Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2016, Article ID 4536426, 15 pages
http://dx.doi.org/10.1155/2016/4536426

Research Article

Minimizing the Cycle Time in Two-Sided Assembly Lines with Assignment Restrictions: Improvements and a Simple Algorithm

Zixiang Li, Qiuhua Tang, and Liping Zhang

Industrial Engineering Department, Wuhan University of Science and Technology, Wuhan, Hubei 430081, China

Correspondence should be addressed to Qiuhua Tang; tangqiuhua@wust.edu.cn

Received 10 December 2015; Accepted 23 May 2016

Academic Editor: Anna M. Gil-Lafuente

The two-sided assembly line balancing problem type-II (TALBP-II) is of major importance for the reconfiguration of the two-sided assembly lines which are widely utilized to assemble large-size high-volume products. The TALBP-II is NP-hard, and some assignment restrictions in real applications make this problem much more complex. This paper provides an integer programming model for solving the TALBP-II with assignment restrictions optimally and utilizes a simple and effective iterated greedy (IG) algorithm to address large-size problems. This algorithm utilizes a new local search by considering precedence relationships between tasks in order to reduce the computational time. In particular, a priority-based decoding scheme is developed to handle these assignment restrictions and reduce sequence-dependent idle times by adjusting the priority values. Experimental comparison among the proposed decoding scheme and other published ones demonstrates the efficiency of the priority-based decoding. A comprehensive computational comparison among the IG algorithm and other eight recent algorithms proves effectiveness of the proposed IG algorithm.

1. Introduction

A two-sided assembly line consists of a set of sequential mated-stations connected by the material handling system. It is widely used to produce large-size high-volume products, such as cars, trucks, and automobiles. An example of two-sided assembly lines is shown in Figure 1. Both sides are utilized for performing a set of tasks. A pair of face-to-face stations like station $(n, 1)$ and station $(n, 2)$ is called a mated-station and one of them is the companion of the other. The idle time on each station can be divided into two types: sequence-dependent idle time such as idle time behind task 1 [1] and the remaining idle time existing at the rear of the last task such as idle time behind 3.

Compared with the traditional one-sided assembly line, the two-sided assembly line balancing problem (TALBP) is more complex due to the existence of restrictions on the operation direction. Some tasks have to be operated on a predefined side of the line, while others can be allocated to any side of the line. Thus, the task preferred directions can be classified into three types, namely, L (left), R (right), and E (either). In summary, there are three restrictions that need to be satisfied, including the precedence restriction, cycle time restriction, and direction restriction.

Besides these basic restrictions, other assignment restrictions which exist in real applications need to be considered, including zoning restriction, synchronous restriction, positional restriction, distance restriction, and resource restriction [2]. In this paper, we focus on zoning restriction, synchronous restriction, and position restriction which are usually found in real applications. The zoning restriction can be divided into two types: positive zoning restriction and negative zoning restrictions. Positive zoning restriction means that some tasks need to be assigned to the same station while negative zoning restriction prohibits some tasks being allocated to the same mated-station [3]. Synchronous restriction restricts two operators to perform a pair of tasks simultaneously on both sides of the same mated-station for collaboration [4]. The positional restriction indicates that certain tasks need to be allocated on predetermined stations or mated-stations [1].

The TALBP, so far as the objectives are concerned, can be divided into two versions: TALBP-I that minimizes the number of stations with a given cycle time, and TALBP-II

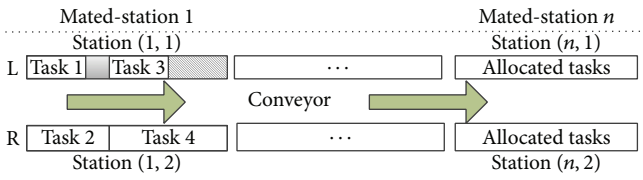

Figure 1: A typical example of two-sided assembly line.

that minimizes the cycle time for given mated-stations. The TALBP-I is more appropriate for the first-time installation of the assembly line, while the TALBP-II is proper for the reconfiguration of the assembly line [5]. Note that the TALBP-II is more complex than the TALBP-I, since the cycle time is unknown and needs to be optimized.

To our knowledge, limited papers considered the TALBP-II [2, 6–8] even though the TALBP-II is of major importance in industry [5, 7]. In addition, no paper introduces decoding schemes for the TALBP-II with assignment restrictions. To enhance research on the TALBP-II with assignment restrictions, this paper proposes a simple and effective iterated greedy (IG) algorithm. In particular, this algorithm proposes a new priority-based decoding in which the assignment restrictions are satisfied by adjusting the priority values. The IG algorithm is a local search method, and it has obtained excellent results for some discrete optimization problems [9–13]. This research is the first attempt to adopt this IG algorithm for the TALBP-II with assignment restrictions.

The organization of this paper is introduced as follows. Section 2 presents the recent method for TALBP-II and assignment restrictions. The mathematical model for TALBP-II with assignment restrictions is described in Section 3. The decoding scheme for TALBP-II with assignment restrictions is explained in Section 4. The improved IG algorithm is introduced in Section 5. Section 6 demonstrates the effectiveness of IG by comparison with other well-known methods. Finally, the conclusions are given in the last section.

2. Literature Review

The two-sided assembly line balancing has become an active field of research since the two-sided assembly line balancing problem (TALBP) was first introduced by Bartholdi [14]. The TALBP can be divided into TALBP-I that minimizes the number of stations and TALBP-II that minimizes the cycle time.

Most attention is paid to the TALBP-I, and exact, heuristic, and meta-heuristic methods have been applied. Exact methods include station-oriented enumerative algorithm [15] and branch-and-bound algorithms [16, 17]. Heuristic methods include the group assignment procedure [18] and the enhanced priority-based heuristic [19]. Additional research was based on meta-heuristic methods, and they are a genetic algorithm [1], a tabu search algorithm [20], a bee algorithm [21], simulated annealing algorithms [22–24], particle swarm optimization algorithms [25, 26], and a hybrid honey bee mating optimization [27] and multineighborhood based path relinking [28].

As for the TALBP-II, Kim et al. [6] adopted a mathematical model and utilized a genetic algorithm. Purnomo et al. [2] also proposed a genetic algorithm to handle the TALBP-II with assignment restrictions. Purnomo and Wee [7] proposed a harmony search algorithm to address multiobjective TALBP-II. More recently, Lei and Guo [8] proposed a variable neighborhood search and they developed a new method to determine the allocated sides of the tasks by utilizing a side string. Tang et al. [29] developed a discrete artificial bee colony algorithm and they also provided some methods to reduce the idle times on two-sided assembly lines. Except for Purnomo et al. [2], all the other papers ignore the assignment restrictions in real application. And Purnomo et al. [2] did not exhibit the detailed decoding procedure, but the detailed decoding procedure is very important for the replication of the method.

Now we focus on additional contributions dealing with assignment restrictions. Apart from the above literature of Purnomo et al. [2], all the researches belong to TALBP-I. Kim et al. [1] focused on positional restrictions. Baykasoglu and Dereli [3], Özbakir and Tapkan [21], and Purnomo and Wee [7] considered zoning restrictions. Simaria and Vilarinho [4] considered the zoning and synchronism restrictions. Tapkan et al. [30, 31] handled the positional restrictions, zoning restrictions, and synchronism restrictions. Yuan et al. [32], Wang et al. [33], Li et al. [34], and Tang et al. [35] also employed different methods to handle the above three restrictions. Tuncel and Aydin [36] considered additional assignment restrictions as an industrial assembly system in practice. Among these researches, few papers presented their decoding schemes except for a similar decoding scheme by four papers [32–35]. Yet this decoding method ignores the reduction of sequence-dependent idle times in the decoding procedure and the methods to deal with assignment restrictions can be possibly improved.

From the literature review, it is observed that only one paper deals with TALBP-II with assignment restrictions and no detailed encoding procedure for TALBP-II with assignment restrictions is presented. Therefore this paper focuses on the TALBP-II with assignment restrictions by utilizing a new algorithm and also provides new decoding schemes to handle assignment restrictions and reduce sequence-dependent idles by adjusting the priority values.

3. Mathematical Model for TALBP-II with Assignment Restrictions

3.1. Problem Assumptions

(1) A single model is taken into account and the travel time between stations is ignored.

(2) Precedence diagrams are known and deterministic.

(3) The task with a positional restriction should be assigned to the predefined station.

(4) The tasks in the positive zoning restriction should be assigned to the same station.

(5) The tasks in the negative zoning restrictions are prohibited to be allocated to the same mated-station.

