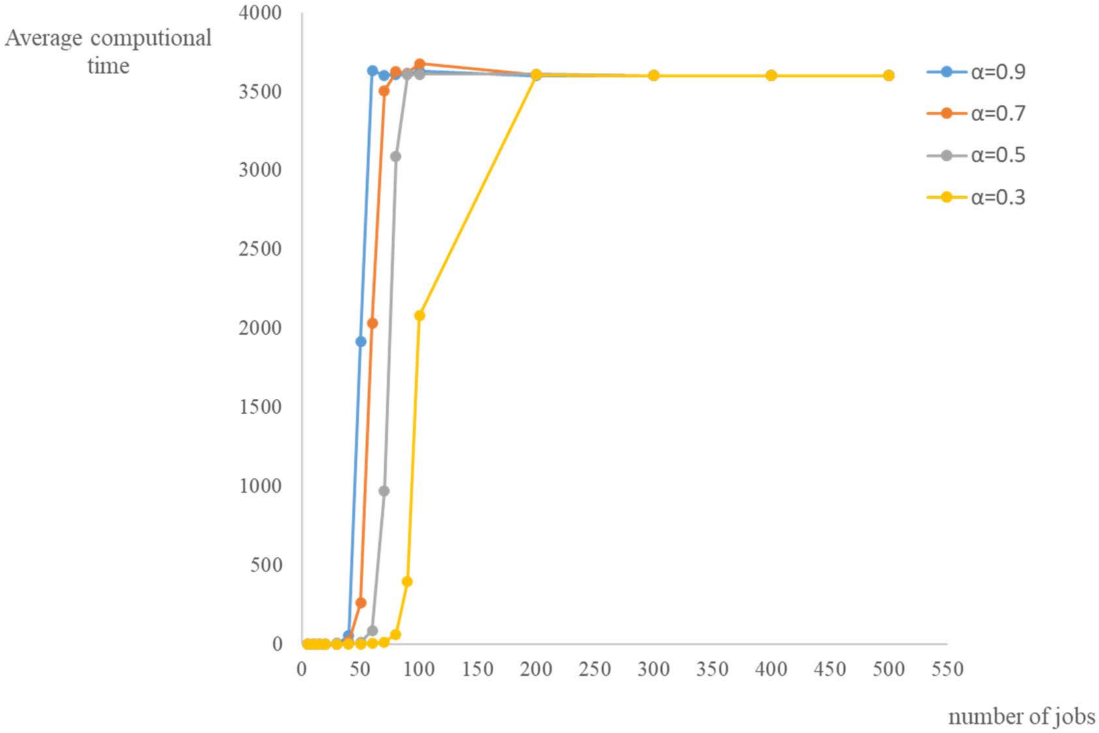

FIGURE 7. The average computational time of model_2(WSPT) under different α values.

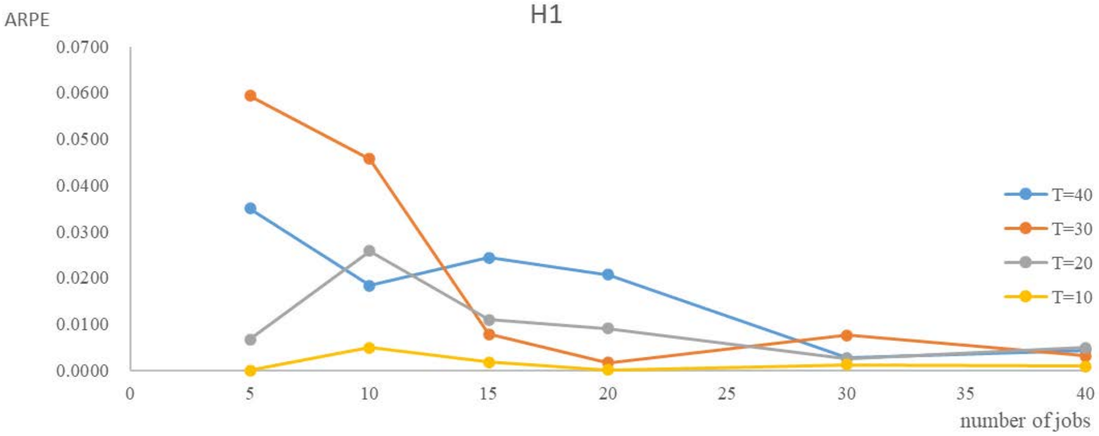

FIGURE 8. ARPE obtained by H1 for different T values.

