Received December 24, 2019, accepted January 25, 2020, date of publication February 10, 2020, date of current version February 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2972619

An Improved Lexicographical Whale Optimization Algorithm for the Type-II Assembly Line Balancing Problem Considering Preventive Maintenance Scenarios

KAI MENG, QIUHUA TANG, ZIKAI ZHANG, AND XINBO QIAN
Key Laboratory of Metallurgical Equipment and Control Technology, Ministry of Education, Wuhan University of Science and Technology, Wuhan 430081, China
Hubei Key Laboratory of Mechanical Transmission and Manufacturing Engineering, Wuhan University of Science and Technology, Wuhan 430081, China

Corresponding author: Qiuhua Tang (tangqiuhua@wust.edu.cn)

This work was supported by the National Natural Science Foundation of China, under Grant 51875421.

ABSTRACT In the traditional assembly line balancing, all the workstations are assumed available and hence the unavailability of any workstation brings about the stoppage of the whole line and the waste of the production capacity in the rest workstations. Considering the planning characteristic of preventive maintenance, this paper proposes a novel methodology of integrating the preventive maintenance scenarios into assembly line balancing problems to bypass the unavailable workstation. A lexicographic model is formulated to generate multiple task assignment plans that ensure primarily the high productivity under regular operation scenario and guarantee secondarily the production continuity under preventive maintenance scenarios. And, an improved whale optimization algorithm (IWOA) with three modifications is proposed to solve this problem. Specifically, a combined crossover operator enhances better combination in exploration; three best search agents promote the exploitation; partial regeneration avoids being trapped in local optima. More than five thousand experiments demonstrate that the joint of three modifications endows the IWOA significant superiority over six variants and other six well-known algorithms. Moreover, integrating preventive maintenance scenarios into the assembly line balancing problem increases the production efficiency by 1% at the cost of small production adjustment.

INDEX TERMS Assembly line balancing, lexicographical optimization, preventive maintenance, whale optimization algorithm.

I. INTRODUCTION

An assembly line is a special flow-line manufacturing system to integrate the prefabricated parts into a finished product. It comprises of a cluster of successive workstations connected by a material handling system. The assembly line balancing problem (ALBP) aims to assign assembly tasks to these workstations with the optimization of one or more objectives and satisfaction of precedence relationship and cycle time constraints. An effective assignment of the assembly tasks contributes largely to the productivity and smoothness of the line.

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

According to the adopted objectives, the ALBP is generally divided into four categories: type-I, type-II, type-E and type-F [1]. The type-I minimizes the number of workstations under a known cycle time [2], [3] and the type-II focuses on minimizing the cycle time of predefined workstations [4], [5]. The type-E tries to maximize line efficiency and productivity with a variable number of workstations and a variable cycle time [6]. And, the type-F aims to find a feasible balance for a given number of stations and a given cycle time is known [7].

However, in the above ALBPs, there is a common assumption that machine breakdown never occurs in the production process, which is obviously impractical. In practice, due to fatigue, wear or aging et al., all the machines in the workstations tend to degrade and even fail to work, and all the machines must be maintained to guarantee the reliability and availability of the production line. Normally, there are two types of maintenance actions: corrective maintenance and preventive maintenance. Corrective maintenance corrects the defects as soon as possible once a machine is breakdown, but it cannot be planned due to the random occurrence of failures [8]. Preventive maintenance (PM) is a planned action to improve the reliability of machines. It keeps all machines in a satisfactory state and reduces the unexpected shutdowns. It is worth noting that when one machine in any workstation is maintained, the unavailability of this workstation might result in the blockage in upstream workstations and starvation in downstream workstations, and further lead to the stoppage of the whole assembly line [9]. In this case, the production capacity of the rest workstations without maintenance is wasted unfortunately. If the workstations to be maintained can be bypassed strategically, the productivity and stability of the line may be promoted. Hence, it is necessary to consider PM into the ALBP.

Till now, there has been no literature concerning the issue of ALBP considering PM. In contrast, existing literature has demonstrated that integrating PM into the production scheduling within flexible flow shops can reduce the running cost and system unavailability time. Specifically, Yu and Seif [10] demonstrated the integration of them can shorten the total maintenance and tardiness costs. Feng *et al.* [11] addressed a sequence-dependent group scheduling problem within a flexible flow shop and they claimed that implementing PM at the idle time between groups can shorten the production cycle time and save the manufacturing cost. Moradi *et al.* [12] claimed that the joint of them could effectively reduce the makespan and the unavailability of the production system. Since an assembly line system is a special case of flow shop [13], the joint optimization of PM and ALBP may be effective, and thereafter ALBP considering PM is studied in this paper in detail.

When PM is included in ALBP, besides the traditional objectives, new objectives such as minimizing total task alteration should be considered concurrently. To coordinate these objectives, three procedures including weighted summation, Pareto-front optimization, and lexicographic optimization are generally adopted. Among them, the weighted summation approach [14] endows the more important objectives with higher weights and further combines all the objectives into one to reduce the computational complexity. It should be noted that it is difficult to quantify the weights of objectives for the problem under consideration. The Pareto-front optimization [15], [16] probes a set of non-dominated solutions where no objective can be improved without sacrificing others. Each objective in this method has equal importance; while the objectives in this paper show a significant hierarchy relationship owing to their impact on the productivity of the line. This means the Pareto-front optimization is not fit for this paper. In contrast, the lexicographic optimization approach ranks the objectives by qualitative but not quantitative importance; subsequently, it derives the upper-level objective first and then optimizes the lower-level one [17], [18]. Hence, lexicographic optimization is utilized in this paper to handle the hierarchy objectives.

To solve the complex ALBP-related problem, various solution procedures are developed, which include exact algorithms, heuristic, and meta-heuristic algorithms. The exact algorithms, such as the branch-and-bound method [19]–[21], are effective to solve small-scaled cases but can hardly obtain the optimal solutions for large-scale cases. Heuristic algorithms, which include Hoffman method [22] and ranked positional weight method [23], [24], can obtain an efficient solution easily and quickly, but they are problem-dependent and easy to fall into the local optimum. Instead, meta-heuristic algorithms show strong strengths like flexibility, derivation-free mechanisms, and local optima avoidance [25]. And, a group of promising meta-heuristic algorithms is utilized to solve the ALBP-related problems, which include grey wolf algorithm [26], discrete artificial swarm algorithm [27], multi-objective cellular genetic algorithm [28], and hybrid particle swarm optimization algorithm [29]. Among the emerging meta-heuristic algorithms, whale optimization algorithm (WOA) shows advantages such as simple concept, less adjustment parameter, and easy implementation [30]. Meanwhile, it has been successfully applied to other engineering optimization problems like mobile robot scheduling [31], image segmentation [32], and constrained engineering design problems [33]. Thus, the WOA algorithm is hired here to solve the ALBP considering PM.

In summary, it is sufficient and necessary to integrate the planned PM into the type-II assembly line balancing problem with the predefined workstations. If so, the workstation with maintained machines can be bypassed and other workstations can continue to assemble products. The contributions of this paper are as follows:

(1) The ALBP considering PM is proposed to bypass the workstation under maintenance and guarantee the production continuity in other workstations, and hence to improve the productivity of the whole assembly line.

(2) A lexicographic programming model is formulated to primarily ensure the productivity under regular operation scenario and secondarily recover the production under PM scenarios.

(3) An improved whale optimization algorithm (IWOA) with three modifications, which include crossover operator, three leaders, and partial regeneration, is proposed to solve the large-scale problems due to its strong NP-hardness nature.

The rest of this paper is structured as follows. Section II analyzes the characteristics of ALBP considering PM scenarios. Section III formulates a lexicographic nonlinear mathematical model. Section IV details an improved whale optimization algorithm. Section V analyzes the experimental results and Section VI presents the conclusions and future work.

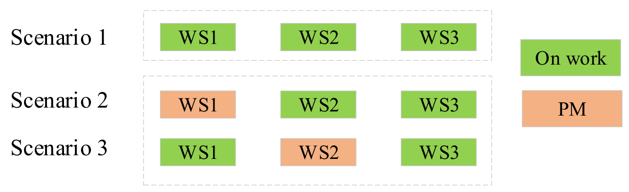


FIGURE 1. Illustration of regular operation scenario and PM scenarios.

II. ALBP CONSIDERING PM SCENARIOS

A. PREVENTIVE MAINTENANCE SCENARIOS

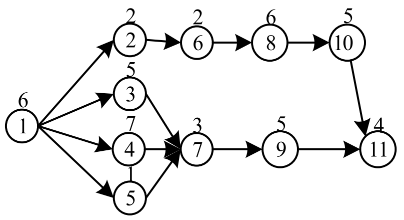
Normally, the preventive maintenance of machines is planned in advance. Take a simple assembly line with three workstations shown in Fig. 1 as an illustrative example. In general, all three workstations are in normal production and none of them is under maintenance. This scenario is called the regular operation scenario and marked as Scenario 1 in Fig. 1. According to the PM plan, some workstations will be maintained in the following month or even week. Obviously, as long as the maintenance capacity and personnel are competent, more than one workstation can be maintained simultaneously. For simplicity, only one workstation can be maintained in each scenario. As shown in Fig. 1, there are three scenarios including one regular operation scenario (Scenario 1) and two PM scenarios (Scenarios 2 and 3).

B. ASSIGNMENT PLANS AND TASK ALTERATIONS

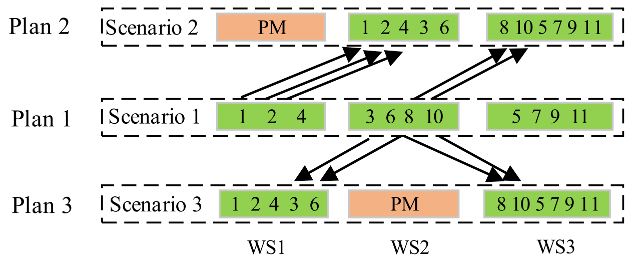
Due to the serial feature of the assembly line system, as long as a workstation is in PM, the upstream workstations will be blocked while the downstream ones will be starved, and subsequently, the whole line will stop running [34]. A challenging idea is that if the workstation under maintenance can be bypassed, other workstations can go on working and the production within an assembly line can continue. In this case, the key to promote the production continuity is to make multiple assignment plans with each plan for a scenario specifically. As explained by Fig. 2, for the regular operation scenario (Scenario 1), all tasks are allocated to the total of three workstations just like the traditional ALBP. For two other PM scenarios in which workstation 1 or 2 is under maintenance respectively, all the tasks are reallocated into the available workstations.

