Hindawi Publishing Corporation
Discrete Dynamics in Nature and Society
Volume 2016, Article ID 5413520, 13 pages
http://dx.doi.org/10.1155/2016/5413520

Research Article

An Improved Multiobjective PSO for the Scheduling Problem of Panel Block Construction

Zhi Yang,[1,2] Cungen Liu,[1,2] Xuefeng Wang,[1,2] and Weixin Qian[1,2]

[1] *State Key Laboratory of Ocean Engineering, Shanghai Jiao Tong University, Shanghai 200240, China*
[2] *Collaborative Innovation Center for Advanced Ship and Deep-Sea Exploration, Shanghai 200240, China*

Correspondence should be addressed to Cungen Liu; cgliu@sjtu.edu.cn

Received 1 February 2016; Revised 27 April 2016; Accepted 10 May 2016

Academic Editor: Seenith Sivasundaram

Uncertainty is common in ship construction. However, few studies have focused on scheduling problems under uncertainty in shipbuilding. This paper formulates the scheduling problem of panel block construction as a multiobjective fuzzy flow shop scheduling problem (FSSP) with a fuzzy processing time, a fuzzy due date, and the just-in-time (JIT) concept. An improved multiobjective particle swarm optimization called MOPSO-M is developed to solve the scheduling problem. MOPSO-M utilizes a ranked-order-value rule to convert the continuous position of particles into the discrete permutations of jobs, and an available mapping is employed to obtain the precedence-based permutation of the jobs. In addition, to improve the performance of MOPSO-M, archive maintenance is combined with global best position selection, and mutation and a velocity constriction mechanism are introduced into the algorithm. The feasibility and effectiveness of MOPSO-M are assessed in comparison with general MOPSO and nondominated sorting genetic algorithm-II (NSGA-II).

1. Introduction

Large bulk carriers, tankers, and container ships are characterized by large block coefficients and long parallel middle bodies. Consequently, there is a significant demand for panel blocks. To improve the efficiency of panel block construction, most large shipyards establish an assembly line for panel blocks. However, many shipyards in China face the problem that panel block assembly line scheduling often does not work for actual production. The scheduling problem of panel block construction is a type of flow shop scheduling problem (FSSP). Normally, FSSPs consist of determining the sequence for processing n jobs on m machines, where each job is processed on all of the machines in the same order. FSSPs are a type of nondeterministic polynomial-hard (NP-hard) combinational optimization problems. Heuristic or metaheuristic algorithms are considered to be suitable for solving FSSPs.

In most cases, FSSPs are considered in deterministic environments where the parameters, including the processing time and due date, are taken as crisp values. Nevertheless, the temporal parameters cannot be evaluated precisely in real-world production because of machine and human factors. This could be the major reason that flow shop scheduling often does not apply to actual production. Thus, it is more reasonable to model FSSPs with imprecise and vague parameters. Approaches to model this type of problem based on the concept of fuzzy sets have been widely studied in recent decades (e.g., Tsujimura et al. [1]; Itoh and Ishii [2]; Wu [3]; Huang et al. [4]). These problems are called fuzzy FSSPs because the imprecise and vague parameters are expressed as fuzzy parameters. In general, fuzzy FSSPs can be classified into three main classes: fuzzy FSSPs with a fuzzy due date, fuzzy FSSPs with a fuzzy processing time, and fuzzy FSSPs with both a fuzzy processing time and a fuzzy due date.

Multiple objectives should be taken into account in FSSPs. Multiple objectives increase the complexity of FSSPs but make them more similar to actual production. Many studies have examined multiobjective FSSPs. Sun et al. [5] and Yenisey and Yagmahan [6] provided two independent reviews and reported details about the development of multiobjective FSSPs and methods for solving such problems. However, few studies have been devoted to multiobjective fuzzy FSSPs. Kahraman et al. developed a new artificial immune system

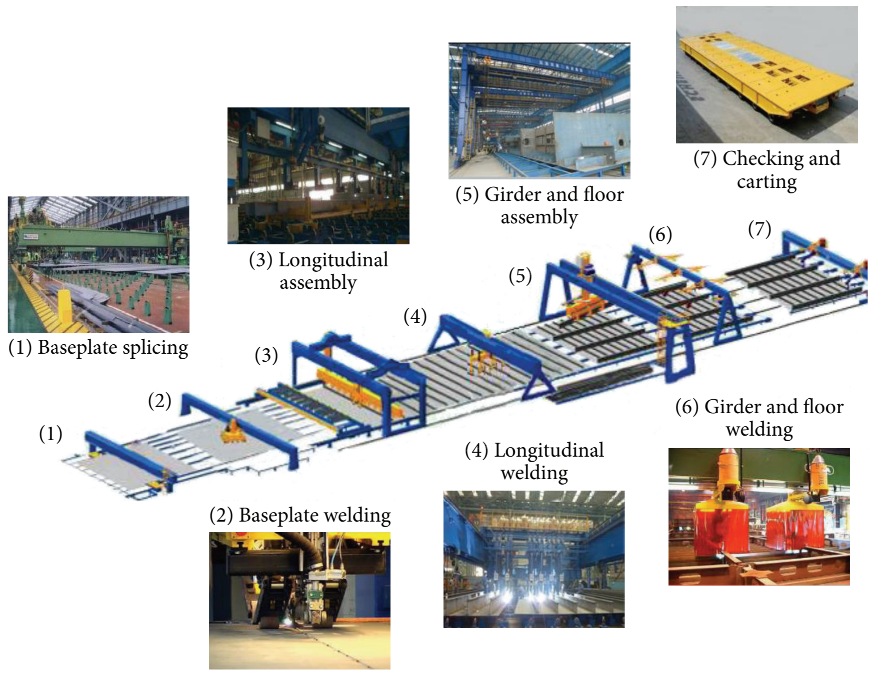

FIGURE 1: A typical assembly line for panel blocks.

(AIS) algorithm to solve a multiobjective fuzzy FSSP with both a fuzzy processing time and a fuzzy due date. The objectives were to minimize the average tardiness and the number of tardy jobs [7]. Engin et al. [8] proposed a scatter search (SS) method to solve a multiobjective fuzzy FSSP that is similar to Kahraman's method. Nakhaeinejad and Nahavandi integrated the technique for order preference by similarity to an ideal solution (TOPSIS) method with the interactive resolution method to solve a multiobjective fuzzy FSSP with a fuzzy processing time. The objectives were to minimize the completion time, the mean flow time, and the machine idle time [9]. Several studies on multiobjective fuzzy job shop scheduling problems (JSSPs) are applicable because FSSPs are a special case of JSSPs. Sakawa and Kubota [10] employed genetic algorithms to solve a multiobjective fuzzy JSSP with a fuzzy processing time and a fuzzy due date. Xing et al. [11] and González-Rodríguez et al. [12] also used genetic algorithms to solve multiobjective fuzzy JSSPs. Generally, the objectives of multiobjective fuzzy JSSPs include minimizing the maximum fuzzy completion time, minimizing the number of tardy jobs, maximizing the minimum agreement index of the fuzzy due date and fuzzy completion time, and maximizing the average agreement index. These objectives can also be considered in multiobjective fuzzy FSSPs.

Multiobjective fuzzy FSSPs can be considered to be similar to a host of actual flow shop production cases. However, other conditions should be applied to some multiobjective fuzzy FSSPs. For example, in an assembly line for panel blocks, the just-in-time (JIT) idea, which requires the necessary products to be produced in the necessary quantities at the necessary times, should be taken into account because panel blocks are intermediate products in hull construction systems. In this paper, we formulate the scheduling problem of panel block construction as a multiobjective fuzzy FSSP with a fuzzy processing time, a fuzzy due date, and the JIT idea. The JIT concept determines the existence of precedence relations among the panel blocks to be constructed as well as the expression of the fuzzy due date. To solve the multiobjective complex FSSP, we propose an improved algorithm called MOPSO-M that introduces mutation and a velocity constriction mechanism to particle swarm optimization (PSO) and implements a hybrid procedure to combine archive maintenance with global best position selection.

The remainder of this paper is organized as follows. Section 2 describes the scheduling problem of panel block construction. Section 3 introduces operations on fuzzy numbers that are needed to formulate scheduling problems. Section 4 introduces the proposed algorithm for solving the scheduling problem of panel block construction. Computational results are reported in Section 5 and are followed by the conclusions in Section 6.

2. Scheduling Problem of Panel Block Construction

2.1. Problem Description. Hull construction systems are multilevel production systems. Generally, a ship hull is assembled from dozens of hull blocks. A hull block is composed of several subblocks, most of which are panel blocks. Moreover, every panel block is constructed with steel plants and sections. As shown in Figure 1, a typical assembly line for panel blocks that can assemble and weld various types of steel plants and sections usually consists of seven main processes, including baseplate splicing, baseplate welding, longitudinal assembly, longitudinal welding, girders and floors assembly, girders and floors welding, and checking and carting. Each process is implemented at its corresponding station. Every panel block to be constructed must visit the stations one by one.

