Journal of Cleaner Production 135 (2016) 508–522

Contents lists available at ScienceDirect

Journal of Cleaner Production

journal homepage: www.elsevier.com/locate/jclepro

Minimizing energy consumption and cycle time in two-sided robotic assembly line systems using restarted simulated annealing algorithm

Zixiang Li, Qiuhua Tang[*], LiPing Zhang

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

A R T I C L E I N F O

Article history:
Received 12 January 2016
Received in revised form
27 May 2016
Accepted 21 June 2016
Available online 23 June 2016

Keywords:
Assembly line balancing
Two-sided robotic assembly line
Energy consumption
Simulated annealing
Pareto-optimal set

A B S T R A C T

Two-sided assembly lines are usually utilized to produce large-sized high-volume products. Recently robots are widely utilized in this line to replace the manual handling and manage the allocated tasks. For a robotic assembly line, the energy consumption is a major expense and the increased energy cost draws much more attentions from manufacturing enterprises. To the best knowledge of the authors, there is no research reported on the energy consumption of two-sided robotic assembly line. This paper presents a new mixed-integer programming model to minimize the energy consumption and cycle time simultaneously. A restarted simulated annealing algorithm is developed to deal with the complexity of the model, which utilizes new local search with three neighbor structures and restart phase based on the crowding distance assignment procedure to obtain well-spread Pareto-optimal set. Testing cases are designed to measure the performance of the proposed method and the restarted simulated annealing algorithm is compared with the fast elitist non-dominated sorting genetic algorithm. The computational results demonstrate that the proposed model is useful to reduce the total energy consumption and the restarted simulated annealing algorithm outperforms the non-dominated sorting genetic algorithm in both convergence and spread criteria.

1. Introduction

Two-sided assembly lines are widely utilized to assemble cars, trucks, automobiles and buses. This line differs from the traditional one-sided assembly line. In two-sided lines, both left and right sides are utilized in parallel. Some tasks are preferred to be operated on the left side (L-type) or right side (R-type) while others should be operated on either side (E-type). In the face of a competitive market and diverse customers' demands, the demand of flexible production becomes urgent. Robots are the essential parts to increase flexibility by being programmed to perform different tasks, and thus robots are allocated to the stations to replace the manual resource (Gao et al., 2009). Robots can perform the tasks continually without worries of fatigue and the assembly lines with robots are called robotic assembly lines.

A layout of two-sided robotic assembly line is depicted in Fig. 1. In this line, each mated-station is composed with two facing stations, and each station is allocated with a robot to operate different tasks. Two-sided assembly lines have several advantages over the traditional one-sided assembly lines, such as a shorter line length, less material handling, and reduced cost of tools (Bartholdi, 1993). To balance this line, two-sided robotic assembly line balancing (TRALB) problems are designed. TRALB problems are divided into two types. Type I TRALB problem deals with minimizing the number of stations with a given cycle time whereas type II TRALB minimizes the cycle time on a set of mated-stations (Kim et al., 2009; Purnomo et al., 2013).

Robotic assembly lines have several advantages over manual assembly lines, such as high productivity, manufacturing flexibility, less skilled labor and good quality of products (Levitin et al., 2006). In a robotic assembly line, the energy consumption is a major expense, and the increase of the energy price makes it non-ignorable. Fysikopoulos et al. (2012) indicate that the energy cost during a car manufacturing process is about 9–12% of the total manufacturing cost, and 20% reduction in energy consumption results in about 2–2.4% reduction in the final manufacturing cost.

[*] Corresponding author.
E-mail address: tangqiuhua@wust.edu.cn (Q. Tang).

http://dx.doi.org/10.1016/j.jclepro.2016.06.131
0959-6526/© 2016 Elsevier Ltd. All rights reserved.

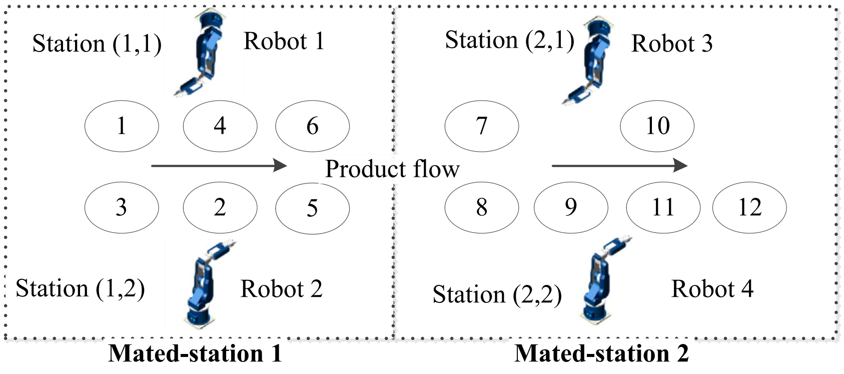

Fig. 1. The layout of two-sided robotic assembly line.