It should be mentioned that task alterations from a PM scenario to the regular operation or vice versa are expected to be as small as possible to promote production stability. Specifically, in Fig. 2, if workstation 1 is to be maintained, tasks (1, 2, 4) are reallocated into workstations 2 and tasks (8, 10) into workstation 3. After maintenance, all these tasks return immediately and the production mode under PM scenario 2 is changed back to the regular operation scenario. Obviously, before or after the quick adjustment, all the tasks are performed by the total available workstations, and productivity continuity is hence assured.

Taking into account the maintenance probability of machines within the assembly line, the direct adjustment from



(a) Precedence graph



(b) Assignment plans and task alteration

FIGURE 2. Illustration of precedence graph, task assignment plans and alterations.

a PM scenario to another PM one is rare. Hence, only the direct adjustment between a PM scenario and the regular one is considered for simplicity. Based on this, the following section tries to formulate the problem of ALB considering PM scenarios.

III. PROBLEM DESCRIPTION

As mentioned above, the ALBP considering PM scenarios is to generate multiple task assignment plans for the regular operation and PM scenarios to ensure the efficiency and stability of production. In each task assignment plan, all the tasks are allocated to the available workstations under the constraint of precedence relations. Obviously, the resulted cycle time for each task assignment plan is unique and it is expected to minimize the task alterations between plans. Hence, a multi-objective model for the type-II assembly line balancing problem considering PM scenarios is established first and is further reformulated into a lexicographical optimization model to handle the hierarchic objectives. The assumptions considered in the model are provided as follows:

(1) Each workstation can perform all the tasks.
(2) Only one product is assembled in this line.
(3) The processing times of tasks are deterministic.
(4) The precedence relationships among the tasks are known.
(5) Each task cannot be further subdivided and must be allocated to exactly one workstation in each task assignment plan.
(6) The walking time in each workstation is negligible.
(7) In each PM scenario, exactly one workstation can be maintained and is predefined.

A. NOTATIONS

1) INDICES AND SETS

k, s, i	Indices of tasks, $\{k, s, i = 1, \ldots, N\}$.
J	Index of workstations,$\{j = 1, \ldots, M\}$.
sc	Index of scenarios,$\{sc = 1, \ldots Q\}$.$sc = 1$ presents the regular operation scenario and the others are PM scenarios.
W	Set of precedence relationships. If k is an immediate predecessor of s, $(k, s) \in W$.
J_{sc}	Set of workstations to be maintained in scenario sc.

2) PARAMETERS

N	The number of tasks.
M	The number of workstations on the line.
Q	The number of scenarios, which equals the number of PM scenarios plus one.
t_i	Processing time of task i.

3) VARIABLES

$X_{ij,sc}$	Binary variable. It equals 1 if task i is assigned to workstation j in scenario sc; otherwise, 0.
TA	Integer variable, total task alteration.
CT_{sc}	Integer variable, the cycle time under scenario s c.

B. MATHEMATICAL MODEL

Similar to the traditional type-II ALBP proposed by Çil *et al.* [5], a mathematical model is proposed as below to generate multiple task assignment plans for the regular operation and PM scenarios.

$$f_1 = \min CT_1 = \min\max_j \sum\nolimits_{i=1}^{N} \left(t_i \cdot X_{ij,1}\right) \tag{1}$$

$$f_2 = \min \left(\left(\sum\nolimits_{sc=2}^{Q} CT_{sc}\right) \Big/ CT^*_{\max} + TA \big/ TA_{\max} \right) \tag{2}$$

Subject to:

$$\sum\nolimits_{i=1}^{N} X_{ij,sc} = 0, \quad \forall j \in J_{sc}, \quad sc \in \{2, \ldots, Q\} \tag{3}$$

$$\sum\nolimits_{j=1}^{J-J_{sc}} X_{ij,sc} = 1, \quad \forall i \in \{1, \ldots, N\}, \quad sc \in \{1, \ldots, Q\} \tag{4}$$

$$\sum\nolimits_{i=1}^{N} X_{ij,sc} \geq 1, \quad \forall j \in (J - J_{sc}), \quad \forall sc \in \{1, \ldots, Q\} \tag{5}$$

$$\sum\nolimits_{i=1}^{N} t_i \cdot X_{ij,sc} \leq CT_{sc}, \quad \forall j \in (J - J_{sc}), \quad \forall sc \in \{1, \ldots, Q\} \tag{6}$$

$$\sum\nolimits_{j=1}^{J-J_{sc}} j \cdot \left(X_{kj,sc} - X_{sj,sc}\right) \leq 0, \quad \forall (k, s) \in W, \quad sc \in \{1, \ldots, Q\} \tag{7}$$

$$TA = \sum\nolimits_{sc=2}^{Q} \sum\nolimits_{i=1}^{N} \left| \sum\nolimits_{j=1}^{J-J_{sc}} (j \cdot X_{ij,1} - j \cdot X_{ij,sc}) \right| \tag{8}$$

In this model, Equation (1) shows that for the regular operation scenario, the resulted cycle time should ensure all assembly tasks to be allocated evenly and efficiently in all workstations. On the other hand, since the minimum cycle time under different PM scenarios may be different, the summation of all cycle time under PM scenarios is minimized. Meanwhile, the total task alteration is calculated to reduce the adjustment from one PM scenario to the regular operation scenario. It is worth noting that as shown in Equation (2), the total cycle time and total task alteration are both normalized by dividing their respective maximums and are endowed with the same importance for simplicity.

With respect to constraints, Equation (3) emphasizes that the workstation under maintenance is not allowed to undertake any task; while Equation (4) guarantees each task to be allocated into exactly one available workstation under any scenario. Obviously, all the workstations except the one in maintenance will undertake at least one task as illustrated in Equation (5) for fairness. Equation (6) restricts the total processing time in each workstation to be less than the respective cycle time under any scenario. Equation (7) insists to satisfy the precedence relationship between any two tasks. Equation (8) calculates the total task alteration that is represented by the index difference of allocated workstation between regular operation scenario and PM scenario.

With Equations (1-8), the type-II assembly line balancing problem considering PM scenarios is formulated as a multi-objective non-linear programming model.

C. LEXICOGRAPHICAL OPTIMIZATION

To our best knowledge, the assembly line is mostly working under the regular operation scenario in daily production, and thus the first objective, minimizing the cycle time under the regular operation scenario, is far more important than that under PM scenarios. Hence, the lexicographical optimization is introduced to reformulate the above model.

For the upper-level model, Equation (1) is regarded as the objective to ensure productivity under the regular operation scenario. In addition, the constraints involve Equations (4-7) in which only the regular operation scenario is involved. The resulted linear mathematical model for the regular operation scenario is similar to the traditional type-II ALBP proposed by Çil, *et al.* [5]. After solving this upper-level model, a minimum cycle time ,$CT_1^{\min}$, is obtained for the regular operation scenario.

For the lower-level model, the objective in Equation (2) is considered to reduce the production adjustment for PM. With regard to constraints in this case, Equations (3-8) are utilized as constraints. Besides, the optimal result of the upper level, $CT_1^{\min}$, is hired as an upper bound as shown in Equation (9). Due to this, for the regular operation scenario, the resulted productivity is still high, but the task reallocation under the regular operation scenario reduces the task alterations among task assignment plans. As a result, the obtained task assignment plans for regular operation and PM scenarios guarantee the productivity and stability simultaneously.

$$\sum\nolimits_{i=1}^{N} \left(t_i \cdot X_{ij,sc}\right) \leq CT_1^{\min}, sc = 1 \tag{9}$$

D. MODEL LINEARIZATION

Due to the non-differentiation caused by the absolute value in Equation (8), the lower-level model cannot be solved directly by a mixed integer linear programming method and hence it needs to be linearized. In this situation, a positive variable ,Num_{ic}, is introduced to record the value of the task assignment alteration as defined in Equation (10).

$$Num_{ic} = \left|\sum_{j=1}^{J-J_{sc}} \left(j \cdot X_{ij,1} - j \cdot X_{ij,sc}\right)\right| \tag{10}$$

Equation (10) is transformed into Equations (11-12) to eliminate the non-differentiation caused by the absolute value. Sequentially, Equation (8) is finally changed into Equation (13).

$$Num_{ic} \geq \sum_{j=1}^{J-J_{sc}} \left(j \cdot X_{ij,1}\right) - \sum_{j=1}^{J-J_{sc}} \left(j \cdot X_{ij,sc}\right), \quad \forall i \in \{1, \ldots, N\}, sc \in \{1, \ldots, Q\} \tag{11}$$

$$Num_{ic} \geq \sum_{j=1}^{J-J_{sc}} \left(j \cdot X_{ijsc}\right) - \sum_{j=1}^{J-J_{sc}} \left(j \cdot X_{ij,1}\right), \quad \forall i \in \{1, \ldots, N\}, sc \in \{1, \ldots, Q\} \tag{12}$$

$$\min TA = \sum_{sc=2}^{Q} \sum_{i=1}^{N} Num_{ic} \tag{13}$$

Hence, the lower-level optimization model contains Equations (2-7, 10-13). It should be mentioned that the total cycle time and total task alteration are both normalized by dividing $TA_{\max}$ and $CT^*_{\max}$ respectively. For $TA_{\max}$, it is assumed that all tasks should be adjusted from the first workstation to the last, and the alteration of each task is equal to $M - 1$. Hence the $TA_{\max}$ under all PM scenarios can be expressed by Equation (14).

$$TA_{\max} = \sum_{sc=2}^{Q} \sum_{i=1}^{N} (M - 1) \tag{14}$$

For $CT^*_{\max}$, suppose that under every PM scenario, each available workstation except the last one holds only one task with the smallest processing time and the remaining tasks are put into the last workstation. The total processing time in the last available workstation is set as the maximum cycle time under the current scenario. Hence $CT^*_{\max}$ is the sum of the maximum cycle time of all PM scenarios.

IV. AN IMPROVED WHALE OPTIMIZATION ALGORITHM

Enlightened by the bubble-net hunting strategy of humpback whales, the whale optimization algorithm (WOA) [30] is presented as a novel nature-inspired meta-heuristic optimization algorithm. Due to its advantages of simple concept, less adjustment parameter, and easy implementation, it is employed in this paper to solve the ALBP considering PM scenarios.

