IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 5, 2019, accepted July 15, 2019, date of publication July 18, 2019, date of current version August 7, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2929582

Scheduling Jobs of Two Competing Agents on a Single Machine

CHEN-YANG CHENG[1], SHU-FEN LI[2], KUO-CHING YING[1], AND YU-HSI LIU[1,3]
[1]Department of Industrial Engineering and Management, National Taipei University of Technology, Taipei 10608, Taiwan
[2]Department of Industrial Engineering and Management, National Chin-Yi University of Technology, Taichung 41170, Taiwan
[3]LITE-ON Technology Corporation, Taipei 11492, Taiwan

Corresponding author: Kuo-Ching Ying (kcying@ntut.edu.tw)

This work was supported in part by the Ministry of Science and Technology of the Republic of China, Taiwan, under Grant MOST106-2221-E-027-085.

ABSTRACT This paper studies a single-machine scheduling problem with a two competing agents in which the performance criteria of the first and second agents are to minimize the mean lateness and number of tardy jobs, respectively. Due to the non-deterministic polynomial-time hardness of this problem, we propose an effective and efficient algorithm, denominated as the SPT-M algorithm, to generate the non-dominated solutions of the Pareto set. Computational results conducted on a test problem set reveal that the proposed SPT-M algorithm can generate an efficient Pareto frontier in remarkably short computing time. The contribution of this paper could help practitioners to determine the tradeoffs between the jobs of two agents competing for a single resource.

INDEX TERMS Scheduling, single-machine, two competing agents.

I. INTRODUCTION

In practice, various performance measures, such as capacity utilization, production cost, and hit rate, are used to monitor the efficiency and effectiveness of operations in manufacturing shops. Production scheduling is a key method for improving resource allocation and thereby increasing capacity utilization and manufacturing shop hit rate and decreasing production cost. To achieve these goals, planners typically use more than one criterion (e.g., makespan, total completion time, lateness, earliness, tardiness, and number of tardy jobs) for evaluation and determining a proper solution for production scheduling [1]. Multi-objective scheduling problems require multiple performance criteria, which may contradict one another, to solve scheduling problems. Because optimizing a single performance criterion does not usually suit the practical needs of real-world production scheduling, multi-objective scheduling problems have been heavily discussed in the literature [1]. So far, thousands of optimization and approximation methods for solving different multi-objective scheduling problems have been proposed [2]–[5].

For the traditional multi-objective scheduling problems, a decision maker typically tries to satisfy multiple criteria on the same set of jobs. However, in many applications, jobs are owned by different entities (agents) and must be processed using the same resource. Therefore, multi-agent scheduling problems in which different entities (agents) compete on the usage of shared resources have received more and more attention in recent decades [6]. For example, Wu *et al.* [7] investigated the two-stage assembly flowshop with two-agent; Lin *et al.* [8] studied the two-agent multi-facility customer order scheduling problem; Yin *et al.* [9] considered the just-in-time scheduling problem with two competing agents on unrelated parallel machines; and Yin *et al.* [10] focused on the two-agent flowshop scheduling problem.

Motivated by practical applications, this study focuses on the single-machine scheduling problem (SMSP) with two competitive agents. The first agent has a set J^1 of n_1 jobs, and the second agent has a set J^2 of jobs, and the two must compete to perform on one shared processing resource. We define $n = n_1 + n_2$ and $J = J^1 \cup J^2$, where $J^1 \cap J^2 = \emptyset$. Each job $j \in J^s (s = 1, 2)$ must be processed on the machine exactly once with given processing time p_j^s. Each job $j \in J^s (s = 1, 2)$ has a predetermined due date d_j^s. All agents aim at optimizing an individual performance criterion based only on their own jobs. The performance criteria of the first and second agents considered herein are minimizing mean lateness of the n_1 jobs in set J^1 ($\sum_{j \in J^1} L_j/n_1$) and the number of tardy jobs of the n_2 jobs in set J^2 ($\sum_{j \in J^2} U_j$), respectively. The objective of this

---

The associate editor coordinating the review of this manuscript and approving it for publication was Zhiwu Li.

investigation is to generate an efficient Pareto frontier using non-dominated solutions, which could help decision-makers to determine the tradeoffs between the jobs of two agents competing for a single machine. A sequence is considered a non-dominated solution if no other sequence $\pi \in \Pi$ exists for the first objective function value $z_1(\pi) \leq z_1(\pi^*)$ and the second objective function value $z_2(\pi) \leq z_2(\pi^*)$, in which at least one of the inequalities is strict. By using the well-known three-field notation [11], the two-agent SMSP considered herein can be denoted as $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$, where "1" indicates that the shop environment is a single machine; $ND(^{*,**})$ represents the objective to obtain a non-dominated solution set with respect to the performance criteria of the two agents. The performance criterion of the first agent (*) is mean lateness $(\sum_{j\in J^1} L_j/n_1)$ of its jobs $j \in J^1$, whereas the performance criterion of the second agent (**) is the number of tardy jobs $(\sum_{j\in J^2} U_j)$ of its jobs $j \in J^2$. Moreover, the critical assumptions of this two-agent SMSP are as follows:

- All job release dates are at the beginning (time zero) of the scheduling period.
- No preemptive priorities are assigned; that is, the jobs are not allowed to be split.
- The processing times for jobs on the machine are typically not the same.
- The machine can execute no more than one job at a time.
- The machine is always available for processing jobs throughout the planning horizon.
- The setup time of the machine is negligible.

The practical applications of the addressed SMSP in which two agents compete for the usage of one shared processing resource and each agent has its own criterion to optimize are considerable. Not only can devices, such as lathes, be considered a single machine; an entire manufacturing facility, individual, workstation, or production line with a bottleneck may also be treated as a single machine. In this study, an effective and efficient algorithm, referred to as the SPT-M algorithm, which combines and extends thoughts from the shortest processing time (SPT) rule and Moore's algorithm, is proposed to yield non-dominated solutions of a Pareto set for the addressed problem. The aim of this study focuses directly on generating a set of alternative schedules to provide an operational basis for negotiation the usage of one shared processing resource. Once a set of non-dominated solutions has been generated by using the proposed SPT-M algorithm, decision makers could choose one of these non-dominated solutions to arrange jobs of two customers (agents) based on their trading contracts.

The remainder of this paper is organized as follows: Section 2 presents a review of the relevant literature with respect to the bi-objective SMSPs to establish the basis of this study. Section 3 describes in detail the procedures of the proposed SPT-M algorithm, and Section 4 discusses the performance of the proposed SPT-M algorithm. Finally, Section 5 presents conclusions and recommendations for future studies.

II. RELATED WORKS

The first study on bi-criteria SMSPs was by Smith in 1956 [12]. Since then, several researchers have continued to study the problem proposed by Smith, whereas others have investigated different performance criteria SMSPs. As surveyed by Nagar *et al.* [13], before 1995, researchers typically employed conventional and well-established techniques, such as branch-and-bound [14] and dynamic programming [15] methods, to resolve bi-objective SMSPs. Because both techniques are affected by computational time complexity, which restricts the scope of these exact methods when solving large problems, heuristic algorithms have become the primary technique for solving bi-criteria SMSPs since 1995. These heuristic algorithms are designed to consider the tradeoffs between computational cost and solution quality for real-world sized problems.

Numerous problem-specific heuristic algorithms have been proposed to solve different bi-criteria SMSPs. Köksalan and Keha [16] considered two bi-criteria SMSPs with the objectives of minimizing flowtime and number of tardy jobs and minimizing flowtime and maximum earliness, respectively. The authors presented a promising heuristic algorithm and a genetic algorithm (GA) for the first problem and adapted the GA for the second problem by utilizing its special structure. Azizoglu *et al.* [17] proposed two general procedures to generate all efficient schedules and identify the most efficient schedule, respectively, for the bi-criteria SMSP of minimizing both maximum earliness and the number of tardy jobs. Jolai *et al.* [18] focused on the same problem and proposed a GA that used a heuristic algorithm to improve the initial population. Harald *et al.* [19] focused on a bi-criteria SMSP with both traditional and nontraditional requirements and proposed an experimental approach and random key GA to identify non-dominated solutions. Eren and Güner [20] considered a bi-criteria SMSP with sequence-dependent setup times. The authors proposed an integer programming model to minimize the weighted sum of the total completion time and total tardiness in solving problems with up to 12 jobs. Moreover, a special heuristic algorithm and a Tabu search-based heuristic algorithm were introduced for solving large problems. Later, Chen [21] investigated a bi-criteria SMSP with periodic maintenance. A highly accurate and efficient heuristic algorithm was proposed to minimize the total flow time and maximum tardiness by providing a small set of efficient sequences. The heuristic algorithms mentioned in the aforementioned studies can yield high-quality non-dominated solutions of the Pareto set in reasonable computing times. However, none of them considered the case of SMSPs with competing agents.