Reduced usage of energy can also keep the industries competitive and reduce pollution.

However, the research on the reduction of energy consumption in the assembly line systems is limited, and they are only related to one-sided or U-type assembly lines (Nilakantan et al., 2015a,b, 2016). To the author's best knowledge, there is no research reported where energy consumption in a two-sided robotic assembly line is considered. In addition, the energy consumption and the balance of the assembly line are not positively correlated strictly and they may be conflicted as proven in the research of Nilakantan et al. (2015a), but no multi-objective optimization algorithm is applied to obtain a Pareto-optimal set. Therefore this paper focuses on the two-sided robotic assembly line with the objectives of minimizing the energy consumption and cycle time simultaneously. Three main contributions of this research are:

(1) A new type II two-sided robotic assembly line with the objective of minimizing the energy consumption is defined and new benchmark problems are generated.
(2) A new mixed-integer programming model is developed to minimize the energy consumption and cycle time simultaneously. Nilakantan et al. (2015a) proposed a model to minimize the maximum of the energy consumptions on all stations in a straight robotic assembly line. In this paper, the sum of energy consumption on all stations is minimized. This new objective is more practical when there is no limitation on the peak energy consumption of the stations.
(3) A restarted simulated annealing (SA) algorithm as a metaheuristic method is developed to solve the multi-objective type II two-sided robotic assembly line balancing (TRALB-II) problem. New local search and a restart phase are put forward to enhance the performance of the SA algorithm. The SA is selected due to its simplicity and effectiveness, which are proved in literature (Arkat et al., 2007; Khorasanian et al., 2013; Özcan and Toklu, 2009b; Özcan, 2010).

The remainder of this paper is organized as follows. Section 2 presents the literature review, and the mathematical model is described in Section 3. Section 4 provides the encoding, decoding and the proposed SA algorithm. Section 5 gives the computational results and the findings of this research are concluded in Section 6.

2. Literature review

The literature review in this part consists of three categories: two-sided assembly line balancing, robotic assembly line balancing and the energy consumption in production.

Since the two-sided assembly line balancing (TALB) problem is first presented by Bartholdi (1993), an increasing number of methods are developed, which can be divided into exact methods, heuristic methods and metaheuristic methods. For exact methods, Hu et al. (2008) presented a station-oriented enumerative algorithm, Wu et al. (2008) and Hu et al. (2010) developed branch-and-bound algorithms. For heuristic methods, Lee et al. (2001) proposed a group assignment procedure and Özcan and Toklu (2010) developed a heuristic approach based on sequencing operations for assembly lines. Among the metaheuristic methods, Kim et al. (2000) developed a genetic algorithm, Özcan and Toklu (2009a) utilized a tabu search algorithm, Özbakir and Tapkan (2011) proposed a bee algorithm, Khorasanian et al. (2013), Özcan and Toklu (2009b) and Özcan (2010) utilized simulated annealing algorithms to solve two-sided balancing problems with different objectives.

Faced with diverse customers' demands, the concept of robotic assembly line draws growing attentions in recent days. In this line, robots rather than human beings assemble the products on each station, which was first come up by Rubinovitz and Bukchin (1991). Later, Rubinovitz et al. (1993) extended a branch-and-bound algorithm based heuristic approach to design and balance the robotic assembly line. Tsai and Yao (1993) utilized a heuristic approach for series-type robotic assembly line and Kim and Park (1995) developed a cutting plane algorithm for the robotic assembly line balancing with the objective of minimizing the total number of robot cells. Levitin et al. (2006) considered the type II robotic assembly line balancing (RALB-II) problem and they proposed a genetic algorithm. Gao et al. (2009) also considered the RALB-II problem and a genetic algorithm hybridized with local search was utilized to solve it. Yoosefelahi et al. (2012) proposed an evolution strategies algorithm for a multi-objective RALB-II. More recently, Aghajani et al. (2014) balanced the robotic mixed-model two-sided assembly line with robot setup times. Among all the above researches, only the paper by Aghajani et al. (2014) dealt with the two-sided robotic assembly line.

With regard to the energy consumption in manufacturing systems, the research is limited (Dai et al., 2013) and the research on energy consumption on an assembly line is very minimal. Fysikopoulos et al. (2012) presented a study of energy consumption in an automotive assembly and they showed that modeling an assembly by including energy considerations can save energy and cost. He et al. (2012) provided modeling method of task-oriented energy consumption for machining manufacturing system and they utilized SIMULINK simulation environment to solve this problem. Nilakantan et al. (2015a) investigated the total energy consumption in straight robotic assembly line systems. They developed two models to minimize the cycle time and energy consumption respectively. Later, Nilakantan et al. (2015b) considered the energy consumption in U-shaped robotic assembly line with a bio-inspired algorithm, and then they proposed a differential evolution to minimize total energy consumption in U-shaped robotic assembly line (Nilakantan et al., 2016).