A. BASIC WOA

The hunting process of humpback whales contains two stages: exploration and exploitation. In the exploration stage, whales probe the ocean and search for prey. If the prey is found, the exploitation stage starts to hunt it. At this stage, humpback whales first dive into the ocean and then swim upward at a shrinking circle and along a spiral-shaped path around the prey simultaneously. In the meantime, they eject bubbles continuously as a net to trap the prey. This unique mechanism is called bubble-net feeding method. Mimicking the unique feeding behavior of humpback whales, the WOA algorithm comprises of three procedures: search for prey, shrinking encircling prey, and spiral updating position. These procedures are selected automatically according to a random number p in [0, 1] and a positive variable $|A|$.

- Search for prey ($p < 0.5$and $|A| \geq 1$)

Whales (search agents) search for prey globally under the guidance of randomly selected agent according to Equations (15-18).

$$\vec{D} = \left|\vec{C} \cdot \overrightarrow{X_{rand}}(t) - \vec{X}(t)\right| \tag{15}$$

$$\vec{X}(t+1) = \overrightarrow{X_{rand}}(t) - \vec{A} \cdot \vec{D} \tag{16}$$

$$\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a} \tag{17}$$

$$\vec{C} = 2 \cdot \vec{r} \tag{18}$$

where, t represents the current iteration. $\vec{X}$ and $\overrightarrow{X_{rand}}$ are the location of the current agent and the randomly chosen search agent. $\vec{r}$ is a random vector with uniform distribution in [0, 1]. The parameter $\vec{a}$ linearly decreases from two to zero over the course of iterations. And, the location of a search agent at the next iteration is determined by the current location of this agent and a randomly chosen one. Owning to randomly chosen search agents, global search is enabled for the WOA algorithm.

Obviously, $\vec{A}$ is in $[-a, a]$ and there is a 50% probability that $\left|\vec{A}\right| \geq 1$ or $\left|\vec{A}\right| < 1$ at the beginning of iterations. When the parameter $\vec{a}$ arrives at one, the probability that $\left|\vec{A}\right| \geq 1$ is zero and the process of searching for prey is terminated.

- Shrinking encircling prey ($p < 0.5$ *and* $|A| < 1$)

Since the best search agent is the nearest to the prey, other search agents may reposition themselves according to the distance to the best as represented by Equations (19-20).

$$\vec{X}(t+1) = \overrightarrow{X^*}(t) - \vec{A} \cdot \vec{D} \tag{19}$$

$$\vec{D} = \left|\vec{C} \cdot \overrightarrow{X^*}(t) - \vec{X}(t)\right| \tag{20}$$

where, $\vec{X}^*$ is the position of the best whale. Clearly, as the parameter $\vec{a}$ decreases from one to zero, $\left|\vec{A}\right|$ goes down to zero in trend, and thus other search agents become closer to the best one at the next iteration.

- Spiral updating position ($p \geq 0.5$)

Meanwhile, other search agents approach the current best one by following the spiral-shaped trajectory as illustrated by Equations (21-22).

$$\vec{X}(t+1) = \vec{D}' \cdot e^{bl} \cdot \cos(2\pi l) + \vec{X}^*(t) \tag{21}$$

$$\vec{D}' = \left|\overrightarrow{X^*}(t) - \vec{X}(t)\right| \tag{22}$$

TABLE 1. Generating the processing sequence of tasks for PM scenario1.

PM scenario 1: {0.97, 0.96, 0.49, 0.80, 0.14, 0.42, 0.91, 0.79, 0.96, 0.66, 0.04}			
Assignable task set	Task priority in set	Selected task	Processing sequence
set={1}	{**0.97**}	1	{1}
set={2,3,4,5}	{**0.96**, 0.49, 0.80, 0.14}	2	{1,2}
set={3,4,5,6}	{0.49, **0.80**, 0.14, 0.42}	4	{1,2,4}
set={3,5,6}	{**0.49**, 0.14, 0.42}	3	{1,2,4,3}
set={5,6}	{0.14, **0.42**}	6	{1,2,4,3,6}
set={5,8}	{0.14,**0.79**}	8	{1,2,4,3,6,8}
set={5,10}	{0.14, **0.66**}	10	{1,2,4,3,6,8,10}
set={5}	{**0.14**}	5	{1,2,4,3,6,8,10,5}
set={7}	{**0.91**}	7	{1,2,4,3,6,8,10,5,7}
set={9}	{**0.96**}	9	{1,2,4,3,6,8,10,5,7,9}
set={11}	{**0.04**}	11	{1,2,4,3,6,8,10,5,7,9,11}

where, b is a constant for defining the shape of the logarithmic spiral. l is a random number in [−1, 1]. $\vec{D}'$ is the distance vector between the current best one and each of other search agents. Through the location updating, other search agents in the next iteration will move toward the current best along a spiral-shaped trajectory.

B. SCENARIO-ORIENTED ENCODING AND DECODING

Due to the discrete nature of the problem under consideration, the original WOA algorithm cannot be applied directly and thus the random key method [26] is introduced for encoding and decoding. For each search agent, the number of segments equals to that of scenarios, and the sequence of these segments represents sequentially the regular operation scenario, PM scenario 1, PM scenario 2, etc. In each segment, the number of elements is equivalent to the number of tasks. The value of each element, randomly generated in [0, 1], represents the priority of the corresponding task. Taking a small case with three PM scenarios and eleven tasks as an example, it has four segments with eleven elements for each as illustrated in Fig. 3.

For each scenario, the decoding process involves two parts: generating the processing sequence of tasks and assigning the sequenced tasks into available workstations. The processing sequence should be generated according to the task priorities and precedence relations. Hence, a task, whose immediate predecessors have all been completed, can be put into the assignable task set. Only the task with the largest priority value in this set is selected and added into the processing sequence. Then, the assignable task set is updated; once it is empty, all the tasks are ordered in a processing sequence. Based on the code provided in Fig 3 and the precedence relations among tasks in Fig 2(a), the process to generate the processing sequence for PM scenario 1 is illustrated in Table 1.

Since all the tasks are sequenced in order, the sequenced tasks are to be assigned to the available workstations.

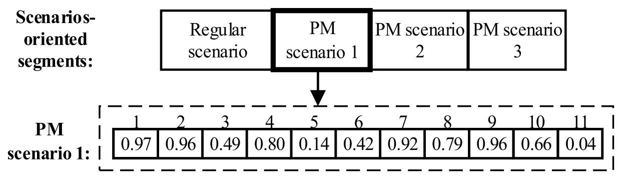


FIGURE 3. The scenario-oriented encoding method.

```
Scenario-oriented decoding
Input: a code for all scenarios.
Output: multiple task assignment plans.
Begin:
 1: For $sc$=1 to $Q$ do
 2:     Processing sequence $\mathcal{H}=\{\emptyset\}$.
 3:     Construct the assignable task set $\Omega$.
 4:     While ($\Omega \neq \emptyset$) do
 5:         Put the task with the largest priority in $\Omega$ to the end of $\mathcal{H}$.
 6:         Update the assignable task set $\Omega$.
 7:     End while
 8:     Set the initial cycle time $CT_{sc} = \sum_{i=1}^{I} t_i / |J - J_{sc}|$.
 9:     Set the processing time of the last workstation $CT_{last} = inf$.
10:     While $CT_{last} > CT_{sc}$ do
11:         $CT_{sc} = CT_{sc} + 1$
12:         Allocate the sequenced tasks into $J - J_{sc}$ except the last one under the constraint of $CT_{sc}$.
13:         Put all the remaining tasks into the last available workstation.
14:         Calculate $CT_{last}$.
15:     End while
16: End For
```

FIGURE 4. Scenario-oriented decoding mechanism.

In this process, the initial cycle time CT_{sc} is set through the division of the total processing time of tasks by the number of all available workstations as formulated in $\left(CT_{sc} = \sum_{i=1}^{I} t_i \Big/ |J - J_{sc}|\right)$. Then, supposing that the current cycle time constraint is satisfied, sequentially allocate as many tasks as possible according to the processing sequence into all the available workstations except the last one. Finally, put all the remaining tasks into the last available workstation. As long as the total processing time in the last workstation is larger than the current cycle time, set the current cycle time plus a constant and repeat the allocating process; otherwise, terminate this process, and report the minimum cycle time and task assignment plan for the given scenario.

As the above mentioned, a unique task assignment plan is obtained for each scenario, and hence, the task alteration from regular operation scenario to each PM scenario can be calculated. The task alteration of one PM scenario is the absolute difference of workstation index of each task. The total task alteration is the summation of task alterations of all PM scenarios. To sum up, the pseudo code of scenario-oriented decoding is shown in Fig 4.

C. EXPLOITATION WITH THREE LEADERS

In the basic WOA algorithm, only the current best search agent leads the evolution of the whole population in the exploitation stage. If the current best search agent locates at a local optimum, the population has a great possibility of being trapped into it, especially in the later stage of iterations. To accelerate convergence speed but overcome premature convergence, the concept of more leaders is introduced. According to grey wolf optimization proposed by Mirjalili *et al.* [35], three best search agents so far are selected simultaneously for the exploitation stage. Hence, the shrinking encircling prey mechanism is changed as illustrated in Equations (23-25).

$$\begin{aligned}\overrightarrow{D_1} &= \left|\vec{C}\cdot\overrightarrow{X_1^*}(t)-\vec{X}(t)\right|;\\ \overrightarrow{D_2} &= \left|\vec{C}\cdot\overrightarrow{X_2^*}(t)-\vec{X}(t)\right|;\\ \overrightarrow{D_3} &= \left|\vec{C}\cdot\overrightarrow{X_3^*}(t)-\vec{X}(t)\right| \end{aligned}\tag{23}$$

$$\begin{aligned}\overrightarrow{X_1} &= \overrightarrow{X_1^*}(t)-\vec{A}\cdot\overrightarrow{D_1};\\ \overrightarrow{X_2} &= \overrightarrow{X_2^*}(t)-\vec{A}\cdot\overrightarrow{D_2};\\ \overrightarrow{X_3} &= \overrightarrow{X_3^*}(t)-\vec{A}\cdot\overrightarrow{D_3}\end{aligned}\tag{24}$$

$$\vec{X}(t+1) = \left(\overrightarrow{X_1}+\overrightarrow{X_2}+\overrightarrow{X_3}\right)/3 \tag{25}$$

where, $\overrightarrow{X_1^*}$, $\overrightarrow{X_2^*}$, and $\overrightarrow{X_3^*}$ are the three best search agents obtained so far. $\overrightarrow{X_1}$, $\overrightarrow{X_2}$, and $\overrightarrow{X_3}$ represent the possible position of the current search agent $\vec{X}(t)$ according to the three best ones respectively. $\vec{X}(t+1)$ is the updated position of the search agent for the next iteration.