3.2. Mathematical Model. In the TALBP-II, minimizing the cycle time is the widely used objective [6] and we employ it in this paper. Based on the research by Kim et al. [6] and Purnomo et al. [2], the other constraints are listed as follows:

$$\sum_{j\in J}\sum_{k\in K(i)} x_{ijk} = 1 \quad \forall i \in I, \tag{1}$$

$$\sum_{g\in J}\sum_{k\in K(h)} g\cdot x_{hgk} \le \sum_{j\in J}\sum_{k\in K(i)} j\cdot x_{ijk} \quad \forall i \in I - P_0,\ h \in P(i), \tag{2}$$

$$t_i^f - t_h^f + \psi\left(1 - \sum_{k\in K(h)} x_{hjk}\right) + \psi\left(1 - \sum_{k\in K(i)} x_{ijk}\right) \ge t_h \quad \forall i \in I - P_0,\ h \in P(i),\ j \in J, \tag{3}$$

$$t_p^f - t_i^f + \psi\left(1 - x_{ijk}\right) + \psi\left(1 - x_{pjk}\right) + \psi\left(1 - z_{ip}\right) \ge t_p \tag{4}$$

for all $i \in I,\ p \in \{r \mid r \in I - (P_a(i) \cup S_a(i) \cup C(i)),\ i < r\},\ j \in J,\ k \in K(i) \cap K(p)$,

$$t_i^f - t_p^f + \psi\left(1 - x_{ijk}\right) + \psi\left(1 - x_{pjk}\right) + \psi \cdot z_{ip} \ge t_i \tag{5}$$

for all $i \in I,\ p \in \{r \mid r \in I - (P_a(i) \cup S_a(i) \cup C(i)),\ i < r\},\ j \in J,\ k \in K(i) \cap K(p)$,

$$t_i^f \le \mathrm{CT} \quad \forall i \in I, \tag{6}$$

$$x_{ijk} = 1 \quad \forall (i,(j,k)) \in \mathrm{PC},\ k \in K(i), \tag{7}$$

$$x_{ijk} - x_{hjk} = 0 \quad \forall (i,h) \in \mathrm{PZ},\ k \in K(i) \cap K(h), \tag{8}$$

$$\sum_{k\in K(i)} x_{ijk} + \sum_{k\in K(h)} x_{hjk} \le 1 \quad \forall (i,h) \in \mathrm{NZ}, \tag{9}$$

$$x_{ijf} - x_{hjk} = 0 \quad \forall (i,h) \in \mathrm{SC},\ k \in K(h),\ f \in K(i),\ k \ne f, \tag{10}$$

$$t_i^f - t_i = t_h^f - t_h \quad \forall (i,h) \in \mathrm{SC}. \tag{11}$$

Constraint (1) is the occurrence restriction which guarantees that each task must be allocated to only one side of a mated-station. Constraint (2) is the precedence restriction which is usually found in the assembly line balancing problem. Constraints (3)–(5) consider the sequence-dependent finishing time of tasks. For a pair of tasks (i,h), if they are allocated to the same mated-station and task i is an immediate successor of task h, then constraint (3) becomes active and it is reduced to $t_i^f - t_h^f \ge t_h$. For a pair of tasks (i,h), if they are allocated to the same station and have no precedence relationships, then constraints (4)-(5) become active. If task i is allocated earlier than task p, then $z_{ip} = 1$ and constraint (4) is reduced to $t_p^f - t_i^f \ge t_p$. Otherwise, $z_{ip} = 0$ and constraint (5) is reduced to $t_i^f - t_p^f \ge t_i$. Constraint (6) is the cycle time restriction which ensures that each task is finished within cycle time. Constraint (7) is the positional restriction which makes sure that task i is assigned to station (j,k). Constraint (8) is the positive zoning restriction which ensures that tasks i, h are operated on the same station. Constraint (9) is the negative zoning restriction which forbids tasks i, h being assigned to the same mated-station. Constraints (10)-(11) are the synchronous restrictions. Constraint (10) ensures that task i and task h have the same starting time and constraint (11) ensures that task i and task h are allocated to the two sides of a mated-station.

3.3. New Objective Function for Evolution. The general objective of TALBP-II is minimizing the cycle time [6], which also means minimizing the total idle times, including the sequence-dependent idle time and the remaining idle time existing at the rear of the last task. Nevertheless, in experiments, a large number of gained solutions by the meta-heuristics have the same cycle time and the meta-heuristics cannot determine the best one among these solutions, causing their poor performance. Thus, this paper proposes the following objective based on the fitness function in Özbakir and Tapkan [21] to distinguish the solutions:

$$\begin{aligned}\mathrm{Fit} = {} & w_1 \sum_{j\in J}\sum_{k\in K(i)} \frac{\left(\mathrm{CT} - \mathrm{SF}_{jk}\right)}{2\mathrm{nm}} \\ & + w_2 \sum_{j\in J}\sum_{k\in K(i)} \frac{\left(\mathrm{SF}_{jk} - \mathrm{ST}_{jk}\right)}{2\mathrm{nm}} \\ & + w_3 \sqrt{\sum_{j\in J}\sum_{k\in K(i)} \frac{\left(\mathrm{CT} - \mathrm{SF}_{jk}\right)^2}{2\mathrm{nm}}} \\ & + w_4 \sqrt{\sum_{j\in J}\sum_{k\in K(i)} \frac{\left(\mathrm{SF}_{jk} - \mathrm{ST}_{jk}\right)^2}{2\mathrm{nm}}},\end{aligned} \tag{12}$$

where SF_{jk} and ST_{jk} are the completion time of the last task and the sum of task times on station (j,k), respectively, and w_1, w_2, w_3, w_4 are coefficients which need to be determined. The first part in fitness function minimizes the remaining idle time, the second part aims at minimizing the sequence-dependent idle time, and the third and fourth parts balance the remained idle time and the sequence-dependent idle time on each station.

4. Encoding and Decoding Scheme for TALBP-II with Assignment Restrictions

In the TALBP-II, the number of mated-stations is fixed and the station-oriented encoding and decoding are widely utilized [1, 2]. Due to the utilization of two sides of a mated-station, both the sequence of allocating tasks and the side allocating a task need be determined to obtain a feasible solution. Kim et al. [6] utilize a heuristic method to allocate tasks and this method ignores the sequence-dependent idle times in the decoding process, which may result in an amount of idle time and a large cycle time. The priority-based encoding and decoding [22], on the contrary, allocate the tasks based on the priority values of the tasks and the sequence-dependence idle times can be reduced by the adjustment of the priority values which can be achieved by the meta-heuristics. In this paper, we develop a new priority-based encoding and decoding based on the work of Khorasanian et al. [22].

The proposed priority-based decoding first selects the side of the current mated-station and then selects a task based on the priority values. Also, the task assignment rule is embedded into the decoding schemes to balance the workload and reduce sequence-dependent idle times. In contrast to the work of Yuan et al. [32] and Li et al. [34], new methods that deal with positional restriction and positive zoning restriction are proposed. All these task assignment rules and the decoding schemes are introduced as follows.

4.1. Initial Priority Values Adjustment. In the proposed priority-based decoding, all the initial priority values are limited to a range of $[0, 1]$. Taking the well-known RPW heuristic [37] as example, the initial task priority values are equal to the sum of processing times of a task and its successors. We first sort all the tasks in a decreasing order of the priority values to obtain a task permutation and then calculate priority values PL$[i]$ for tasks as follows:

$$\mathrm{PL}[i] = 1 - \frac{(2\mathrm{TS}_i - 1)}{2\mathrm{nt}} \quad \mathrm{TS}_i \in \{1, 2, \ldots, \mathrm{nt}\}, \tag{13}$$

where TS_i is the sequence of task i in the task permutation and nt is the number of tasks. This equation makes the former tasks in the task permutation have higher priority values. Since the task permutation is obtained in decreasing order of the initial priority values, the task with the largest initial priority values also has the largest priority value after priority values adjustment. Since some metaheuristic algorithms update the task permutations in the search process, such as simulated annealing [22], (13) can also transfer the task permutation into priority values.

4.2. Task Assignment Rule. The task assignment rule is applied to reduce the sequence-dependent idle time and reduce the workload deviation. First, we seek to reduce the sequence-dependent idle time by boosting the priorities of tasks without generating sequence-dependent idle times. Then we try to balance the workload on each station by boosting the priority values of tasks whose finishing times on the station are within a range [Cm, CT], where Cm is average workload which can be calculated with $\mathrm{Cm} = \sum_{i \in I} t_i / 2\mathrm{nm}$. In this way, we try to make the workload on each station vary within [Cm, CT] and the deviation is thus reduced.

Step 1. Calculate priority values for all the tasks.

Step 2. Set PL$[i] \leftarrow$ PL$[i]$ + 1 when task i satisfies cycle time, direction, and precedence restrictions while $j \neq$ nm or satisfies only direction and precedence restrictions while $j =$ nm.

Step 3. If PL$[i] > 1$ and task i in the assignable task set can begin at the earliest start time of the current station, then PL$[i] \leftarrow$ PL$[i]$ + 1.

