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

Copyright © 2016 Zhi Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

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) 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).
(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. Nakhaiinejad 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 \( \tilde{p}_{i,j} = (\tilde{p}_{i,j}^O, \tilde{p}_{i,j}^P, \tilde{p}_{i,j}^P) \), which includes three parameters: the optimistic value \( (\tilde{p}_{i,j}^O) \), the most plausible value \( (\tilde{p}_{i,j}^M) \), and the pessimistic value \( (\tilde{p}_{i,j}^P) \). The membership function of the triangular fuzzy processing time is formulated as in the following equation and as shown in Figure 2(a):

\[
\mu_{\tilde{p}_{i,j}}(t) = \begin{cases} 
0, & t \leq \tilde{p}_{i,j}^O, \ t \geq \tilde{p}_{i,j}^P \\
\frac{t - \tilde{p}_{i,j}^O}{\tilde{p}_{i,j}^P - \tilde{p}_{i,j}^O}, & \tilde{p}_{i,j}^O < t < \tilde{p}_{i,j}^P \\
\frac{\tilde{p}_{i,j}^P - t}{\tilde{p}_{i,j}^P - \tilde{p}_{i,j}^O}, & \tilde{p}_{i,j}^O < t \leq \tilde{p}_{i,j}^P
\end{cases}
\]  

(1)

The fuzzy due date is considered as a trapezoidal fuzzy number. For a trapezoidal fuzzy due date that is denoted as \( \tilde{d}_i = (d_i^L, d_i^E, d_i^M, d_i^U) \), \( d_i^L \) and \( d_i^U \) are the lower and upper bounds of the fuzzy due date, respectively, and \( d_i^E \) and \( d_i^M \) represent the expected due date interval \( (d_i^E, d_i^M) \). 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:

\[
\mu_{\tilde{d}_i}(t) = \begin{cases} 
0, & t \leq d_i^L, \ t \geq d_i^U \\
\frac{t - d_i^L}{d_i^M - d_i^L}, & d_i^L < t < d_i^E \\
1, & d_i^E < t \leq d_i^M \\
\frac{d_i^U - t}{d_i^U - d_i^E}, & d_i^E < t \leq d_i^U
\end{cases}
\]  

(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 \( k \)th position of \( \pi \). The fuzzy completion times of the panel blocks can be calculated using the following formulas:

\[
\tilde{C}_{\pi_1} = \tilde{p}_{\pi_1,1},
\]

(3)

\[
\tilde{C}_{\pi_j} = \tilde{C}_{\pi_{j-1}} + \tilde{p}_{\pi_j,j}, \ j \in \{2, \ldots, m\},
\]

(4)

\[
\tilde{C}_{\pi_k} = \tilde{C}_{\pi_{k-1}} + \tilde{p}_{\pi_k,k}, \ k \in \{2, \ldots, n\},
\]

(5)

\[
\tilde{C}_{\pi_{k,j}} = \max(\tilde{C}_{\pi_{k-1,j}}, \tilde{C}_{\pi_{k,j-1}}) + \tilde{p}_{\pi_{k,j}},
\]

(6)

\[
\tilde{C}_i = \tilde{C}_{\pi_m}.
\]

(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 \( \tilde{C}_i = (\tilde{C}_i^O, \tilde{C}_i^P, \tilde{C}_i^P) \), also includes three parameters: the optimistic value \( (\tilde{C}_i^O) \), the most plausible value \( (\tilde{C}_i^M) \), and the pessimistic value \( (\tilde{C}_i^P) \).

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 \( \tilde{d}_i \):

\[
\text{AI}_i = \frac{\text{area}(\tilde{C}_i \cap \tilde{d}_i)}{\text{area} \tilde{C}_i}.
\]  

(8)
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:

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

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:

\[
\text{minimize } f_1 = \text{makespan} = \max_{i=1,2,\ldots,n} C_i, 
\]

\[
\text{maximize } f_2 = \overline{AI} = \frac{1}{n} \sum_{i=1}^{n} AI_i, 
\]

\[
\text{maximize } f_3 = \text{AI}_{\text{min}} = \min_{i=1,2,\ldots,n} AI_i. 
\]

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

$$\mu_{\bar{A} + \bar{B}}(z) = \sup_{x+y = z} \min \{\mu_{\bar{A}}(x), \mu_{\bar{B}}(y)\}.$$  \hspace{1cm} (13)

For two triangular fuzzy numbers $\bar{A} = (a_1, a_2, a_3)$ and $\bar{B} = (b_1, b_2, b_3)$,

$$\bar{A} + \bar{B} = (a_1 + b_1, a_2 + b_2, a_3 + b_3).$$  \hspace{1cm} (14)

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

$$\mu_{\bar{A} \vee \bar{B}}(z) = \sup_{x \vee y = z} \min \{\mu_{\bar{A}}(x), \mu_{\bar{B}}(y)\}.$$  \hspace{1cm} (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 Mor [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 $\bar{A}$ and $\bar{B}$; according to Lei's criterion, the approximate max is either $\bar{A}$ or $\bar{B}$. The two criteria are given below.