Similarly, the spiral updating position mechanism is also changed as illustrated in Equations (26-28).

$$\begin{aligned}\overrightarrow{D_1'} &= \left|\overrightarrow{X_1^*}(t)-\vec{X}(t)\right|;\\ \overrightarrow{D_2'} &= \left|\overrightarrow{X_2^*}(t)-\vec{X}(t)\right|;\\ \overrightarrow{D_3'} &= \left|\overrightarrow{X_3^*}(t)-\vec{X}(t)\right|\end{aligned}\tag{26}$$

$$\begin{aligned}\overrightarrow{X_1} &= \overrightarrow{D_1'}\cdot e^{bl}\cdot\cos(2\pi l)+\overrightarrow{X_1^*}(t);\\ \overrightarrow{X_2} &= \overrightarrow{D_2'}\cdot e^{bl}\cdot\cos(2\pi l)+\overrightarrow{X_2^*}(t);\\ \overrightarrow{X_3} &= \overrightarrow{D_3'}\cdot ye^{bl}\cdot\cos(2\pi l)+\overrightarrow{X_3^*}(t)\end{aligned}\tag{27}$$

$$\vec{X}(t+1) = \left(\overrightarrow{X_1}+\overrightarrow{X_2}+\overrightarrow{X_3}\right)/3 \tag{28}$$

where, $\overrightarrow{D_1'}$, $\overrightarrow{D_2'}$, and $\overrightarrow{D_3'}$ are the distance vector from the current search agent to the best three search agents obtained so far.

Obviously, the updated agent $\vec{X}(t+1)$ guided by three best leaders is of more robustness than that by a single leader.

D. ENHANCED EXPLORATION VIA CROSSOVER

In the exploration stage of the basic WOA, each search agent updates its position by a randomly-chosen one, and causes the loss of efficient partial information. Hence, this paper employs a crossover operator to exchange the gene segments between two parent chromosomes and further explore a better combination of efficient partial information. Specifically, for each search agent, another search agent is selected randomly from individuals with better performance. Subsequently for each scenario, two intersections are randomly generated and the segment between them is exchanged with that of another agent.

E. PARTIAL REGENERATION

If the best agent remains unchanged for a number of successive iterations, it is probable that the current best agent is a local optimum. To avoid being trapped in a local optimum, as long as the number of this kind of iterations reaches a predefined threshold T_r, partial regeneration of search agents is executed. Specifically, a certain number of agents except the best one in the current population are replaced by the newly generated. Note that, the percentage of newly generated search agents, which is set as S_r, is expected to be relatively small so as to keep the stability of the population.

F. IWOA FRAMEWORK

Three above-mentioned improvements are hereafter integrated into the basic WOA to improve the performance of the algorithm. First, the exploration stage is enhanced by using two-point crossover operator, which aims to explore better combinations of efficient partial information. Second, the top three rather than the best one of the search agents are selected to overcome premature convergence in the exploitation stage. Third, to avoid being trapped in a local optimum, partial regeneration is introduced to refresh search agents in the population. The flowchart of the proposed IWOA is depicted in Fig. 5.

It is worth noting that two objectives need to be lexicographically optimized in this paper. When comparing the performance of two search agents, the one is better if:

(1) It has a smaller upper-level objective, the cycle time under the regular operation scenario;
(2) Otherwise, it has a smaller lower-level objective.

V. COMPUTATIONAL EXPERIMENTS AND RESULTS

In this section, four types of experiments are designed to verify the effectiveness of the proposed mathematical model and the improved IWOA algorithm. Specifically, the necessity of ALBP considering PM scenarios is first analyzed graphically and numerically. Then the validity of the mathematical model is confirmed on small-scaled benchmark cases. After calibrating the parameters of each algorithm, the performance of the proposed algorithm is compared with its variants and other six algorithms.

It should be mentioned that the mathematical model is programmed with GAMS/Cplex 24.8.5, and all the algorithms are coded with MATLAB R2016a and run on a computer with Intel(R) Core(TM) i5 CPU, 2.80 GHz, and 4.00GB RAM. The termination criterion for each run is the elapsed CPU time set as $N*N*10$ milliseconds, where N is the number

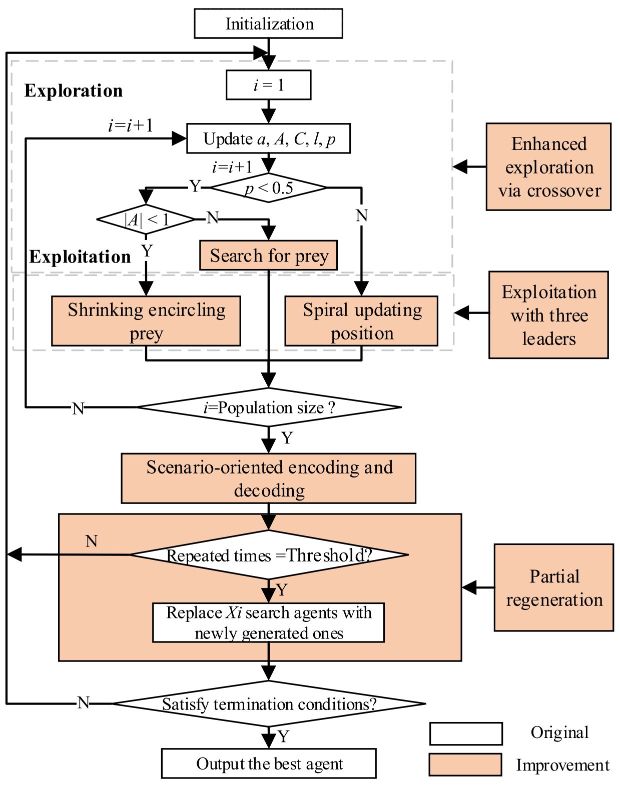

Initialization
Exploration
i = 1
i=i+1
Update a, A, C, l, p
i=i+1
p < 0.5
|A| < 1
Search for prey
Exploitation
Shrinking encircling prey
Spiral updating position
Enhanced exploration via crossover
Exploitation with three leaders
i=Population size ?
Scenario-oriented encoding and decoding
Repeated times =Threshold?
Replace Xi search agents with newly generated ones
Partial regeneration
Satisfy termination conditions?
Output the best agent
Original
Improvement

FIGURE 5. The general framework of IWOA.

of tasks. Meanwhile, to evaluate the result of each experiment, the relative percentage deviation (*RPD*) similar to Zhang *et al.* [36] is introduced. It can be calculated by Equation (29).

$$RPD = \frac{([f_1]+f_2)_{sol} - ([f_1]+f_2)_{best}}{([f_1]+f_2)_{best}} \times 100\% \qquad (29)$$

where $(\cdot)_{sol}$ and $(\cdot)_{best}$ show the solution obtained by an algorithm and the best solution among all the algorithms. f_1 and f_2 are objective values of the upper- and lower-level optimization problems, respectively. It is known that the value of the upper-level objective is much greater than one while that of the lower-level is less than one. Taking into account that the upper-level objective is much more important than the lower-level one, when evaluating the performance of each solution, the former is rounded and placed in the integer while the latter is placed in the decimal place, which is shown by $([f_1]+f_2)$ for simplicity.

Since PM scenarios are not included in traditional benchmark cases as observed in the literature, a set of benchmarks is established on the ground of the well-known ALBPs. The number of tasks and stations, processing times and precedence relationships are from the website <https://assembly-line-balancing.de/>. It should be mentioned that workstations to be maintained are randomly selected. Specific information about these benchmark cases is listed in TABLE 2. The first two columns represent the index

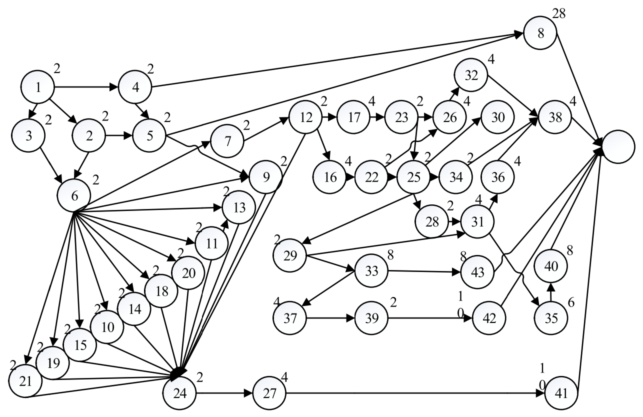


FIGURE 6. Processing times and precedence relations of the aircraft part assembly.

and name of cases respectively and the third column shows the number of tasks. The fourth and fifth column informs the number of workstations and the set of respective workstations to be maintained. In total, more than 5000 experiments are conducted based on these benchmark cases.

A. NECESSITY OF CONSIDERING PM SCENARIOS INTO ALBP

To illustrate the significance of the ALBP considering PM scenarios, a real aircraft part assembly line with 6 workstations and 44 tasks is taken as an example. Among these workstations, the workstations 1 and 4 are maintained respectively. The processing time and precedence relations of tasks are described in Fig 6. In the figure, the number in the circle represents the index of tasks, the arrows between circles represent precedence relations, and the number over the circle represents the processing time (h) of the corresponding task.

According to the need of the work field, the following assumptions are given.

(1) The assembly line works two shifts with eight hours for each and thirty days a month.

(2) Limited by technological and personnel requirements, each PM lasts 4 hours.

(3) The production adjustment between the regular operation scenario and any PM scenario takes 30 minutes.

Fig. 7 reports the final task assignment plans. In this figure, each box shows a task and the number in the box represents the index of tasks. The yellow boxes illustrate that a task will be allocated to a different workstation under the given PM scenario compared with that under the regular operation scenario. Fig. 7 (a) depicts the optimal result of the upper level problem in which PM scenarios are not considered. Obviously, the obtained cycle time is 30h and the total processing time in each workstation is relatively balanced.