Step 4. If PL$[i] > 1$ and the finishing time of task i in the assignable task set is in [Cm, CT], then PL$[i] \leftarrow$ PL$[i]$ + 1.

Step 5. After Step 4, there are three conditions: (1) task i cannot be allocated when $0 <$ PL$[i] < 1$; (2) task i can be allocated and its priority is not boosted when $1 <$ PL$[i] < 2$; (3) task i can be allocated and its priority is boosted when PL$[i] > 2$.

In Step 2, only direction and precedence restrictions need to be satisfied while $j =$ nm, which make sure that we can acquire a feasible solution. After Step 2, task i can be assigned only when PL$[i] > 1$ and task i violates at least one of the above restrictions when PL$[i] < 1$. In Step 4, the tasks whose finishing times are within [Cm, CT] have higher priorities for the sake of undertaking some sequence-dependent idle time and holding more workload.

4.3. Improved Positional Restriction Handling. Yuan et al. [32] and Li et al. [34] ignore the positional restriction during the decoding process. This paper improves their decoding by preventing the tasks in the positional restriction being assigned to the former mated-station. In addition, if the current station is the predetermined station (j^*, k^*), the priority of the task is increased. To achieve the above goal, we define the matrix TS$[i][j][k]$ in (14) and the remaining values are set to zero. Then, the propriety of task i is updated with expression (15):

$$\begin{aligned} &\mathrm{TS}[i][j][k] = -\psi, \quad \forall j < j^*,\ k \in K(i); \\ &\mathrm{TS}[i][j][k] = 1, \quad \forall j = j^*,\ k = k^*, \end{aligned} \tag{14}$$

$$\mathrm{PL}[i] = \mathrm{PL}[i] + \mathrm{TS}[i][j][k]. \tag{15}$$

After adjusting the priorities, when the former mated-station is the current mated-station, PL$[i] < 1$ and task i cannot be assigned. If the current mated-station is the predetermined station, the priority of task i is boosted to increase the possibility of finding a solution which satisfies the positional restriction. If the current mated-station is the latter mated-station, then the priority of task i remains unchanged for the sake of gaining a feasible solution.

4.4. Improved Positive Zoning Restriction Handling. Yuan et al. [32] and Li et al. [34] allocate the task in the positive zoning restriction as a whole and at an unchangeable sequence to a station. This paper improves their method by assigning the tasks in the positive zoning restriction one by one, which helps reduce sequence-dependent idle time. For this purpose, the relationship matrix among tasks TT[i][h] is defined, and the initial values are set by following expression (16) and the remaining ones are set to zero:

$$\begin{aligned} &\mathrm{TT}\,[i]\,[h] = 1, \\ &\mathrm{TT}\,[h]\,[i] = 1, \\ &\qquad \forall\,(i,h) \in \mathrm{PZ}. \end{aligned} \tag{16}$$

For task i, if PL[i] > 1, its priority is updated as follows:

$$\mathrm{Pl}\,[i] = \mathrm{Pl}\,[i] + \sum_{h=1}^{\mathrm{NT}} \mathrm{TT}\,[i]\,[T_h]\,, \tag{17}$$

where NT is the number of tasks that have been allocated to station (j,k) and T_h is the hth task on station (j,k). With this adjustment, if there is at least one task in the positive zoning restriction allocated to station (j,k), then the priorities of other tasks are boosted.

4.5. Decoding Scheme for the TALBP-II with Assignment Restrictions. The procedure of the decoding scheme is introduced as follows.

Step 1. Set the initial cycle time $\mathrm{CT}_{\mathrm{initial}}$.

Step 2. Open a new mated-station.

Step 3. Choose a side with greater capacity of the current mated-station. If the capacities of both sides are equal, select the left side by default.

Step 4. Update the priority values with task assignment rule.

Step 5. If task i, $i \in I$, is not in the negative zoning restriction, go to Step 5. Else, if PL[i] > 1 and the other task has been allocated to the current mated-station, then PL[i] ← PL[i] − ψ.

Step 6. If PL[i] > 1, then PL[i] = PL[i] + TS[i][j][k].

Step 7. If PL[i] > 1, then Pl[i] = Pl[i] + $\sum_{h=1}^{\mathrm{NT}}$ TT[i][T_h].

Step 8. If task i, $i \in$ I, is not in the synchronous restriction, go to Step 8. Else, if tasks i, h in synchronous restriction can be assigned, then assign the two tasks and go to Step 10; otherwise, PL[i] ← PL[i] − ψ and go to Step 9.

Step 9. If no task whose priority value is larger than one exists, select another side and execute Steps 2–7. If no tasks can be assigned to either side, go to Step 2; else, select the task with the largest priority and allocate the tasks to the selected station, and then go to Step 10.

Step 10. If some tasks are still not assigned, go to Step 3; if all the tasks are allocated, go to Step 11.

Step 11. Choose the largest finishing time of the stations as the current cycle time and update the initial cycle time.

In this procedure, Step 4 considers the cycle time, direction, and precedence restrictions. Steps 5, 6, 7, and 8 deal with negative zoning restrictions, positional restrictions, positive zoning restrictions, and synchronous restrictions, respectively. After executing Steps 4–8, the task can be assigned only if PL[i] > 1. Note that the initial cycle time is set to $\mathrm{CT}_{\mathrm{initial}} \leftarrow a \times \mathrm{Cm}$ $(a > 1)$; then the cycle time is updated if the current cycle time is better than the incumbent one with $\mathrm{CT}_{\mathrm{initial}} \leftarrow \mathrm{CT}_{\mathrm{current}} - 1$. Therefore, the cycle time is reduced gradually.

4.6. Cost Function. In the decoding scheme, the synchronism restriction and negative zoning restriction are satisfied while positive zoning restriction and positional restrictions may be violated. Therefore, the two factors are considered in the cost function as follows:

$$f \longleftarrow f + w_p \mathrm{np} + w_{\mathrm{pz}} \mathrm{npz}, \tag{18}$$

where np and npz are the number of the violated positional restrictions and the number of the violated positive zoning restrictions and w_p and w_{pz} are the penalty coefficients. w_p and w_{pz} should be large enough to obtain a solution which satisfies all the restriction, and thus w_p and w_{pz} are both set equal to 1000.

5. Iterated Greedy Approach for TALBP-II with Assignment Restrictions

The iterated greedy (IG) algorithm is a local search method and it is can be regarded as a simple approach with less sophisticated parameters than other hybrid algorithms [38]. Despite its simplicity, the IG algorithm has shown promising results for different scheduling problems. IG starts with constructing a high performing initial solution. Then random destruction is proposed to remove some tasks and reconstruction is utilized to reinserts these tasks back. After reconstruction, a different solution can be obtained and then the acceptance criterion is applied to determine whether the incumbent can be substituted. Optionally, after the acceptance criterion phase, a local search procedure aims at finding better solutions locally. The random destruction, reconstruction, acceptance criterion, and local search together make up a loop. The procedure of IG for TALBP-II with assignment restrictions is explained as follows.

5.1. Initialization with Modified NEH Heuristic. The NEH heuristic [39] has been shown to be effective for the flow shop scheduling problem, but it has been not applied to the TALBP-II, which can be regarded as a special flow shop scheduling problem. This paper develops a modified NEH (mNEH) for TALBP-II explained as follows.

```
Procedure Original local search (π)
  improve := true;
  While (improve = true) do
    improve := false;
    For i := 1 to n do
      remove a task k at random from π (Different from the last selected one);
        π' := best permutation by inserting k into all possible positions of π;
      If Fit(π') < Fit(π);
          π := π'
          improve := true;
      End if
    End for
  End while
```

Algorithm 1: Original local search (LS1).

Step 1. Generate an initial task permutation π ($\pi = [\pi_1, \pi_2, \ldots, \pi_n]$) with RPW heuristic [37].

Step 2. Remove the 2nd task i from π and obtain the solution by inserting it into position 1. Then the better one is selected.

Step 3. Remove the remaining tasks i, $i = 3, \ldots, n$, and obtain the best solution by inserting it to all the positions before its incumbent.

The mNEH differs from the original NEH in two aspects: (1) the well-known ranked positional weight in assembly line balancing is proposed to gain initial task permutation; (2) the tasks are not inserted one by one and the following parts of the task permutation are maintained, guaranteeing that a feasible solution can be obtained each time. It is worth noting that the mNEH heuristic finds much better solutions than just using the RPW heuristic. The difference is so clear that no statistical test is needed to prove the effectiveness of the proposed mNEH heuristic.

5.2. Improved Local Search. The insert operator is selected as the neighbor structure after checking the swap operator and insert operator. Based on the work of Ruiz and Stützle [38], a local search is developed and shown in Algorithm 1. In Algorithm 1, a task k is removed from the current position and inserted into all the possible positions. The new generated schedule will replace the incumbent one when an improvement can be achieved. The entire procedure iterates until no improvement can be achieved.