Sakawa’s criterion is

$$\text{max}(\bar{A}, \bar{B}) = \bar{A} \vee \bar{B} = (a_\max \vee b_\max, a_\max \vee b_\max, a_\max \vee b_\max).$$  \hspace{1cm} (16)

Lei’s criterion is as follows:

if $\bar{A} > \bar{B}$, then $\text{max}(\bar{A}, \bar{B}) = \bar{A} \vee \bar{B} = \bar{A}$; else $\bar{A} \vee \bar{B} = \bar{B}$.  \hspace{1cm} (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

$$C_1(\bar{A}) = \frac{a_1^3 + 2a_2^2 + a_3^3}{4}$$  \hspace{1cm} (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

$$C_2(\bar{A}) = a_2$$  \hspace{1cm} (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

$$C_3(\bar{A}) = a_3 - a_1$$  \hspace{1cm} (20)

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

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, 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:

$$V_q(k+1) = \omega V_q(k) + r_1 c_1 [X_q^{\text{best}}(k) - X_q(k)]$$

$$+ r_2 c_2 [X_q^{\text{gbest}}(k) - X_q(k)],$$  \hspace{1cm} (21)

where $X_q(k) = \{x^j_q(k), x_{q,2}(k), \ldots, x_{q,D}(k)\}$ and $X_q(k) = \{x^j_q(k), x_{q,2}(k), \ldots, x_{q,D}(k)\}$ represent the velocity and position of the $q$th particle at the $k$th iteration, respectively, $X_q^{\text{best}}(k) = \{x^j_q^{\text{best}}(k), x_{q,2}^{\text{best}}(k), \ldots, x_{q,D}^{\text{best}}(k)\}$ denotes the best previous position of the $q$th particle, $X_q^{\text{gbest}}(k) = \{x^j_q^{\text{gbest}}(k), x_{q,2}^{\text{gbest}}(k), \ldots, x_{q,D}^{\text{gbest}}(k)\}$ represents the global best position that has been detected in the swarm, $\omega$ 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 $\chi$ that is known as the constriction factor. The velocity of every particle is updated through the following equation:

$$V_q(k+1) = \chi [V_q(k) + r_1 c_1 [X_q^{\text{best}}(k) - X_q(k)]$$

$$+ r_2 c_2 [X_q^{\text{gbest}}(k) - X_q(k)]],$$  \hspace{1cm} (22)

where $\chi = 2/|2 - \phi - \sqrt{\phi^2 - 4\phi}|, \phi = c_1 + c_2, \phi > 4$. 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
\[ X_q(k + 1) = X_q(k) + V_q(k + 1). \]  
(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. Liu 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,
\[
\text{minimize } F(X) = \left[ f_1(X), f_2(X), \ldots, f_L(X) \right],
\]  
(24)
where \( X \in \Theta \in \mathbb{R}^T, \Theta \) 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 \( X_0 > X_1 \), if and only if \( f_j(X_0) \leq f_j(X_1) \forall j \in \{1, 2, \ldots, L\}, f_j(X_0) < f_j(X_1) \exists j \in \{1, 2, \ldots, L\} \). \( X_0 \) 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 \( \exists X_1 \in \Theta : X_1 > X_0 \).

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.

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 \( \pi^{PR} \), 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. Identify the \( d \)th dimension value of \( \pi \), which is represented by \( \pi(d) \); if the job that precedes \( \pi(d) \) has not yet been added to \( \pi^{PR} \), then go to (4); else, go to (5).
4. Let \( d = d + 1 \); go to (3).
5. Add \( \pi(d) \) to \( \pi^{PR} \) and take it as the last dimension value; delete \( \pi(d) \) from \( \pi \); if \( \pi = [ ] \), then output \( \pi^{PR} \), 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:
\[
v_{q,d}(k + 1) = \begin{cases} 
\text{delta}_d, & \text{if } v_{q,d}(k + 1) > \text{delta}_d, \\
-\text{delta}_d, & \text{if } v_{q,d}(k + 1) < \text{delta}_d, \\
V_d(k + 1), & \text{otherwise}, 
\end{cases}
\]  
(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
the search process. Ar is updated at every iteration, and $X_{gbest}^k$ 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 [28]. 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 $n_{q}^{PR}$ with respect to each solution; evaluate the objective vector of each solution and store the nondominated individuals of $S_{q}$ (i.e., the set of solutions) in $Ar_{q}$; determine $X_{gbest}^0$ 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 $n_{q}^{PR}$ 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 $S_{k+1}^{C}$ to $S_{k+1}^{M}$ and produce neighboring solutions; rename $S_{k+1}^{C}$ to $S_{k+1}^{M}$; obtain $n_{q}^{PR(M)}$ 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 $X_{gbest}^k(k + 1)$ for every particle; $X_{gbest}^q(k + 1)$ is updated with $X_q(k)$ if $X_q(k) > X_{gbest}^q(k)$.