Fig. 7(b-d) reports the task assignment plans under regular operation and PM scenarios when PM is executed on workstations 1 and 4 respectively and tasks are not allowed to assign to it. For the regular operation scenario, the resulted cycle time reported in Fig. 7(b) is the same as

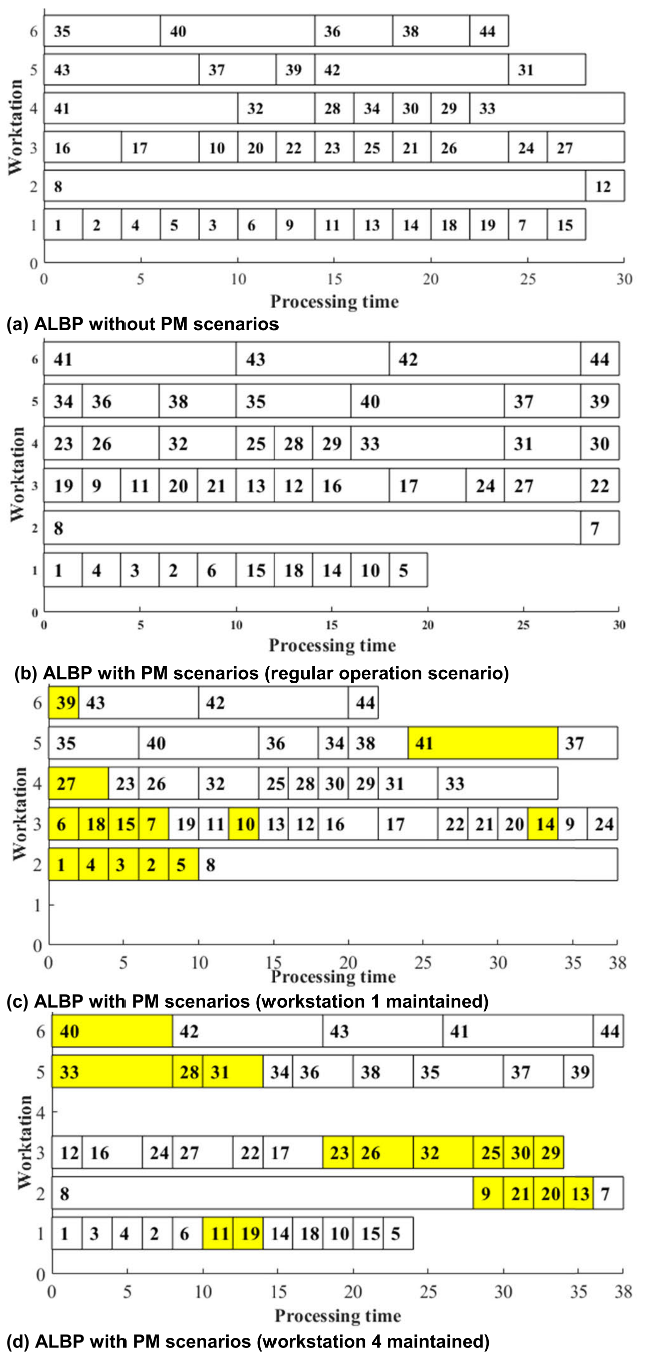
TABLE 2. Benchmark cases information.

No.	Case name	Tasks	Workstations	Workstations in PM	No.	Case name	Tasks	Workstations	Workstations in PM
1	Mertens	7	3	{2}	23	Hahn	53	10	{7; 9; 10}
2	Mertens	7	3	{2; 3}	24	Hahn	53	14	{4; 5; 9}
3	Mertens	7	4	{2; 3}	25	Tonge	70	7	{4; 5; 7}
4	Mertens	7	4	{3; 4}	26	Tonge	70	10	{1; 4; 10}
5	Bowmon	8	3	{2}	27	Tonge	70	14	{9; 10; 12}
6	Bowmon	8	3	{2; 3}	28	Tonge	70	19	{4; 5; 11}
7	Bowmon	8	4	{2; 3}	29	Lutz	89	8	{3; 7; 8}
8	Bowmon	8	4	{3; 4}	30	Lutz	89	12	{3; 4; 5}
9	Jackson	11	3	{2}	31	Lutz	89	16	{2; 9; 12}
10	Jackson	11	3	{2; 3}	32	Lutz	89	21	{3; 15; 18}
11	Jackson	11	4	{2; 3}	33	Acrus	111	9	{3; 6; 8}
12	Jackson	11	4	{3; 4}	34	Acrus	111	13	{1; 2; 13}
13	Roszieg	25	3	{1; 2; 3}	35	Acrus	111	17	{10; 12; 17}
14	Roszieg	25	4	{2; 3; 4}	36	Acrus	111	22	{6;10;12}
15	Roszieg	25	6	{1; 2; 3}	37	Barthol	148	10	{4; 8; 9}
16	Roszieg	25	9	{1; 3; 9}	38	Barthol	148	14	{4; 7; 14}
17	Gunther	35	4	{1; 2; 4}	39	Barthol	148	21	{11; 16; 17}
18	Gunther	35	5	{1; 2; 4}	40	Barthol	148	29	{11; 19; 22}
19	Gunther	35	7	{1; 4; 6}	41	Scholl	297	19	{4; 17; 18}
20	Gunther	35	12	{1; 3; 8}	42	Scholl	297	29	{4; 7; 13}
21	Hahn	53	5	{1; 2; 4}	43	Scholl	297	38	{10; 16; 23}
22	Hahn	53	7	{3; 6; 7}	44	Scholl	297	50	{11; 31; 35}

TABLE 3. Computational results by GAMS/Cplex and IWOA.

No.	Case	M	Q − 1	J_{sc}	$CT^*_{\max}$	$TA_{\max}$	GAMS/Cplex					IWOA			
							CT_1	CT^*	TA	CPU(s)	Gap	CT_1	CT^*	TA	CPU(s)
1	Mertens	3	1	2	28	14	10	15	2	0.376	0	10	15	2	0.0426
2		3	2	2; 3	56	28	10	15;15	5	0.469	0	10	15;15	5	0.0638
3		4	2	2; 3	50	42	9	10;10	4	0.565	0	9	10;10	4	0.0624
4		4	2	3; 4	50	42	9	10;10	4	0.5	0	9	10;10	4	0.0598
5	Bowmon	3	1	2	72	16	28	38	3	0.265	0	28	38	3	0.0423
6		3	2	2; 3	144	32	28	38;38	7	0.5	0	28	38;38	7	0.0597
7		4	2	2; 3	134	48	22	28;28	5	0.516	0	22	28;28	5	0.0674
8		4	2	3; 4	134	48	22	28;28	9	0.483	0	22	28;28	9	0.0670
9	Jackson	3	1	2	45	22	16	23	3	0.422	0	16	23	3	0.0526
10		3	2	2; 3	90	44	16	23;23	7	0.609	0	16	23;23	7	0.0729
11		4	2	2; 3	86	66	12	17;17	5	0.688	0	12	17;17	5	0.0762
12		4	2	3; 4	86	66	12	17;17	6	0.609	0	12	17;17	6	0.0728

that in Fig. 7(a), but their task assignment plans are different. This observation means that there is no loss in production productivity under regular operation conditions. Moreover, the task assignment plan in Fig. 7(b) is more convenient for production adjustment. The cycle times under PM scenarios are 38h, which is significantly larger than that under regular operation scenario. Even if workstation 1 or 4 is in maintenance, the whole line can be still in production. This results in $(4 - 0.5 \times 2) \times 2/38 \approx 0.158$ more aircraft parts to be produced in a month. Compared with the total monthly production without PM scenarios that equals to $(16 \times 30 - 4 \times 2)/30 \approx 15.73$, the proposed methodology of integrating PM scenarios into the ALBP achieves obviously 1% higher production efficiency, which is of large economic value given the total production quantity in the plant.



(a) ALBP without PM scenarios
(b) ALBP with PM scenarios (regular operation scenario)
(c) ALBP with PM scenarios (workstation 1 maintained)
(d) ALBP with PM scenarios (workstation 4 maintained)

FIGURE 7. Tasks assignment plan of ALBP with or without PM scenarios.

It can be concluded that it is necessary to consider PM scenarios into assembly line balancing, and this integration can improve production efficiency largely.

B. VERIFICATION OF MATHEMATICAL MODEL

To test the validity of the proposed lexicographical optimization model, the small-scaled benchmarks illustrated in TABLE 2 are utilized. For each case, the lexicographical optimization model is first solved to optimality, and then, the obtained upper- and lower-level objective values are set

TABLE 4. Orthogonal array through the taguchi experiment for IWOA.

No.	Levels of parameters			Average Result	
	Population size	Iteration threshold	Selective percentage	f_1	f_2
1	30	10	10%	1513	0.079
2	30	20	20%	1515	0.086
3	30	30	30%	1517	0.074
4	50	10	20%	1516	0.091
5	50	20	30%	1507	0.070
6	50	30	10%	1519	0.088
7	70	10	30%	1512	0.076
8	70	20	10%	1512	0.087
9	70	30	20%	1517	0.070

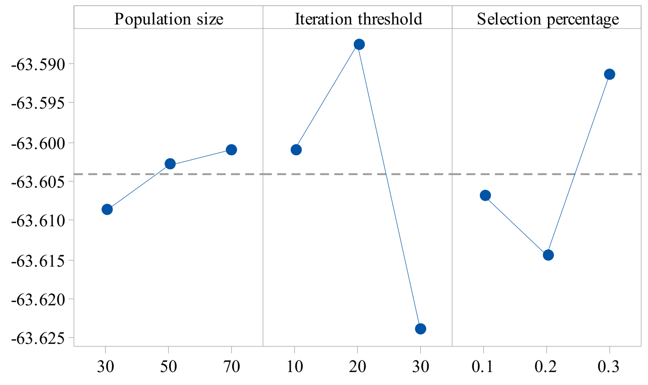


FIGURE 8. Signal-to-noise ratio main effect map.

as termination criteria of IWOA. The computational results are shown in TABLE 3. It should be mentioned that the first five columns introduce the basic information of cases sequentially, including the case index, the case name, the number of workstations, the number of PM scenarios and the maintained workstation for each PM scenario. Columns 6-7 introduce $CT^*_{\max}$ and $TA_{\max}$ for each case. Columns 8-10 report the resulted cycle time CT_1 for regular operation scenario, the corresponding cycle time CT^* for each PM scenario, and the total task alteration TA for all PM scenarios respectively. Columns 11-12 represent CPU time (s) and the obtained relative gap. Columns 13-16 report the results calculated by the IWOA.

Obviously, both GAMS/Cplex and IWOA obtain the optimal solution for each case, but the computational time spent by GAMS/Cplex is far more than that by the IWOA. This demonstrates that the proposed IWOA algorithm can achieve optimal results with smaller computational efforts.

C. PARAMETER CALIBRATION

Since the controlled parameters have a significant influence on the stability and efficiency of algorithms, calibration

TABLE 5. Best parameter combination calibrated for each algorithm.