The local search of Algorithm 1 aims at finding a local optimum, but it may lose the local optimum since some tasks may not be tested, and it requires a lot of computational time due to inserting a task in all the possible positions. To overcome the first drawback, all the tasks are selected in a given permutation (π^{rp}). Initial experiments show that the best permutation performs better than a random task permutation, and thus the best permutation is selected as the referred permutation. As for the second drawback, the key point is eliminating invalid repetitions of the decoding while preserving the ability of finding the best permutation obtained by inserting a selected task in all possible positions. After checking the solutions of inserting a task in all possible positions, we can see that a lot of solutions have some fitness. In fact, some new task permutation may conflict with the precedence restriction, and this kind of task permutations corresponds to a task permutation satisfying precedence restriction, which indicates that the new task permutation conflicted with the precedence restriction is not needed. Therefore, the improved local search inserts a task into a new position only when it follows all its predecessors and precedes all its successors [22]. The improved local search is depicted in Algorithm 2.

5.3. Destruction, Reconstruction, and Acceptance Criterion. In order to escape local optima, destruction and reconstruction are applied to obtain a new solution. Then, the acceptance criterion is utilized to determine whether the new solution replaces the incumbent. In the destruction phase of Ruiz and Stützle [38], some randomly selected tasks are removed from the incumbent solution π and inserted into a task list π_R which consists of removed tasks. We introduce a small modification in which the tasks are moved to the back of the task permutation, which makes sure that each task permutation can acquire a feasible solution. And in the reconstruction phase, the back tasks are reinserted into the former stations one by one. Once a new solution is obtained after the destruction, reconstruction, and local search phases, the acceptance criterion is adopted to check whether this one is acceptable. We employ the acceptance criterion similar to the one suggested by Pan et al. [10] and the constant temperature is calculated with temperature $= T \cdot \sum_i^n t_i/100\text{nm}$. The procedure of IG is explained in Algorithm 3.

Note that the number of removed tasks (d) and the parameter T are important parameters and need to be determined. A large d may result in a completely new solution while a small d cannot help the algorithm escape from a local optimum. Similarly, a large T may accept poor-quality solutions, while a small T can result in no acceptance of new solutions.

```
Procedure LS2 (π, π^rp)
  counter := 0; i := 1;
  While (counter < nt)
    Remove task π_i^rp from π;
    for j := 1 to n do
      If task i satisfies precedence restriction when inserted into position j then
          π' := solution by inserting task i into position j;
        If Fit(π') < Fit(π);
            π := π'; counter := 0;
          Else if Fit(π') = Fit(π);
              π := π';
          End if
        Else
          Continue;
      End for
      counter := counter + 1;
      i := mod(i + 1, nt)
  End while
```

Algorithm 2: Improved local search with insertion operator (LS2).

```
Procedure of IG (d, T)
  π := mNEH_heuristic initialization;
  π_b := π;
  While (termination criterion is not satisfied) do
    π' := π;                                                          % Destruction phase
    For i := 1 to d do
        π' := remove one task of π' randomly to position n − d + i;
    End for
    For i := 1 to d do                                                % Destruction phase
        π' := best permutation by inserting task π(n − d + i) in all possible position before n − d + i of π';
    End for
    π'' := Local search (π')                                          % Local search
    If Fit(π'') < Fit(π) then                                         % Acceptance criterion
        π := π'';
        If Fit(π) < Fit(π_b) then                                     % Check whether new best solution
            π_b := π;
        End if
    Else if rand(0, 1) ≤ exp{−(Fit(π'') − Fit(π))/Temperature} then
        π := π'';
    End if
  End while
  Return π_b
End
```

Algorithm 3: IG algorithm for the TALBP-II.

6. Computational Results

The computational tests are carried out to prove the effectiveness of the priority-based decoding schemes for TALBP-II with assignment restrictions and the high performance of the improved IG algorithm. All the benchmark problems with different cycle times are solved, which range from small-size problems, P9, P12, P16, and P24, to large-size problems, P65, P148, and P205. P9, P12, and P24 are taken from Kim et al. [1]. P16, P65, and P205 are taken from Lee et al. [18]. P148 from Bartholdi [14] is modified just like Kim et al. [6]. As for the TALBP-II with assignment restrictions, the additional restrictions are generated for the study and they are listed in Table 1. For the coefficients, w_1, w_2, w_3, w_4, in the objective function, (1, 1, 1, 1), (10, 10, 1, 1), (10, 5, 2, 1), and (10, 5, 1, 1) are checked and (10, 5, 1, 1) are chosen. Since the sizes of different problems differ from each other, we set the elapsed CPU time $t = \text{nt} \times \text{nt} \times \rho$ ms as a termination criterion for all cases and ρ is set to 15.

6.1. Evaluation of the Proposed Priority-Based Decoding Scheme. This section tests the performance of the proposed

Table 1: List of assignment restrictions.

Problem	Number of mated-station	Positional restriction {task, (mated-station, side)}	Positive zoning restriction {task, task}	Negative zoning restriction {task, task}	Synchronism restriction {task, task}
P9	2	{1, (1, 1)}	{6, 9}	{3, 9}	{4, 5}
	2	{2, (1, 2)}	{6, 9}	{3, 9}	{4, 5}
P12	2	{4, (1, 1)}	{1, 4}	{2, 3}	{10, 11}
	3	{4, (1, 1)}, {2, (1, 2)}	{1, 4}	{2, 3}	{10, 11}
P16	2	{2, (1, 1)}	{3, 6}	{8, 9}	{11, 12}
	3	{2, (1, 1)}, {9, (2, 2)}	{3, 6}	{8, 9}	{11, 12}
P24	2	{2, (1, 1)}, {4, (1, 2)}	{12, 13}	{9, 13}	{1, 4}, {17, 19}
	3	{2, (1, 1)}, {21, (3, 1)}	{12, 13}	{9, 13}	{1, 4}, {17, 19}
	4	{2, (1, 1)}, {21, (4, 1)}	{12, 13}	{9, 13}	{1, 4}, {17, 19}
P65	4	{4, (1, 2)}, {13, (1, 2)}, {9, (2, 1)}	{7, 8}, {31, 32}	{13, 14}, {29, 30}	{22, 23}, {51, 52}
	6	{4, (1, 2)}, {7, (2, 2)}, {17, (4, 1)}, {61, (6, 2)}	{7, 8}, {31, 32}	{13, 14}, {29, 30}	{22, 23}, {51, 52}
	8	{4, (1, 2)}, {7, (2, 2)}, {17, (5, 1)}, {38, (7, 2)}, {61, (8, 2)}	{7, 8}, {31, 32}	{13, 14}, {29, 30}	{22, 23}, {51, 52}
P148	5	{2, (1, 2)}, {15, (3, 1)}, {27, (4, 1)}, {135, (5, 2)}	{38, 39}, {45, 46}, {142, 145}	{11, 12}, {42, 43}, {119, 120}	{9, 10}, {30, 32}, {132, 138}
	7	{2, (1, 2)}, {13, (3, 1)}, {87, (6, 2)}, {122, (7, 2)}	{47, 48}, {50, 52}, {142, 145}	{11, 12}, {42, 43}, {119, 120}	{9, 10}, {26, 28}, {132, 138}
	9	{2, (1, 2)}, {65, (3, 1)}, {68, (4, 1)}, {79, (8, 2)}, {126, (9, 2)}	{47, 48}, {50, 52}, {142, 145}	{11, 12}, {42, 43}, {119, 120}	{9, 10}, {26, 28}, {132, 138}
P205	6	{2, (1, 2)}, {86, (3, 1)}, {140, (5, 1)}, {191, (6, 2)}	{14, 18}, {56, 57}, {154, 160}, {173, 185}	{21, 22}, {109, 110}, {114, 115}	{19, 20}, {96, 97}, {145, 151}, {194, 195}
	8	{2, (1, 2)}, {36, (2, 2)}, {91, (4, 2)}, {126, (6, 2)}, {189, (8, 1)}	{14, 18}, {56, 57}, {154, 160}, {173, 185}	{21, 22}, {109, 110}, {114, 115}	{19, 20}, {96, 97}, {145, 151}, {194, 195}
	10	{3, (1, 2)}, {11, (2, 2)}, {87, (4, 1)}, {140, (8, 1)}, {191, (10, 2)}	{14, 18}, {56, 57}, {121, 122}, {166, 167}	{21, 22}, {109, 110}, {114, 115}	{19, 20}, {96, 97}, {145, 151}, {194, 195}

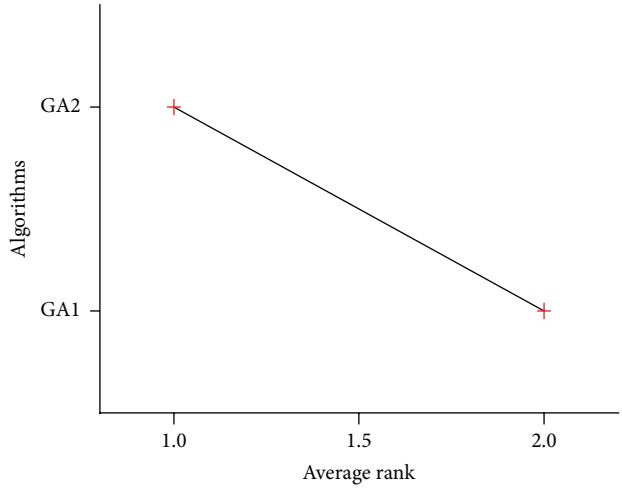

Figure 2: Mean plot of average ranks with 95% confidence intervals.

priority-based decoding scheme by comparing it with the reported ones. There are two components to be considered: (1) the performance of the priority-based decoding on the TALBP-II and (2) the performance of the new methods to deal with assignment restrictions.