3600 seconds, and k is the number of instances. Another index, a hit number, calculates the number of optimal solutions obtained by the heuristic algorithms. The hit number is only used for small problems since all optimal solutions can be reached by model_2(WSPT) within the limit of 3600 seconds when the number of jobs is less than 50. The ARPE and hit number are tabulated in Table 6 for small jobs.

From Table 6, it can be observed that H1 performs considerably well in terms of solution quality and hit number, and the whole average ARPE and hit rate are 0.0124% and 87.0%, respectively. H1 appears to be more accurate for large values of α (the adjusting coefficient of weight is larger) and for small values of T (the maintenance time is shorter). However, the impact of different α and T on the optimality gaps was considerably improved by SWAP and INSERT procedures for H1, as shown in Figures 8-13. In summary, the average ARPE and hit rate for H1+SWAP are 0.0070% and 92.3%, and 0.0005% and 98.2% for HA, respectively.

To compare the performances of H1, H1+SWAP, and HA, we use the signal-to-noise ratio (SNR) which is a statistical inference method in the Taguchi method. For the experiment, the SNR is calculated as follows:

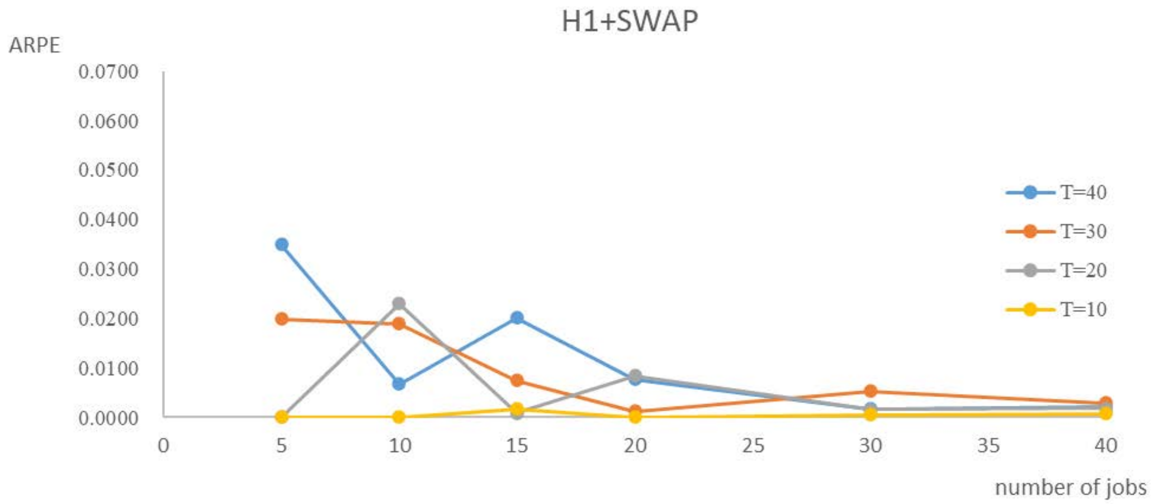

FIGURE 9. ARPE obtained by H1+SWAP for different *T* values.

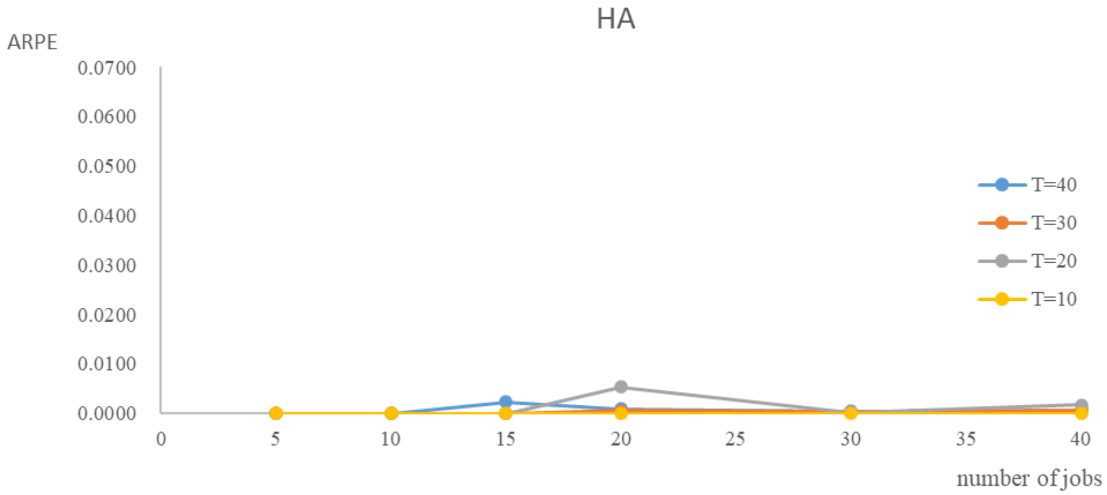

FIGURE 10. ARPE obtained by HA for different *T* values.