Parameters	Range	Value
WOA algorithm		
The number of individuals	30, 50, 70	70
WOA_LD algorithm		
The number of individuals	30, 50, 70	30
WOA_CR algorithm		
The number of individuals	30, 50, 70	70
WOA_PR algorithm		
The number of individuals	30, 50, 70	70
Iteration threshold	10, 20, 30	20
Selective percentage	10%, 20%, 30%	10%
WOA_LD_CR algorithm		
The number of individuals	30, 50, 70	70
WOA_LD_PR algorithm		
The number of individuals	30, 50, 70	50
Iteration threshold	10, 20, 30	10
Selective percentage	10%, 20%, 30%	30%
WOA_CR_PR algorithm		
The number of individuals	30, 50, 70	30
Iteration threshold	10, 20, 30	10
Selective percentage	10%, 20%, 30%	20%
GWO algorithm		
Population size	30, 50, 70	70
GA algorithm		
Population size	30, 50, 70	70
Crossover rate	0.6, 0.7, 0.8	0.6
Mutation rate	0.06, 0.07, 0.08	0.08
Select rate	0.7 0.8 0.9	0.7
PSO algorithm		
Population size	30, 50, 70	70
Whale swarm algorithm		
Population size	30, 50, 70	70
MFO algorithm		
Population size	30, 50, 70	30
HHO algorithm		
Population size	30, 50, 70	50

experiments are performed for determining the best combination of parameters of each algorithm. For the proposed IWOA, the tested parameters are the population size (PS), iteration threshold (T_r), regeneration percentage (S_r). The levels of these parameters are: PS {30, 50, 70}, T_r{10, 20, 30}, and S_r{10%, 20%, 30%}. To obtain the corresponding response value, the orthogonal array is employed for the experiments as shown in Table 4. Considering that the parameters for the large-scaled problems are also effective for the middle-scaled and small-scaled ones, the largest case of P297 with 50 workstations is utilized. And the experiment of each parameter combination runs 10 times.

The calibration results are presented in Table 4. The two rightmost columns in Table 4 report the average value of 10 times under each parameter combination. Among them, the penultimate column reports the average cycle time (f_1) of the upper-level objective for regular operation scenario,

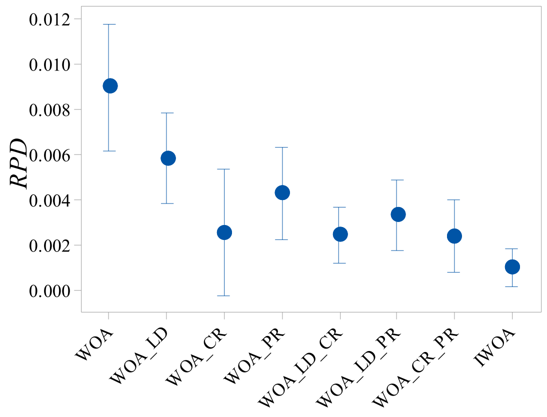


FIGURE 9. 95% Confidence interval for the average RPD of WOA, six variants, IWOA.

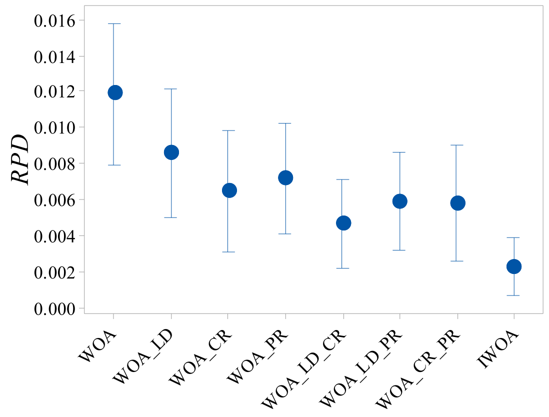


FIGURE 10. 95% Confidence interval for the minimum RPD of WOA, six variants, IWOA.

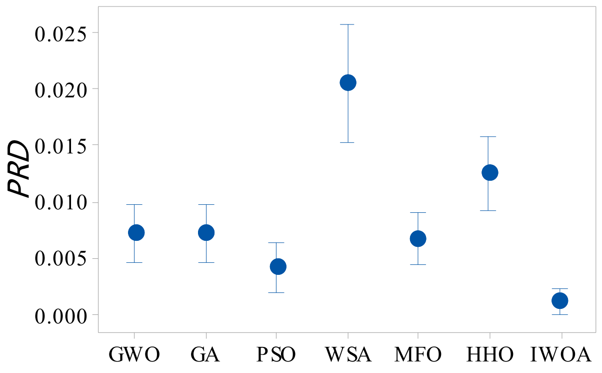


FIGURE 11. 95% Confidence interval for the average RPD of six existing algorithms, IWOA.

and the last one (f_2) is the average value of lower-level objective, $CT^*/CT^*_{\max} + TA/TA_{max}$. As shown in Fig. 8, the best parameter combination of the proposed IWOA is $\{PS = 70,\ T_r = 20,\ S_r = 30\%\}$.

Besides, the parameters of six variants of IWOA algorithm and other compared algorithms are also calibrated. Half of six variants comprise of single modification

TABLE 6. Comparisons of WOA, six variants, IWOA algorithm.

No	WOA		WOA_LD		WOA_CR		WOA_PR		WOA_LD_CR		WOA_LD_PR		WOA_CR_PR		IWOA	
	Avg	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0.002	0.001	0.002	0	0	0	0.001	0	0	0	0.001	0.001	0	0.001	0	0
12	0.001	0.001	0.001	0.001	0	0	0.001	0	0	0	0	0	0	0.001	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0.008	0	0	0
14	0.010	0.010	0.010	0.010	0.010	0	0	0.010	0.010	0.010	0	0.010	0	0	0	0
15	0.014	0	0.014	0.001	0	0	0.014	0	0	0	0.014	0	0	0	0.001	0
16	0.023	0.022	0.023	0.021	0.044	0.021	0.022	0.021	0.001	0.021	0	0	0.022	0.022	0	0.021
17	0.001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0.015	0.016	0.008	0	0.004	0	0.004	0	0.006	0	0.007	0	0.004	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0.010	0.012	0.006	0.006	0	0.002	0.002	0.006	0	0.002	0.002	0.006	0.002	0.002	0	0
22	0.008	0.011	0.006	0.011	0.003	0.003	0	0.003	0.006	0.003	0.003	0.006	0	0	0	0
23	0.008	0.016	0	0.012	0.004	0.023	0.008	0.019	0.007	0	0	0.004	0.004	0.019	0	0.002
24	0.010	0.031	0.010	0.046	0	0.035	0.005	0.010	0.005	0.025	0.005	0.030	0	0.025	0.010	0
25	0.005	0.014	0.005	0.014	0.005	0	0	0.010	0	0.010	0.005	0.005	0	0	0.005	0.005
26	0.014	0.022	0.007	0.015	0	0.007	0.007	0.007	0	0	0.014	0.007	0.007	0.007	0.007	0.007
27	0.019	0.019	0.009	0.010	0.001	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0.009	0	0
28	0.012	0.036	0	0.012	0	0.024	0	0.024	0	0.012	0	0.024	0	0.024	0	0
29	0.008	0.007	0.009	0.008	0	0.001	0.003	0.007	0	0.002	0.002	0	0.001	0.002	0	0.002
30	0.019	0.022	0.011	0.012	0	0.015	0.009	0.010	0.011	0.002	0.007	0.014	0.006	0.012	0	0
31	0.033	0.024	0.016	0.008	0.003	0	0.018	0.001	0.007	0	0.010	0	0	0.001	0	0
32	0.022	0.038	0.010	0.029	0.001	0.021	0.008	0.038	0.003	0	0.012	0.016	0.003	0.028	0	0.002
33	0.005	0.007	0.002	0.005	0.002	0.004	0.002	0.007	0	0.002	0	0.005	0	0	0	0.002
34	0.007	0.007	0.007	0.010	0	0.003	0.004	0	0.003	0	0.003	0.007	0.003	0	0	0
35	0.010	0.010	0.005	0.014	0	0.004	0	0.005	0	0.005	0	0.009	0	0	0	0.005
36	0.013	0.026	0.006	0.020	0.001	0.019	0.013	0.013	0.006	0.019	0.006	0.019	0	0.020	0.006	0
37	0.002	0.003	0.001	0.004	0	0.002	0	0	0.002	0.002	0	0	0.001	0.002	0.001	0.001
38	0.004	0.011	0.004	0.002	0	0.005	0.003	0.011	0	0.007	0	0	0.003	0.006	0	0.010
39	0.007	0.010	0.010	0	0	0.005	0.002	0.012	0.001	0.010	0.004	0.009	0	0	0	0.007
40	0.007	0.007	0.005	0.005	0.003	0	0.002	0.007	0.002	0.009	0	0.007	0.003	0.003	0	0.008

respectively and hence are named as WOA_LD, WOA_CR, and WOA_PR respectively; while each of another half includes two modifications simultaneously as denoted by WOA_LD_CR, WOA_LD_PR, and WOA_CR_PR. Meanwhile, six existing algorithms including grey wolf optimizer (GWO) [35], genetic algorithm (GA) [37], particle swarm optimization algorithm (PSO) [3], whale swarm algorithm (WSA) [38], moth flame optimization (MFO), harris hawks optimization (HHO) are introduced for comparison. The best parameter combinations of these algorithms are shown in TABLE 5.

D. PERFORMANCE ANALYSIS OF IWOA ALGORITHM

To test the effectiveness of three modifications and the resulted IWOA algorithm, comparison experiments are conducted with six variant algorithms and other six well-known algorithms.

1) EFFECTIVENESS OF THREE MODIFICATIONS

To verify the effectiveness of modifications in the proposed IWOA, the original WOA, and six variant algorithms are hired with each of them tested on 32 benchmark cases. For each case, each algorithm runs 10 times within the same CPU time limit of $N * N{*}10\ ms$ for fairness. The comparative results are reported in TABLE 6 and Figs (9-10) in which the least significant difference interval is the 95% confidence level. It can be observed that the value of *RPD* obtained by WOA_LD, WOA_CR, and WOA_PR are smaller than that by the original WOA in most cases, which demonstrates that each of three modifications is an effective improvement.

TABLE 7. Comparison result of GWO, GA, PSO, WSA, MFO, HHO, IWOA.