In recent decades, two-agent SMSPs have begun attracting a great deal of research interest. This problem arises when two customers, each owning a set of jobs, compete to perform their jobs on one shared processing resource and both customers want to optimize an individual performance criterion concerning only their own jobs. Baker and Smith [6] first considered an SMSP with two competing customers, in which jobs belonging to different customers were evaluated based on their individual criteria. The authors demonstrated that when minimizing a combination of three basic scheduling criteria (e.g., makespan, lateness, and total weighted completion time) the problem was non-deterministic polynomial-time hardness (*NP*-hard). Agnetis *et al*. [22], [23] considered several cases of two-agent SMSPs arising from different combinations of three criteria for the two agents: maximum of a regular function, weighted total completion time, and number of tardy jobs. The authors addressed the problem of finding the optimal solution for one agent with a constraint on the other agent's cost function. Moreover, the authors generated all non-dominated solutions and analyzed the computational complexity of each case. Ng *et al.* [24] further revealed that a two-agent SMSP, where the objective is to minimize the total completion time of the first agent with the restriction that the number of tardy jobs of the second agent cannot exceed a given number can be solved in pseudo-polynomial time using binary encoding.

Wan *et al.* [25] considered several two-agent SMSPs with controllable job processing times, the objective is to minimize the objective function of one agent subject to a given upper bound on the objective function of the other agent. The authors presented polynomial-time algorithms for several special cases of these two-agent SMSPs. Leung *et al.* [26] generalized the results of some of the aforementioned two-agent SMSPs by including preemption and release dates. The authors further established the relationships between two-agent SMSPs with rescheduling and scheduling subject to availability constraints. Lee *et al.* [27] proposed a branch-and-bound algorithm and three heuristic algorithms to solve a two-agent SMSP with linear deterioration jobs, the objective being to minimize the total weighted completion time of jobs from the first agent, while not allowing tardy jobs for the second agent. Liu *et al.* [28] presented optimal properties and polynomial-time algorithms to solve two two-agent SMSPs of increasing job linear deterioration processing times. Their objective was to minimize the performance criterion of one agent with the restriction that the objective function value of the other agent remained less than or equal to a fixed level.

Yin *et al.* [29] proposed optimal properties and complexity results for two-agent SMSPs with a linear non-increasing deterioration processing times of jobs. Three objective functions, namely maximum, total, and total weighted earliness costs, are considered for the first agent subject to the condition that the maximum earliness cost of the second agent is less than an established upper bound. The shortest processing time (SPT) and earliest due date (EDD) rules sequence jobs in the ascending order of their processing times and due dates, respectively. These two rules have been prominently used in literature to solve SMSPs [30]. Khowala *et al.* [31] discussed a two-agent SMSP that aimed to minimize the total weighted completion time and maximum lateness and proposed a forward SPT-EDD heuristic algorithm to obtain non-dominated solutions for the Pareto set. Oron *et al.* [32] designed faster polynomial-time optimization algorithms for various two-agent SMSPs with equal job processing times. The objective functions of the two agents were to minimize either the weighted sum of completion times or the weighted number of tardy jobs.

Recently, Yin *et al.* [33] considered SMSPs involving two agents, where the due dates of the first agent are determined by using the common (CON) or slack (SLK) due date assignment methods. The objective is to minimize the performance criterion of the first agent while keeping the objective value of the second agent no greater than a given limit. The authors analyzed the computational complexity and presented a dynamic programming algorithm to solve the problem. Yin *et al.* [34] investigated an integrated production and batch delivery scheduling problem on a single machine with two competing agents. Each of the agents wants to minimize an objective function depending on the completion time of its own jobs. The authors proposed polynomial-time algorithms and pseudo-polynomial dynamic programming algorithms to minimize the objective function of one agent while keeping the objective function value of the other agent below or at a given value. Wang *et al.* [35] studied a two-agent single-machine due date assignment and scheduling problem. The authors presented a fully polynomial-time approximation scheme to minimize the objective of one agent, subject to an upper bound on the objective of the other agent. Yin *et al.* [36] considered the problem of scheduling n non-resumable and simultaneously available jobs on a single machine with several agents, in which each job belongs to one of the agents. The authors proposed pseudo-polynomial dynamic programming algorithms and mixed integer linear programming (MILP) formulations to minimize the last agent's criterion, while keeping each of the other agents' criterion values no greater than a given limit. Zhang and Wang [37] presented optimal polynomial and pseudo-polynomial time algorithms to solve a two-agent SMSP with different scenarios. The objective was to determine a schedule that minimized the total weighted late work of the first agent while ensuring that the maximum cost of the second agent did not exceed a specified bound. Their algorithms are valid if the processing times and due dates of all jobs are arbitrary positive real numbers. More recently, Yin *et al.* [38] studied several integrated production, inventory, and batch delivery SMSPs with due date assignment and two competing agents. The authors proposed exact and/or approximation solution algorithms for each of the problems considered.

Related studies have indicated that two-agent SMSPs with the objective of enumerating all non-dominated solutions are the most difficult problems to solve, and designing approaches for determining the non-dominated solution set for two-agent SMSPs is difficult. For a more detailed literature review, a systematic survey and classification of existing contributions in terms of the complexity of two-agent SMSPs and the proposed algorithms can be found in research published by Perez-Gonzalez and Framinan [39]. Although the optimal Pareto set of the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem can be obtained via exact methods such as complete enumeration (CE) and branch and bound methods, these approaches take a prohibitive amount of computational time even for moderate size problems. For the practical purpose, it is often more appropriate to look for heuristic algorithms that generate a near-optimal Pareto set at the relatively minor computational expense. This leads to the development of the SPT-M algorithm in this study.

III. SPT-M ALGORITHM

The following sections discuss in detail the proposed SPT-M algorithm for solving the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem. First, we present the structure of the non-dominated solutions of the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem. Second, the detailed procedure of the proposed SPT-M algorithm is described. Finally, we provide a numerical example of the proposed SPT-M algorithm.

A. STRUCTURE OF THE NON-DOMINATED SOLUTIONS OF $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$

Some well-known scheduling theorems for SMSPs with a single criterion have been developed for yielding optimal sequences. For example, mean lateness is minimized if jobs are sequenced according to the SPT rule [40], [41], whereas the number of tardy jobs is minimized if jobs are sequenced according to Moore's algorithm [30]. Based on these theorems, we can use the following lemmas to explore the structure of the non-dominated solutions of the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem.

Lemma 1: For all strongly non-dominated solutions of the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem, there exists an optimal schedule for the problem in which the n_1 jobs of the first agent (set J^1) are sequenced according to the SPT rule.

Proof: Consider a sequence π in which the n_1 jobs of the first agent (set J^1) are not sequenced according to the SPT rule. That is, somewhere in π there must exist a pair of adjacent jobs, i and i' ($i, i' \in J^1$), with i' following i, such that $p_i^1 > p_{i'}^1$. Now construct a new sequence, π', in which jobs i and i' are interchanged in sequence and all other jobs finish at the same time as in π. Let B denotes the set of jobs preceding jobs i and i' in both schedules π and π', and A denotes the set of jobs following i and i' in both schedules π and π'. In addition, let $C_j(\pi)$ and $C_j(\pi')$ denote the completion time of job j under schedule π and π', respectively, and $S_j(\pi)$ and $S_j(\pi')$ denote the start time of job j under schedule π and π', respectively. Evidently, $S_i(\pi) = S_{i'}(\pi')$. Also, we temporarily adopt the notations $\sum_{j\in J^1} L_j/n_1(\pi)$ and $\sum_{j\in J^1} L_j/n_1(\pi')$ to represent $\sum_{j\in J^1} L_j/n_1$ of schedules π and π', respectively.

We first show that $\sum_{j\in J^1} L_j/n_1(\pi')$ is smaller than $\sum_{j\in J^1} L_j/n_1(\pi)$.

$$
\begin{aligned}
&\sum\nolimits_{j\in J^1} L_j/n_1(\pi)\\
&\quad = (\sum\nolimits_{j\in J^1 \text{and } j\in B} L_j + L_i + L_{i'} + \sum\nolimits_{j\in J^1 \text{and } j\in A} L_j)/n_1\\
&\quad = (\sum\nolimits_{j\in J^1 \text{and } j\in B} L_j + C_i(\pi) - d_i + C_{i'}(\pi) - d_{i'}\\
&\qquad + \sum\nolimits_{j\in J^1 \text{and } j\in A} L_j)/n_1
\end{aligned}
$$

And,

$$
\begin{aligned}
&\sum\nolimits_{j\in J^1} L_j/n_1(\pi')\\
&\quad = (\sum\nolimits_{j\in J^1 \text{and } j\in B} L_j + L_{i'} + L_i + \sum\nolimits_{j\in J^1 \text{and } j\in A} L_j)/n_1\\
&\quad = (\sum\nolimits_{j\in J^1 \text{and } j\in B} L_j + C_{i'}(\pi') - d_{i'} + C_i(\pi') - d_i\\
&\qquad + \sum\nolimits_{j\in J^1 \text{and } j\in A} L_j)/n_1
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
&\sum\nolimits_{j\in J^1} L_j/n_1(\pi) - \sum\nolimits_{j\in J^1} L_j/n_1(\pi')\\
&\quad = (C_i(\pi) - d_i + C_{i'}(\pi) - d_{i'})/n_1\\
&\qquad - (C_{i'}(\pi') - d_{i'} + C_i(\pi') - d_i)/n_1\\
&\quad = (S_i(\pi) + p_i^1 - d_i + S_i(\pi) + p_i^1 + p_{i'}^1 - d_{i'})/n_1\\
&\qquad - (S_{i'}(\pi') + p_{i'}^1 - d_{i'} + S_{i'}(\pi') + p_{i'}^1 + p_i^1 - d_i)/n_1\\
&\quad = (p_i^1 - p_{i'}^1)/n_1 > 0
\end{aligned}
$$

In other words, the interchange in sequence of jobs i and i' reduces the value of $\sum_{j\in J^1} L_j/n_1$. Therefore, any sequence of the n_1 jobs of the first agent that is not an SPT sequence can be improved with respect to $\sum_{j\in J^1} L_j/n_1$ by interchanging a pair of jobs. It follows that the SPT sequence of the n_1 jobs of the first agent itself must be optimal.