From the literature, it can be seen that all the works reported are for straight and U-shaped assembly lines with the objective of minimizing energy consumption. Most of the researchers focused optimization algorithms with one objective. It is observed that no papers dealt with energy consumption in two-sided robotic assembly line with multi-objective optimization algorithm. Hence this paper focuses on proposing multi-objective optimization algorithm for two-sided robotic assembly line and this is very relevant in this field of study.

3. Mathematical model for TRALB problem

In two-sided robotic assembly lines, tasks are allocated to each station satisfying the precedence constraints and direction constraints. A set of robots are assigned to stations to perform the allocated tasks. Before introducing the mathematical model, some basic assumptions considered in this paper are provided as follows:

1) Power consumption of each robot is assumed and energy consumption is computed with the power consumption of each robot.
2) One type of product is assembled.
3) The operation time of a task depends on the type of the assigned robot and it is deterministic.
4) Each station is allocated with one robot and the number of stations is equal to the number of available robots.
5) Each task can be performed by any robot and each robot can be allocated to any station.
6) Setup times between tasks are ignored.
7) No work-in-process inventory is considered.
8) Planning horizon and maintenance operations are not considered in this model.

3.1. Notations

Indices:

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

Parameters:

Nt: Number of tasks.
Nm: Number of mated-stations.
Nr: Number of robots.
I: Set of tasks, $\mathrm{I} = \{1,2,\cdots,i,\cdots,Nt\}$.
J: Set of mated-stations, $\mathrm{J} = \{1,2,\cdots,j,\cdots,Nm\}$.
R: Set of robot types, $R = \{1,2,\cdots,r,\cdots,Nr\}$.
t_{ir}: Processing time of task i by robot r.
PC_r: Operation power consumption of the robot r per time unit.
SPC_r: Standby power consumption of the robot r per time unit.
AL: Set of tasks with left direction, $\mathrm{AL} \subseteq \mathrm{I}$.
AR: Set of tasks with right direction, $\mathrm{AR} \subseteq \mathrm{I}$.
AE: Set of tasks with either direction, $\mathrm{AE} \subseteq \mathrm{I}$.
P_0: Set of tasks that have no immediate predecessors.
$\mathrm{P}(i)$: Set of immediate predecessors of the task i.
$\mathrm{P}_a(i)$: Set of all predecessors of the task i.
$\mathrm{S}_a(i)$: Set of successors of the task i.
$\mathrm{S}(i)$: Set of immediate successors of the task i.
ψ: Large positive number.
$\mathrm{C}(i)$: Set of tasks whose operation directions are opposite to that of task i; $\mathrm{C}(i) = \begin{cases} \mathrm{AL} & \text{if } i \in \mathrm{AR} \\ \mathrm{AR} & \text{if } i \in \mathrm{AL} \\ \varnothing & \text{if } i \in \mathrm{AE} \end{cases}$
$\mathrm{K}(i)$: Set of integers which indicate the preferred directions of the task i; $\mathrm{K}(i) = \begin{cases} \{1\} & \text{if } i \in \mathrm{AL} \\ \{2\} & \text{if } i \in \mathrm{AR} \\ \{1,2\} & \text{if } i \in \mathrm{AE} \end{cases}$
PC:Set of pairs of tasks and predetermined stations for positional constraint.
PZ:Set of pairs of tasks for positive zoning constraint.
NZ:Set of pairs of tasks for negative zoning constraint.
SC:Set of pair of tasks for synchronism constraint.

Decision variables:

CT: Cycle time.
TE: Total energy consumption.
x_{ijk}:1, if task i is assigned to mated-station j at side k; 0, otherwise.
y_{rjk}:1, if robot r is assigned to mated-station j at side k; 0, otherwise.
t_i^f: Finishing time of task i.
EC_{jk}: Energy consumption on k side of mated-station j.