For the first component, two genetic algorithms (GA) with different decoding schemes are compared and all the algorithms have the same parameters. They are GA1 with the station-oriented decoding scheme [6] and GA2 with the proposed decoding scheme. All the large-size problems are tested and each case runs 20 times. The average relative deviation index (RDI) for each case is utilized for comparison and the RDI can be calculated using the following expression:

$$\text{Relative deviation index (RDI)} = \frac{\text{Some}_{\text{sol}} - \text{Best}}{\text{Worst} - \text{Best}} \times 100, \tag{19}$$

where, Some_{sol} is the fitness obtained by one running and Best and Worst are the best fitness and worst fitness of 40 individuals for a same case, respectively.

In order to check whether there is statically significant difference between the two decoding schemes, we use the parametric t test, but these RDI values show a strong deviation from the normality, which is caused by the difference of the benchmark problems. Therefore, we propose a nonparametric Wilcoxon matched-pairs signed rank test. In addition, all results of the decoding schemes are transferred in the following way: the best result is assigned to rank 1 while the worst result is marked with rank 2. The analysis shows there is a significant difference between the two decoding schemes with P values close to zero and the means plot with 95% minimal significant confidence intervals is depicted in Figure 2. It can be seen that the average rank for GA2 with the proposed decoding is 1, which proves the superiority of the proposed decoding scheme.

In fact, the original station-oriented decoding scheme cannot reduce the sequence-dependent idle time effectively since it only uses a heuristic method. The proposed priority-based decoding, on the contrary, takes the sequence-dependent idle times and the balance of the stations into account, which can further reduce the idle times on the stations.

For the method to deal with assignment restrictions, we also utilized two genetic algorithms: (1) GA3 with the published method of dealing with assignment restrictions in Yuan et al. [32] and Li et al. [34] and (2) GA4 with the proposed method of dealing with assignment restrictions. Based on the computational test, the GA4 outperforms the GA3 for all the cases; namely, the improved method outperforms the original one. Also, there is statically significant difference between the two algorithms. There are mainly two reasons leading to the high performance of the new method of dealing with assignment restrictions: (1) new positional restriction handling increases the possibility of finding a solution which satisfies positional restriction and (2) new positive zoning restriction handling can reduce the sequence-dependent idle time caused by assigning the tasks in positive zoning restriction together.

6.2. Calibration of the Iterated Greedy Algorithm. In this part, we calibrate the iterated greedy algorithm and determine d in the destruction phase and T in the acceptance criterion. All the possible combinations of the two factors are tested and the full factorial design is listed as follows:

(i) d: 4 levels (4, 8, 12, 16);

(ii) T: 4 levels (0.1, 0.5, 1, 5).

Since different problems are involved, we select the best combination for each problem. Taking the largest-size problem, P205 [18], as an example, the calibration procedure is introduced as follows. To guarantee the validity of the calibration, 5 cases for P205 are solved and average results of each case are recorded. After finishing all the experiments, we also propose the RDI to measure the solutions. Then, the average RDI value of each case is proposed for statistical analysis. The algorithm is coded in C++ programming language with Microsoft Visual Studio 2012 which is running on a set of personal computers with Intel® Core2™ CPU 2.33 GHZ, 3.036 GB RMA.

After carrying out all the runs, the analysis of variance (ANOVA) technique [40] is utilized to analyze the date, after checking the independence of the residuals, homogeneity of variance, and normality of the residuals. For ANOVA results, there is a statistically significant difference among the different levels if the P value of one factor is less than a predetermined number α (α is set equal to 0.05). Table 2 reports the results for RDI values, and d has a statistically significant effect on RDI at the 95.0% confidence. And the mean plots for the factors d and T are depicted in Figure 3. According to the ANOVA results, we choose $d = 4$ at first and, then, the value of d is fixed and $T = 1$ is chosen.

Notice that it is more convenient to divide all the problems into several groups and decide the parameters for each group. This paper selects the parameters for each problem due to different performance of an algorithm on different problems, and the homogeneity of variance and normality of

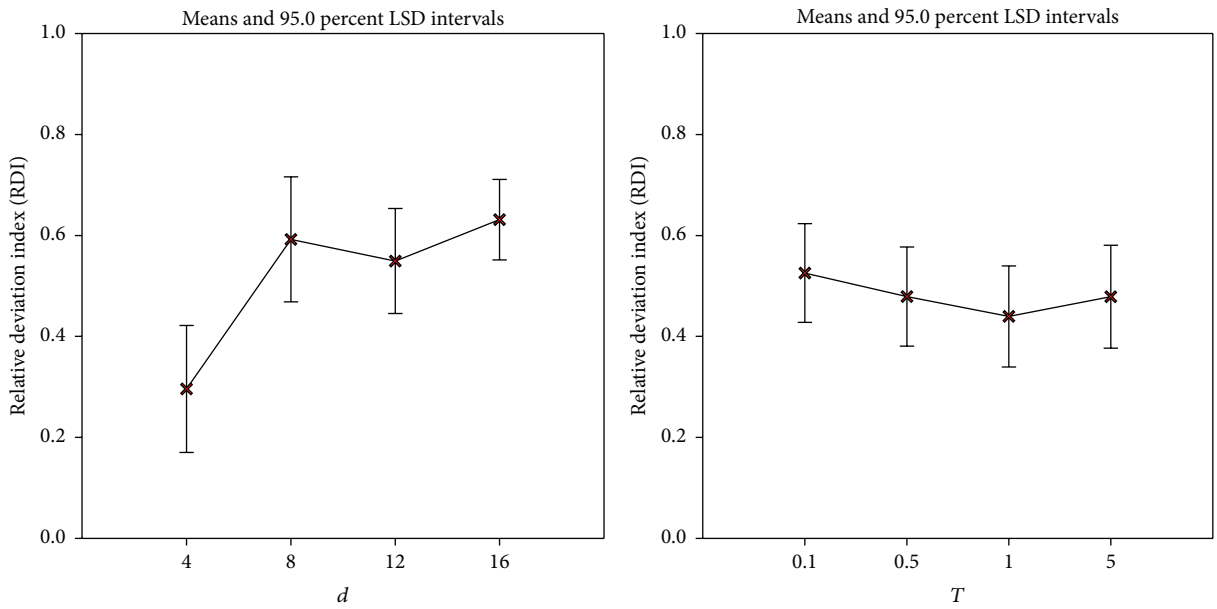

Figure 3: Mean plot for the factors d and T for the IG ANOVA calibration experiment.

Table 2: ANOVA results for RDI values.

Source of variation	Df	Sum of squares	Mean square	F-ratio	P value
d	3	1.375	0.458	8.151	**<0.001**
T	3	0.102	0.034	0.604	0.615
$d \times T$	9	0.491	0.0546	0.971	0.472
Residual	64	3.599	0.0562		
Total	79	5.568	0.0705		

the residuals are violated if we calibrate the parameters for several problems together.

6.3. Performance Comparison among Algorithms. This section aims at showing the effectiveness of the IG on TALBP-II and TALBP-II with additional restrictions. The IG is compared with several recent algorithms: a tabu search algorithm (TS) [20], a simulated annealing algorithm (SA) [22], a late acceptance hill-climbing algorithm (LAHC) [32], a two-ant colony optimization algorithm (2-ANTBAL) [4], a particle swarm optimization algorithm with negative knowledge (PSONG) [25], a genetic algorithm (GA) [1], a teaching-learning-based optimization algorithm (TLBO) [34], and a bee optimization algorithm (BA) [21]. In order to show the superiority of the improved local search, IG with LS1 is marked with IG1 and the improved IG with LS2 is marked with IG2. All the algorithms are calibrated with the same procedure in Section 6.2, and all the parameters and the code in C++ language are available upon request from the authors.