Lemma 2: For all strongly non-dominated solutions of the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem, there exists an optimal schedule for the problem in which all non-tardy jobs in the set J^2 are sequenced in EDD order [24].

Lemma 3: For the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem, there exists a non-delay schedule (one that prohibits unforced idleness) for all strongly non-dominated solutions on the Pareto-optimal frontier [31].

Based on the lemmas presented, the structure of any non-dominated solution of the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem can be denoted by a permutation containing three subset jobs J_{I}, J_{II}, and J_{III}, where J_{I} contains n_1 jobs from J^1 that are sequenced in SPT order, J_{II} contains all non-tardy jobs from J^2 that are sequenced in EDD order and J_{III} contains all tardy jobs from J^2 that are sequenced in any order. The permutation of jobs in J_{I} and J_{II} may be mixed. Because interference only exists between jobs in J_{I} and J_{II}, the jobs in J_{III} can be scheduled after jobs J_{I} and J_{II} without affecting the performance criteria. For example, if $J_I = \{j_3, j_1, j_2\}$, $J_{\text{II}} = \{j_7, j_4, j_5\}$, and $J_{III} = \{j_6, j_8\}$, then a non-dominated solution may be denoted by the permutation $\pi = \{j_3, j_7, j_4, j_1, j_2, j_5, j_6, j_8\}$.

Consider the graph shown in Fig. 1, which represents the efficient Pareto frontier for the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem. The points $\pi_0, \pi_1, \ldots, \pi_{k-1}, \pi_k$ in the graph denote the strongly non-dominated solutions on the efficient Pareto frontier, in which π_0 gives the minimum number of tardy jobs for n_2 jobs in the set J^2, and π_k gives the minimum value of mean lateness for n_1 jobs in the set J^1. The points $\pi_{0'}$ and $\pi_{k'}$ are weakly non-dominated solutions. The minimum value for $\sum_{j\in J^2} U_j$ of the boundary solution π_0 can be obtained by sequencing the n_2 jobs in the set J^2 using Moore's algorithm,

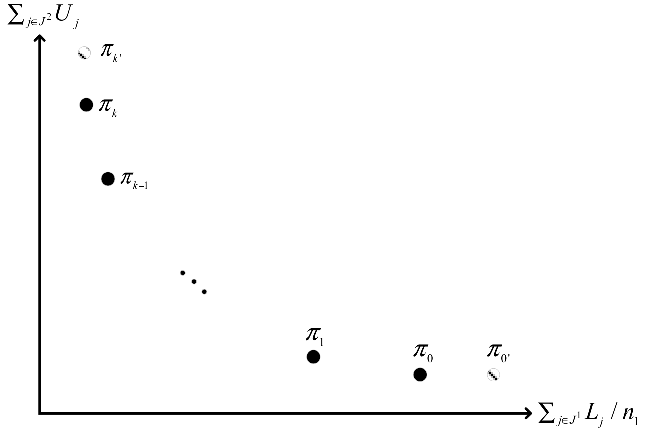


FIGURE 1. **Efficient Pareto frontier for the 1 $\|ND\left(\sum_{j\in J^1} L_j/n_1, \Sigma_{j\in J^2} U_j\right)$ problem.**

and the minimum $\sum_{j\in J^1} L_j/n_1$ value of the boundary solution π_k can be obtained by sequencing the n_1 jobs in set J^1 in SPT order.

B. PROCEDURE OF THE SPT-M ALGORITHM

Based on Lemmas 1–3 and the discussion presented on the structure of non-dominated solutions and the efficient Pareto frontier, this study proposes an effective and efficient SPT-M algorithm to generate the non-dominated solutions of the Pareto set for the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem. The proposed SPT-M algorithm attempts to generate the non-dominated solutions from the initial boundary solution π_0 and then sequentially finds the subsequent non-dominated solution by moving the job in the set J_{II} with the longest processing time to set J_{III} until the final boundary solution π_k is achieved. The reason for moving the job in the set J_{II} with the longest processing time to the set J_{III} is to get a maximum decrease in $\sum_{j\in J^1} L_j/n_1$ by adding one tardy job of the set J^2. The detailed steps of the proposed SPT-M algorithm are explained as follows.

Step 1. Initialize

Input data of the first and second agents' jobs in sets J^1 and J^2, respectively.

Step 2. Calculate the minimum objective function values of the two boundary solutions

Sequence the jobs in sets J^1 and J^2 alone according to the SPT rule and EDD rule (see Fig. 2 for the detailed procedure), respectively, to calculate the minimum objective function values for $\sum_{j\in J^1} L_j/n_1$ and $\sum_{j\in J^2} U_j$ of the boundary solutions π_k and π_0, respectively.

Step 3: Obtain the initial non-dominated solution

Use the sequences obtained in step 2 to set initial subsets J_I, J_{II}, and J_{III}, where the jobs in the set J_{II} are rearranged in EDD order. Identify the initial non-dominated solution π_0 by placing the jobs in set J_{II} on the scheduling horizon in a manner that ensures that they are all completed on their due dates. Subsequently, use the insertion mechanism (see Fig. 3) to individually insert the jobs in the set J_I into the remaining available period of the time horizon

```
Moore's algorithm
Step 1: Arrange the $n_2$ jobs in set $J^2$ in EDD order. Let $T$ denotes the completion time of all jobs in $J_{\text{II}}$,
        Initially, $J_{\text{II}} = \varnothing$, $J_{\text{III}} = \varnothing$, and $T = 0$.
Step 2: If all jobs have been examined, go to step 6; otherwise selected the first unexamined job, call it job $j$, in
        the order established in step 1. Let $T_{temp} = T + p_j^2$. If $T_{temp} \le d_j^2$, go to step 3; otherwise, go to step 4.
Step 3: Add job $j$ into $J_{\text{II}}$, and let $T = T_{temp}$. Go back to step 2.
Step 4: Select the job with a maximum processing time from $J_{\text{II}}$, call it job $j'$. If $p_j^2 < p_{j'}^2$, go to step 5;
        otherwise, add job $j$ into $J_{\text{III}}$ and go to back to step 2.
Step 5: Move job $j'$ from $J_{\text{II}}$ to $J_{\text{III}}$ and add job $j$ into $J_{\text{II}}$, if the job $j$ will be able to complete before its
        due date after replacing $j'$ with $j$. Then, let $T = T_{temp} - p_{j'}^2$ and go back to step 2; otherwise, add job $j$
        into $J_{\text{III}}$ and go back to step 2.
Step 6: If all the jobs have been examined. The set $J_{\text{III}}$ gives the minimum number of tardy jobs.
```

FIGURE 2. **The detailed procedure of Moore's algorithm.**

```
The insertion mechanism
1: Sequence jobs set $J_{\text{I}}$ by the SPT rule.
2: While the processing time of job $i$ is less than the available time between start time to the job $j$ on the time
   horizon, for all jobs $i$ belong to the job set $J_{\text{I}}$, and all jobs $j$ belong to job set $J_{II}$
3: Arrange job $i$ after the earlier job $j$ of $J_{\text{I}}$ and reschedule jobs of the job set $J_{II}$ by the forward rule
4: Endwhile
5: Arrange the unscheduled jobs of $J_{\text{I}}$ after jobs of $J_{II}$.
```