During production, the processing time of each process is often affected by uncertainty, imprecision, and vagueness due to both machine and human factors. In this situation, it is more appropriate to estimate both the processing time and the due date while considering the uncertainty. As intermediate products of hull blocks, panel blocks must prioritize the requirements of the hull block assembly because of geometric and processing constraints. The JIT concept includes precedence relations among the panel blocks that are to be constructed. Additionally, the JIT concept defines an uncertain due date as follows: "in principle, due dates are expected to be met, but certain earliness and tardiness limits can be tolerated, and longer ones will have lower values."

Thus, the scheduling of panel block construction focuses on finding proper sequences for processing the required panel blocks on the assembly line with a fuzzy processing time, a fuzzy due date, and precedence relations to attain specific objectives. This problem can be summarized as a multiobjective complex FSSP and is considered and analyzed below.

A set of panel blocks to be constructed with precedence relations will be processed sequentially at station 1, station 2, and so on until the final station. The stations are continuously available. At any time, each station can process a maximum of one panel block, and each panel block can be processed at a maximum of one station. Preemption is not allowed; that is, the processing of a panel block at a station cannot be interrupted. All of the panel blocks are available for processing at time zero. The set-up times at the stations are included in the processing time, while the transportation times between the stations are negligible. The fuzzy processing time and fuzzy due date are represented by fuzzy numbers. The notations of the scheduling problem of the panel block construction are as follows:

n: number of panel blocks to be constructed,

m: number of stations,

i: index of panel blocks, $i \in \{1, 2, \ldots, n\}$,

j: index of stations, $j \in \{1, 2, \ldots, m\}$,

$\tilde{p}_{i,j}$: fuzzy processing time of panel block i at station j,

$\tilde{d}_i$: fuzzy due date of panel block i,

$\tilde{C}_{i,j}$: fuzzy completion time of panel block i at station j,

$\tilde{C}_i$: final fuzzy completion time (makespan) of panel block i.

2.2. Problem Formulation. In this paper, the fuzzy processing time is taken as a triangular fuzzy number and is denoted as $\tilde{p}_{i,j} = (p_{i,j}^O, p_{i,j}, p_{i,j}^P)$, which includes three parameters: the optimistic value ($p_{i,j}^O$), the most plausible value ($p_{i,j}$), and the pessimistic value ($p_{i,j}^P$). The membership function of the triangular fuzzy processing time is formulated as in the following equation and as shown in Figure 2(a):

$$\mu_{\tilde{p}_{i,j}}(t) = \begin{cases} 0, & t \le p_{i,j}^O,\ t \ge p_{i,j}^P, \\ \dfrac{t - p_{i,j}^O}{p_{i,j} - p_{i,j}^O}, & p_{i,j}^O < t \le p_{i,j}, \\ \dfrac{p_{i,j}^P - t}{p_{i,j}^P - p_{i,j}}, & p_{i,j} < t < p_{i,j}^P. \end{cases} \quad (1)$$

The fuzzy due date is considered as a trapezoidal fuzzy number. For a trapezoidal fuzzy due date that is denoted as $\tilde{d}_i = (d_i^L, d_i^{E_1}, d_i^{E_2}, d_i^U)$, d_i^L and d_i^U are the lower and upper bounds of the fuzzy due date, respectively, and $d_i^{E_1}$ and $d_i^{E_2}$ represent the expected due date interval $(d_i^{E_1}, d_i^{E_2})$. The membership function of the trapezoidal fuzzy due date is given by (2) and is shown in Figure 2(b); it represents the degree of satisfaction with respect to the final completion time:

$$\mu_{\tilde{d}_i}(t) = \begin{cases} 0, & t \le d_i^L,\ t \ge d_i^U, \\ \dfrac{t - d_i^L}{d_i^{E_1} - d_i^L}, & d_i^L < r < d_i^{E_1}, \\ 1, & d_i^{E_1} \le r \le d_i^{E_2}, \\ \dfrac{d_i^U - t}{d_i^U - d_i^{E_2}}, & d_i^{E_2} < r < d_i^U. \end{cases} \quad (2)$$

Let $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$ denote a permutation of jobs (i.e., the panel blocks to be constructed). Suppose that the job i is allocated at the kth position of π. The fuzzy completion times of the panel blocks can be calculated using the following formulas:

$$\tilde{C}_{\pi_1,1} = \tilde{p}_{\pi_1,1}, \quad (3)$$

$$\tilde{C}_{\pi_1,j} = \tilde{C}_{\pi_1,j-1} + \tilde{p}_{\pi_1,j}, \quad j \in \{2, \ldots, m\}, \quad (4)$$

$$\tilde{C}_{\pi_k,1} = \tilde{C}_{\pi_{k-1},1} + \tilde{p}_{\pi_k,1}, \quad k \in \{2, \ldots, n\}, \quad (5)$$

$$\tilde{C}_{\pi_k,j} = \max\left(\tilde{C}_{\pi_k,j-1}, \tilde{C}_{\pi_{k-1},j}\right) + \tilde{p}_{\pi_k,j}, \quad k \in \{2, \ldots, n\},\ j \in \{2, \ldots, m\}, \quad (6)$$

$$\tilde{C}_i = \tilde{C}_{\pi_k} = \tilde{C}_{\pi_k,m}. \quad (7)$$

The fuzzy completion time of each panel block has the same structure as the fuzzy processing time. The final fuzzy completion time, which is denoted as $\tilde{C}_i = (C_i^O, C_i, C_i^P)$, also includes three parameters: the optimistic value (C_i^O), the most plausible value (C_i), and the pessimistic value (C_i^P).

The completion time is always expected to meet the due date. The agreement index (AI) of the fuzzy completion time with respect to the fuzzy due date is often used to represent the portion of $\tilde{C}_i$ that meets $\tilde{d}_i$. The AI, which is defined in (8) and is shown in Figure 2(c), indicates the degree of compliance between $\tilde{C}_i$ and $\tilde{d}_i$:

$$\mathrm{AI}_i = \frac{\operatorname{area}\left(\tilde{C}_i \cap \tilde{d}_i\right)}{\operatorname{area} \tilde{C}_i}. \quad (8)$$

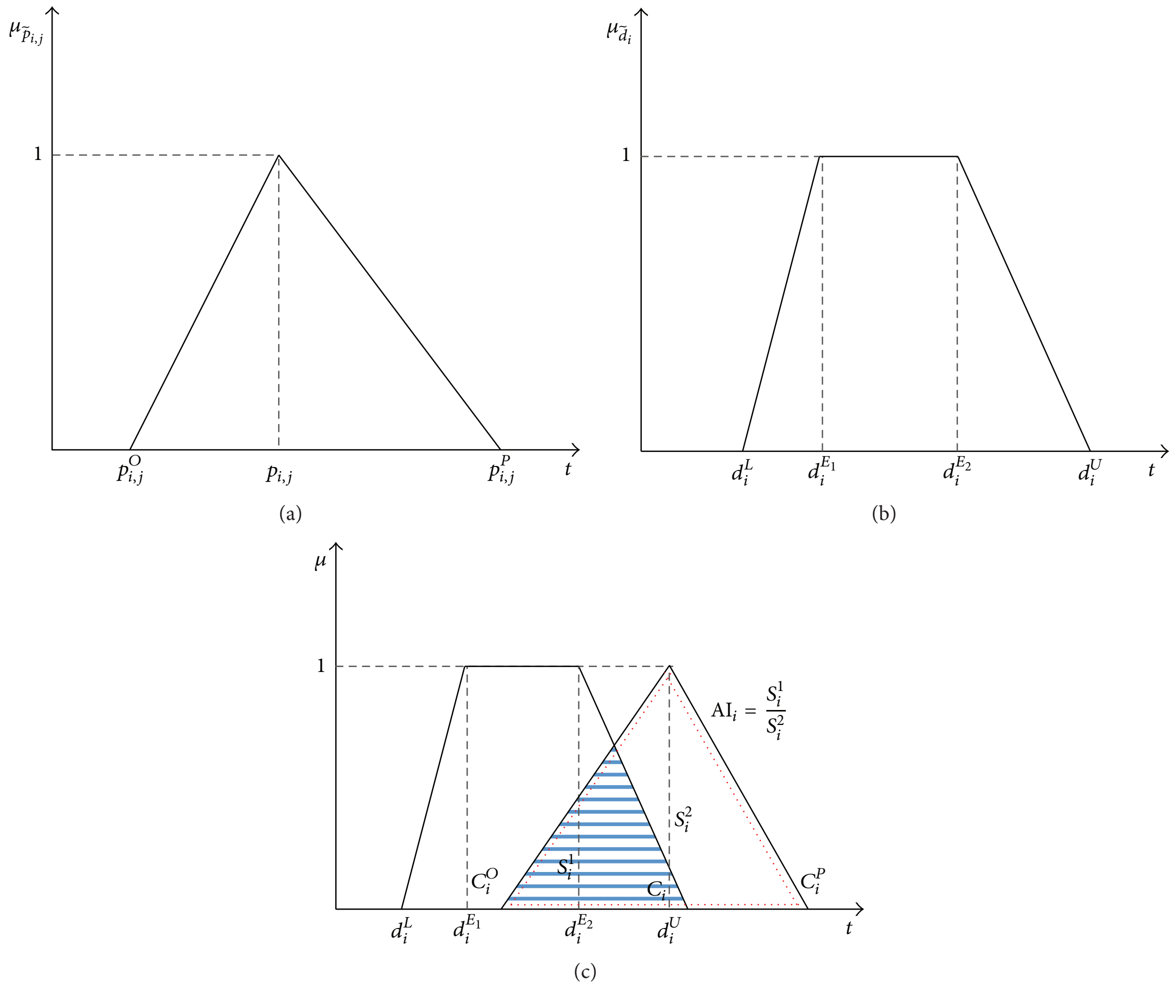

Figure 2: (a) Membership function of the triangular fuzzy processing time; (b) membership function of the trapezoidal fuzzy due date; (c) agreement index (AI).