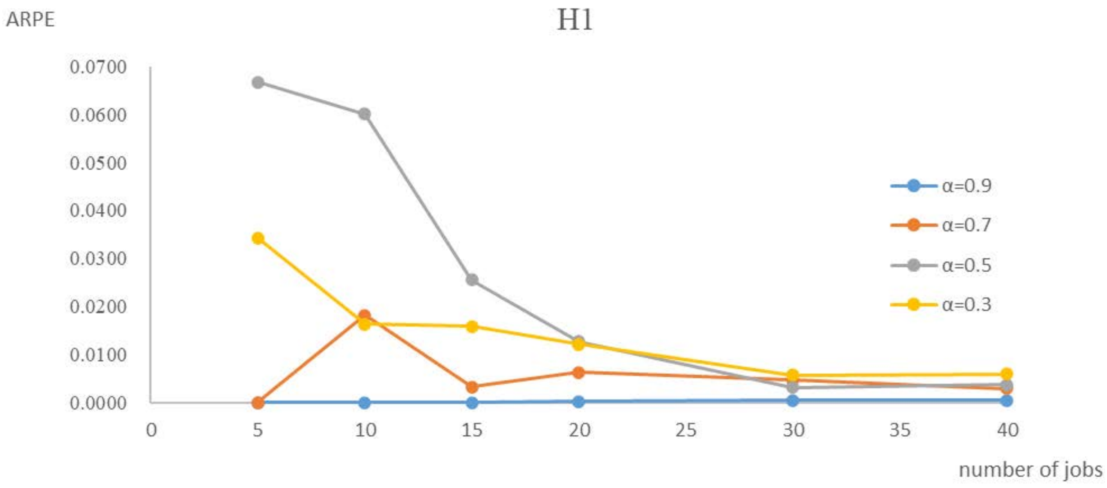

FIGURE 11. ARPE obtained by H1 for different α values.

$\eta = -10\cdot \log(\sum_{i=1}^{k} \frac{x_i^2}{k})$ where x_i is the relative percentage of error (RPE) index (i.e., $(hs - ms)\big/ms)$ for each instance i and k is equal to 960. The largest SNR means a smaller loss, which implies that the largest SNR is better. The SNRs

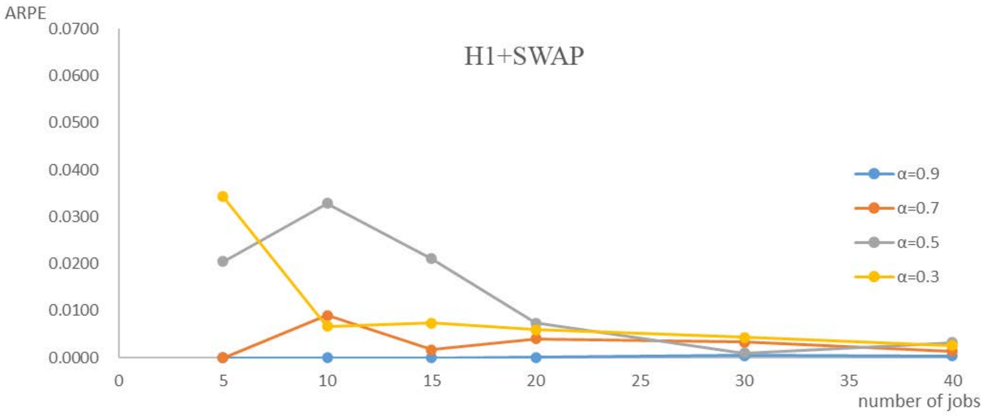

FIGURE 12. ARPE obtained by H1+SWAP for different α values.