FIGURE 3. **The pseudo code of insertion mechanism.**

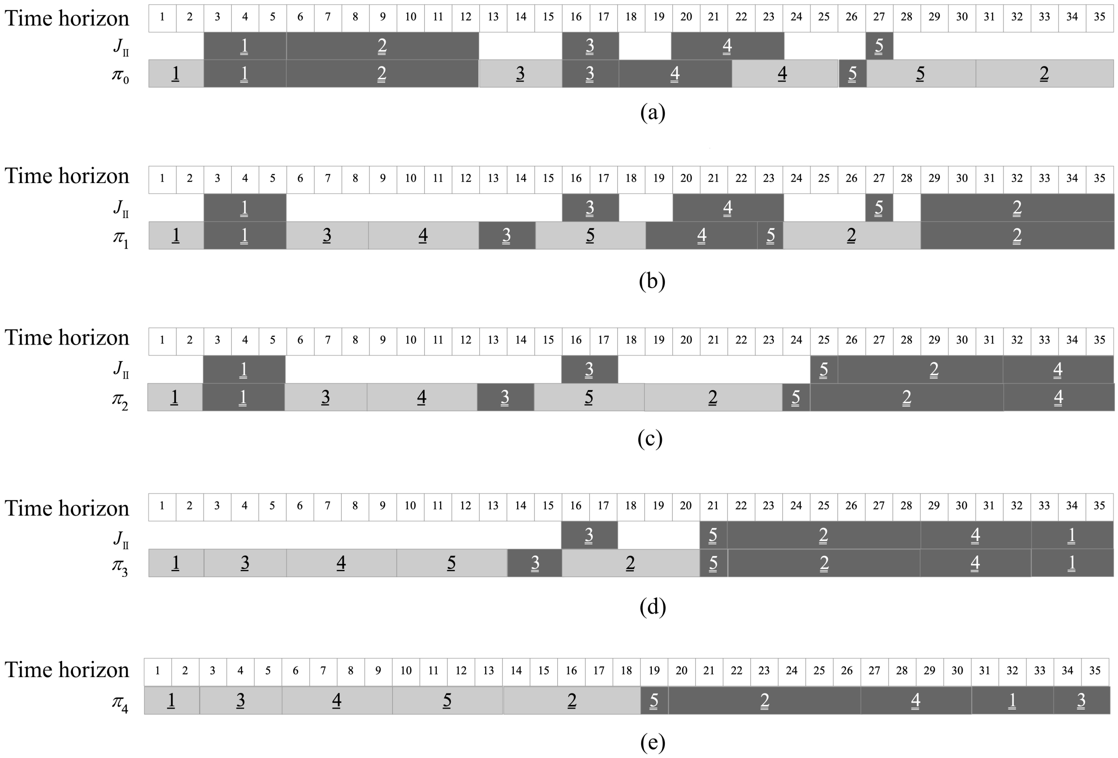


FIGURE 4. Gantt chart for scheduling the numerical example.

that is greater than or equal to the processing time of the inserted job. During the insertion process, after each job in J_I is inserted, the jobs in the set J_{II} are moved forward to avoid unforced idleness of jobs in the set J_I. Finally, place the jobs in set J_{III} on the scheduling horizon after jobs in sets J_I and J_{II}.

Step 4: Obtain the subsequent non-dominated solution

Select the job with the longest processing time from the set J_{II} and move it to set J_{III} using Moore's algorithm. Identify the subsequent non-dominated solution by placing the remaining jobs in set J_{II} on the scheduling horizon in a manner that ensures that all of them are completed on their due dates, and use the insertion mechanism to insert the jobs in the set J_I into the remaining available period of time horizons. Finally, place the jobs in set J_{III} on the scheduling horizon after jobs in sets J_I and J_{II}.

Step 5: Stopping criterion

If the mean lateness value $\sum_{j\in J^1} L_j/n_1$ of the subsequent non-dominated solution is equal to the minimum value obtained in step 2, then the algorithm is terminated; otherwise, return to step 4.

C. NUMERICAL EXAMPLE

Fig. 4 presents the Gantt charts for scheduling the numerical example using the proposed SPT-M algorithm, in which the jobs in the set J^1 and J^2 are indicated in light gray and dark gray, respectively. Using the data presented in Table 1, the detailed steps for solving the numerical example are explained as follows.

Step 1: Initialize

Assume that the first agent has a set J^1 of five jobs and the second agent has a set J^2 of five jobs. The processing time p_j^s and due date d_j^s of each job $j \in J^s$ $(s = 1, 2)$ are given as follows.

Step 2: Calculate the minimum objective function values of the two boundary solutions

Sequence the jobs in set J^1 alone according to the SPT rule; a permutation [$\underline{1}$ $\underline{3}$ $\underline{4}$ $\underline{5}$ $\underline{2}$] with $\sum_{j\in J^1} L_j/n_1 = -7.8$ (see iteration 0 of Table 1) is obtained.

Sequence the jobs in set J^2 alone using Moore's algorithm; a permutation [$\underline{\underline{1}}$ $\underline{\underline{2}}$ $\underline{\underline{3}}$ $\underline{\underline{4}}$ $\underline{\underline{5}}$] with $\sum_{j\in J^2} U_j = 0$ (see iteration 0 of Table 1) is obtained.

Therefore, the minimum objective function values of $\sum_{j\in J^1} L_j/n_1 = -7.8$ and $\sum_{j\in J^2} U_j = 0$ of the boundary solutions π_k and π_0 are set, respectively. As a result, the algorithm is terminated when a non-dominated solution with $\sum_{j\in J^1} L_j/n_1 = -7.8$ is obtained.

Step 3: Obtain the initial non-dominated solution

Use the sequences obtained in step 2 to set subsets $J_I = \{\underline{1}\ \underline{3}\ \underline{4}\ \underline{5}\ \underline{2}\}$, $J_{\mathrm{II}} = \{\underline{\underline{1}}\ \underline{\underline{2}}\ \underline{\underline{3}}\ \underline{\underline{4}}\ \underline{\underline{5}}\ \}$, and $J_{\mathrm{III}} = \{\text{null}\}$ of the initial non-dominated solution.

Iteration 1^{st}:

As shown in Fig. 4(a) and Table 1, the initial non-dominated solution π_0 =[$\underline{1}$ $\underline{\underline{1}}$ $\underline{\underline{2}}$ $\underline{3}$ $\underline{\underline{3}}$ $\underline{\underline{4}}$ $\underline{4}$ $\underline{\underline{5}}$ $\underline{5}$ $\underline{2}$] with

TABLE 1. Detailed data of the numerical example.

	J^1						J^2					
Iteration No.	j	p_j^1	d_j^1	C_j^1	L_j^1	$\sum_{j\in J^1} L_j / n_1$	j	p_j^2	d_j^2	C_j^2	U_j^2	$\sum_{j\in J^2} U_j$
0	1	2	5	2	- 3		1	3	5	2	0	
	3	3	17	5	- 12		2	7	12	5	0	
	4	4	23	9	- 14		3	2	17	12	0	
	5	4	28	13	- 15		4	4	23	19	0	
	2	5	13	18	5	- 7.8	5	1	27	26	0	0
1	1	2	5	2	- 3		1	3	5	5	0	
	3	3	17	15	- 2		2	7	12	12	0	
	4	4	23	25	2		3	2	17	17	0	
	5	4	28	30	2		4	4	23	21	0	
	2	5	13	35	22	4.2	5	1	27	26	0	0
2	1	2	5	2	- 3		1	3	5	5	0	
	3	3	17	8	- 9		3	2	17	14	0	
	4	4	23	12	- 11		4	4	23	22	0	
	5	4	28	18	- 10		5	1	27	23	0	
	2	5	13	28	15	- 3.6	2	7	12	35	1	1
3	1	2	5	2	- 3		1	3	5	5	0	
	3	3	17	8	- 9		3	2	17	14	0	
	4	4	23	12	- 11		5	1	27	24	0	
	5	4	28	18	- 10		2	7	12	31	1	
	2	5	13	23	10	- 4.6	4	4	23	35	1	2
4	1	2	5	2	- 3		3	2	17	15	0	
	3	3	17	5	- 12		5	1	27	21	0	
	4	4	23	9	- 14		2	7	12	28	1	
	5	4	28	13	- 15		4	4	23	32	1	
	2	5	13	20	7	- 7.4	1	3	5	35	1	3
5	1	2	5	2	- 3		5	1	27	19	0	
	3	3	17	5	- 12		2	7	12	26	1	
	4	4	23	9	- 14		4	4	23	30	1	
	5	4	28	13	- 15		1	3	5	33	1	
	2	5	13	18	5	- 7.8	3	2	17	35	1	4

$\sum_{j\in J^1} L_j/n_1 = -4.2$ and $\sum_{j\in J^2} U_j = 0$ can be found using the algorithm. After inserting jobs 3 and 4 of J_I, jobs 4 and 5 of J_{II} are moved forward, respectively, to create a non-delay schedule.

Step 4: Obtain the subsequent non-dominated solution

Iteration 2^{nd}:

As shown in Fig. 4(b) and Table 1, the subsequent non-dominated solution $\pi_1 = [\underline{1}\ \underline{\underline{1}}\ \underline{3}\ \underline{4}\ \underline{\underline{3}}\ \underline{5}\ \underline{\underline{4}}\ \underline{\underline{5}}\ \underline{2}\ \underline{\underline{2}}\ ]$ with $\sum_{j\in J^1} L_j/n_1 = -3.6$ and $\sum_{j\in J^2} U_j = 1$ can be determined using the algorithm after moving job 2 from J_{II} to J_{III}. After inserting jobs 3 and 4 of J_I, job 3 of J_{II} is moved forward, and after inserting job 5 of J_I, jobs 4 and 5 of J_{II} are moved forward to create a non-delay schedule.