No	GWO		GA		PSO		WSA		MFO		HHO		IWOA	
	Avg	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*	*Avg*	*Min*
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0.001	0	0	0	0	0	0	0
11	0.001	0	0.001	0	0.001	0	0.003	0.002	0.002	0.001	0.002	0.001	0	0
12	0	0	0.001	0	0.001	0.001	0.002	0.002	0.001	0.001	0.002	0.001	0	0
13	0	0	0	0	0	0	0.017	0.008	0	0	0.008	0	0	0
14	0.010	0.010	0.010	0.010	0	0.001	0.011	0.011	0	0.001	0.010	0.010	0	0
15	0.014	0	0.014	0	0	0	0.028	0.015	0.014	0.001	0.014	0.001	0	0
16	0.023	0.023	0.001	0.023	0.001	0.022	0.046	0.023	0.024	0.023	0.023	0.023	0	0
17	0	0	0	0	0	0	0	0	0	0	0.001	0	0	0
18	0.017	0	0.020	0	0	0	0.016	0	0.006	0	0.019	0	0.012	0
19	0.004	0	0.012	0	0	0	0.027	0	0	0	0.008	0	0	0
20	0.001	0	0	0	0.002	0	0.003	0	0.002	0.004	0.003	0	0.002	0
21	0.002	0.002	0.002	0	0	0	0.018	0.010	0.004	0.002	0.010	0.004	0.002	0
22	0.006	0.009	0.006	0.006	0.003	0.006	0.011	0.020	0.008	0.014	0.008	0.009	0	0
23	0.011	0.027	0.008	0.016	0.004	0.008	0.015	0.027	0.008	0.023	0.015	0.027	0	0
24	0.005	0.005	0.015	0.030	0.010	0.015	0.035	0.041	0.010	0.026	0.020	0.015	0	0
25	0	0	0	0.010	0.005	0.010	0.010	0.019	0	0.010	0.005	0.019	0	0.005
26	0.007	0	0.007	0	0	0	0.028	0.015	0.007	0.001	0.014	0.014	0.007	0
27	0.010	0.010	0.009	0.019	0	0	0.028	0.029	0	0.010	0.019	0.019	0	0
28	0.012	0	0	0	0	0.012	0.035	0.048	0.012	0.012	0.012	0.024	0.011	0
29	0.003	0.008	0.003	0.006	0.003	0.006	0.019	0.015	0.004	0.009	0.010	0.006	0	0
30	0.002	0	0.005	0.001	0.005	0.011	0.028	0.033	0.011	0.017	0.013	0.009	0	0.004
31	0.027	0.028	0.017	0.018	0.023	0.023	0.051	0.057	0.019	0.016	0.031	0.029	0	0
32	0.019	0.025	0.019	0.026	0.017	0.027	0.055	0.061	0.018	0.032	0.039	0.032	0	0
33	0.002	0.002	0.005	0.005	0.005	0.002	0.012	0.010	0.005	0.005	0.007	0.005	0	0
34	0.003	0.007	0.003	0.004	0.003	0.003	0.016	0.016	0.003	0.004	0.010	0.010	0	0
35	0.005	0	0.005	0.001	0	0.001	0.019	0.025	0.010	0.010	0.015	0.015	0	0
36	0.007	0.013	0.013	0.013	0.007	0.007	0.020	0.026	0.007	0.007	0.013	0.019	0	0
37	0.002	0.005	0.004	0.005	0.002	0.003	0.011	0.011	0.002	0.005	0.006	0.006	0	0
38	0.008	0.009	0.008	0.009	0.008	0.008	0.017	0.019	0.006	0.007	0.011	0.010	0	0
39	0.011	0.011	0.018	0.021	0.013	0.008	0.032	0.027	0.012	0.012	0.022	0.025	0	0
40	0.017	0.001	0.023	0.022	0.021	0.003	0.038	0.035	0.018	0.019	0.026	0.023	0	0

Meanwhile, when two modifications are combined, WOA_LD_CR, WOA_LD_PR, and WOA_CR_PR obtain an even lower *RPD* compared with the single modification. And the proposed IWOA combining three modifications has the best performance, which indicates the effectiveness of the proposed improvements.

2) COMPARISON WITH EXISTING ALGORITHMS

To further test the performance of the algorithms, the IWOA is compared with six existing algorithms including GWO, GA, PSO, WSA, MFO, HHO on 32 benchmark cases. The average and minimum *RPD* of these algorithms under the CPU time limit are reported in TABLE 7 and Figs (11-12). It can be observed that the proposed IWOA algorithm outperforms other six well-known algorithms. Specifically, when compared with WSA, IWOA obtains 30 better results in terms of average *RPD* and 26 better ones in terms of minimum *RPD*, implying that IWOA is superior to WSA in almost all cases. When compared with GWO, GA, PSO, MFO, HHO, the proposed IWOA yields a total of 40, 41, and 38, 47, 55 better results respectively, demonstrating the significant superiority of IWOA. The above conclusions can also be proved by Figs (11-12) from a statistical viewpoint.

It can be concluded that with three effective modifications, the proposed IWOA achieves the optimal solutions for all

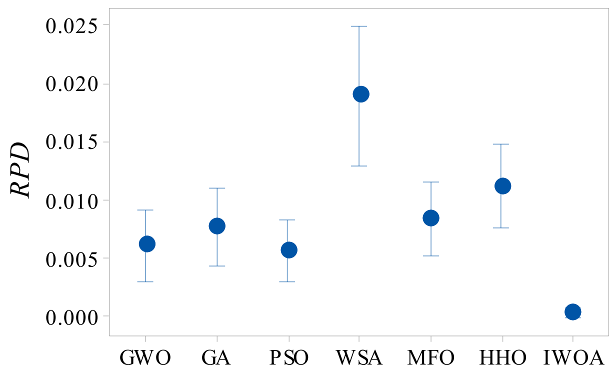


FIGURE 12. 95% Confidence interval for the minimum RPD of six existing algorithms, IWOA.

small-scaled cases within shorter computational time. More importantly, it outperforms the original WOA, six variant algorithms, and six well-known algorithms.

VI. CONCLUSION AND FUTURE RESEARCH

Since the planned preventive maintenance in workstations can be known, considering PM scenarios into the assembly line balancing problem is sufficient and necessary for fully exploiting the production capacity of all available workstations. Hence, this paper first proposes a model for the type-II assembly line balancing problem considering PM scenarios, and then reformulates it as a lexicographical optimization problem to highlight the importance of regular operation scenario and reveal the associated relationships between PM scenarios and regular operation scenario. Further, an improved whale optimization algorithm with three modifications is proposed to tackle this problem, which includes exploitation via three search agents, enhanced exploration via crossover operator and partial regeneration. More than 5000 experiments lead to the following conclusions:

(1) The two-level lexicographical optimization approach guarantees productivity under regular operation scenario. More importantly, small task alterations help to bypass the workstation under maintenance and increase production efficiency by 1%.

(2) The joint of three modifications endows IWOA significant superiority over six variants and other six well-known algorithms.

In the future, the integration of mixed-model assembly line balancing and preventive maintenance will be studied for practical purposes. And, further research will be extended to the integration of assembly scheduling with period-based or condition-based preventive maintenance and even with corrective maintenance.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their helpful comments and constructive suggestions.

REFERENCES

[1] N. Boysen, M. Fliedner, and A. Scholl, "A classification of assembly line balancing problems," *Eur. J. Oper. Res.*, vol. 183, no. 2, pp. 674–693, Dec. 2007, doi: 10.1016/j.ejor.2006.10.010.

[2] N. Manavizadeh, N.-S. Hosseini, M. Rabbani, and F. Jolai, "A simulated annealing algorithm for a mixed model assembly u-line balancing type-I problem considering human efficiency and just-in-time approach," *Comput. Ind. Eng.*, vol. 64, no. 2, pp. 669–685, Feb. 2013, doi: 10.1016/j.cie.2012.11.010.

[3] Q. Tang, Z. Li, and L. Zhang, "An effective discrete artificial bee colony algorithm with idle time reduction techniques for two-sided assembly line balancing problem of type-II," *Comput. Ind. Eng.*, vol. 97, pp. 146–156, Jul. 2016, doi: 10.1016/j.cie.2016.05.004.

[4] Q. Zheng, M. Li, Y. Li, and Q. Tang, "Station ant colony optimization for the type 2 assembly line balancing problem," *Int. J. Adv. Manuf. Technol.*, vol. 66, nos. 9–12, pp. 1859–1870, Jun. 2013, doi: 10.1007/s00170-012-4465-9.

[5] Z. A. Çil, S. Mete, E. Özceylan, and K. Ağpak, "A beam search approach for solving type II robotic parallel assembly line balancing problem," *Appl. Soft Comput.*., vol. 61, pp. 129–138, Dec. 2017, doi: 10.1016/j.asoc.2017.07.062.

[6] R. Esmaeilbeigi, B. Naderi, and P. Charkhgard, "The type E simple assembly line balancing problem: A mixed integer linear programming formulation," *Comput. Oper. Res.*, vol. 64, pp. 168–177, Dec. 2015, doi: 10.1016/j.cor.2015.05.017.

[7] Y. Keun Kim, Y. Ju Kim, and Y. Kim, "Genetic algorithms for assembly line balancing with various objectives," *Comput. Ind. Eng.*, vol. 30, no. 3, pp. 397–409, Jul. 1996, doi: 10.1016/0360-8352(96)00009-5.

[8] W. Cui, Z. Lu, C. Li, and X. Han, "A proactive approach to solve integrated production scheduling and maintenance planning problem in flow shops," *Comput. Ind. Eng.*, vol. 115, pp. 342–353, Jan. 2018, doi: 10.1016/j.cie.2017.11.020.

[9] Z. Wu, S. Sun, and S. Xiao, "Risk measure of job shop scheduling with random machine breakdowns," *Comput. Oper. Res.*, vol. 99, pp. 1–12, Nov. 2018, doi: 10.1016/j.cor.2018.05.022.

[10] A. J. Yu and J. Seif, "Minimizing tardiness and maintenance costs in flow shop scheduling by a lower-bound-based GA," *Comput. Ind. Eng.*, vol. 97, pp. 26–40, Jul. 2016, doi: 10.1016/j.cie.2016.03.024.

[11] H. Feng, C. Tan, T. Xia, E. Pan, and L. Xi, "Joint optimization of preventive maintenance and flexible flowshop sequence-dependent group scheduling considering multiple setups," *Eng. Optim.*, vol. 51, no. 9, pp. 1529–1546, Sep. 2019, doi: 10.1080/0305215x.2018.1540696.