Indicator variables:

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

3.2. Mathematical model

Based on the research by Kim et al. (2009) and Nilakantan et al. (2015a), a mathematical model for type II TRALB problem is developed with two objectives:minimizing the cycle time and the sum of energy consumptions on all stations.

$$\text{Min} \quad CT \tag{1}$$

$$\text{Min} \quad TE = \sum_{j=1}^{nm} \sum_{k=1}^{2} EC_{jk} \tag{2}$$

$$EC_{jk} = \sum_{r=1}^{nr} \sum_{i=1}^{nt} PC_r t_{ir} x_{ijk} y_{rjk} + \left(\sum_{r=1}^{nr} SPC_r \cdot y_{rjk} \right) \cdot \left(CT - \sum_{r=1}^{nr} \times \sum_{i=1}^{nt} t_{ir} x_{ijk} y_{rjk} \right) \tag{3}$$

$$\sum_{j \in \mathrm{J}} \sum_{k \in \mathrm{K}(i)} x_{ijk} = 1 \quad \forall i \in \mathrm{I} \tag{4}$$

$$\sum_{g \in \mathrm{J}} \sum_{k \in \mathrm{K}(h)} g \cdot x_{hgk} \le \sum_{j \in \mathrm{J}} \sum_{k \in \mathrm{K}(i)} j \cdot x_{ijk} \quad \forall i \in \mathrm{I} - \mathrm{P}_0, h \in \mathrm{P}(i) \tag{5}$$

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

$$t_i^f - t_h^f + \psi\left(1 - \sum_{k\in \mathrm{K}(h)} x_{hjk}\right) + \psi\left(1 - \sum_{k\in \mathrm{K}(i)} x_{ijk}\right) \ge \sum_{r=1}^{Nr} t_{ir} y_{rjk} \quad \forall i \in \mathrm{I} - \mathrm{P}_0, h \in \mathrm{P}(i), j \in \mathrm{J} \tag{7}$$

$$t_p^f - t_i^f + \psi\left(1 - x_{ijk}\right) + \psi\left(1 - x_{pjk}\right) + \psi(1 - z_{ip}) \ge \sum_{r=1}^{Nr} t_{pr} y_{rjk} \quad \forall i \in \mathrm{I}, p \in \{r | r \in \mathrm{I} - (\mathrm{P}_a(i) \cup \mathrm{S}_a(i) \cup \mathrm{C}(i)) \text{ and } i < r\}, j \in \mathrm{J}, k \in \mathrm{K}(i) \cap \mathrm{K}(p) \tag{8}$$

$$t_i^f - t_p^f + \psi\left(1 - x_{ijk}\right) + \psi\left(1 - x_{pjk}\right) + \psi \cdot z_{ip} \ge \sum_{r=1}^{Nr} t_{ir} y_{rjk} \quad \forall i \in \mathrm{I}, p \in \{r | r \in \mathrm{I} - (\mathrm{P}_a(i) \cup \mathrm{S}_a(i) \cup \mathrm{C}(i)) \text{ and } i < r\}, j \in \mathrm{J}, k \in \mathrm{K}(i) \cap \mathrm{K}(p) \tag{9}$$

$$t_i^f + \psi\left(1 - x_{ijk}\right) \ge \sum_{r=1}^{Nr} t_{ir} y_{rjk} \quad \forall i \in I, j \in J, k \in \mathrm{K}(i) \tag{10}$$

$$\sum_{r}^{Nr} y_{rjk} = 1 \quad \forall j \in J, k = 1, 2 \tag{11}$$

$$\sum_{j=1}^{nm} \sum_{k=1}^{2} y_{rjk} = 1 \quad \forall r \in R \tag{12}$$

$$x_{ijk} = 1 \quad \forall (i, (j, k)) \in \mathrm{PC}), k \in K(i) \tag{13}$$

$$x_{ijk} - x_{hjk} = 0 \quad \forall (i, h) \in \mathrm{PZ}, k \in \mathrm{K}(i) \cap \mathrm{K}(h) \tag{14}$$

$$\sum_{k\in \mathrm{K}(i)} x_{ijk} + \sum_{k\in \mathrm{K}(h)} x_{hjk} \le 1 \quad \forall (i, h) \in \mathrm{NZ} \tag{15}$$

$$x_{ijf} - x_{hjk} = 0 \quad \forall (i, h) \in \mathrm{SC}, k \in \mathrm{K}(h), f \in \mathrm{K}(i), k \ne f \tag{16}$$