The precedence relations between the panel blocks to be constructed can be depicted in matrix form as in (9). This equation can express the precedence relations between two arbitrary panel blocks that are to be constructed:

$$\mathrm{PR}_{i,k} = \begin{cases} 1, & \text{if the completion of block } i \text{ must precede that of block } k \ (i \gg k), \\ -1, & \text{if the completion of block } i \text{ must } \forall i,k \in \{1,2,\ldots,n\} \text{ lag behind that of block } k \ (i \ll k), \ i \neq k, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

To more accurately reflect real-world situations, we formulate the scheduling problem as a three-objective problem that not only minimizes the fuzzy makespan but also maximizes the average agreement index and the minimum agreement index:

$$\text{minimize } f_1 = \text{makespan} = \max_{i=1,2,\ldots,n} \widetilde{C}_i, \tag{10}$$

$$\text{maximize } f_2 = \overline{\mathrm{AI}} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{AI}_i, \tag{11}$$

$$\text{maximize } f_3 = \mathrm{AI}_{\min} = \min_{i=1,2,\ldots,n} \mathrm{AI}_i. \tag{12}$$

In summary, this paper formulates the scheduling problem of panel block construction as a multiobjective fuzzy flow shop scheduling problem with precedence relations.

3. Operations on Fuzzy Numbers

Equations (4)–(6) and (10) show that some operations on fuzzy numbers are essential to the formulation of the scheduling problem. These operations involve the addition operation, the max operation, and the ranking method for two or more fuzzy numbers.

According to the extension principle of Zadeh, the membership function of the addition operation (+) for two fuzzy numbers is given by

$$\mu_{\widetilde{A}+\widetilde{B}}(z) = \sup_{z=x+y} \min\{\mu_{\widetilde{A}}(x), \mu_{\widetilde{B}}(y)\}. \quad (13)$$

For two triangular fuzzy numbers $\widetilde{A} = (a^1, a^2, a^3)$ and $\widetilde{B} = (b^1, b^2, b^3)$,

$$\widetilde{A} + \widetilde{B} = (a^1 + b^1, a^2 + b^2, a^3 + b^3). \quad (14)$$

The membership function of the max operation ($\vee$) for two fuzzy numbers is defined as

$$\mu_{\widetilde{A}\vee\widetilde{B}}(z) = \sup_{z=x\vee y} \min\{\mu_{\widetilde{A}}(x), \mu_{\widetilde{B}}(y)\}. \quad (15)$$

However, based on the extension principle, the fuzzy number that is obtained as the result of the max operation ($\vee$) for two triangular fuzzy numbers is not always a triangular structure. Two approximations for the max operation, which were proposed by Sakawa and Mori [13] and Lei [14], are widely used in fuzzy processing time studies. Sakawa's criterion states that the approximate max is a triple composed of $\widetilde{A}$ and $\widetilde{B}$; according to Lei's criterion, the approximate max is either $\widetilde{A}$ or $\widetilde{B}$. The two criteria are given below.

Sakawa's criterion is

$$\max(\widetilde{A}, \widetilde{B}) = \widetilde{A} \vee \widetilde{B} \simeq (a^1 \vee b^1, a^2 \vee b^2, a^3 \vee b^3). \quad (16)$$

Lei's criterion is as follows:

$$\text{if } \widetilde{A} > \widetilde{B}, \text{ then } \max(\widetilde{A}, \widetilde{B}) = \widetilde{A} \vee \widetilde{B} \simeq \widetilde{A}; \text{ else } \widetilde{A} \vee \widetilde{B} \simeq \widetilde{B}. \quad (17)$$

As shown in (10) and (17), obtaining the fuzzy makespan and approximating the fuzzy max using Lei's criterion both require a ranking method for fuzzy numbers. This paper uses the following three criteria to rank triangular fuzzy numbers [10].

Criterion 1. The greatest associate ordinary number

$$C_1(\widetilde{A}) = \frac{a^1 + 2a^2 + a^3}{4} \quad (18)$$

is used as the first criterion to rank the triangular fuzzy numbers.

Criterion 2. If C_1 does not rank the fuzzy numbers, then the best maximal presumption

$$C_2(\widetilde{A}) = a^2 \quad (19)$$

is chosen as the second criterion.

Criterion 3. If C_1 and C_2 do not rank the fuzzy numbers, then the difference of the spreads

$$C_3(\widetilde{A}) = a^3 - a^1 \quad (20)$$

is utilized as the third criterion. The three criteria allow almost all triangular fuzzy numbers to be ranked. For example, if $C_1(\widetilde{A}) > C_1(\widetilde{B})$, then $\widetilde{A} > \widetilde{B}$; if $C_1(\widetilde{A}) = C_1(\widetilde{B})$ and $C_2(\widetilde{A}) > C_2(\widetilde{B})$, then $\widetilde{A} > \widetilde{B}$; if $C_1(\widetilde{A}) = C_1(\widetilde{B})$, $C_2(\widetilde{A}) = C_2(\widetilde{B})$, and $C_3(\widetilde{A}) = C_3(\widetilde{B})$, then $\widetilde{A} > \widetilde{B}$.

Lei's criterion employs the three criteria that are described above to obtain the approximate max, while Sakawa's criterion forms the approximate max by comparing three pairs of special points. Lei's criterion provides a better approximation to the real max than Sakawa's criterion [14]. Accordingly, we employ Lei's criterion to approximate the fuzzy max in this paper.

4. MOPSO-M for the Scheduling Problem

4.1. PSO Algorithm. Particle swarm optimization is a population-based stochastic optimization technique that was proposed by Kennedy and Eberhart [15]. In PSO, each potential solution is treated as a particle that possesses two attributes: position and velocity. Each particle flies in the search space (i.e., the solution space) at a certain velocity, which is dynamically adjusted according to the flying experiences of it and its companions. In a D-dimensional search space, the velocity of every particle is updated in accordance with the following equation:

$$V_q(k+1) = \omega V_q(k) + r_1 c_1 [X_q^{pbest}(k) - X_q(k)] + r_2 c_2 [X^{gbest}(k) - X_q(k)], \quad (21)$$

where $V_q(k) = \{v_{q,1}(k), v_{q,2}(k), \ldots, v_{q,D}(k)\}$ and $X_q(k) = \{x_{q,1}(k), x_{q,2}(k), \ldots, x_{q,D}(k)\}$ represent the velocity and position of the qth particle at the kth iteration, respectively, $X_q^{pbest}(k) = \{x_{q,1}^{pbest}(k), x_{q,2}^{pbest}(k), \ldots, x_{q,D}^{pbest}(k)\}$ denotes the best previous position of the qth particle, $X^{gbest}(k) = \{x_1^{gbest}(k), x_2^{gbest}(k), \ldots, x_D^{gbest}(k)\}$ represents the global best position that has been detected in the swarm, ω is the inertia weight that controls the impact of the current velocity on the new velocity, c_1 and c_2 are learning factors that represent the relative influences of the self-cognition and social-interaction, respectively, and r_1 and r_2 are uniform random numbers in the interval $(0, 1)$. In this paper, we consider a modified version of PSO that was proposed by Clerc and Kennedy [16], which incorporates a parameter χ that is known as the constriction factor. The velocity of every particle is updated through the following equation:

$$V_q(k+1) = \chi \{V_q(k) + r_1 c_1 [X_q^{pbest}(k) - X_q(k)] + r_2 c_2 [X^{gbest}(k) - X_q(k)]\}, \quad (22)$$

where $\chi = 2/|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|$, $\varphi = c_1 + c_2$, $\varphi > 4$. The main role of the constriction factor is to alleviate the swarm explosion effect and ensure convergence of the PSO. The position of every particle is updated as

$$X_q(k+1) = X_q(k) + V_q(k+1). \quad (23)$$

It is clear that the velocity and position of every particle are updated continuously and that the PSO is suited to a continuous solution space.

Due to its advantages, including its simplicity, easy implementation, and low computational cost, PSO has been used in a wide variety of optimization problems. Several researchers have extended PSO for scheduling problems that are set in discrete spaces. Liao et al. developed a PSO algorithm for FSSPs with three incommensurable objectives. They also attempted to incorporate a local search scheme into the proposed algorithm [17]. Rahimi-Vahed and Mirghorbani [18] utilized a PSO algorithm in a bicriteria FSSP. Sha and Hung Lin [19] provided a PSO-based multiobjective algorithm for FSSPs. Applications of PSO have also been reported in the area of fuzzy scheduling problems. Lei [20] proposed a Pareto archive PSO algorithm for multiobjective JSSPs with both fuzzy processing times and fuzzy due dates. Niu et al. [21] redefined and modified PSO by introducing genetic operators, and Li and Pan [22] hybridized PSO with a tabu search (TS) to solve FSSPs with fuzzy processing times. The major issue in successfully applying PSO to scheduling problems is to develop an effective problem mapping and solution generation mechanism [23].