Iteration 3^{rd}:

As shown in Fig. 4(c) and Table 1, the subsequent non-dominated solution $\pi_2 = [\underline{1}\ \underline{\underline{1}}\ \underline{3}\ \underline{4}\ \underline{\underline{3}}\ \underline{5}\ \underline{2}\ \underline{\underline{5}}\ \underline{\underline{2}}\ \underline{\underline{4}}\ ]$ with $\sum_{j\in J^1} L_j/n_1 = -4.6$ and $\sum_{j\in J^2} U_j = 2$ can be obtained using the algorithm after moving job 4 from J_{II} to J_{III}. After inserting jobs 3 and 4 of J_I, job 3 of J_{II} is moved forward to create a non-delay schedule.

Iteration 4^{th}:

As shown in Fig. 4(d) and Table 1, the subsequent non-dominated solution $\pi_3 = [\underline{1}\ \underline{3}\ \underline{4}\ \underline{5}\ \underline{\underline{3}}\ \underline{2}\ \underline{\underline{5}}\ \underline{\underline{2}}\ \underline{\underline{4}}\ \underline{\underline{1}}\ ]$ with $\sum_{j\in J^1} L_j/n_1 = -7.4$ and $\sum_{j\in J^2} U_j = 3$ can be obtained using the algorithm after moving job 1 from J_{II} to J_{III}. After inserting jobs 1, 3, 4, and 5 of J_I, job 3 of J_{II} is moved forward to create a non-delay schedule.

Iteration 5^{th}:

As shown in Fig. 4(e) and Table 1, the subsequent non-dominated solution $\pi_4 = [\underline{1}\ \underline{3}\ \underline{4}\ \underline{5}\ \underline{2}\ \underline{\underline{5}}\ \underline{\underline{2}}\ \underline{\underline{4}}\ \underline{\underline{1}}\ \underline{\underline{3}}\ ]$ with $\sum_{j\in J^1} L_j/n_1 = -7.8$ and $\sum_{j\in J^2} U_j = 4$ can be identified using the algorithm after moving job 3 from J_{II} to J_{III}. The algorithm is terminated when a non-dominated solution with $\sum_{j\in J^1} L_j/n_1 = -7.8$, that is, the stopping criterion has been achieved. Eventually, five strongly non-dominated solutions, (4.2, 0), (−3.6,1), (−4.6, 2), (−7.4, 3), and (−7.8, 4), are obtained by using the SPT-M algorithm. Then, decision makers could choose one of these non-dominated solutions to arrange jobs of two customers (agents) based on their trading contracts.

IV. EXPERIMENTAL RESULTS

This section compares the experimental results obtained by applying the proposed SPT-M algorithm on a set of test instances with those obtained using the complete enumeration (CE) method. The following subsections elucidate the test instances, performance indices, and experimental results of this study.

A. TEST INSTANCES

The test problem set has 120 randomly generated test instances, which has 24 combinations of the number of jobs and range of due dates. Each agent has two types of numbers of jobs: symmetric and asymmetric. In the symmetric type, each agent has four values for the number of jobs: 6, 7, 10, and 15. In the asymmetric type, the first and second agents have two values for the number of jobs: (n_1, n_2) =(6, 18) and (18, 6). Therefore, the two agents have a total of six values for the number of jobs: (6, 6), (7, 7), (10, 10), (15, 15), (6, 18), and (18, 6) jobs. The processing time p_j^s of each job $j \in J^s$ $(s = 1, 2)$ is an integer that is randomly generated from the discrete uniform distribution $U[1, 20]$.

The range of due dates is generated using the approach adopted by Abdul-Razaq *et al.* [42] and Keha *et al.* [43]. To be more specific, the due date d_j^s of each job $j \in J^s$ $(s = 1, 2)$ is an integer that is randomly generated from the discrete uniform distribution $U[P(L - R/2), P(L + R/2)]$, where the first parameter combination is $P = 0.5 \times \sum_j p_j^1 + \sum_j p_j^2$ and $L \in \{0.5, 0.7\}$, and the second parameter combination is $P = \sum_j p_j^1 + 0.5 \times \sum_j p_j^2$ and $L \in \{0.4, 0.8\}$. Therefore, four ranges of due dates are identified: $[0.3P, 0.7P]$, $[0.1P, 0.9P]$, $[0.5P, 0.9P]$, and $[0.3P, 1.1P]$. To evaluate the computational efficiency and solution quality of the proposed algorithm, five test instances are generated for each combination of the number of jobs and range of due dates, yielding a total of $6 \times 4 \times 5 = 120$ test instances for the experiment.

B. PERFORMANCE INDICES

To evaluate the solution quality of the proposed SPT-M algorithm and the CE method, we adopted the following six indices that have commonly been used [44]–[46].

- A: the number of non-dominated solutions obtained using a certain approach.
- B: the number of non-dominated solutions in net non-dominated Pareto set obtained using a certain approach.
- B/A: the quotient between B and A (i.e., the ratio of non-dominated solutions in the net non-dominated Pareto set obtained using a certain approach).
- C: the coverage rate of the two sets of non-dominated solutions obtained using two approaches. Let Π_1 and Π_2 be two sets of non-dominated solutions obtained using two approaches, respectively. Thus, C maps the ordered pair (Π_1, Π_2) into the interval [0, 1] as follows:

$$C(\Pi_1, \Pi_2) = \frac{|\pi_a \in \Pi_1/\exists \pi_b \in \Pi_2 : \pi_a \succ \pi_b|}{|\Pi_1|}$$

Accordingly, a larger value of C reflects favorable non-dominated solutions of Π_1. If all non-dominated solutions in Π_2 are dominated by Π_1, then $C(\Pi_1, \Pi_2) = 1$; however, if none of the non-dominated solutions in Π_2 are dominated by Π_1, then $C(\Pi_1, \Pi_2) = 0$.

- Υ: the convergence metric, which denotes the mean value of the minimum Euclidean distances of all non-dominated solutions obtained using a certain approach to net non-dominated solutions. A Υ value closer to zero indicates a higher convergence level of the obtained non-dominated solutions.
- Ω: the dominance metric, which represents the percentage of the number of solutions in the net non-dominated Pareto set obtained using a certain approach.

C. RESULTS AND DISCUSSION

The proposed SPT-M algorithm and CE method were coded using Visual C++ and run on a personal computer with an Intel Core i3-4030U 1.9-GHz CPU with 4 GB random-access memory and on the Visual Studio 2015 integrated-development-environment software verification platform. The maximum computational time of the CE method was set at 3,600 s. Tables 2–7 summarize the six performance indices obtained using the proposed SPT-M algorithm and the compared CE method, in which the average (Ave), minimum (Min), and maximum (Max) values for the five test instances are provided.

As shown in Table 2, the proposed SPT-M algorithm is more effective than the CE method in terms of the six performance indices for the test instances with (6, 6) jobs. The total average of the Ave, Min, and Max values obtained using the proposed SPT-M algorithm with respect to the performance indices A, B, B/A, C, Υ, and Ω are (6.200, 5.000, 6.750), (4.850, 3.500, 6.500), (0.791, 0.529, 1.000), (0.742, 0.479, 1.000), (0.420, 0.000, 1.103), and (0.833, 0.607, 1.000), respectively. The corresponding values for the CE method are (4.550, 3.250, 6.000), (2.450, 1.250, 4.750), (0.514, 0.242, 0.810), (0.404, 0.2030, 0.661), (1.943, 0.555, 4.516), and (0.403, 0.203, 0.705), respectively. The proposed SPT-M algorithm demonstrated favorable performance compared with the CE method across most Ave, Min, and Max values with respect to the performance indices A, B, B/A, C, Υ, and Ω. The analytical results of these performance indices also reveal that the majority of solutions in the net non-dominated Pareto set were obtained using the SPT-M algorithm. Furthermore, the non-dominated solutions obtained using the SPT-M algorithm exhibited high coverage rates and convergence levels.

Similar statistical results were also achieved for the test instances with (7, 7), (10, 10), (15, 15), (6, 18), and (18, 6) jobs. As indicated in Tables 3–7, the performance

TABLE 2. Performance comparison between SPT-M and CE in test instances with (6, 6) jobs.