$$t_i^f - t_i = t_h^f - t_h \quad \forall (i, h) \in \mathrm{SC} \tag{17}$$

Objective function (1) minimizes the cycle time and objective function (2) minimizes the sum of the power consumptions on all stations. Eq. (3) calculates the energy consumption on k side of mated-station j. The first part of Eq. (3) computes the operation power consumption in station (j,k) and the second part obtains the standby power consumption. Eq. (4) ensures that each task is allocated exactly to one side of a mated-station. Eq. (5) checks the precedence constraints of the tasks and Eq. (6) is the cycle time constraint. Eqs (7)–(9) control the sequence-dependent finishing time. For a pair of task i and task h, if task h is an immediate predecessor of task i on a same mated-station, then Eq. (7) becomes active and it is reduced to $t_i^f - t_h^f \ge \sum_{r=1}^{Nr} t_{ir} y_{rjk}$. If a pair of task i and task h has no precedence relations and they are allocated to one same station, then Eqs (8) and (9) become active. If task i is assigned earlier than task p, then Eq. (8) becomes active and it is reduced to $t_p^f - t_i^f \ge \sum_{r=1}^{Nr} t_{pr} y_{rjk}$. Otherwise, Eq. (9) becomes active and it is reduced to $t_i^f - t_p^f \ge \sum_{r=1}^{Nr} t_{ir} y_{rjk}$. Eq. (10) makes sure that the finishing time of task i is larger than or equal to the operation time of task i. Eq. (11) guarantees that only one robot is allocated to one station and Eq. (12) enforces that each robot can be assigned to only one station. Eq. (13) handles with positional constraint. Eqs (14) and (15) deal with the positive zoning constraint and the negative zoning constraint. Eqs (16) and (17) enforce that each pair of tasks in synchronism constraint are allocated to the opposite sides of a same mated-station with the same starting time. Detailed explanation of the multiple Eqs (13)–(17) refers to recent researches by Yuan et al. (2015) and Tang et al. (2015).

In this paper, the power consumption of a robot is classified into two modes, production mode and standby mode. The production mode corresponds to operation power consumption described with the first part of Eq. (3). The standby mode means the intervals between operations and it corresponds to the standby power consumption explained with the second part of Eq. (3). A detailed example is shown in Section 4.6.

4. Restarted simulated annealing algorithm

Simulated annealing (SA) algorithm is first designed by Kirkpatrick et al. (1983) to combinatorial optimization. Gradually, this algorithm becomes one of the most popular metaheuristics for real world problems due to easy utilization and high performance (Arkat et al., 2007). This simple and effective method has also been extended to two-sided assembly line (Khorasanian et al., 2013; Özcan and Toklu, 2009b; Özcan, 2010). SA is a local search method, and it escapes from being trapped into local optima by accepting a worse solution with a probability function. It starts with an initial solution, S, then a neighbor solution of the original solution, S', is generated. If the new objective $f(S')$ is smaller than the incumbent objective, then the new solution is accepted to replace the current one. Otherwise, the new solution is accepted with a probability of $\exp^{-(f(S')-f(S))/T}$. Where, T is a control parameter and it is reduced with the searching process proceeding.

In this study, a multi-objective restarted simulated annealing algorithm (RSA) is developed to optimize two objectives simultaneously. In order to obtain Pareto-optimal set, the proposed RSA algorithm utilizes new acceptance and a restart mechanism as a diversification strategy. The procedure of the restarted SA algorithm is outlined in Fig. 2. The procedure starts with generating a solution randomly for initialization using the encoding and decoding as explained in Section 4.1 and then a main loop circulates until the termination criterion is reached. The main loop consists of obtaining a solution with a neighbor structure, updating Pareto-optimal set, application of acceptance rule, restart mechanism and cooling schedule. The details of the RSA are explained in the following sections.

4.1. Encoding and decoding

To obtain an initial solution, the allocation of the robots and the assignment of tasks should be determined. This paper develops three integer vectors for the encoding of the TRALB-II: robot assignment vector, task assignment to mated-station vector and task sequence vector.

Robot assignment vector determines the assignment of the robots and it is a (1 × Nr) vector. Task assignment vector and task sequence vector together determine the assignment of tasks on the stations. Task assignment vector is a (1 × Nt) vector and each element is an integer between 1 and Nm. If the ith element is j, the task i is supposed to be assigned to mated-station j. Task assignment vector can determine the assignment of the tasks to mated-stations, but the assignment of tasks on each mated-station remains unsolved and task sequence vector is thus utilized. Task

```
Algorithm RSA for RTALB-II problem
Input: RTALB-II data, RSA parameters, dn (Number of moves before restart)
Output: Pareto-optimal set (POS)
Begin:
    Generate an initial solution S;
    d:= 0; ns:= 0;
    While (termination criterion is not satisfied) do
        While (ns<NS (Number of repetitions at each temperature level))
            Obtain a new solution S' with a neighbor structure;
            ns:=ns+1;
            If S' is not dominated by the solutions in POS,
                Update POS;
            Else
                d:= d+1;
                if (d>dn )
                    Select a new solution S" with the restart mechanism and $S \leftarrow S''$;
                Else
                    select one objective randomly (f) and $\Delta = f(S') - f(S)$;
                    If ($\Delta \le 0$)
                        $S \leftarrow S'$;
                    Else
                        $S \leftarrow S'$ with a probability of $\exp^{-\Delta/(T\times f(S))}$;
        Endwhile
        T:=αT;
    Endwhile
```

Fig. 2. The procedure of restarted simulated annealing algorithm.

sequence vector is also a (1 × *Nt*) vector, and it decides the sequence of tasks on each mated-station. With the task assignment and the sequence of tasks on each mated-station, the detail assignment of all the tasks on each mated-station can be determined based on a new decoding scheme explained as follows.