4.2. The Proposed Algorithm. This paper proposes a multiobjective algorithm (we call it MOPSO-M) to solve the scheduling problem of panel block construction that possesses fuzzy processing times, fuzzy due dates, precedence relations, and multiple objectives. As a multiobjective algorithm, MOPSO-M is developed based on the concept of Pareto optimality as described below.

For a multiobjective optimization problem with T decision variables and L objectives,

$$\text{minimize } F(X) = [f_1(X), f_2(X), \ldots, f_L(X)], \quad (24)$$

where $X \in \Theta \in \mathbb{R}^T$, Θ is the search space, and $F(X) \in \mathbb{R}^L$. A solution $X_0 \in \Theta$ is said to dominate another solution $X_1 \in \Theta$, which is represented as $X_0 \succ X_1$, if and only if $f_l(X_0) \leq f_l(X_1)\ \forall l \in \{1, 2, \ldots, L\}$, $f_l(X_0) < f_l(X_1)\ \exists l \in \{1, 2, \ldots, L\}$. X_0 is said to be nondominated regarding a given set if X_0 is not dominated by any solution in the set. X_0 is said to be a Pareto optimal solution if and only if $\neg\exists X_1 \in \Theta : X_1 \succ X_0$.

The proposed algorithm is described in detail below.

4.2.1. Solution Representation. Finding a suitable mapping between the position of the particles and the job sequence is crucial to the application of PSO to FSSPs. In MOPSO-M, a ranked-order-value (ROV) rule [24] that is based on a random key representation is utilized to convert the continuous positions of particles to the discrete permutations of jobs. In particular, for a position $X_q = [x_{q,1}, x_{q,2}, \ldots, x_{q,n}]$, the position values from smallest to largest are mapped to rank values from 1 to n, which generates a permutation of jobs $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$. If there are two or more identical position values, the one with the smaller dimension has priority to be mapped to the rank value. Table 1 presents a simple example that illustrates the ROV rule. Note that the ROV rule is more available for a flow shop scheduling problem with a relatively large number of jobs (e.g., $n = 10$, 20, or larger), for in such cases the problem that different positions of the particles map to the same permutations of jobs is pretty rare and has little or no adverse effect on the effectiveness of the optimization algorithms.

Table 1: Example of the mapping between the position of the particle and the permutation of jobs.

Dimension number (d)	1	2	3	4
Position value ($x_{q,d}$)	0.138	1.542	2.306	1.542
Rank value	1	2	4	3

Note: $n = 4$, $X_q = [0.138, 1.542, 2.036, 1.542]$, and $\pi = [1, 2, 4, 3]$.

In MOPSO-M, the precedence relations among the required panel blocks are not handled based on the particle position but on the permutation of jobs. To obtain the precedence-based permutation π^{PR}, an available mapping is constructed using a five-step process:

(1) Initialize $\pi^{PR} = [\,]$, $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$.

(2) Let $d = 1$.

(3) Identity the dth dimension value of π, which is represented by $\pi(d)$; if the job that precedes $\pi(d)$ has not yet been added to π^{PR}, then go to (4); else, go to (5).

(4) Let $d = d + 1$; go to (3).

(5) Add $\pi(d)$ to π^{PR} and take it as the last dimension value; delete $\pi(d)$ from π; if $\pi = [\,]$, then output π^{PR}; else, go to (2).

Table 2 shows an example that uses the mapping to obtain the precedence-based permutation.

4.2.2. Outline of MOPSO-M. In order to improve the search capability of MOPSO-M, we apply the mechanism that was introduced in SMPSO [25] for further bounding the accumulated velocity of each variable d (in each particle) using the following velocity constriction equation:

$$v_{q,d}(k+1) = \begin{cases} \text{delta}_d, & \text{if } v_{q,d}(k+1) > \text{delta}_d, \\ -\text{delta}_d, & \text{if } v_{q,d}(k+1) > \text{delta}_d\ \forall d \in \{1, 2, \ldots, D\}, \\ v_{q,d}(k+1), & \text{otherwise,} \end{cases} \quad (25)$$

where $\text{delta}_d = (\text{upper\_limit}_d - \text{lower\_limit}_d)/2$. In summary, the velocity of every particle is calculated by (22), and the resulting velocity is then constrained by (25). The position of every particle is updated through (23).

In MOPSO-M, an external archive (Ar) is used to store the nondominated solutions that are produced during

Table 2: Example of the mapping for obtaining the precedence-based permutation.

Initialization	Precedence	Process of mapping
$\pi^{\mathrm{PR}} = []$,	$\mathrm{PR}_{4,2} = 1$,	(1) $d = 1, \pi(1) = 1, \pi^{\mathrm{PR}} = [1], \pi = [2, 4, 3]$
$\pi = [1, 2, 4, 3]$	$\mathrm{PR}_{2,4} = -1$,	(2) $d = 1, \pi(1) = 2$, go to (4), $\pi^{\mathrm{PR}} = [1]$
	$(4 \gg 2)$	(3) $d = 2, \pi(2) = 4, \pi^{\mathrm{PR}} = [1, 4], \pi = [2, 3]$
		(4) $d = 1, \pi(1) = 2, \pi^{\mathrm{PR}} = [1, 4, 2], \pi = [3]$
		(5) $d = 1, \pi(1) = 3, \pi^{\mathrm{PR}} = [1, 4, 2, 3], \pi = []$,
		Output $\pi^{\mathrm{PR}} = [1, 4, 2, 3]$

the search process. Ar is updated at every iteration, and X^{gbest} of the particles are selected from the nondominated solutions in Ar. The algorithm may get stuck in local optima if the members of Ar lack diversity. This motivates the introduction of mutation to potentially produce new nondominated solutions and to provide new members for Ar. Mutation operator has been usually used in MOSPO. SMPSO applies a polynomial mutation to the 15% of the particles [25]. OMOPSO utilizes a combination of uniform and nonuniform mutation to the particle swarm [26]. PAPSO performs mutation on archive members [20]. In MOPSO-M, the mutation operators, including SWAP, INSERT, and INVERSE, are applied to the copy of the solutions at every iteration to generate neighboring solutions and to improve the performance of the neighborhood search. The above three operators are described below.

SWAP. Randomly select two different elements from a sequence and then swap them.

INSERT. Randomly choose two different elements from a sequence and then insert the back one before the front one.

INVERSE. Invert the subsequence between two different random positions of a sequence.

MOPSO-M is outlined as follows.

(1) Iteration = 0: initialize a population of Ps particles; obtain π_0^{PR} with respect to each solution; evaluate the objective vector of each solution and store the nondominated individuals of S_0 (i.e., the set of solutions) in Ar_0; determine $X^{pbest}(0)$ and $X^{gbest}(0)$ for each particle.

(2) Iteration = $k+1$: update $V(k+1)$ and $X(k+1)$ of each particle using (22), (25), and (23); obtain π_{k+1}^{PR} with respect to each solution; evaluate the objective vector of each solution, find the nondominated individuals of S_{k+1}, and store them in set Nd_{k+1}.

(3) Copy the members of S_{k+1} to S_{k+1}^C; perform mutation on the members of S_{k+1}^C and produce neighboring solutions; rename S_{k+1}^C to S_{k+1}^M; obtain $\pi_{k+1}^{\mathrm{PR}(M)}$ with respect to each solution; evaluate the objective vector of each solution, find the nondominated individuals of S_{k+1}^M, and store them in set Nd_{k+1}^M.

(4) Maintain Ar_{k+1} and select $X^{gbest}(k+1)$ for every particle; $X_q^{pbest}(k+1)$ is updated with $X_q(k)$ if $X_q(k) \succ X_q^{pbest}(k)$.

(5) If the terminal condition is met, then output the optimal solutions and the optimal objective vectors; else, let iteration $\leftarrow$ iteration + 1 and go to (2).

The procedure of archive maintenance and X^{gbest} selection is detailed in Section 4.2.3. The mutation operators, including SWAP, INSERT, and INVERSE, are randomly implemented on the members of S_{k+1}^C.

4.2.3. Archive Maintenance and X^{gbest} Selection. The number of nondominated solutions in Ar is limited by the predetermined maximum archive size s_M. When the actual size of Ar, which is denoted as s_A, reaches s_M, Ar must decide which solution should be replaced by a new nondominated solution. The crowding distance, which is defined as a density-estimation metric [27], is usually used to select which solution to replace and to promote the diversity of the stored solutions in multiobjective PSO (e.g., Nebro et al. [25]; Raquel and Naval Jr. [28]). Generally, when Ar is full, the solution that has the smallest crowding distance is preferably replaced.