[12] E. Moradi, S. Fatemi Ghomi, and M. Zandieh, "Bi-objective optimization research on integrated fixed time interval preventive maintenance and production for scheduling flexible job-shop problem," *Expert Syst. Appl.*, vol. 38, no. 6, pp. 7169–7178, Jun. 2011, doi: 10.1016/j.eswa.2010.12.043.

[13] W. Qin, Z. Zhuang, Y. Liu, and O. Tang, "A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly," *Comput. Ind. Eng.*, vol. 138, Dec. 2019, Art. no. 106115, doi: 10.1016/j.cie.2019.106115.

[14] F. M. Defersha and F. Mohebalizadehgashti, "Simultaneous balancing, sequencing, and workstation planning for a mixed model manual assembly line using hybrid genetic algorithm," *Comput. Ind. Eng.*, vol. 119, pp. 370–387, May 2018, doi: 10.1016/j.cie.2018.04.014.

[15] H. Babazadeh, M. Alavidoost, M. Fazel Zarandi, and S. Sayyari, "An enhanced NSGA-II algorithm for fuzzy bi-objective assembly line balancing problems," *Comput. Ind. Eng.*, vol. 123, pp. 189–208, Sep. 2018, doi: 10.1016/j.cie.2018.06.014.

[16] Z. Zhang, K. Wang, L. Zhu, and Y. Wang, "A Pareto improved artificial fish swarm algorithm for solving a multi-objective fuzzy disassembly line balancing problem," *Expert Syst. Appl.*, vol. 86, pp. 165–176, Nov. 2017, doi: 10.1016/j.eswa.2017.05.053.

[17] Y. Rasekhipour, I. Fadakar, and A. Khajepour, "Autonomous driving motion planning with obstacles prioritization using lexicographic optimization," *Control Eng. Pract.*, vol. 77, pp. 235–246, Aug. 2018, doi: 10.1016/j.conengprac.2018.04.014.

[18] M. Cococcioni, M. Pappalardo, and Y. D. Sergeyev, "Lexicographic multi-objective linear programming using grossone methodology: Theory and algorithm," *Appl. Math. Comput.*, vol. 318, pp. 298–311, Feb. 2018, doi: 10.1016/j.amc.2017.05.058.

[19] M. VilàÂă and J. Pereira, "A branch-and-bound algorithm for assembly line worker assignment and balancing problems," *Comput. Oper. Res.*, vol. 44, pp. 105–114, Apr. 2014, doi: 10.1016/j.cor.2013.10.016.

[20] L. Borba and M. Ritt, "A heuristic and a branch-and-bound algorithm for the assembly line worker assignment and balancing problem," *Comput. Oper. Res.*, vol. 45, pp. 87–96, May 2014, doi: 10.1016/j.cor.2013.12.002.

[21] Z. Li, I. Kucukkoc, and Z. Zhang, "Branch, bound and remember algorithm for U-shaped assembly line balancing problem," *Comput. Ind. Eng.*, vol. 124, pp. 24–35, Oct. 2018, doi: 10.1016/j.cie.2018.06.037.

[22] M. Kayar and M. Akalin, "Comparing heuristic and simulation methods applied to the apparel assembly line balancing problem," *Fibes Text East Eur.*, vol. 24, no. 2, pp. 131–138, Jan. 2016, doi: 10.5604/12303666.1191438.

[23] P. Sadeghi, R. D. Rebelo, and J. S. Ferreira, "Balancing mixed-model assembly systems in the footwear industry with a variable neighbourhood descent method," *Comput. Ind. Eng.*, vol. 121, pp. 161–176, Jul. 2018, doi: 10.1016/j.cie.2018.05.020.

[24] S. Akpınar and G. M. Bayhan, "A hybrid genetic algorithm for mixed model assembly line balancing problem with parallel workstations and zoning constraints," *Eng. Appl. Artif. Intell.*, vol. 24, no. 3, pp. 449–457, Apr. 2011, doi: 10.1016/j.engappai.2010.08.006.

[25] H. Yapici and N. Cetinkaya, "A new meta-heuristic optimizer: Pathfinder algorithm," *Appl. Soft Comput.*., vol. 78, pp. 545–568, May 2019, doi: 10.1016/j.asoc.2019.03.012.

[26] Z. Zhang, Q. Tang, and L. Zhang, "Mathematical model and grey wolf optimization for low-carbon and low-noise U-shaped robotic assembly line balancing problem," *J. Cleaner Prod.*, vol. 215, pp. 744–756, Apr. 2019, doi: 10.1016/j.jclepro.2019.01.030.

[27] Y. Zhong, Z. Deng, and K. Xu, "An effective artificial fish swarm optimization algorithm for two-sided assembly line balancing problems," *Comput. Ind. Eng.*, vol. 138, Dec. 2019, Art. no. 106121, doi: 10.1016/j.cie.2019.106121.

[28] B. Zhang, L. Xu, and J. Zhang, "A multi-objective cellular genetic algorithm for energy-oriented balancing and sequencing problem of mixed-model assembly line," *J. Cleaner Prod.*, vol. 244, Jan. 2020, Art. no. 118845, doi: 10.1016/j.jclepro.2019.118845.

[29] M. Şahin and T. Kellegöz, "A new mixed-integer linear programming formulation and particle swarm optimization based hybrid heuristic for the problem of resource investment and balancing of the assembly line with multi-manned workstations," *Comput. Ind. Eng.*, vol. 133, pp. 107–120, Jul. 2019, doi: 10.1016/j.cie.2019.04.056.

[30] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, Feb. 2016, doi: 10.1016/j.advengsoft.2016.01.008.

[31] M. Petrović, Z. Miljković, and A. Jokić, "A novel methodology for optimal single mobile robot scheduling using whale optimization algorithm," *Appl. Soft Comput.*., vol. 81, Aug. 2019, Art. no. 105520, doi: 10.1016/j.asoc.2019.105520.

[32] C. Lang and H. Jia, "Kapur's entropy for color image segmentation based on a hybrid whale optimization algorithm," *Entropy*, vol. 21, no. 3, p. 318, Mar. 2019, doi: 10.3390/e21030318.

[33] H. Chen, Y. Xu, M. Wang, and X. Zhao, "A balanced whale optimization algorithm for constrained engineering design problems," *Appl. Math. Model.*, vol. 71, pp. 45–59, Jul. 2019, doi: 10.1016/j.apm.2019.02.004.

[34] H. H. Miyata, M. S. Nagano, and J. N. Gupta, "Integrating preventive maintenance activities to the no-wait flow shop scheduling problem with dependent-sequence setup times and makespan minimization," *Comput. Ind. Eng.*, vol. 135, pp. 79–104, Sep. 2019, doi: 10.1016/j.cie.2019.05.034.

[35] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Jan. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[36] Z. Zhang, Q. Tang, D. Han, and Z. Li, "Enhanced migrating birds optimization algorithm for U-shaped assembly line balancing problems with workers assignment," *Neural Comput. Appl.*, vol. 31, no. 11, pp. 7501–7515, Nov. 2019, doi: 10.1007/s00521-018-3596-9.

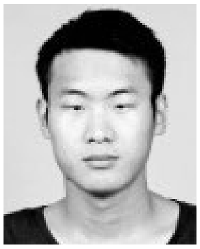
[37] Y. K. Kim, Y. Kim, and Y. J. Kim, "Two-sided assembly line balancing: A genetic algorithm approach," *Prod. Planning Control*, vol. 11, no. 1, pp. 44–53, Jan. 2000, doi: 10.1080/095372800232478.

[38] H. Zhong, C. Hu, X. Li, L. Gao, B. Zeng, and H. Dong, "Kinematic calibration method for a two-segment hydraulic leg based on an improved whale swarm algorithm," *Robot. Comput.-Integr. Manuf.*, vol. 59, pp. 361–372, Oct. 2019, doi: 10.1016/j.rcim.2019.05.002.

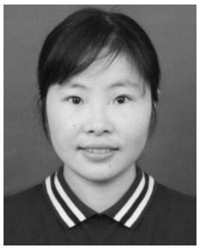
KAI MENG was born in Shanxi, China, in 1996. He received the B.S. degree in industrial engineering from the Wuhan University of Science and Technology, in 2018. He is currently pursuing the Ph.D. degree in mechanical engineering. His research interest includes intelligent optimization algorithms within shop scheduling problems.

QIUHUA TANG was born in Lichuan, Hubei, China, in 1970. She received the B.S. degree in process and equipment of machinery manufacturing from Northeastern University, in 1992, the master's degree in mechanical design and theory from the Wuhan University of Science and Technology, and the Ph.D. degree from the Wuhan University of Technology.

Since 1992, she has been working with the Wuhan University of Science and Technology, and was promoted as a Professor, in 2009. She has undertaken three general research projects supported by the National Natural Science Foundation of China (NSFC), and won the 2016 honorable mention in mechanical engineering discipline of NSFC, in December 2017. Her research interests include production process planning and scheduling, manufacturing process monitoring and control, and modern optimization methods and algorithms. She has published more than 100 articles in academic journals and conferences. She received awards like the Wuhan Excellent Paper Award, excellent paper of Chinese Mechanical Engineering Society.



ZIKAI ZHANG was born in Xiangyang, Hubei, China, in 1994. He received the B.S. degree in industrial engineering from the Wuhan University of Science and Technology, in 2016, and the master's degree. He is currently pursuing the Ph.D. degree in mechanical engineering with the Wuhan University of Science and Technology. His research interests include production process control and intelligent algorithms.



XINBO QIAN was born in Nantong, Jiangsu, China, in 1986. She received the B.S. degree in thermal energy and power engineering from Hohai University, in 2008, and the Ph.D. degree in hydraulic and hydropower engineering from the Huazhong University of Science and Technology, in 2014.

From 2012 to 2013, she was a Visiting Ph.D. Student with the University of Toronto, Canada, where she had a research work with the Center for maintenance optimization and reliability engineering (CMORE). From 2014 to 2018, she was a Lecturer with the Department of Mechatronic Engineering, Wuhan University of Science and Technology (WUST). Since 2018, she has been an Associate Professor with WUST. She has undertaken one research project supported by the National Natural Science Foundation of China (NSFC), and is participating in three other NSFC projects. She has published more than ten articles and one book. She is also participating in the development and revision of national standards about reliability testing and estimation for hydraulic components. Her research interests include prognostics and health management, maintenance optimization, and reliability testing for mechanical system and components.