Since the IG algorithms have not been applied to the TALBP-II without additional restrictions, we first compare the performances of these algorithms on the TALBP-II without additional restrictions. All the cases are solved for 20 times, and the average results on 14 small-size cases are reported in Table 3. The optimal solution (OPT) is also included which can be obtained by General Algebraic Modeling System 23.0 (GAMS)/CPLEX. It can be observed from Table 3 that all the algorithms can find the optimal solution in each run. Computational results demonstrate the effectiveness of the proposed IG2 for small-size problems.

As for the large-size problem, the average RDI value of each case is reported in Table 4 and the best cycle time of each case is shown in Table 5. In Table 4, we observe that three local search algorithms, TS, SA, and LAHC, do not obtain satisfactory results, while the IG2 as a local search method can still gain remarkable results. In particular, the IG2 acquires the best RDI values for almost all the largest cases and its average RDI is only 4.1%. Specifically, the IG2 outperforms the TS, SA, and LAHC for 24, 24, and 24 cases, respectively. As for swarm-based algorithms, the PSONG and 2-ANTBAL obtain remarkable results. Table 5 shows the best result by each algorithm with twenty runs. The lower bound (LB) of the cycle time [6] and the best solutions by a neighborhood genetic algorithm (n-GA) [6] is also reported. Contrary to the results on the average results, the local search algorithms can obtain better results than swarm-based algorithms partly. IG2 outperforms the compared algorithms regarding the best solutions for most cases and it improves the results by n-GA for 17 cases. Among all 25 large-size cases, IG2 outperforms TS, SA, and LAHC for 20, 11, and 15 cases.

Based on the above computational results, we can draw the following conclusions. First, some swarm-based algorithms cannot obtain satisfying results on the best solutions since they lack the ability of strong local search. Second, the local search methods, TS, SA, and LAHC, perform worse than population-based methods on the average results. Third, the destruction and reconstruction phase are an effective method to escape from local optima, which is confirmed by

Table 3: Average result comparison for small-size problems.

Problem	nm	OPT	TS	SA	LAHC	2-ANTBAL	PSONG	GA	TLBO	BA	IG1	IG2
P9	2	5	5	5	5	5	5	5	5	5	5	**5**
	3	3	3	3	3	3	3	3	3	3	3	**3**
P12	2	7	7	7	7	7	7	7	7	7	7	**7**
	3	5	5	5	5	5	5	5	5	5	5	**5**
	4	4	4	4	4	4	4	4	4	4	4	**4**
	5	3	3	3	3	3	3	3	3	3	3	**3**
P16	2	22	22	22	22	22	22	22	22	22	22	**22**
	3	16	16	16	16	16	16	16	16	16	16	**16**
	4	15	15	15	15	15	15	15	15	15	15	**15**
	5	11	11	11	11	11	11	11	11	11	11	**11**
P24	2	35	35	35	35	35	35	35	35	35	35	**35**
	3	24	24	24	24	24	24	24	24	24	24	**24**
	4	18	18	18	18	18	18	18	18	18	18	**18**
	5	16	16	16	16	16	16	16	16	16	16	**16**

Table 4: Average relative deviation index (RDI) for the large-size cases. Best results in bold.

Problem	nm	TS	SA	LAHC	2-ANTBAL	PSONG	GA	TLBO	BA	IG1	IG2
65	4	100.00	75.00	51.09	38.04	4.35	18.48	15.22	21.74	13.04	**0.00**
	5	100.00	73.79	43.45	15.86	**0.00**	10.34	11.03	13.10	16.55	1.38
	6	100.00	93.87	50.92	15.34	5.52	23.93	0.00	33.74	13.50	**0.00**
	7	84.35	100.00	74.78	8.70	9.57	23.48	13.04	31.30	20.87	**0.00**
	8	82.28	100.00	72.15	17.72	3.80	23.42	5.70	22.15	15.82	**0.00**
148	4	57.58	100.00	18.18	**0.00**	**0.00**	10.61	0.00	0.00	4.55	**0.00**
	5	100.00	96.43	50.00	**0.00**	**0.00**	26.79	0.00	8.93	21.43	10.71
	6	89.89	100.00	13.30	1.06	**0.00**	6.38	0.00	0.00	3.19	2.13
	7	72.16	100.00	47.42	**0.00**	**0.00**	16.49	0.00	0.00	12.37	11.34
	8	100.00	98.65	59.46	**0.00**	6.76	41.89	20.27	21.62	28.38	18.92
	9	96.04	100.00	60.40	2.97	**0.00**	21.78	2.97	0.00	10.89	8.91
	10	100.00	82.91	64.56	**0.00**	**0.00**	33.54	0.00	0.00	21.52	18.99
	11	79.77	100.00	63.58	1.16	**0.00**	15.03	2.89	3.47	14.45	8.67
	12	81.44	100.00	70.10	1.03	**0.00**	25.77	6.70	0.00	18.56	17.01
205	4	100.00	84.57	56.35	17.24	33.48	14.34	17.51	48.91	54.08	**0.00**
	5	95.66	100.00	63.30	13.98	23.61	26.80	5.63	27.41	43.76	**0.00**
	6	100.00	80.81	50.36	15.06	20.24	21.70	28.74	27.85	40.81	**0.00**
	7	100.00	73.04	68.97	16.30	16.22	14.42	26.18	45.69	33.62	**0.00**
	8	84.32	100.00	73.64	3.36	11.11	13.18	20.84	6.55	33.07	**0.00**
	9	83.55	100.00	56.52	18.36	15.90	14.39	17.65	40.22	41.97	**0.00**
	10	95.99	100.00	72.43	**0.00**	9.42	30.63	17.89	23.91	42.93	4.54
	11	96.91	100.00	65.40	6.65	7.21	14.89	8.08	11.24	23.20	**0.00**
	12	100.00	78.69	71.81	16.43	13.84	18.63	20.92	32.77	40.04	**0.00**
	13	100.00	93.14	88.76	0.15	0.00	3.21	0.00	0.00	15.04	**0.00**
	14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	**0.00**
Avg		88.00	89.24	56.28	8.38	7.24	18.80	9.65	16.82	23.35	**4.10**

the superiority of the IG2 over TS, SA, and LAHC about the average results. Fourth, the new local search is proved efficient by the superiority of the IG2 over the IG1.

In order to confirm that the IG2 is statistically better than the compared ones for the average results, we also carry out statistical analysis. Since the normality of the average RDI values is violated, we carry out a nonparameter Friedman rank-based test [41]. In addition, the results of different algorithms are transferred so that the best result is marked with a rank of 1 and the worst result on the average RDI is with 10 since there are 10 algorithms. The results of the analysis show that the P values for the average ranks are close to zero,

Table 5: Best results of large-size problems. Best results in bold.

Problem	nm	LB	n-GA	TS	SA	LAHC	2-ANTBAL	PSONG	GA	TLBO	BA	IG1	IG2
65	4	638	641	641	**638**	**638**	639	639	**638**	639	640	639	**638**
	5	510	515	514	512	513	512	**511**	**511**	512	513	512	512
	6	425	432	431	427	427	427	427	**426**	427	430	**426**	427
	7	365	372	369	**367**	368	**367**	368	368	368	370	368	**367**
	8	319	327	326	322	323	322	**321**	323	323	325	322	**321**
148	4	641	**641**	**641**	**641**	**641**	**641**	**641**	**641**	**641**	**641**	**641**	**641**
	5	513	514	**513**	**513**	**513**	**513**	**513**	**513**	**513**	**513**	**513**	**513**
	6	427	**428**	**428**	**428**	**428**	**428**	**428**	**428**	**428**	**428**	**428**	**428**
	7	366	368	**367**	**367**	**367**	**367**	**367**	**367**	**367**	**367**	**367**	**367**
	8	321	323	323	**321**	322	**321**	**321**	322	**321**	322	322	**321**
	9	285	287	287	**286**	**286**	**286**	**286**	**286**	**286**	**286**	**286**	**286**
	10	257	259	259	**258**	259	**258**	**258**	259	**258**	**258**	**258**	**258**
	11	233	237	236	235	236	**234**	**234**	**234**	**234**	235	235	**234**
	12	214	218	218	**215**	**215**	**215**	**215**	216	**215**	**215**	216	216
205	4	2919	2946	2943	2945	2941	2933	2946	2933	2937	2948	2947	**2927**
	5	2335	2364	2369	2360	2347	2350	2362	2351	2349	2353	2359	**2345**
	6	1946	1984	1995	1966	1969	1961	1969	1966	1972	1975	1968	**1956**
	7	1668	1709	1692	1691	1690	1689	1690	1687	1698	1709	1692	**1682**
	8	1460	1507	1485	1479	1486	**1472**	1482	1481	1490	1481	1486	1474
	9	1297	1337	1338	1325	1323	1325	1319	1314	1325	1340	1328	**1311**
	10	1168	1189	1214	1193	1196	1182	1188	1197	1195	1201	1198	**1181**
	11	1062	1095	1093	1082	1101	1081	1082	1082	1085	1089	1082	**1078**
	12	973	1039	1002	995	1000	995	994	994	999	1004	1000	**984**
	13	944	**944**	953	**944**	**944**	**944**	**944**	**944**	**944**	**944**	**944**	**944**
	14	944	**944**	**944**	**944**	**944**	**944**	**944**	**944**	**944**	**944**	**944**	**944**