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 $S_{k+1}$.

### 4.2.3. Archive Maintenance and $X_{gbest}$ Selection

The number of nondominated solutions in Ar is limited by the pre-defined 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 $X_{gbest}^q(k) \leftarrow X_{gbest}^q(k + 1)$.

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 $X_{gbest}^q(k + 1)$ of all of the particles in the set $\{ q \mid X_{gbest}^q(k + 1) = X_q(k + 1), X_N(k + 1) > X_q(k + 1) \in Ar_{k+1} \}$. 

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

<table>
<thead>
<tr>
<th>Initialization</th>
<th>Precedence</th>
<th>Process of mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\pi^{PR} = []$</td>
<td>$\pi_{k+1}^{PR} = 1$, $\pi_{k+1}^{PR} = 1$, $\pi = [2, 4, 3]$</td>
<td>(1) $d = 1, \pi(1) = 1, \pi^{PR} = [1], \pi = [2, 4, 3]$</td>
</tr>
<tr>
<td>$\pi = [1, 2, 4, 3]$</td>
<td>$\pi_{2,4}^{PR} = -1$, $\pi_{0}^{PR} = 1$</td>
<td>(2) $d = 1, \pi(1) = 2$, go to (4), $\pi^{PR} = [1]$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3) $d = 2, \pi(2) = 4$, $\pi^{PR} = [1, 4, 3], \pi = [2, 3]$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(4) $d = 1, \pi(1) = 2$, $\pi^{PR} = [1, 4, 2], \pi = [3]$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(5) $d = 1, \pi(1) = 3$, $\pi^{PR} = [1, 4, 2, 3], \pi = []$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output $\pi^{PR} = [1, 4, 2, 3]$</td>
</tr>
</tbody>
</table>
(4) For each new member $X_N(k+1)$, if it does not dominate any member of $A_{R_{k+1}}$, 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 $A_{R_{k+1}}$.

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

(4.2.1) Compute $u = \min_{i \in [1 \ldots n]} \{ N(X_i(k+1)) \}$, $\forall X_i(k+1) \in A_{R_{k+1}}$, where $N(X_i(k+1))$ represents the number of particles whose value of $X^{gbest}(k+1)$ is $X_i(k+1)$; if $u > g$ ($g$ is an integer, and $g \in [0.025Ps, 0.05Ps]$), then let $u = g$.

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

(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 $H \leftarrow H \setminus \{ X_e(k+1) \}$; if $N(X_e(k+1)) = 0$, let $f = f-1$; if $N(X_e(k+1)) < u$ and $f < C$, repeat (4.2.3); if $N(X_e(k+1)) < u$ 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 ($10 \times 7$ and $20 \times 7$ 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 ($p'_{i,j}$) and the pessimistic value ($p''_{i,j}$) 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, $\delta_{11}$, $\delta_{12}$, $\delta_{21}$, and $\delta_{22}$ 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.

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 ($I^1_{\epsilon}$) [30], hypervolume indicator (HV), and coverage indicator (C) [31].

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

$$I_{\epsilon}^1(\text{OF}) = \inf_{\epsilon \in \mathbb{R}} \left\{ \exists Z^1 \in \text{PF}^* : D^1 \cap Z^1 \neq \emptyset \right\},$$

(26)

where $Z^1 \in \mathbb{R}^L$ if and only if $\forall 1 \leq l \leq L : z^1_l < \epsilon + z^2_l$, $z^1 = (z^1_1, \ldots, z^1_L)$ and $Z^2 = (z^2_1, \ldots, z^2_L)$ 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 $Z^2 \in \text{OF}$, a hypercube $\gamma_i$ is constructed with a reference point $R$ and the objective vector $Z^2$ 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:

$$HV = \text{volume} \left( \bigcup_{i=1}^{\lvert \text{OF} \rvert} \gamma_i \right).$$

(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 $I_{1+}$ and HV indicators.