Archive maintenance and X^{gbest} selection are two important procedures in the PSO-based multiobjective algorithm. MOPSO-M combines these two procedures by referring to Lei's method [20]. The hybrid procedure of archive maintenance and X^{gbest} selection is presented below.

(1) Assign all members of Ar_k to Ar_{k+1}; let $X_q^{gbest}(k) \leftarrow X_q^{gbest}(k+1)$.

(2) For each solution $X_N(k+1) \in Nd_{k+1}$ or Nd_{k+1}^M, if it is dominated by any member of Ar_{k+1}, then exclude it from the archive; else, first insert it into Ar_{k+1} and take it as a new member; go to (3) or (4).

(3) For each new member $X_N(k+1)$, if it dominates some members of Ar_{k+1}, then remove the dominated members from the archive and substitute $X_N(k+1)$ for the $X^{gbest}(k+1)$ of all of the particles in the set $\{q \mid X_q^{gbest}(k+1) = X_D(k+1), X_N(k+1) \succ X_D(k+1) \in \mathrm{Ar}_{k+1}\}$.

(4) For each new member $X_N(k+1)$, if it does not dominate any member of Ar_{k+1}, then one has the following:

(4.1) If $s_A = s_M$, remove member $X_C(k+1)$ with the smallest crowding distance; if $X_N(k+1) \neq X_C(k+1)$, then replace $X^{gbest}(k+1)$ of all of the particles in the set $\{q \mid X_q^{gbest}(k+1) = X_C(k+1)\}$ with $X_N(k+1)$; else, remove $X_N(k+1)$ from Ar_{k+1}.

(4.2) If $s_A < s_M$, one has the following:

(4.2.1) Compute $u = \min_{e=1,2,\ldots,s_A}\{N(X_e(k+1))\}$, $\forall X_e(k+1) \in \mathrm{Ar}_{k+1}$, where $N(X_e(k+1))$ represents the number of particles whose value of $X^{gbest}(k+1)$ is $X_e(k+1)$; $N(X_N(k+1)) = 0$; if $u > g$ (g is an integer, and $g \in [0.025\mathrm{Ps}, 0.05\mathrm{Ps}]$), then let $u \leftarrow g$.

(4.2.2) Let $H = \{X_e(k+1) \mid N(X_e(k+1)) > u\}$, $C = |H|$, $f = 0$.

(4.2.3) Select the solution $X_e(k+1) \in H$ that is nearest to $X_N(k+1)$; substitute $X_N(k+1)$ for the new $X^{gbest}(k+1)$ of one particle whose current $X^{gbest}(k+1)$ is $X_e(k+1)$; let $H \leftarrow H \setminus \{X_e(k+1)\}$; let $N(X_N(k+1)) \leftarrow N(X_N(k+1)) + 1$; let $f \leftarrow f + 1$;
if $N(X_N(k+1)) < u$ and $f < C$, repeat (4.2.3); if $N(X_N(k+1)) < u$ and $f = C$, go to (4.2.2); if $N(X_N(k+1)) = u$, go to the end.

This hybrid procedure ensures that each archive member serves as X^{gbest} of at least one particle. Thus, all of the members, especially the new individuals, can participate in the search process and guide particles towards new regions of the search space. The implementation of the hybrid procedure, the introduction of mutation, and the application of the velocity constriction mechanism of SMPSO are expected to make the optimal solutions that are generated by MOPSO-M better approximate the Pareto optimal solutions.

5. Computational Results

Because the scheduling problem of panel block construction is usually complex and requires higher quality optimal solutions, a serviceable algorithm with stronger optimization capability is needed. In this study, real-time production data are used to test the performance of the proposed algorithm. The real-time data of the fuzzy processing time and fuzzy due date of two sets of panel blocks to be constructed (10 × 7 and 20 × 7 fuzzy FSSPs) come from a large shipyard in Shanghai, China, and are shown in Tables 8 and 10. The most plausible value ($p_{i,j}$) of the fuzzy processing time is determined as the mean value of the historical processing times of the same or very similar panel blocks. The optimistic value ($p_{i,j}^{O}$) and the pessimistic value ($p_{i,j}^{P}$) are often randomly obtained from $[\delta_{11}P_{i,j}, \delta_{12}P_{i,j}]$ and $[\delta_{21}P_{i,j}, \delta_{22}P_{i,j}]$, respectively [29]. In this paper, δ_{11}, δ_{12}, δ_{21}, and δ_{22} are set to 0.85, 0.90, 1.10, and 1.25, respectively, based on historical data and the advice of experienced workers. The fuzzy due date of each panel block is provided by the hull block assembly shop, which is the demand side. Additionally, precedence relations among the panel blocks are provided by the hull block assembly shop and are presented in Tables 9 and 11.

Table 3: Main parameter settings of the three algorithms.

MOPSO-M	General MOPSO	NSGA-II
Ps = 60	Ps = 60	Ps = 60
$s_M = 15$	$s_M = 15$	$s_M = 15$
Fe = 30000	Fe = 30000	Fe = 30000
$c_1 = 2.05, c_2 = 2.25$	$c_1 = 2.05$	$p_c = 0.80$
$p_m^C = 1$	$c_2 = 2.25$	$p_m = 0.05$

Notes: Ps denotes the population scale; Fe represents function evaluations; p_c and p_m denote the crossover probability and the mutation probability, respectively.

MOPSO-M is compared with the general MOPSO (without proposed modifications) and nondominated sorting genetic algorithm-II (NSGA-II). Each algorithm uses the ROV representation rule. The parameter settings of the three algorithms are shown in Table 3. All of these algorithms are implemented in MATLAB 8.1.

For evaluating the performance of the algorithms, we consider three quality indicators: unary additive epsilon indicator ($I_{\varepsilon+}^{1}$) [30], hypervolume indicator (HV), and coverage indicator (C) [31].

The $I_{\varepsilon+}^{1}$ indicator, which is defined in (26), equals the minimum factor ε such that any objective vector in an obtained front (OF) is ε-dominated by at least one objective vector in Pareto optimal front (PF*):

$$I_{\varepsilon+}^{1}(\mathrm{OF}) = \inf_{\varepsilon\in\mathbb{R}} \left\{\forall Z^2 \in \mathrm{PF}^* \ \exists Z^1 \in \mathrm{OF} : Z^1 \succ_{\varepsilon+} Z^2\right\}, \quad (26)$$

where $Z^1 \succ_{\varepsilon+} Z^2$ if and only if $\forall 1 \leq l \leq L : z_l^1 < \varepsilon + z_l^2 Z^1 = (z_1^1, \ldots, z_L^1)$ and $Z^2 = (z_1^2, \ldots, z_L^2)$ are two objective vectors of a minimization problem. Because the PF* for each aforementioned fuzzy FSSP is not known, a reference front constituted by gathering all obtained fronts of all algorithms is used in this paper.

As far as the HV indicator is concerned, it measures the volume, in the objective space, covered by the obtained front. Mathematically, for objective vector $Z^t \in \mathrm{OF}$, a hypercube v_t is constructed with a reference point R and the objective vector Z^t as the diagonal corners of the hypercube [32]. The vector of worst objective function values is usually used as the reference point. The HV indicator is calculated as the volume of the union of all hypercubes:

$$\mathrm{HV} = \mathrm{volume}\left(\bigcup_{t=1}^{|\mathrm{OF}|} v_t\right). \quad (27)$$

The C indicator is a binary indicator. Let E_A and E_B represent two sets of approximate Pareto optimal solutions that are generated by algorithm A and algorithm B, respectively. $C(E_A, E_B)$, which is defined in (28), measures the fraction of

Table 4: Median and IQR of the $I^1_{\varepsilon+}$ and HV indicators.

	MOPSO-M		General MOPSO		NSGA-II	
	Median	IQR	Median	IQR	Median	IQR
$I^1_{\varepsilon+}$						
10 × 7	$1.34e-02$	$1.04e-02$	$1.20e-01$	$9.66e-02$	$1.20e-01$	$3.26e-01$
20 × 7	$1.87e-01$	$1.11e-01$	$1.00e-00$	$1.13e-01$	$9.11e-01$	$2.67e-01$
HV						
10 × 7	$9.78e-01$	$6.54e-03$	$9.26e-01$	$4.57e-02$	$9.15e-01$	$2.93e-01$
20 × 7	$7.76e-01$	$1.65e-01$	$7.84e-02$	$1.13e-01$	$1.12e-01$	$1.58e-01$

Table 5: Results of the Wilcoxon rank-sum test for the $I^1_{\varepsilon+}$ and HV indicators.