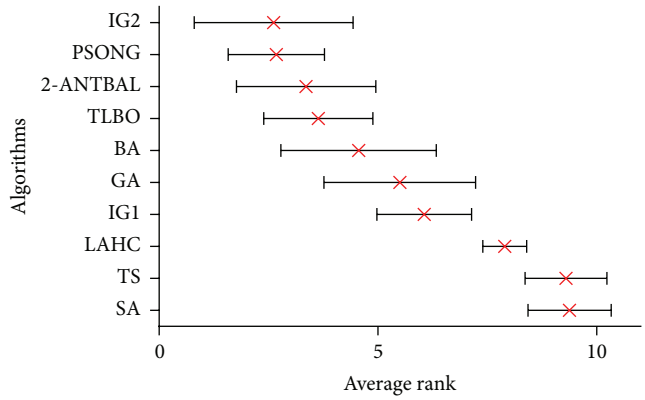

Figure 4: Means plot of the ranks of the average solutions with 95% confidence level.

which indicates there is a statistically significant difference among the average ranks of these compared algorithms. And we also depict the means plot with 95% minimal significant difference confidence intervals in Figure 4 for further studying the difference.

As you see, a total of 10 algorithms are compared. If ordering these algorithms from the best ranking to the worst ranking, an order of algorithms can be achieved: IG2, PSONG, 2-ANTBAL, TLBO, BA, GA, IG1, LAHC, TS, and SA. As stated above, the local search algorithms, SA, TS, and LAHC, are statistically worse than the swarm-based algorithms. Still, as a local search method, the IG2 is statistically better than all the compared algorithms on the average results. According to these results, it is suggested that the proposed IG2 statistically outperforms all the compared algorithms.

As for TALBP-II with additional restrictions, we also utilize all the algorithms to solve the TALBP-II with additional restrictions. The computational results are also analyzed with statistical techniques and these algorithms show similar

Table 6: Result comparisons forassignment restrictions.

Problem	nm	LB	Without restrictions		With restrictions				
			IG2		GAMS		IG2		
			Min	Mean	Min	CPU (s)	Min	Mean	CPU (s)
P9	2	4.25	5	5	5	<1	5	5	<1
	3	2.83	3	3	4	<1	4	4	<1
P12	2	6.25	7	7	7	<1	7	7	<1
	3	4.17	5	5	5	<1	5	5	<1
P16	2	20.5	22	22	24	<1	24	24	<1
	3	13.67	16	16	16	<1	16	16	<1
P24	2	35	35	35	36	>7200	36	36	<1
	3	23.33	24	24	25	>7200	25	25	<1
	4	17.5	18	18	18	152 s	18	18	<1
P65	4	637.4	638	638.80	—	—	640	643.75	63.375
	6	424.9	427	427.55	—	—	428	430.60	63.375
	8	318.7	321	323.60	—	—	324	324.95	63.375
P148	5	512.4	513	513.00	—	—	515	515.85	328.56
	7	366	367	367.35	—	—	368	369.9	328.56
	9	284.7	286	286.40	—	—	287	287.7	328.56
P205	6	1945.4	1962	1969.20	—	—	1968	1980.65	630.375
	8	1459.1	1479	1487.10	—	—	1487	1492.65	630.375
	10	1167.3	1187	1199.70	—	—	1187	1200.5	630.375

results on the TALBP-II without additional restrictions. In fact, only the decoding schemes of the algorithms are slightly changed, and other parts of the algorithms remain unaltered. For simplification, we only show the best results by IG2 in Table 6.

For small-size problems, the results by IG2 are compared with optimal results by GAMS and the runs are interrupted when the optimal solution is discovered. For large-size problems, the results of TALBP-II with additional restrictions are compared with those of TALBP-II without additional restrictions. As can be seen in Table 6, the IG2 and GAMS can both find optimal solutions for all the small-size problems (P9, P12, P16, and P24). But for P24, GAMS needs much more computational time and IG2 can find the optimal solutions for small-size problems within 1 second. As for the large-size problems, GAMS cannot solve them owing to the tremendous compute. However, IG2 as an effective metaheuristic can also find promising results. Taking the P205 (CT = 10) as an example, you can see that the IG2 can find the same best cycle time for the TALBP-II with additional restrictions as that for TALBP-II without additional restrictions. All the computational results suggest the superior performance of IG2 for TALBP-II with additional restrictions.

7. Summary and Conclusion

In this paper, a simple and effective iterated greedy (IG) algorithm is developed for the two-sided assembly line balancing problem type-II (TALBP-II) with assignment restrictions. The NEH heuristic is modified for TALBP-II as the initialization procedure. A new local search with referred permutation is developed, and acceleration by eliminating the insert operator that conflicted with precedence restrictions is developed to speed up the search process while preserving the ability of finding a local optimum.

A new priority-based decoding scheme is also proposed to reduce sequence-dependence idle time, balance the workload, and deal with assignment restrictions. To be specific, the task assignment rules are embedded into the decoding scheme to reduce sequence-dependent idle time and balance the workload on each station. A new method to deal with the positional restriction increase the possibility of finding a feasible solution by preventing assigning the tasks with positional restriction to the former mated-station of the predetermined one. And a new method to deal with positive zoning restriction is applied to further reduce the sequence-dependent idle time by allocating the tasks with positive zoning restriction separately.

Computational studies are carried out to demonstrate the superiority of the developed priority-based decoding scheme and the effectiveness of the proposed IG algorithm. The priority-based decoding scheme is compared with published ones, and computational results show that the priority-based decoding scheme can reduce the sequence-dependent idle times, balance the workload, and deal with assignment restrictions effectively. The computational results of IG on the TALBP-II are compared with those of eight recent algorithms, and the IG obtains superior results on both the average results and the best results, which prove its strong local search and the remarkable ability of escaping from local optima. As for the TALBP-II with additional restrictions, IG can find all the optimal solutions for small-size problems and also obtain promising results for large-size problems.

Future research can apply the IG algorithm to solve other complex assembly line balancing problems, such as the mixed-model assembly line. The priority-based encoding and decoding, especially the methods to reduce idle times and deal with assignment restrictions, may be modified to address other two-sided assembly line problem.

Notations

Indices

i, h, p, r: Tasks
j, g: Mated-stations
k: A side of the line; $k = \{1, \text{left side}; 2, \text{righ side}\}$
(j, k): Station of mated-station j at side k.

Parameters

nt: Number of tasks
nm: Number of mated-stations
CT: Cycle time
I: Set of tasks, $I = \{1, 2, \ldots, i, \ldots, \text{nt}\}$
J: Set of mated-stations; $J = \{1, 2, \ldots, j, \ldots, \text{nm}\}$
t_i: Processing time of task i
AL: Set of tasks with left direction, AL $\subseteq I$
AR: Set of tasks with right direction, AR $\subseteq I$
AE: Set of tasks either direction, AE $\subseteq I$
P_0: Set of tasks that have no immediate predecessors
$P(i)$: Set of immediate predecessors of the task i
$P_a(i)$: Set of all predecessors of the task i
$S_a(i)$: Set of successors of the task i
$S(i)$: Set of immediate successors of the task i
ψ: Large positive number
$C(i)$: Set of tasks whose operation directions are opposite to that of task i; $C(i) = \{\text{AL if } i \in \text{AR}; \text{AR if } i \in \text{AL}; \varnothing \text{ if } i \in \text{AE}\}$
$K(i)$: Set of integers which indicate the preferred directions of the task i; $K(i) = \{\{1\} \text{ if } i \in \text{AL}; \{2\} \text{ if } i \in \text{AR}; \{1, 2\} \text{ if } i \in \text{AE}\}$
PC: Set of pairs of tasks and predetermined station for positional restriction
PZ: Set of pairs of tasks for positive zoning restriction
NZ: Set of pairs of tasks for negative zoning restriction
SC: Set of pair of tasks for synchronism restriction.

Decision Variables

x_{ijk}: 1, if task i is assigned to mated-station j at side k; 0, otherwise
t_i^f: Finishing time of task i.

Indicator Variables

z_{ip}: 1, if task i is assigned earlier than task p in the same station; 0, otherwise.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This research work is funded by the National Natural Science Foundation of China (Grant nos. 51275366 and 51305311).