<table>
<thead>
<tr>
<th></th>
<th>MOPSO-M</th>
<th>General MOPSO</th>
<th>NSGA-II</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Median</td>
<td>IQR</td>
<td>Median</td>
</tr>
<tr>
<td>$I_{1+}$</td>
<td>10 x 7</td>
<td>1.34e-02</td>
<td>1.04e-02</td>
</tr>
<tr>
<td></td>
<td>20 x 7</td>
<td>1.87e-01</td>
<td>1.11e-01</td>
</tr>
</tbody>
</table>

Table 5: Results of the Wilcoxon rank-sum test for the $I_{1+}$ and HV indicators.

<table>
<thead>
<tr>
<th></th>
<th>General MOPSO</th>
<th>NSGA-II</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10 x 7</td>
<td>20 x 7</td>
</tr>
<tr>
<td>$I_{1+}$</td>
<td>10 x 7</td>
<td>102.5 (5.87e-08)</td>
</tr>
<tr>
<td>HV</td>
<td>MOPSO-M</td>
<td>87.5 (2.45e-08)</td>
</tr>
</tbody>
</table>

Notes: for the $I_{1+}$ indicator, we apply wilcox.test ($I_{1+}$ (MOPSO-M), $I_{1+}$ (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.

In summary, the results of the $I_{1+}$ and HV values suggest that MOPSO-M outperforms 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 x 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:

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 $I_{1+}$, 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 $20 \times 7$ fuzzy FSSP using MOPSO-M.

<table>
<thead>
<tr>
<th>Index</th>
<th>$\alpha_1$</th>
<th>$\alpha_2$</th>
<th>$\alpha_3$</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
<th>Optimal solutions (processing sequence)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.55</td>
<td>0.30</td>
<td>0.15</td>
<td>4931.50</td>
<td>0.898</td>
<td>0.090</td>
<td>3-2-7-4-9-12-10-5-15-16-8-11-14-20-13-18</td>
</tr>
<tr>
<td>2</td>
<td>0.40</td>
<td>0.30</td>
<td>0.30</td>
<td>4954.00</td>
<td>0.974</td>
<td>0.723</td>
<td>3-2-7-1-4-9-10-5-12-17-15-16-3-8-11-14-20-18</td>
</tr>
<tr>
<td>3</td>
<td>0.30</td>
<td>0.35</td>
<td>0.35</td>
<td>4969.25</td>
<td>0.990</td>
<td>0.889</td>
<td>3-1-2-7-4-9-10-5-12-17-15-16-8-11-14-20-18</td>
</tr>
</tbody>
</table>

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

Figure 4(a) shows the Epsilon indicator for the $20 \times 7$ fuzzy FSSP, and Figure 4(b) shows the Epsilon indicator for the $10 \times 7$ fuzzy FSSP. 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 $20 \times 7$ 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:

$$w_{s,t} = \begin{cases} \frac{f_{s,t}^\text{max} - f_{s,t}}{f_{s,t}^\text{max} - f_{s,t}^\text{min}}, & t = 1, \\ \frac{f_{s,t} - f_{s,t}^\text{min}}{f_{s,t}^\text{max} - f_{s,t}^\text{min}}, & t = 2, 3, \end{cases}$$

(29)
Table 8: Fuzzy processing time and fuzzy due date of the 10 × 7 fuzzy FSSP.

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

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.

<table>
<thead>
<tr>
<th>i</th>
<th>k</th>
<th>( PR_{i,k} )</th>
<th>( PR_{k,i} )</th>
<th>Graphical representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>1</td>
<td>(-1)</td>
<td>1 \textgreater 7</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>1</td>
<td>(-1)</td>
<td>5 \textgreater 9</td>
</tr>
</tbody>
</table>

where \( w_{i,s} \) is the \( s \)-th nondimensional objective value of the \( i \)-th optimal solution, \( f_{i,s} \) is the \( s \)-th objective value of the \( i \)-th optimal solution, \( f_{i}^{\text{max}} = \max\{f_{1,1}, f_{1,2}, \ldots, f_{25,5}\} \), and \( f_{i}^{\text{min}} = \min\{f_{1,1}, f_{1,2}, \ldots, f_{25,5}\} \). The weighted sum of the nondimensional objective values can be calculated by

\[
W_i = \sum_{s=1}^{3} \alpha_s w_{i,s},
\]

where \( \sum_{s=1}^{3} \alpha_s = 1, \ 0 \leq \alpha_s \leq 1 \). The optimal solution with the maximum value of \( W_i \) 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...
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


Submit your manuscripts at http://www.hindawi.com