		General MOPSO		NSGA-II	
		10 × 7	20 × 7	10 × 7	20 × 7
$I^1_{\varepsilon+}$	MOPSO-M	87.5 ($2.45e-08$)	102.5 ($5.87e-08$)	41.5 ($5.77e-10$)	90 ($3.93e-08$)
HV	MOPSO-M	800 ($1.19e-07$)	807 ($4.78e-09$)	871 ($1.95e-13$)	840 ($5.43e-11$)

Notes: for the $I^1_{\varepsilon+}$ indicator, we apply wilcox. test ($I^1_{\varepsilon+}$ (MOPSO-M), $I^1_{\varepsilon+}$ (general MOPSO or NSGA-II), and alternative = "less") in R. For the HV indicator, we apply wilcox. test (HV (MOPSO-M), HV (general MOPSO or NSGA-II), and alternative = "greater"). Values in ( ) are the p values for the test statistics.

the members of E_B that are dominated by members of E_A, reflecting the dominance relation between the two sets:

$$C(E_A, E_B) = \frac{\left|\left\{X^B \in E_B \mid \exists X^A \in E_A,\ X^A \succ X^B\right\}\right|}{|E_B|}. \quad (28)$$

The C indicator maps the ordered pair (E_A, E_B) to the interval $[0, 1]$. $C(E_A, E_B) = 1$ indicates that all of the solutions in E_B are dominated by individuals in E_A, while $C(E_A, E_B) = 0$ implies that none of the solutions in E_B are dominated by members of E_A.

In the performance evaluation experiment, each algorithm is independently run 30 times for each fuzzy FSSP. The median and interquartile range (IQR) of the $I^1_{\varepsilon+}$ and HV indicators are reported in Table 4. In the calculation of the two indicators, the objective values are normalized into values in the interval $[1, 2]$. For the $I^1_{\varepsilon+}$ indicator, the lower the value the better the obtained front, while, for the HV indicator, the lower the value the better the obtained front. Thus, depending on the data contained in Table 4, we see that MOPSO-M achieves the best values for both the $I^1_{\varepsilon+}$ and HV indicators in both the fuzzy FSSPs. Table 5 summarizes the results of statistical pairwise comparisons by applying Wilcoxon test to the $I^1_{\varepsilon+}$ and HV values. These results suggest that MOPSO-M obtains better fronts than general MOPSO and NSGA-II with statistical confidence, regarding the two indicators in both the fuzzy FSSPs. More information can be obtained from the box plots in Figure 3, for the box plots visualize the distributions of the $I^1_{\varepsilon+}$ and HV values. Attending to this figure, we can also observe the fact that MOPSO-M outperforms the other two algorithms concerning the two indicators.

We turn now to the C indicator. Let $A1$, $A2$, and $A3$ denote MOPSO-M, MOPSO, and NSGA-II, respectively, and E_{Ai} $(i = 1, 2, 3)$ denote the set constituted by gathering all nondominated solutions that are produced by Ai $(i = 1, 2, 3)$ in 30 independent runs. Table 6 lists the results of the C indicator, and Figure 4 shows the distribution of the corresponding objective vectors of the solutions in E_{Ai}. Here the makespan which is assumed to be a triangular fuzzy number is defuzzified using (18). For the smaller-scale 10 × 7 fuzzy FSSP with simpler precedence relations, MOPSO-M exhibits better average performance than the general MOPSO and NSGA-II. All of the optimal solutions of MOPSO and NSGA-II are covered by those of MOPSO-M, and the number of optimal solutions of MOPSO-M is 20% greater than those of the other two algorithms. For the larger-scale 20 × 7 fuzzy FSSP with more complex precedence relations, MOPSO-M is significantly superior to the other two algorithms. Its superiority is manifested in two ways:

Table 6: Results of the C indicator.

	10 × 7	20 × 7		10 × 7	20 × 7
$C(E_{A1}, E_{A2})$	0.000	0.571	$\|E_{A1}\|$	6	25
$C(E_{A2}, E_{A1})$	0.000	0.000	$\|E_{A2}\|$	5	21
$C(E_{A1}, E_{A3})$	0.000	1.000	$\|E_{A3}\|$	5	16
$C(E_{A3}, E_{A1})$	0.000	0.000			

Notes: $|E_{Ai}|$ denotes the number of optimal solutions generated by algorithm Ai in 30 independent runs.

(1) MOPSO-M generates more optimal solutions.

(2) All of the optimal solutions of NSGA-II and the most optimal solutions of general MOPSO are dominated by those of MOPSO-M, while none of the optimal solutions of MOPSO-M are dominated by those of the other two algorithms.

In summary, the results of the $I^1_{\varepsilon+}$, HV, and C indicators demonstrate that MOPSO-M outperforms the general MOPSO and NSGA-II in solving the scheduling problem of panel block construction in terms of the quality of the optimal solutions. The outperformance can clearly be attributed to the implementation of the hybrid procedure, the introduction

Table 7: Three examples of determining the processing sequence for the 20 × 7 fuzzy FSSP using MOPSO-M.

Index	α_j			Optimal objective vectors			Optimal solutions (processing sequence)
	α_1	α_2	α_3	f_1	f_2	f_3	
1	0.55	0.30	0.15	4931.50	0.898	0.090	3-2-7-4-1-9-12-10-5-17-15-19-16-6-8-11-14-20-13-18
2	0.40	0.30	0.30	4954.00	0.974	0.723	3-2-7-1-4-9-10-5-12-17-15-19-6-16-13-8-11-14-20-18
3	0.30	0.35	0.35	4969.25	0.990	0.889	3-1-2-7-4-9-10-5-12-17-15-19-6-13-16-8-11-14-20-18

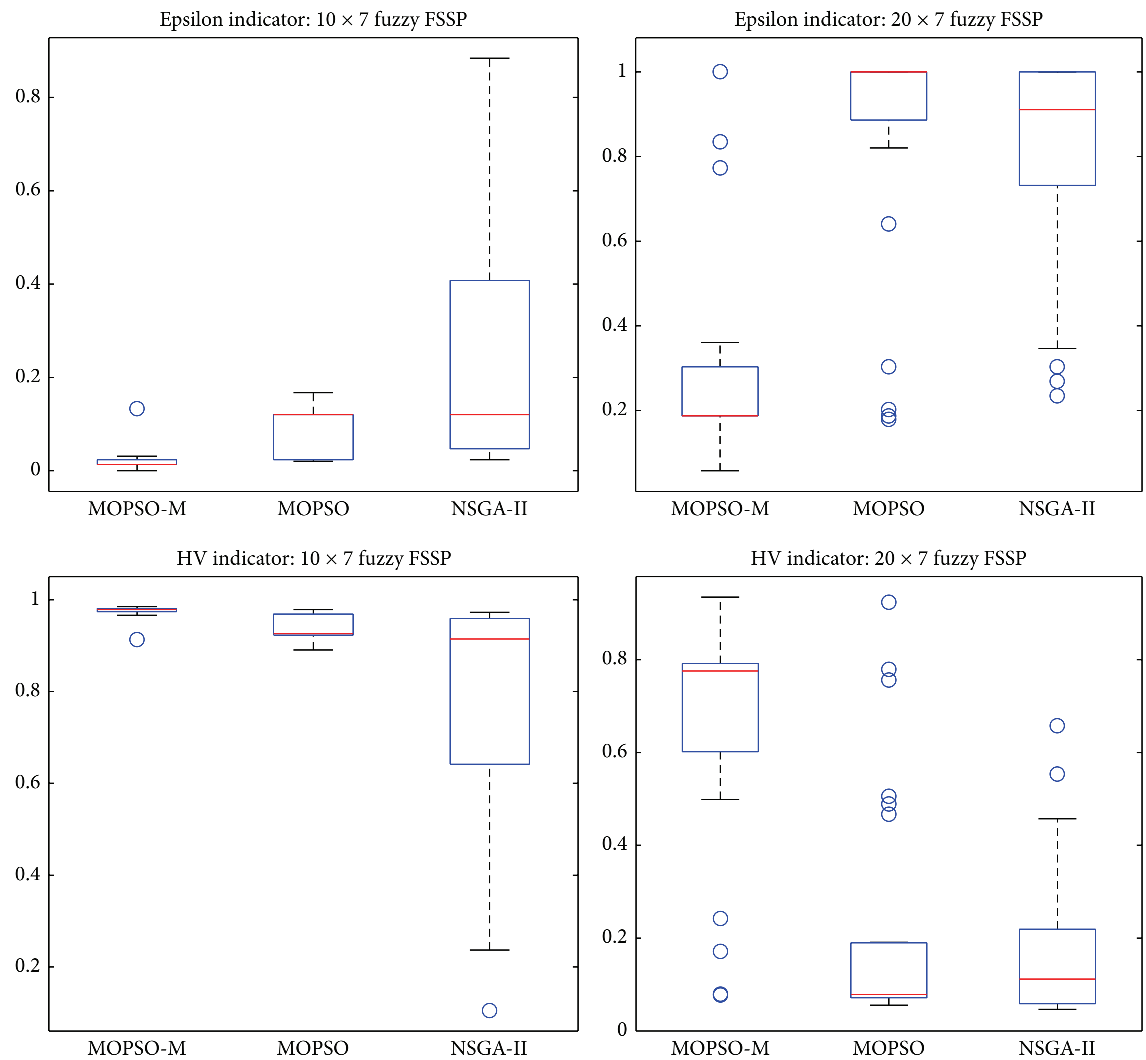

Figure 3: Box plots of the $I^1_{\varepsilon+}$ and HV indicators obtained by MOPSO-M, MOPSO, and NSGA-II in the two fuzzy FSSPs.

of mutation, and the application of the velocity constriction mechanism of SMPSO.