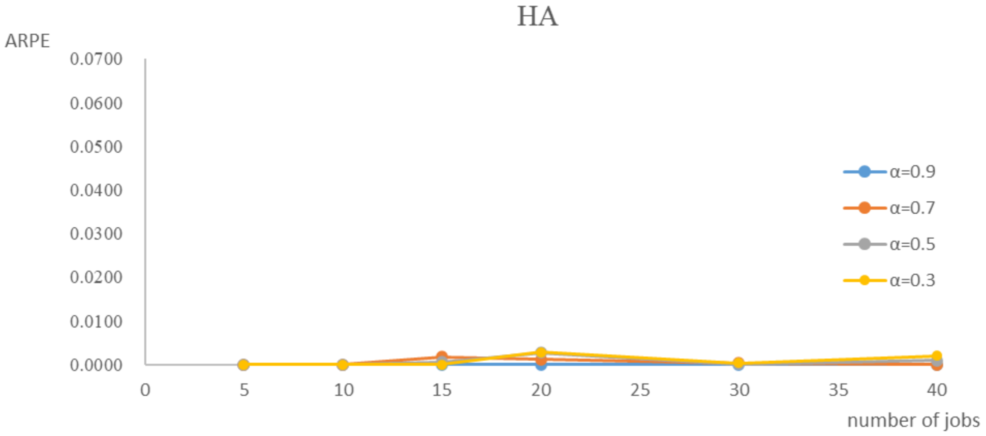

FIGURE 13. ARPE obtained by HA for different α values.

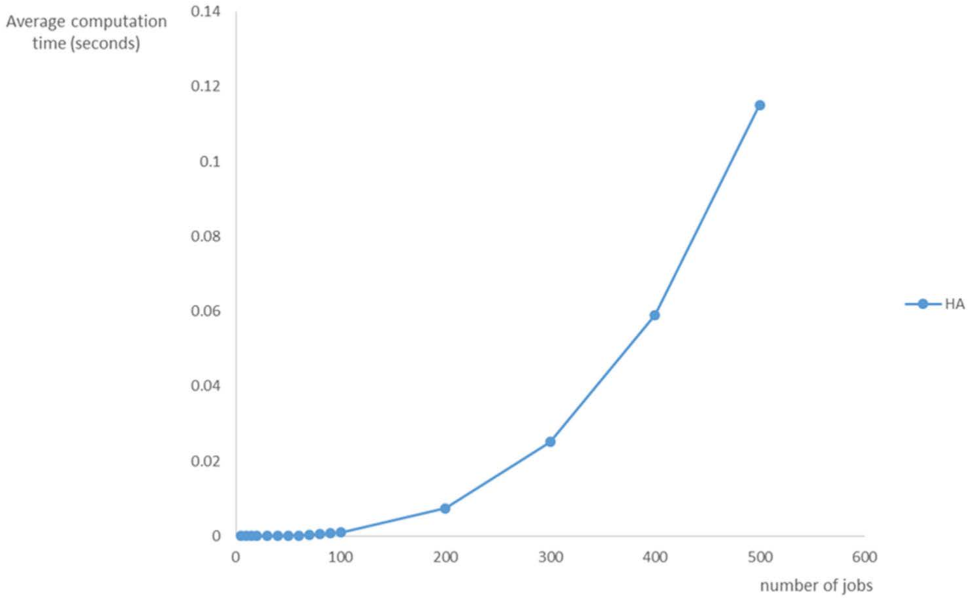

FIGURE 14. The average computational time spent by the proposed HA.

calculated for H1, H1+SWAP, and HA are 21.96, 25.04, and 45.30, respectively; hence, the results obtained by HA are substantially better than those of H1 and H1+SWAP in terms of solution quality.

TABLE 6. Comparison results for the heuristic algorithm with optimal solutions.

	ARPE(%)			Hit number		
n	H1	H1+SWAP	H1+SWAP+INSERT (HA)	H1	H1+SWAP	H1+SWAP+INSERT (HA)
5	0.0253	0.0137	0.0000	154	156	160
10	0.0237	0.0121	0.0000	140	152	160
15	0.0112	0.0076	0.0006	143	148	158
20	0.0078	0.0044	0.0017	139	148	154
30	0.0035	0.0023	0.0002	129	139	156
40	0.0033	0.0019	0.0007	130	143	155

TABLE 7. The results for the proposed methods compared with lower bounds.