References

[1] Y. K. Kim, Y. Kim, and Y. J. Kim, "Two-sided assembly line balancing: a genetic algorithm approach," *Production Planning & Control*, vol. 11, no. 1, pp. 44–53, 2000.

[2] H. D. Purnomo, H.-M. Wee, and H. Rau, "Two-sided assembly lines balancing with assignment restrictions," *Mathematical and Computer Modelling*, vol. 57, no. 1-2, pp. 189–199, 2013.

[3] A. Baykasoglu and T. Dereli, "Two-sided assembly line balancing using an ant-colony-based heuristic," *International Journal of Advanced Manufacturing Technology*, vol. 36, no. 5-6, pp. 582–588, 2008.

[4] A. S. Simaria and P. M. Vilarinho, "2-ANTBAL: an ant colony optimisation algorithm for balancing two-sided assembly lines," *Computers & Industrial Engineering*, vol. 56, no. 2, pp. 489–506, 2009.

[5] E. Falkenauer, "Line balancing in the real world," in *Proceedings of the International Conference on Product Lifecycle Management*, vol. 1, pp. 360–370, 2005.

[6] Y. K. Kim, W. S. Song, and J. H. Kim, "A mathematical model and a genetic algorithm for two-sided assembly line balancing," *Computers and Operations Research*, vol. 36, no. 3, pp. 853–865, 2009.

[7] H. D. Purnomo and H.-M. Wee, "Maximizing production rate and workload balancing in a two-sided assembly line using Harmony Search," *Computers and Industrial Engineering*, vol. 76, no. 1, pp. 222–230, 2014.

[8] D. Lei and X. Guo, "Variable neighborhood search for the second type of two-sided assembly line balancing problem," *Computers & Operations Research*, vol. 72, pp. 183–188, 2016.

[9] R. Ruiz and T. Stützle, "An Iterated Greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives," *European Journal of Operational Research*, vol. 187, no. 3, pp. 1143–1159, 2008.

[10] Q.-K. Pan, L. Wang, and B.-H. Zhao, "An improved iterated greedy algorithm for the no-wait flow shop scheduling problem with makespan criterion," *International Journal of Advanced Manufacturing Technology*, vol. 38, no. 7-8, pp. 778–786, 2008.

[11] J. M. Framinan and R. Leisten, "Total tardiness minimization in permutation flow shops: a simple approach based on a variable greedy algorithm," *International Journal of Production Research*, vol. 46, no. 22, pp. 6479–6498, 2008.

[12] I. Ribas, R. Companys, and X. Tort-Martorell, "An iterated greedy algorithm for the flowshop scheduling problem with blocking," *Omega*, vol. 39, no. 3, pp. 293–301, 2011.

[13] G. Minella, R. Ruiz, and M. Ciavotta, "Restarted Iterated Pareto Greedy algorithm for multi-objective flowshop scheduling problems," *Computers & Operations Research*, vol. 38, no. 11, pp. 1521–1533, 2011.

[14] J. J. Bartholdi, "Balancing two-sided assembly lines: a case study," *International Journal of Production Research*, vol. 31, no. 10, pp. 2447–2461, 1993.

[15] X.-F. Hu, E.-F. Wu, and Y. Jin, "A station-oriented enumerative algorithm for two-sided assembly line balancing," *European Journal of Operational Research*, vol. 186, no. 1, pp. 435–440, 2008.

[16] E.-F. Wu, Y. Jin, J.-S. Bao, and X.-F. Hu, "A branch-and-bound algorithm for two-sided assembly line balancing," *International Journal of Advanced Manufacturing Technology*, vol. 39, no. 9-10, pp. 1009–1015, 2008.

[17] X.-F. Hu, E.-F. Wu, J.-S. Bao, and Y. Jin, "A branch-and-bound algorithm to minimize the line length of a two-sided assembly line," *European Journal of Operational Research*, vol. 206, no. 3, pp. 703–707, 2010.

[18] T. O. Lee, Y. Kim, and Y. K. Kim, "Two-sided assembly line balancing to maximize work relatedness and slackness," *Computers and Industrial Engineering*, vol. 40, no. 3, pp. 273–292, 2001.

[19] S. D. Lapierre and A. B. Ruiz, "Balancing assembly lines: an industrial case study," *Journal of the Operational Research Society*, vol. 55, no. 6, pp. 589–597, 2004.

[20] U. Özcan and B. Toklu, "A tabu search algorithm for two-sided assembly line balancing," *The International Journal of Advanced Manufacturing Technology*, vol. 43, no. 7, pp. 822–829, 2009.

[21] L. Özbakir and P. Tapkan, "Bee colony intelligence in zone constrained two-sided assembly line balancing problem," *Expert Systems with Applications*, vol. 38, no. 9, pp. 11947–11957, 2011.

[22] D. Khorasanian, S. R. Hejazi, and G. Moslehi, "Two-sided assembly line balancing considering the relationships between tasks," *Computers & Industrial Engineering*, vol. 66, no. 4, pp. 1096–1105, 2013.

[23] U. Özcan and B. Toklu, "Balancing of mixed-model two-sided assembly lines," *Computers & Industrial Engineering*, vol. 57, no. 1, pp. 217–227, 2009.

[24] U. Özcan, "Balancing stochastic two-sided assembly lines: a chance-constrained, piecewise-linear, mixed integer program and a simulated annealing algorithm," *European Journal of Operational Research*, vol. 205, no. 1, pp. 81–97, 2010.

[25] P. Chutima and P. Chimklai, "Multi-objective two-sided mixed-model assembly line balancing using particle swarm optimisation with negative knowledge," *Computers and Industrial Engineering*, vol. 62, no. 1, pp. 39–55, 2012.

[26] Y. Delice, E. K. Aydogan, U. Özcan, and M. S. İlkay, "A modified particle swarm optimization algorithm to mixed-model two-sided assembly line balancing," Journal of Intelligent Manufacturing. *In press*.

[27] B. Yuan, C.-Y. Zhang, X.-Y. Shao, and Z.-B. Jiang, "An effective hybrid honey bee mating optimization algorithm for balancing mixed-model two-sided assembly lines," *Computers & Operations Research*, vol. 53, pp. 32–41, 2015.

[28] Z. Yang, G. Zhang, and H. Zhu, "Multi-neighborhood based path relinking for two-sided assembly line balancing problem," *Journal of Combinatorial Optimization*, pp. 1–20, 2015.

[29] 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," *Computers & Industrial Engineering*, vol. 97, pp. 146–156, 2016.

[30] P. Tapkan, L. Ozbakir, and A. Baykasoglu, "Modeling and solving constrained two-sided assembly line balancing problem via bee algorithms," *Applied Soft Computing Journal*, vol. 12, no. 11, pp. 3343–3355, 2012.

[31] P. Tapkan, L. Ozbakir, and A. Baykasoglu, "Bees Algorithm for constrained fuzzy multi-objective two-sided assembly line balancing problem," *Optimization Letters*, vol. 6, no. 6, pp. 1039–1049, 2012.

[32] B. Yuan, C.-Y. Zhang, and X.-Y. Shao, "A late acceptance hill-climbing algorithm for balancing two-sided assembly lines with multiple constraints," *Journal of Intelligent Manufacturing*, vol. 26, no. 1, pp. 159–168, 2013.

[33] B. Wang, Z. Guan, D. Li, C. Zhang, and L. Chen, "Two-sided assembly line balancing with operator number and task constraints: a hybrid imperialist competitive algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 74, no. 5-8, pp. 791–805, 2014.

[34] D. Li, C. Zhang, X. Shao, and W. Lin, "A multi-objective TLBO algorithm for balancing two-sided assembly line with multiple constraints," *Journal of Intelligent Manufacturing*, 2014.

[35] Q. Tang, Z. Li, L. Zhang, C. A. Floudas, and X. Cao, "Effective hybrid teaching-learning-based optimization algorithm for balancing two-sided assembly lines with multiple constraints," *Chinese Journal of Mechanical Engineering*, vol. 28, no. 5, pp. 1067–1079, 2015.

[36] G. Tuncel and D. Aydin, "Two-sided assembly line balancing using teaching-learning based optimization algorithm," *Computers and Industrial Engineering*, vol. 74, no. 1, pp. 291–299, 2014.

[37] W. B. Helgeson and D. P. Birnie, "Assembly line balancing using the ranked positional weight technique," *Journal of Industrial Engineering*, vol. 12, pp. 394–397, 1961.

[38] R. Ruiz and T. Stützle, "A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem," *European Journal of Operational Research*, vol. 177, no. 3, pp. 2033–2049, 2007.

[39] M. Nawaz, E. E. Enscore Jr., and I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," *Omega*, vol. 11, no. 1, pp. 91–95, 1983.

[40] D. C. Montgomery, *Design and Analysis of Experiments*, John Wiley & Sons, New York, NY, USA, 5th edition, 2000.

[41] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.