The optimal objective vectors and the corresponding optimal solutions that are generated by MOPSO are more reliable for use in determining a sequence for processing required panel blocks on the assembly line to attain specific objectives. For example, for the 20 × 7 fuzzy FSSP, whose optimal objective vectors that are generated by MOPSO are shown in Figure 4(b), the weighted sum of the nondimensional objective values is utilized to determine the processing sequence. The objective values are nondimensionalized using the following equation:

$$w_{s,t} = \begin{cases} \dfrac{f_t^{\max} - f_{s,t}}{f_t^{\max} - f_t^{\min}}, & t = 1, \\[2ex] \dfrac{f_{s,t} - f_t^{\min}}{f_t^{\max} - f_t^{\min}}, & t = 2, 3, \end{cases} \quad (29)$$

Table 8: Fuzzy processing time and fuzzy due date of the 10 × 7 fuzzy FSSP.

i	$\tilde{P}_{i,1}$	$\tilde{P}_{i,2}$	$\tilde{P}_{i,3}$	$\tilde{P}_{i,4}$	$\tilde{P}_{i,5}$	$\tilde{P}_{i,6}$	$\tilde{P}_{i,7}$	$\tilde{d}_i$
1	115 136 153	189 220 273	120 133 159	126 144 170	139 157 186	212 244 285	105 124 150	900 1300 1900 2300
2	88 101 112	132 152 173	87 97 109	84 98 109	103 116 144	139 158 185	68 79 97	700 800 900 1000
3	121 134 160	205 228 255	122 139 167	140 159 182	143 159 196	204 240 264	108 123 135	800 1000 1600 2000
4	86 100 114	146 162 196	82 94 107	95 107 128	95 112 123	139 160 187	66 75 86	1000 1400 2200 2500
5	116 137 152	190 224 251	119 133 151	134 157 179	143 165 188	197 226 276	105 123 140	1800 2200 2900 3200
6	83 92 112	150 176 195	75 89 97	97 108 121	95 108 134	141 160 195	77 87 104	1000 1800 2300 2500
7	121 134 163	212 244 268	122 141 168	126 148 169	128 149 180	216 248 308	100 118 136	1100 1400 1600 1800
8	92 107 122	158 176 215	86 99 114	87 103 126	106 120 138	139 160 186	68 76 88	2000 2200 2700 3000
9	124 139 174	194 220 246	122 141 167	128 147 163	127 150 186	199 226 253	105 118 140	2000 2400 2700 3500
10	118 137 168	208 236 281	117 136 166	139 156 190	142 159 199	203 236 276	99 116 140	2000 2300 2800 3500

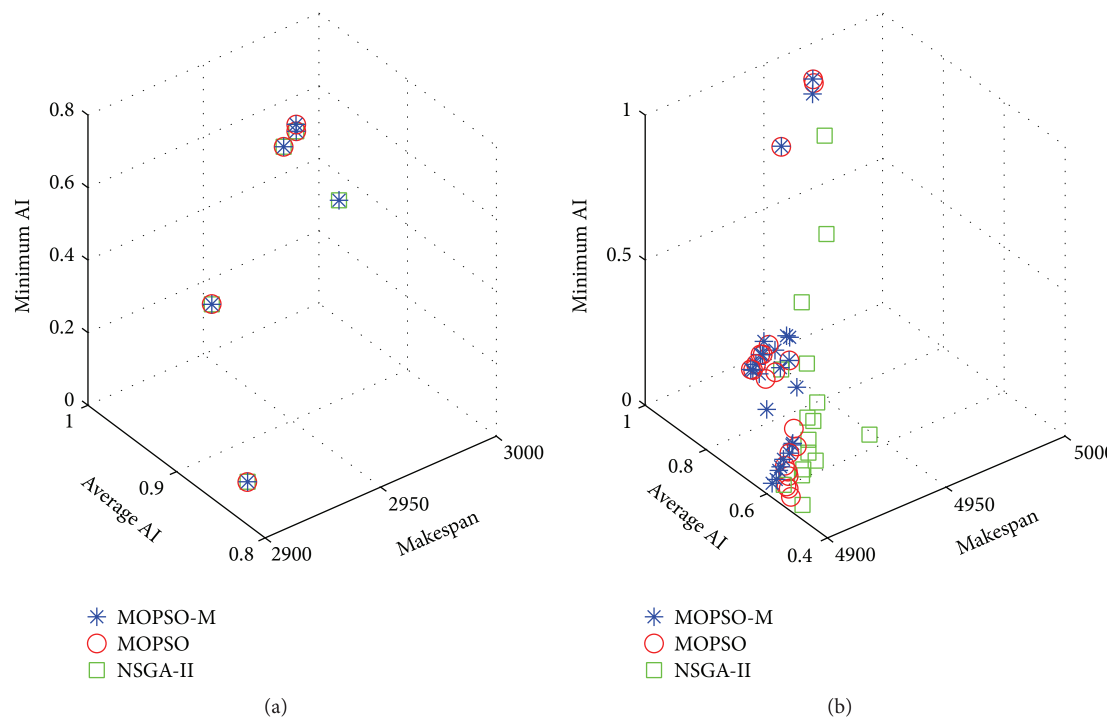

Figure 4: Distribution of optimal objective vectors generated by MOPSO-M, MOPSO, and NSGA-II in the performance evaluation experiment. (a) 10 × 7 fuzzy FSSP; (b) 20 × 7 fuzzy FSSP.

Table 9: Precedence relations among the 10 panel blocks to be constructed.

i	k	$PR_{i,k}$	$PR_{k,i}$	Graphical representation
1	7	1	−1	$1 \gg 7$
5	9	1	−1	$5 \gg 9$

where $w_{s,t}$ is the tth nondimensional objective value of the sth optimal solution, $f_{s,t}$ is the tth objective value of the sth optimal solution, $f_t^{\max} = \max\{f_{1,t}, f_{2,t}, \ldots, f_{25,t}\}$, and $f_t^{\min} = \min\{f_{1,t}, f_{2,t}, \ldots, f_{25,t}\}$. The weighted sum of the nondimensional objective values can be calculated by

$$W_s = \sum_{t=1}^{3} \alpha_t w_{st}, \tag{30}$$

where $\sum_{t=1}^{3} \alpha_t = 1$, $0 \le \alpha_t \le 1$. The optimal solution with the maximum value of W_s can be used as the processing sequence. Three examples of determining the processing sequence are shown in Table 7.

6. Conclusions

In this study, we introduce a typical assembly line for panel blocks in a shipyard. To accurately represent actual production, we formulate the scheduling problem of panel block construction as a multiobjective fuzzy FSSP with a fuzzy processing time, a fuzzy due date, and precedence relations between the panel blocks. An effective multiobjective particle swarm optimization called MOPSO-M is proposed and applied to the scheduling problem. Computational results

Table 10: Fuzzy processing time and fuzzy due date of the 20 × 7 fuzzy FSSP.

i	$\widetilde{P}_{i,1}$	$\widetilde{P}_{i,2}$	$\widetilde{P}_{i,3}$	$\widetilde{P}_{i,4}$	$\widetilde{P}_{i,5}$	$\widetilde{P}_{i,6}$	$\widetilde{P}_{i,7}$	$\widetilde{d}_i$
1	121 139 174	206 240 271	118 131 161	124 146 182	143 166 199	204 240 300	111 125 145	1000 1300 2000 2500
2	88 98 112	134 156 190	77 86 105	89 104 118	103 116 145	144 166 196	74 87 106	800 1200 1700 2000
3	81 90 105	162 180 218	93 104 122	91 105 125	92 105 116	131 150 177	70 82 101	700 800 1000 1100
4	119 132 160	205 238 264	123 137 155	132 152 186	132 155 177	211 248 310	113 125 139	1500 1800 2200 2300
5	81 91 104	146 168 197	75 89 105	99 117 146	89 101 118	144 160 195	73 81 97	2000 2300 3000 3500
6	124 143 157	206 232 285	123 137 159	136 159 179	130 153 180	200 222 255	99 113 127	1600 2300 3800 4700
7	90 100 116	143 162 185	87 98 121	94 105 118	101 119 135	155 180 216	71 83 100	1000 1200 2300 2500
8	128 144 164	194 228 262	120 135 165	135 155 184	152 169 208	211 234 257	101 116 142	3500 4000 4600 4800
9	121 139 153	213 242 300	117 130 144	138 157 190	142 159 193	191 220 271	103 120 142	1500 1800 2500 2800
10	87 97 122	150 172 191	87 97 110	103 114 143	106 120 135	139 164 200	72 80 94	2000 2200 2700 3000
11	83 96 114	141 166 189	88 104 117	102 117 144	100 111 123	139 154 183	69 77 90	3700 4000 4600 5000
12	128 150 185	200 232 255	111 128 151	137 156 192	139 163 194	220 250 295	101 113 138	2000 2800 3800 4000
13	88 102 120	153 176 206	73 86 106	89 100 118	100 115 141	134 154 174	68 80 98	2000 2800 4000 4500
14	82 95 112	155 172 198	84 95 117	96 113 139	99 112 133	136 158 198	73 83 102	3900 4200 4800 5200
15	131 145 168	214 238 283	117 138 173	125 146 175	135 154 171	200 230 265	104 116 138	2200 2800 3400 3700
16	126 145 182	197 224 260	120 139 168	138 163 190	146 167 186	225 250 290	96 113 137	2500 3000 4400 5000
17	116 133 156	221 248 273	114 127 147	131 146 181	144 162 178	197 232 274	110 124 140	2500 2800 3300 3500
18	78 90 104	153 178 201	88 102 117	94 107 133	90 104 128	150 172 189	73 82 93	3500 4000 5000 5500
19	84 95 113	139 156 176	78 87 104	94 108 119	95 111 130	153 178 210	72 80 89	2600 3000 3800 4000
20	118 137 153	210 236 283	127 141 157	124 144 159	149 169 196	216 248 298	99 117 145	3900 4300 5300 5600