Step 1: Decide whether assignable task exists for either side.
Step 2: If all the tasks allocated to the current mated-station are assigned, end this procedure. Otherwise, go to Step 3.
Step 3: Obtain the assignable task set for both sides and go to Step 4.
Step 4: If only one assignable task set is not empty, select this side which has assignable tasks and go to Step 6. Otherwise, go to Step 5.
Step 5: Select the side with larger capacity as the current station. If the capacities of both sides are equal, select the left side by default. Then, go to Step 6.//Station selection mechanism
Step 6: Check whether the tasks which can be operated at the earliest start time of the current station exist in the corresponding assignable task set. If not, go to Step 7. Otherwise, delete the task which cannot be operated at the earliest start time of the current station.//Task selection mechanism
Step 7: Select the task which is in the first position of the task sequence vector from the assignable task set and go to Step 1.

The above decoding procedure can be regarded as an improved version of Kim et al. (2009) to solve the TRALB problem. Kim et al. (2009) utilize the task assignment vector to determine the assignment of the tasks to mated-stations and a heuristic method to determine the task sequence. Thus, the task sequence remains unchanged during the search process. However, different task sequence can result in different sequence-dependent idle times in two-sided assembly line and the optimization of task sequence can further improve the solutions. Therefore, the first possible improvement is applying the task sequence vector to further optimize task sequence. In addition, the decoding procedure proposed by Kim et al. (2009) first selects a task and then decides a side to allocate the selected task. This method ignores the reduction of sequence-dependent idle times in the decoding procedure, and thus the second possible improvement is developing new decoding scheme based on Khorasanian et al. (2013). The decoding method proposed here first utilizes the station selection mechanism to select a side with larger capacity to obtain the balanced workloads on the two sides and then proposes the task selection mechanism to select a task with the earliest start time for the reduction of the sequence-dependent idle times.

An example of 12-task problem (Kim et al., 2000) is depicted in Fig. 3. In Fig. 3, the robot assignment can be obtained directly with the robot assignment vector. The task assignment determines the assignment of tasks to mated-stations. Then the task sequence determines the task assignment on each mated-station with the proposed decoding scheme.

4.2. Neighbor structure

Since three vectors in Section 4.1 are involved, different neighbor structures are developed to obtain neighbor solutions. This paper utilizes three different methods and each responds an integer vector. All the neighbor structures are depicted in Fig. 4 with an example.

1) Neighbor structure for robot assignment: A neighbor solution is obtained by swap operator. In swap operation, two different robots are selected and their positions are exchanged.
2) Neighbor structure for task assignment to mated-station: A neighbor solution is obtained by swap operator and mutation operator and they are selected randomly. In swap operator, two tasks assigned to different mated-stations are randomly

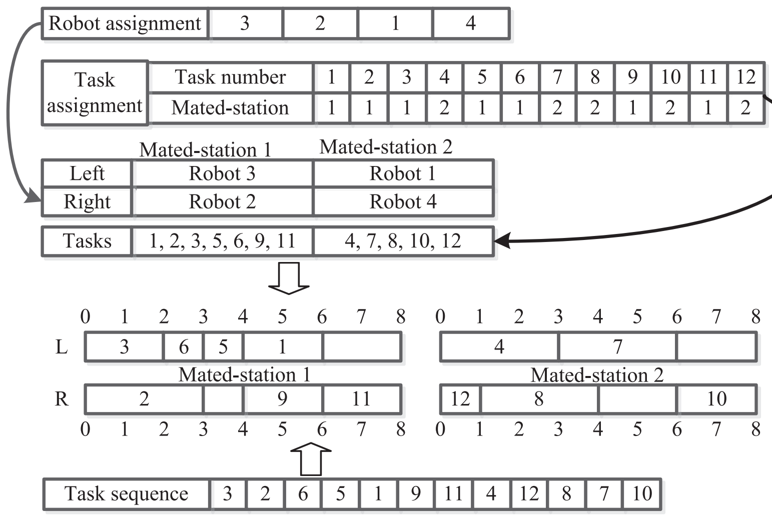

Fig. 3. Solution representation for RALB-II problem.

selected and their corresponding mated-stations are exchanged. In mutation operator, a task is randomly selected and its original mated-station is replaced with a different mated-station.