	ARPE(%)			
n	H1	H1+SWAP	H1+SWAP+INSERT (HA)	model_2(WSPT)
50	0.1635	0.1627	0.1620	0.1618
60	1.1045	1.1040	1.1036	1.1035
70	2.7472	2.7468	2.7464	2.7460
80	4.3028	4.3026	4.3024	4.3023
90	5.7817	5.7816	5.7813	5.7812
100	7.3049	7.3048	7.3047	7.3047
200	24.3736	24.3746	24.3736	24.3735
300	38.0943	38.0943	38.0943	38.0943
400	44.9501	44.9501	44.9501	44.9501
500	48.2551	48.2551	48.2551	48.2559

In Figs. 8-13, it can also be seen that the differences in solution quality of the three heuristic algorithms (H1, H1+SWAP, HA) are trivial as the number of jobs increases. This phenomenon also appears in Table 7, where the proposed methods have almost the same ARPE values that are compared to the lower bounds obtained by model_2(WSPT). This result is reasonable because the impact of the WMA is insignificant when the number of jobs becomes large for this problem; thus, we can conclude that the jobs sorted by the WSPT rule are a considerable policy for the considered problem with large jobs. Additionally, it is worth noting that the proposed HA is very quick to obtain near-optimal solutions, as shown in Fig. 14.

VI. CONCLUSION

In this paper, we consider SMSP with optional weight-modifying activity. The objective function was the total weighted completion time. To the best of our knowledge, the pioneer study was introduced by Mosheiov and Oron [12], who developed a pseudopolynomial dynamic programming algorithm. To date, none of the subsequent studies considered this scheduling problem. This problem has been shown to be NP-hard (Bruno *et al.* [21]). However, our proposed MILP model (model_2) is effective in solving problems with up to 40 jobs, and for the special cases with $\alpha = 0.3$, some instances with up to 100 jobs can obtain optimal solutions by model_2. However, we also propose some foundational optimality properties and based on these properties, a heuristic algorithm is developed where the initial WSPT-based sequence of jobs is improved by SWAP and INSERT steps. The computational experiments show that the average relative percentage of error (ARPE) from optimal solutions and the hit rate are 0.0005% and 98.2%, respectively, for the number of jobs is less than or equal to 40. Regarding the large problems, the initial WSPT-based sequence of jobs has almost the same performance as the proposed heuristic algorithm, which implies that the WSPT policy is still a good dispatching rule for the considered problem.

As asserted by Mosheiov and Oron [12], the complexity of the considered problem is an open question that needs to be resolved for future research. In addition, some extensions of the problems can be studied in the future, such as minimizing makespan, relative due-date objectives, and bicriterion performances on parallel machines or shops.

REFERENCES

[1] M. L. Pinedo, *Scheduling: Theory, Algorithms and Systems*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1995.

[2] V. Strusevich and K. Rustogi, ‘‘Scheduling with time-changing effects and rate-modifying activities,’’ in *International Series on Operations Research & Management Science*, vol. 243. Cham, Switzerland: Springer, 2017.

[3] C.-Y. Lee and V. J. Leon, "Machine scheduling with a rate-modifying activity," *Eur. J. Oper. Res.*, vol. 128, no. 1, pp. 119–128, Jan. 2001.

[4] G. Mosheiov and D. Oron, "Due-date assignment and maintenance activity scheduling problem," *Math. Comput. Model.*, vol. 44, nos. 11–12, pp. 1053–1057, Dec. 2006.

[5] V. Gordon and A. A. Tarasevich, "A note: Common due date assignment for a single machine scheduling with the rate-modifying activity," *Comput. Oper. Res.*, vol. 36, pp. 325–328, Feb. 2009.

[6] Y. Yin, T. C. E. Cheng, D. Xu, and C. C. Wu, "Common due date assignment and scheduling with a rate-modifying activity to minimize the due date, earliness, tardiness, holding, and batch delivery cost," *Comput. Ind. Eng.*, vol. 63, pp. 223–234, Aug. 2012.

[7] G. Mosheiov and D. Oron, "A note on scheduling a rate modifying activity to minimize total late work," *Comput. Ind. Eng.*, vol. 154, Apr. 2021, Art. no. 107138.

[8] M. Sterna, "A survey of scheduling problems with late work criteria," *Omega*, vol. 39, no. 2, pp. 120–129, Apr. 2011.

[9] Z. Zhu, F. Zheng, and C. Chu, "Multitasking scheduling problems with a rate-modifying activity," *Int. J. Prod. Res.*, vol. 55, no. 1, pp. 296–312, 2017.