Table 11: Precedence relations among the 20 panel blocks to be constructed.

i	k	$\mathrm{PR}_{i,k}$	$\mathrm{PR}_{k,i}$	Graphical representation
1	9	1	−1	$1 \gg 9$
3	5	1	−1	$3 \gg 5$
5	13	1	−1	$10 \gg 5$
8	11	1	−1	$5 \gg 13$
10	5	1	−1	$13 \gg 18$
10	17	1	−1	$10 \gg 17$
11	14	1	−1	$8 \gg 11$
12	16	1	−1	$11 \gg 14$
13	18	1	−1	$14 \gg 20$
14	20	1	−1	$12 \gg 16$
15	19	1	−1	$15 \gg 19$

Notes: some $\mathrm{PR}_{i,k}$ values that rely on transitive relations (e.g., $\mathrm{PR}_{10,18}$) are not reported in this table.

that are based on real-time shipbuilding production data indicate that MOPSO-M outperforms the general MOPSO and NSGA-II in terms of the quality of the optimal solutions. The combination of archive maintenance with X^{gbest} selection, the introduction of mutation, and the application of the velocity constriction mechanism of SMPSO greatly increase the optimization capability of MOPSO-M. Further research is required to analyze more complex problems, such as noncompletely hybrid assembly lines in shipyards, which have more complicated constraints.

Appendix

See Tables 8, 9, 10, and 11.

Competing Interests

The authors declare that there are no competing interests regarding the publication of this paper.

Acknowledgments

This work is supported in part by the National Program on Key Basic Research Project of China (973 Program, Grant no. 2013CB036103) and the Program on Implementation Countermeasures and Proposals for Intelligent Manufacturing in Shipbuilding Industry.

References

[1] Y. Tsujimura, S. H. Park, I. S. Chang, and M. Gen, "An effective method for solving flow shop scheduling problems with fuzzy processing times," *Computers & Industrial Engineering*, vol. 25, no. 1–4, pp. 239–242, 1993.

[2] T. Itoh and H. Ishii, "Fuzzy due-date scheduling problem with fuzzy processing time," *International Transactions in Operational Research*, vol. 6, no. 6, pp. 639–647, 1999.

[3] H.-C. Wu, "Solving the fuzzy earliness and tardiness in scheduling problems by using genetic algorithms," *Expert Systems with Applications*, vol. 37, no. 7, pp. 4860–4866, 2010.

[4] C.-S. Huang, Y.-C. Huang, and P.-J. Lai, "Modified genetic algorithms for solving fuzzy flow shop scheduling problems and their implementation with CUDA," *Expert Systems with Applications*, vol. 39, no. 5, pp. 4999–5005, 2012.

[5] Y. Sun, C. Zhang, L. Gao, and X. Wang, "Multi-objective optimization algorithms for flow shop scheduling problem: a review and prospects," *The International Journal of Advanced Manufacturing Technology*, vol. 55, no. 5–8, pp. 723–739, 2011.

[6] M. M. Yenisey and B. Yagmahan, "Multi-objective permutation flow shop scheduling problem: literature review, classification and current trends," *Omega*, vol. 45, pp. 119–135, 2014.

[7] C. Kahraman, O. Engin, and M. K. Yilmaz, "A new artificial immune system algorithm for multiobjective fuzzy flow shop problems," *International Journal of Computational Intelligence Systems*, vol. 2, no. 3, pp. 236–247, 2009.

[8] O. Engin, C. Kahraman, and M. K. Yilmaz, "A scatter search method for multiobjective fuzzy permutation flow shop scheduling problem: a real world application," in *Computational Intelligence in Flow Shop and Job Shop Scheduling*, pp. 169–189, Springer, Berlin, Germany, 2009.

[9] M. Nakhaeinejad and N. Nahavandi, "An interactive algorithm for multi-objective flow shop scheduling with fuzzy processing time through resolution method and TOPSIS," *The International Journal of Advanced Manufacturing Technology*, vol. 66, no. 5–8, pp. 1047–1064, 2013.

[10] M. Sakawa and R. Kubota, "Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms," *European Journal of Operational Research*, vol. 120, no. 2, pp. 393–407, 2000.

[11] Y. J. Xing, Z. Q. Wang, J. Sun, and J. J. Meng, "A multi-objective fuzzy genetic algorithm for job-shop scheduling problems," in *Proceedings of the International Conference on Computational Intelligence and Security (ICCIAS '06)*, vol. 1, pp. 398–401, IEEE, Guangzhou, China, October 2006.

[12] I. González-Rodríguez, J. Puente, and C. R. Vela, "A multiobjective approach to fuzzy job shop problem using genetic algorithms," in *Current Topics in Artificial Intelligence: 12th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2007, Salamanca, Spain, November 12–16, 2007. Selected Papers*, vol. 4788 of *Lecture Notes in Computer Science*, pp. 80–89, Springer, Berlin, Germany, 2007.

[13] M. Sakawa and T. Mori, "Efficient genetic algorithm for job-shop scheduling problems with fuzzy processing time and fuzzy duedate," *Computers & Industrial Engineering*, vol. 36, no. 2, pp. 325–341, 1999.

[14] D. Lei, "Fuzzy job shop scheduling problem with availability constraints," *Computers and Industrial Engineering*, vol. 58, no. 4, pp. 610–617, 2010.

[15] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, Perth, Australia, December 1995.

[16] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[17] C.-J. Liao, C.-T. Tseng, and P. Luarn, "A discrete version of particle swarm optimization for flowshop scheduling problems," *Computers and Operations Research*, vol. 34, no. 10, pp. 3099–3111, 2007.

[18] A. R. Rahimi-Vahed and S. M. Mirghorbani, "A multi-objective particle swarm for a flow shop scheduling problem," *Journal of Combinatorial Optimization*, vol. 13, no. 1, pp. 79–102, 2007.

[19] D. Y. Sha and H. Hung Lin, "A particle swarm optimization for multi-objective flowshop scheduling," *The International Journal of Advanced Manufacturing Technology*, vol. 45, no. 7-8, pp. 749–758, 2009.

[20] D. Lei, "Pareto archive particle swarm optimization for multi-objective fuzzy job shop scheduling problems," *The International Journal of Advanced Manufacturing Technology*, vol. 37, no. 1-2, pp. 157–165, 2008.

[21] Q. Niu, B. Jiao, and X. Gu, "Particle swarm optimization combined with genetic operators for job shop scheduling problem with fuzzy processing time," *Applied Mathematics and Computation*, vol. 205, no. 1, pp. 148–158, 2008.

[22] J.-Q. Li and Y.-X. Pan, "A hybrid discrete particle swarm optimization algorithm for solving fuzzy job shop scheduling problem," *International Journal of Advanced Manufacturing Technology*, vol. 66, no. 1–4, pp. 583–596, 2013.

[23] W. Xia and Z. Wu, "An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems," *Computers and Industrial Engineering*, vol. 48, no. 2, pp. 409–425, 2005.

[24] B. Liu, L. Wang, and Y.-H. Jin, "An effective PSO-based memetic algorithm for flow shop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, vol. 37, no. 1, pp. 18–27, 2007.

[25] A. J. Nebro, J. J. Durillo, G. Nieto, C. A. C. Coello, F. Luna, and E. Alba, "SMPSO: a new pso-based metaheuristic for multi-objective optimization," in *Proceedings of the IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM '09)*, pp. 66–73, Nashville, Tenn, USA, April 2009.

[26] J. J. Durillo, J. García-Nieto, A. J. Nebro, C. A. Coello, F. Luna, and E. Alba, "Multi-objective particle swarm optimizers: an experimental comparison," in *Evolutionary Multi-Criterion Optimization*, vol. 5467 of *Lecture Notes in Computer Science*, pp. 495–509, Springer, Berlin, Germany, 2009.

[27] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[28] C. R. Raquel and P. C. Naval Jr., "An effective use of crowding distance in multiobjective particle swarm optimization," in *Proceedings of the 7th Annual conference on Genetic and Evolutionary Computation*, pp. 257–264, ACM, June 2005.

[29] S. Abdullah and M. Abdolrazzagh-Nezhad, "Fuzzy job-shop scheduling problems: a review," *Information Sciences*, vol. 278, pp. 380–407, 2014.

[30] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[31] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[32] J. J. Durillo and A. J. Nebro, "JMetal: a Java framework for multi-objective optimization," *Advances in Engineering Software*, vol. 42, no. 10, pp. 760–771, 2011.