Performance Index		Range of Due Dates								Total Average	
		[0.3P, 0.7P]		[0.1P, 0.9P]		[0.5P, 0.9P]		[0.3P, 1.1P]			
		SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE
A	Ave	5.600	4.800	6.600	4.800	6.400	4.800	6.200	3.800	6.200	4.550
	Min	5.000	4.000	6.000	4.000	5.000	3.000	4.000	2.000	5.000	3.250
	Max	6.000	5.000	7.000	6.000	7.000	6.000	7.000	7.000	6.750	6.000
B	Ave	4.200	3.200	6.000	2.400	5.200	2.400	4.000	1.800	4.850	2.450
	Min	2.000	3.000	5.000	1.000	4.000	1.000	3.000	0.000	3.500	1.250
	Max	6.000	4.000	7.000	5.000	7.000	4.000	6.000	6.000	6.500	4.750
B/A	Ave	0.740	0.670	0.914	0.477	0.824	0.503	0.686	0.405	0.791	0.514
	Min	0.400	0.600	0.714	0.200	0.571	0.167	0.429	0.000	0.529	0.242
	Max	1.000	0.800	1.000	0.833	1.000	0.750	1.000	0.857	1.000	0.810
C	Ave	0.740	0.573	0.910	0.400	0.667	0.330	0.652	0.314	0.743	0.404
	Min	0.500	0.500	0.750	0.143	0.333	0.167	0.333	0.000	0.479	0.203
	Max	1.000	0.667	1.000	0.833	1.000	0.571	1.000	0.571	1.000	0.661
Υ	Ave	0.659	1.615	0.057	2.195	0.398	1.596	0.564	2.365	0.420	1.943
	Min	0.000	0.500	0.000	1.097	0.000	0.111	0.000	0.512	0.000	0.555
	Max	1.600	2.267	0.238	3.769	1.250	4.708	1.325	7.319	1.103	4.516
Ω	Ave	0.760	0.603	0.938	0.381	0.867	0.392	0.766	0.234	0.833	0.403
	Min	0.500	0.500	0.833	0.143	0.667	0.167	0.429	0.000	0.607	0.203
	Max	1.000	0.750	1.000	0.833	1.000	0.667	1.000	0.571	1.000	0.705

TABLE 3. Performance comparison between SPT-M and CE in test instances with (7, 7) jobs.

Performance Index		Range of Due Dates								Total Average	
		[0.3P, 0.7P]		[0.1P, 0.9P]		[0.5P, 0.9P]		[0.3P, 1.1P]			
		SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE
A	Ave	7.000	3.800	6.000	3.600	7.400	2.800	6.600	2.400	6.700	3.150
	Min	6.000	3.000	5.000	3.000	6.000	2.000	5.000	2.000	5.500	2.500
	Max	8.000	5.000	8.000	5.000	8.000	4.000	8.000	3.000	8.000	4.250
B	Ave	6.600	1.600	5.800	0.600	7.000	1.600	4.600	1.000	6.000	1.200
	Min	5.000	1.000	5.000	0.000	6.000	1.000	4.000	0.000	5.000	0.500
	Max	8.000	4.000	7.000	3.000	8.000	2.000	5.000	2.000	7.000	2.750
B/A	Ave	0.943	0.393	0.975	0.120	0.950	0.617	0.717	0.467	0.896	0.399
	Min	0.714	0.250	0.875	0.000	0.750	0.250	0.625	0.000	0.741	0.125
	Max	1.000	0.800	1.000	0.600	1.000	1.000	1.000	0.000	1.000	0.600
C	Ave	0.920	0.258	0.960	0.120	0.833	0.274	0.733	0.467	0.862	0.280
	Min	0.600	0.125	0.875	0.000	0.500	0.125	0.500	0.000	0.619	0.063
	Max	1.000	0.714	1.000	0.375	1.000	0.375	1.000	0.500	1.000	0.491
Υ	Ave	0.029	1.686	0.014	2.745	0.071	0.937	0.349	1.305	0.116	1.668
	Min	0.000	0.000	0.000	1.541	0.000	0.000	0.000	0.000	0.000	0.385
	Max	0.143	3.478	0.071	4.325	0.196	1.429	0.500	2.286	0.228	2.880
Ω	Ave	0.943	0.258	0.975	0.075	0.916	0.241	0.893	0.230	0.932	0.201
	Min	0.714	0.125	0.875	0.000	0.833	0.111	0.800	0.000	0.806	0.059
	Max	1.000	0.714	1.000	0.375	1.000	0.333	1.000	0.333	1.000	0.456

indices of the proposed SPT-M algorithm are considerably favorable compared with those of the CE method in terms of the number of obtained non-dominated solutions, number of obtained non-dominated solutions in net non-dominated Pareto set, ratio of obtained non-dominated solutions in the net non-dominated Pareto set, coverage rate of the obtained non-dominated solutions, convergence rate of the obtained non-dominated solutions, and percentage of the number of

TABLE 4. Performance comparison between SPT-M and CE in test instances with (10, 10) jobs.

Performance Index		Range of Due Dates									
		[0.3P, 0.7P]		[0.1P, 0.9P]		[0.5P, 0.9P]		[0.3P, 1.1P]		Total Average	
		SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE
A	Ave	10.000	1.000	10.600	1.000	9.200	1.000	8.000	1.000	9.450	1.000
	Min	8.000	1.000	10.000	1.000	6.000	1.000	6.000	1.000	7.500	1.000
	Max	11.000	1.000	11.000	1.000	11.000	1.000	10.000	1.000	10.750	1.000
B	Ave	10.000	0.200	10.600	0.000	9.200	0.200	7.600	0.200	9.350	0.150
	Min	8.000	0.000	10.000	0.000	6.000	0.000	4.000	0.000	7.000	0.000
	Max	11.000	2.000	11.000	0.000	11.000	1.000	10.000	1.000	10.750	1.000
B/A	Ave	1.000	0.200	1.000	0.000	1.000	0.200	0.933	0.200	0.983	0.150
	Min	1.000	0.000	1.000	0.000	1.000	0.000	0.667	0.000	0.917	0.000
	Max	1.000	1.000	1.000	0.000	1.000	1.000	1.000	1.000	1.000	0.750
C	Ave	1.000	0.020	1.000	0.000	1.000	0.018	0.800	0.067	0.950	0.026
	Min	1.000	0.000	1.000	0.000	1.000	0.000	0.000	0.000	0.750	0.000
	Max	1.000	0.100	1.000	0.000	1.000	0.091	1.000	0.333	1.000	0.131
Υ	Ave	0.000	2.068	0.000	3.266	0.000	3.044	0.075	2.683	0.019	2.765
	Min	0.000	0.000	0.000	2.500	0.000	0.000	0.000	0.000	0.000	0.625
	Max	0.000	5.036	0.000	5.000	0.000	4.614	0.377	5.883	0.094	5.133
Ω	Ave	1.000	0.020	1.000	0.000	1.000	0.022	0.960	0.004	0.990	0.012
	Min	1.000	0.000	1.000	0.000	1.000	0.000	0.800	0.000	0.950	0.000
	Max	1.000	0.100	1.000	0.000	1.000	0.111	1.000	0.200	1.000	0.103

TABLE 5. Performance comparison between SPT-M and CE in test instances with (15, 15) jobs.

Performance Index		Range of Due Dates									
		[0.3P, 0.7P]		[0.1P, 0.9P]		[0.5P, 0.9P]		[0.3P, 1.1P]		Total Average	
		SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE
A	Ave	14.800	1.000	13.000	1.000	14.000	1.000	12.000	1.000	13.450	1.000
	Min	13.000	1.000	10.000	1.000	11.000	1.000	9.000	1.000	10.750	1.000
	Max	16.000	1.000	15.000	1.000	16.000	1.000	15.000	1.000	15.500	1.000
B	Ave	14.800	0.000	13.000	0.200	14.000	0.200	12.000	0.000	13.450	0.100
	Min	13.000	0.000	10.000	0.000	11.000	0.000	9.000	0.000	10.750	0.000
	Max	16.000	0.000	15.000	1.000	16.000	1.000	15.000	0.000	15.500	0.500
B/A	Ave	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Min	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Max	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
C	Ave	1.000	0.000	1.000	0.200	0.800	0.200	1.000	0.000	0.950	0.100
	Min	1.000	0.000	1.000	0.000	0.000	0.000	1.000	0.000	0.750	0.000
	Max	1.000	0.000	1.000	1.000	1.000	1.000	1.000	0.000	1.000	0.500
Υ	Ave	0.000	7.528	0.000	6.594	0.000	8.256	0.000	8.720	0.000	7.775
	Min	0.000	3.801	0.000	2.236	0.000	6.017	0.000	6.991	0.000	4.761
	Max	0.000	10.011	0.000	8.885	0.000	10.929	0.000	10.853	0.000	10.170
Ω	Ave	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Min	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Max	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000

obtained solutions in the net non-dominated Pareto set. Evidently, the quality of the solution provided by the proposed SPT-M algorithm was superior to that of the CE method on the test problem set.

Table 8 compares the computational efficiency (average CPU times) of the proposed SPT-M algorithm with that of the CE method, thus revealing that SPT-M is a highly efficient algorithm that can solve each test instance

TABLE 6. Performance comparison between SPT-M and CE in test instances with (6, 18) jobs.