[10] E. J. Lodree and C. D. Geiger, "A note on the optimal sequence position for a rate-modifying activity under simple linear deterioration," *Eur. J. Oper. Res.*, vol. 201, pp. 644–648, Mar. 2010.

[11] C. Zhao and H. Tang, "A note to due-window assignment and single machine scheduling with deteriorating jobs and a rate-modifying activity," *Comput. Oper. Res.*, vol. 39, no. 6, pp. 1300–1303, Jun. 2012.

[12] G. Mosheiov and D. Oron, "Scheduling problems with a weight-modifying-activity," *Ann. Oper. Res.*, vol. 295, pp. 737–745, Sep. 2020.

[13] J. Pang, H. Zhou, Y.-C. Tsai, and F.-D. Chou, "A scatter simulated annealing algorithm for the bi-objective scheduling problem for the wet station of semiconductor manufacturing," *Comput. Ind. Eng.*, vol. 123, pp. 54–66, Sep. 2018.

[14] Y. Chen, L.-H. Su, Y.-C. Tsai, S. Huang, and F.-D. Chou, "Scheduling jobs on a single machine with dirt cleaning consideration to minimize total completion time," *IEEE Access*, vol. 7, pp. 22290–22300, 2019.

[15] L.-H. Su and H.-M. Wang, "Minimizing total absolute deviation of job completion times on a single machine with leaning activities," *Comput. Ind. Eng.*, vol. 103, pp. 242–249, Jan. 2017.

[16] C. J. Liao and W. J. Chen, "Single-machine scheduling with periodic maintenance and nonresumable jobs," *Comput. Oper. Res.*, vol. 30, pp. 1335–1347, Aug. 2003.

[17] W. J. Chen, "Minimizing total flow time in the single-machine scheduling problem with periodic maintenance," *J. Oper. Res. Soc.*, vol. 57, no. 4, pp. 410–415, Apr. 2006.

[18] Y. Ma, C. Chu, and C. Zuo, "A survey of scheduling with deterministic machine availability constraints," *Comput. Ind. Eng.*, vol. 58, pp. 199–211, Mar. 2010.

[19] E. Sanlaville and G. Schmidt, "Machine scheduling with availability constraints," *Acta Inform.*, vol. 35, no. 9, pp. 795–811, 1998.

[20] G. Schmidt, "Scheduling with limited machine availability," *Eur. J. Oper. Res.*, vol. 121, pp. 1–15, Feb. 2000.

[21] J. Bruno, E. G. Coffman, and R. Sethi, "Scheduling independent tasks to reduce mean finishing time," *Commun. ACM*, vol. 17, no. 7, pp. 382–387, Jul. 1974.

[22] W. E. Smith, "Various optimizers for single-stage production," *Naval. Res. Log. Quart.*, vol. 3, no. 1, pp. 59–66, Mar. 1956.

YARONG CHEN received the B.S. and M.S. degrees in management science and engineering from Jiangsu University, Zhenjiang, Jiangsu, China, in 2003. Since 2003, she has been an Assistant Professor and an Associate Professor with the College of Mechanical and Electrical Engineering, Wenzhou University, Wenzhou, China. Her research interests include the simulation and optimization of complicated manufacturing systems, integration optimization of production planning and scheduling, scheduling algorithms, and lean production.

ZAILIN GUAN received the Ph.D. degree in engineering from the Huazhong University of Science and Technology, Wuhan, Hubei, China, in 1997. From 1997 to 1999, he worked as a Postdoctoral Researcher with The Hong Kong University of Science and Technology, Hong Kong, China. He is currently a Professor with the College of Mechanical Science and Engineering, Huazhong University of Science and Technology. His research interests include digital factory, intelligent manufacturing, APS, and production system modeling simulation and optimization.

YA-CHIH TSAI received the M.S. and Ph.D. degrees with the Department of Industrial Engineering and Management, University of Yuan Ze. She is currently an Associate Professor with the Department of Hotel Management, Vanung University. Her current research interests include production scheduling and semiconductor manufacturing management.

FUH-DER CHOU received the M.S. degree in industrial engineering from Chung Yuan Christian University, in 1988, and the Ph.D. degree in industrial engineering and management from the National Chiao Tung University, in 1997. He is currently a Professor with the College of Mechanical and Electronic Engineering, Wenzhou University, China. His research interests include production scheduling, semiconductor manufacturing management, and group technology.