3) Neighbor structure for task sequence: A neighbor solution is obtained by insert operator. A task is selected randomly, and it is removed out and inserted into a different position.

This paper proposes a simple method for selecting the neighbor structure. For each vector, a random number between [0, 1] is generated. If the random number is smaller than 1/3, then the corresponding vector is modified. If all random numbers are larger than 1/3, the first neighbor structure is selected by default. This method allows only one vector to be modified but also permits three vectors to be modified simultaneously. It is necessary to notice that the modification of the robot assignment and task sequence does not affect the feasibility of the solutions, but the modification of task assignment to mated-station vector may result in infeasible solutions. Hence a repair mechanism is necessary and this paper proposes a simple repair mechanism. The assignment of the tasks are checked and modified as follows: if a successor of one task is allocated to the former mated-station, exchange the mated-stations of the two tasks; if the predecessor of one task is allocated to the latter mated-station, exchange the mated-stations of these two tasks. After executing the repair mechanism, the successor of one task is only assigned to the same or latter mated-station. Then, the decoding procedure is applied to obtain a feasible solution satisfying the precedence constraints.

4.3. Pareto-optimal set update and acceptance criterion

After applying the neighbor structures, a neighbor solution S' is obtained, and then this solution is checked whether it is a non-dominated solution and whether it can be accepted to replace the current one. If the new solution is not dominated by any solution in the Pareto-optimal set, the new solution is added into the current Pareto-optimal set and original solutions in the Pareto–optimal set are removed out if they are dominated by this new solution. After updating the Pareto-optimal set, the new solution replaces the incumbent, i.e. $S \leftarrow S'$.

If the new solution is dominated by the solutions in the Pareto-optimal set, a simple multinomial probability mass function (Kulturel-Konak et al., 2006) is applied to judge whether the new one can be accepted. The multinomial probability mass function was previously developed for multi-objective tabu search by Kulturel-Konak et al. (2006). This method is simple and easy to implement, and it is proved to be competitive with the methods of weighting and scaling of each objective. In this method, the multinomial probability mass function selects an objective to compare the new solution with the current solution in each iteration. Note that two objectives are considered in this study and thus the selection probability of each objective in the multinomial probability mass function is equal to 0.5. Supposed that one objective function $f(*)$ is selected, the change in this objective function $(\Delta, \Delta = f(S') - f(S))$ is obtained. If $\varDelta \leq 0$, new solution S' is accepted to replace the incumbent one. Otherwise, the new

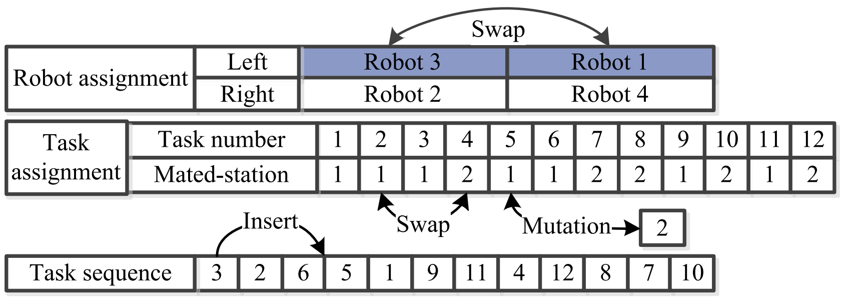

Fig. 4. An example of neighbor structures.

solution is accepted with a probability of $\exp^{-\Delta/(T\times f(S))}$. Notice that the original accept probability $\exp^{-\Delta/T}$ is replaced with $\exp^{-\Delta/(T\times f(S))}$, since the magnitudes of the two objectives are different. This modification makes sure that the two objectives can share a same *T*.

4.4. Restart mechanism

This paper proposes a restart mechanism as a diversification strategy to enhance the capacity of escaping from local optima and obtaining well-spread non-dominated solutions. If a new non-dominated solution cannot be found for a number of *dn* moves, a solution in the Pareto-optimal set is selected. The parameter *dn* is called the number of moves before restart in the following sections. Instead of selecting a solution randomly, this paper proposes a restart mechanism based on the crowding distance operator applied by Deb et al. (2002). The idea behind the crowding distance operator is to select the most isolated solution of the Pareto frontier. If solitary solutions are further explored, the gaps in the objective solution space can be reduced and a better spread of the Pareto-optimal set can be obtained. The main draw of the original crowding distance operator is that it may explore the extreme solutions with the maximum objective or minimum objective again and again. To overcome this drawback, two modifications are made. 1) The distances of the extreme solutions are set equal to 1 rather than infinity (∞). This ensures that the distances of the extreme solutions are larger than other solutions. 2) The selection counter is applied to count the selection times for each solution and the distance is reduced with the number of being selected increasing. After obtaining all the distances for all the solutions in the Pareto-optimal set, the solution with the largest distance is selected. The proposed restart mechanism procedure is depicted in Fig. 5.