Performance Index		Range of Due Dates									
		$[0.3P, 0.7P]$		$[0.1P, 0.9P]$		$[0.5P, 0.9P]$		$[0.3P, 1.1P]$		Total Average	
		SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE
A	Ave	10.400	2.400	8.600	2.200	8.000	1.600	10.200	1.400	9.300	1.900
	Min	9.000	1.000	7.000	1.000	5.000	1.000	8.000	1.000	7.250	1.000
	Max	12.000	4.000	10.000	3.000	13.000	3.000	13.000	2.000	12.500	3.000
B	Ave	9.600	0.800	7.800	0.600	8.000	0.000	9.800	0.600	8.800	0.500
	Min	9.000	0.000	6.000	0.000	5.000	0.000	8.000	0.000	7.000	0.000
	Max	11.000	1.000	10.000	1.000	13.000	0.000	13.000	1.000	11.750	0.750
B/A	Ave	0.930	0.383	1.906	0.367	1.000	0.000	0.964	0.400	1.200	0.288
	Min	0.833	0.000	0.750	0.000	1.000	0.000	0.818	0.000	0.850	0.000
	Max	1.000	1.000	1.000	1.000	1.000	0.000	1.000	1.000	1.000	0.750
C	Ave	0.817	0.092	0.633	0.094	0.840	0.015	0.600	0.072	0.723	0.068
	Min	0.667	0.000	0.000	0.000	0.200	0.000	0.000	0.000	0.217	0.750
	Max	1.000	0.167	1.000	0.250	1.000	0.077	1.000	0.182	1.000	0.169
Υ	Ave	0.056	2.283	0.149	3.237	0.000	3.025	0.040	1.445	0.061	2.498
	Min	0.000	0.000	0.000	0.000	0.000	0.901	0.000	0.000	0.000	0.225
	Max	0.100	5.011	0.333	4.567	0.000	6.000	0.197	2.242	0.1575	4.455
Ω	Ave	0.965	0.057	0.927	0.073	1.000	0.015	0.965	0.055	0.964	0.050
	Min	0.909	0.000	0.857	0.000	1.000	0.000	0.900	0.000	0.917	0.000
	Max	1.000	0.111	1.000	0.143	1.000	0.077	1.000	0.100	1.000	0.108

TABLE 7. Performance comparison between SPT-M and CE in test instances with (18, 6) jobs.

Performance Index		Range of Due Dates									
		$[0.3P, 0.7P]$		$[0.1P, 0.9P]$		$[0.5P, 0.9P]$		$[0.3P, 1.1P]$		Total Average	
		SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE	SPT-M	CE
A	Ave	7.000	1.000	7.000	1.000	6.800	1.000	5.200	1.000	6.500	1.000
	Min	7.000	1.000	7.000	1.000	6.000	1.000	5.000	1.000	6.250	1.000
	Max	7.000	1.000	7.000	1.000	7.000	1.000	6.000	1.000	6.750	1.000
B	Ave	7.000	0.000	7.000	0.200	6.800	0.200	5.200	0.000	6.500	0.100
	Min	7.000	0.000	7.000	0.000	6.000	0.000	5.000	0.000	6.250	0.000
	Max	7.000	0.000	7.000	1.000	7.000	1.000	6.000	0.000	6.750	0.500
B/A	Ave	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Min	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Max	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
C	Ave	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Min	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Max	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
Υ	Ave	0.000	8.322	0.000	8.490	0.000	13.812	0.200	11.347	0.050	10.493
	Min	0.000	5.307	0.000	6.065	0.000	5.860	0.000	4.603	0.000	5.459
	Max	0.000	16.360	0.000	13.268	0.000	19.290	1.000	29.592	0.250	19.628
Ω	Ave	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Min	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000
	Max	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000	1.000	0.000

within approximately 0.2 s, whereas the CE method cannot solve any test instances to optimality within the maximum computing time. Consequently, we can conclude that the proposed SPT-M algorithm is remarkably effective for identifying high-quality non-dominated solutions for the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem.

TABLE 8. Comparison of computational times in different test instances.

		Range of Due Dates				
Instance	Algorithms	[0.3P, 0.7P]	[0.1P, 0.9P]	[0.5P, 0.9P]	[0.3P, 1.1P]	Total Average
(6, 6)	SPT-M	0.016	0.018	0.012	0.016	0.016
	CE	3600.000	3600.000	3600.000	3600.000	3600.000
(7, 7)	SPT-M	0.014	0.071	0.119	0.036	0.060
	CE	3600.000	3600.000	3600.000	3600.000	3600.000
(10, 10)	SPT-M	0.041	0.134	0.033	0.159	0.108
	CE	3600.000	3600.000	3600.000	3600.000	3600.000
(15, 15)	SPT-M	0.048	0.044	0.048	0.054	0.049
	CE	3600.000	3600.000	3600.000	3600.000	3600.000
(6, 18)	SPT-M	0.037	0.042	0.026	0.027	0.033
	CE	3600.000	3600.000	3600.000	3600.000	3600.000
(18, 6)	SPT-M	0.023	0.029	0.043	0.100	0.049
	CE	3600.000	3600.000	3600.000	3600.000	3600.000

V. CONCLUSION

Most practical scheduling problems inherently present multi-objective characteristics; however, relatively few studies have considered such practical characteristics. In this study, we formalize and examine the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem and propose an effective and efficient SPT-M algorithm for obtaining its solution. Experimental results confirm that the proposed SPT-M algorithm outperforms the CE method in terms of the six multi-objective performance indices. Because the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem investigated herein often arises in practical manufacturing systems, this study contributes an effective and efficient algorithm for satisfying real-world scheduling requirements.

Many extension lines in the addressed two-agent SMSPs merit further research. First, designing efficient exact methods for enumerating all of the Pareto optima on the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem are topics that should be addressed in future studies. Second, creating effective and efficient meta-heuristics for solving the $1\|ND(\sum_{j\in J^1} L_j/n_1, \sum_{j\in J^2} U_j)$ problem also constitutes a promising avenue of future research. Third, extensions of the two-agent SMSP involving different performance criteria also serve as a rich area for investigation. Two-agent SMSPs with stochastic processing times are complex, but certainly worthy of further study; moreover, scholars should consider the two-agent SMSPs with special processing constraints such as setup time and breakdown. Finally, three-agent and multi-agent SMSPs pose new challenges for scheduling algorithm design and implementation.

REFERENCES

[1] H. Hoogeveen, "Multicriteria scheduling," *Eur. J. Oper. Res.*, vol. 167, no. 3, pp. 592–623, May 2005.

[2] D. Lei, "Multi-objective production scheduling: A survey," *Int. J. Adv. Manuf. Technol.*, vol. 43, nos. 9–10, pp. 926–938, Aug. 2009.

[3] Q. Peng, M. Zhou, Q. He, Y. Xia, C. Wu, and S. Deng, "Multi-objective optimization for location prediction of mobile devices in sensor-based applications," *IEEE Access*, vol. 6, pp. 77123–77132, 2018.

[4] G. Tian, M. Zhou, and P. Li, "Disassembly sequence planning considering fuzzy component quality and varying operational cost," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 748–760, Apr. 2018.

[5] S. W. Lin and K. C. Yin, "Increasing the total net revenue for single machine order acceptance and scheduling problems using an artificial bee colony algorithm," *J. Oper. Res. Soc.*, vol. 64, no. 2, pp. 293–311, Feb. 2013.

[6] K. R. Baker and J. C. Smith, "A multiple-criterion model for machine scheduling," *J. Sched.*, vol. 6, no. 1, pp. 7–16, Jan. 2003.

[7] C. C. Wu, J. Y. Chen, W. C. Lin, K. Lai, D. Bai, and S. Y. Lai, "A two-stage three-machine assembly scheduling flowshop problem with both two-agent and learning phenomenon," *Comput. Ind. Eng.*, vol. 130, pp. 485–499, Apr. 2019.

[8] W. C. Lin, Y. Yin, S. R. Cheng, T. C. E. Cheng, C. H. Wu, and C. C. Wu, "Particle swarm optimization and opposite-based particle swarm optimization for two-agent multi-facility customer order scheduling with ready times," *Appl. Soft Comput.*, vol. 52, pp. 877–884, Mar. 2017.

[9] Y. Yin, S.-R. Cheng, T. C. E. Cheng, D.-J. Wang, and C.-C. Wu, "Just-in-time scheduling with two competing agents on unrelated parallel machines," *Omega*, vol. 63, pp. 41–47, Sep. 2016.

[10] Y. Yin, T. C. E. Cheng, D. J. Wang, and C. C. Wu, "Two-agent flowshop scheduling to maximize the weighted number of just-in-time jobs," *J. Scheduling*, vol. 20, no. 4, pp. 313–335, Aug. 2017.

[11] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. R. Kan, "Optimization and approximation in deterministic sequencing and scheduling: A survey," *Ann. Math.*, vol. 5, pp. 287–326, 1979.

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

[13] A. Nagar, J. Haddock, and S. Heragu, "Multiple and bicriteria scheduling: A literature survey," *Eur. J. Oper. Res.*, vol. 81, no. 1, pp. 88–104, Feb. 1995.

[14] T. Sen and S. K. Gupta, "A branch-and-bound procedure to solve a bicriterion scheduling problem," *IIE Trans.*, vol. 15, no. 1, pp. 84–88, Mar. 1983.

[15] L. N. Van Wassenhove and K. R. Baker, "A bicriterion approach to time/cost trade-offs in sequencing," *Eur. J. Oper. Res.*, vol. 11, no. 1, pp. 48–54, Sep. 1982.

[16] M. Köksalan and A. B. Keha, "Using genetic algorithms for single-machine bicriteria scheduling problems," *Eur. J. Oper. Res.*, vol. 145, no. 3, pp. 543–556, Mar. 2003.

[17] M. Azizoglu, S. Kondakci, and M. Köksalan, "Single machine scheduling with maximum earliness and number tardy," *Comput. Ind. Eng.*, vol. 45, no. 2, pp. 257–268, Aug. 2003.

[18] F. Jolai, M. Rabbani, S. Amalnick, A. Dabaghi, M. Dehghan, and M. Y. Parast, "Genetic algorithm for bi-criteria single machine scheduling problem of minimizing maximum earliness and number of tardy jobs," *Appl. Math. Comput.*, vol. 194, no. 2, pp. 552–560, Dec. 2007.

[19] U. Haral, R. W. Chen, W. G. J. Ferrell, and M. B. Kurz, "Multiobjective single machine scheduling with nontraditional requirements," *Int. J. Prod. Econ.*, vol. 106, no. 2, pp. 484–574, Apr. 2007.

[20] T. Eren and E. Güner, "A bicriteria scheduling with sequence-dependent setup times," *Appl. Math. Comput.*, vol. 179, no. 1, pp. 378–385, 2006.

[21] W. J. Chen, "An efficient algorithm for scheduling jobs on a machine with periodic maintenance," *Int. J. Adv. Manuf. Technol.*, vol. 34, nos. 11–12, pp. 1173–1182, Nov. 2007.

[22] A. Agnetis, P. B. Mirchandani, D. Pacciarelli, and A. Pacifici, "Scheduling problems with two competing agents," *Oper. Res.*, vol. 52, no. 2, pp. 229–242, Apr. 2004.

[23] A. Agnetis, D. Pacciarelli, and A. Pacifici, "Multi-agent single machine scheduling," *Ann. Oper. Res.*, vol. 150, no. 1, pp. 3–15, 2007.

[24] C. T. Ng, T. C. E. Cheng, and J. J. Yuan, "A note on the complexity of the problem of two-agent scheduling on a single machine," *J. Comb. Optim.*, vol. 12, no. 4, pp. 387–394, Dec. 2006.

[25] G. Wan, R. S. Vakati, J. Y. T. Leung, and M. Pinedo, "Scheduling two agents with controllable processing times," *Eur. J. Oper. Res.*, vol. 205, no. 3, pp. 528–539, Sep. 2010.

[26] J. Y. T. Leung, M. Pinedo, and G. H. Wan, "Competitive two-agent scheduling and its applications," *Oper. Res.*, vol. 58, no. 2, pp. 458–469, Apr. 2010.

[27] W. C. Lee, W. J. Wang, Y. R. Shiau, and C. C. Wu, "A single-machine scheduling problem with two-agent and deteriorating jobs," *Appl. Math. Model.*, vol. 34, no. 10, pp. 3098–3107, Oct. 2010.

[28] P. Liu, N. Yi, and X. Zhou, "Two-agent single-machine scheduling problems under increasing linear deterioration," *Appl. Math. Model.*, vol. 35, no. 5, pp. 2290–2296, May 2011.

[29] Y. Q. Yin, S. R. Cheng, and C. C. Wu, "Scheduling problems with two agents and a linear non-increasing deterioration to minimize earliness penalties," *Inf. Sci.*, vol. 189, pp. 282–292, Apr. 2012.

[30] J. M. Moore, "An n job, one machine sequencing algorithm for minimizing the number of late jobs," *Manag. Sci.*, vol. 15, no. 1, 102-109, Sep. 1968.

[31] K. Khowala, J. Fowler, A. Keha, and H. Balasubramanian, "Single machine scheduling with interfering job sets," *Comput. Oper. Res.*, vol. 45, pp. 97–107, May 2014.

[32] D. Oron, D. Shabtay, and G. Steiner, "Single machine scheduling with two competing agents and equal job processing times," *Eur. J. Oper. Res.*, vol. 244, no. 1, pp. 86–99, Jul. 2015.

[33] Y. Yin, D. Wang, C. Wu, and T. C. E. Cheng, "*CON/SLK* due date assignment and scheduling on a single machine with two agents," *Naval. Res. Log.*, vol. 63, no. 5, pp. 416–429, Aug. 2016.

[34] Y. Yin, Y. Wang, T. C. E. Cheng, D. J. Wang, and C. C. Wu, "Two-agent single-machine scheduling to minimize the batch delivery cost," *Comput. Ind. Eng.*, vol. 92, pp. 16–30, Feb. 2016.

[35] D. Wang, Y. Yin, S. R. Cheng, T. C. E. Cheng, and C. C. Wu, "Due date assignment and scheduling on a single machine with two competing agents," *Int. J. Prod. Res.*, vol. 54, no. 4, pp. 1152–1169, Feb. 2016.

[36] Y. Yin, W. Wang, D. Wang, and T. C. E. Cheng, "Multi-agent single-machine scheduling and unrestricted due date assignment with a fixed machine unavailability interval," *Comput. Ind. Eng.*, vol. 111, pp. 202–215, Sep. 2017.

[37] X. Zhang and Y. Wang, "Two-agent scheduling problems on a single-machine to minimize the total weighted late work," *J. Combinat. Optim.*, vol. 33, no. 3, pp. 945–955, 2017.

[38] Y. Yin, Y. Yang, D. Wang, T. C. E. Cheng, and C.-C. Wu, "Integrated production, inventory, and batch delivery scheduling with due date assignment and two competing agents," *Nav. Res. Logistics*, vol. 65, pp. 393–409, Aug. 2018.

[39] P. Perez-Gonzalez and J. M. Framinan, "A common framework and taxonomy for multicriteria scheduling problems with interfering and competing jobs: Multi-agent scheduling problems," *Eur. J. Oper. Res.*, vol. 235, no. 1, pp. 1–16, May 2014.

[40] M. Pinedo, *Scheduling, Theory, Algorithms, and Systems*. Upper Saddle River, NJ, USA: Prentice Hall, 1995.

[41] K. R. Baker and T. Dan, *Principles of Sequencing and Scheduling*. Hoboken, NJ, USA: Wiley, 2013.

[42] T. Abdul-Razaq, C. N. Potts, and L. N. Van Wassenhove, "A survey of algorithms for the single machine total weighted tardiness scheduling problem," *Discrete Appl. Math.*, vol. 26, nos. 2–3, pp. 235–253, 1990.

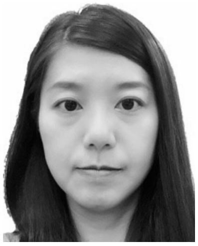
[43] A. B. Keha, K. Khowala, and J. W. Fowler, "Mixed integer programming formulations for single machine scheduling problems," *Comput. Ind. Eng.*, vol. 56, no. 1, pp. 357–367, 2009.

[44] H. M. Cho, S. J. Bae, J. Kim, and I. J. Jeong, "Bi-objective scheduling for reentrant hybrid flow shop using Pareto genetic algorithm," *Comput. Ind. Eng.*, vol. 61, no. 3, pp. 529–541, 2011.

[45] K. C. Ying, S. W. Lin, and S. Y. Wan, "Bi-objective reentrant hybrid flowshop scheduling: an iterated Pareto greedy algorithm," *Int. J. Prod. Res.*, vol. 52, no. 19, pp. 5735–5747, 2014.

[46] S. W. Lin and K. C. Ying, "A multi-point simulated annealing heuristic for solving multiple objective unrelated parallel machine scheduling problems," *Int. J. Prod. Res.*, vol. 53, no. 4, pp. 1065–1076, 2015.

CHEN-YANG CHENG received the Ph.D. degree in industrial and manufacturing engineering from Penn State University. He is currently a Professor with the Department of Industrial Engineering and Management, Taipei University of Technology. His research interests include computer integrated manufacturing, human computer interaction, distributed systems and control, and intelligent systems.



SHU-FEN LI received the Ph.D. degree in industrial engineering and enterprise information from Tunhai University, Taiwan, in 2017. She is currently an Assistant Professor with the Department of Industrial Engineering and Management, National Chin-Yi University of Technology. Her research interests include operations scheduling, computational intelligence, data mining, and lean production.

KUO-CHING YING is currently a Distinguished Professor with the Department of Industrial Engineering and Management, National Taipei University of Technology, Taiwan. He is an Editorial Board of ten prestigious journals and a Reviewer of 45 journals. His research interests include operations scheduling and combinatorial optimization, in which he has published over 100 academic research papers in refereed international journals, such as *Applied Intelligence*, *Applied Soft Computing*, *Computers and Operations Research*, *Computers and Industrial Engineering*, *European Journal of Industrial Engineering*, *European Journal of Operational Research*, the IEEE ACCESS, *International Journal of Production Economics*, *The International Journal of Advanced Manufacturing Technology*, *International Journal of Innovative Computing, Information and Control*, *International Journal of Production Research*, *Journal of the Operational Research Society*, *OMEGA-The International Journal of Management Sciences*, *Production Planning & Control*, *Transportation Research Part E: Logistics and Transportation Review*, among others.

YU-HSI LIU received the master's degree in industrial engineering and management from the National Taipei University of Technology, Taiwan, in 2015. He is currently an Assistant Manager with LITE-ON Technology Corporation.

• • •