4.5. Cooling schedule

In this paper, the temperature *T* is updated with $T=\alpha T$ after executing the local search for *NS* times. Parameter α is the cooling rate and it is a positive number between 0 and 1. *T* decreases during the process of searching which results in a lower probability of accepting worse solutions. This updating method for temperature is adopted from Khorasanian et al. (2013).

In the restarted simulated annealing algorithm, four parameters need to be determined, namely the initial temperature (T_0), the cooling rate (α) and the number of repetitions allowed at each temperature level (*NS*) and number of moves before restart (*dn*). The values of these parameters are determined as a result of calibration experiments which is described in Section 5.

4.6. Numerical example

This section illustrates the process of obtaining a solution for a problem with 12 tasks and 6 robots. The operation times of 12 tasks by six robots are shown in Table 1. Supposed that the operation power consumptions of six robots are 0.25, 0.4, 0.3, 0.35, 0.3 and 0.4 per time unit respectively, the energy consumptions for 12 tasks by six robots can be calculated with $PC_r\times t_{ir}$ and they are shown in Table 2. In this paper, the power consumption of a robot composes of operation power consumption and standby power consumption. The standby power is the power consumed during the intervals of operating tasks and the standby power of robot per time unit is considered to be 10% of the operation power per time unit (Nilakantan et al., 2015a).

An example of total energy consumption is depicted in Fig. 6. Let us focus on the robot 4 which operates task 5 and 9. From Table 1, the operation times of tasks 5 and 9 by robot 4 are 1 and 2 respectively, and then operation power consumption can be calculated with 3×0.35. The idle time behind task 9 is the standby time and thus the standby power can be calculated with $1\times0.35\times0.1$, where 0.1 is obtained since the standby power of robot per time unit is considered to be 10% of the operation power per time unit. The total power consumption can be achieved by summing the power consumption on all the stations, and they are calculated with $1.2+0{+}1+0+1.6+0+1.05+0.035+1.2+0+1.2+0.04=7.325$.

5. Computational results and discussion

In order to test the performance of the proposed restarted simulated annealing (RSA) algorithm, a set of benchmarks are generated and described in the following subsection. The RSA is compared with the famous fast elitist non-dominated sorting genetic algorithm (NSGA-II) (Deb et al., 2002). The parameters of all the algorithms are calibrated carefully with statistical technique. Finally, the results by different algorithms are compared with each other. All the algorithms coded in C++ programming language are running all a set of personal computers with Microsoft Visual Studio 2012. All the computers are equipped with Intel(R) Core2(TM) CPU 2.33 GHZ and 3.036 GB RMA.

5.1. Benchmark of instances

A set of benchmark problems are generated including four small-size problems, P9, P12, P16 and P24, and three large-size problems, P65, P148 and P205. The proposed benchmark problems are taken from real assembly lines in different industries. For example, details of the P65 and P205 are obtained from truck assembly lines of an automobile company. Precedence diagrams and the preferred directions of tasks are taken from the literature directly. To be specific, P9, P12 and P24 are taken from Kim et al. (2000), P16, P65 and P205 are taken from Lee et al. (2001), and the P148 is from Bartholdi (1993). The operation times of task $i(i\in I)$ by robot *r* are randomly generated between $[t_i\times0.8,t_i\times1.2]$ and t_i is the original operation time published. For space limitation, the detail operation times of task $i(i\in I)$ on robot *r* are not exhibited, but they are available upon the request. The operation power consumptions of robots per time unit are randomly generated and the standby power consumption of the robot *r* per time unit is equal to 10% of the operation power consumption of robot *r*. The operation power consumptions of robots are shown in Table 3. In Table 3, *Nm* is the number of mated-stations and R* is the abbreviation of robot *.

5.2. Evaluation methodologies

To evaluate the performance of methods, three metrics are utilized: the ratio of non-dominated solutions, the convergence of the Pareto-optimal solution and the spread metric. These methods have been applied to evaluate multi-objective optimization methods on two-sided assembly line balancing problems by Chutima and Chimklai (2012) and http://www.sciencedirect.com/science/article/pii/S0360835214002186Purnomo and Wee (2014).

For each Pareto-optimal set P_s, the ratio of non-dominated solutions ($RP(P_s)$) can be formulated with Eq. (18), where *P* is the union of the set. Let P_1 and P_2 be two set of Pareto-optimal solutions, then$P=P_1\cup P_2$.$|*|$ is the number of solutions in set*. The higher the ratio $RP(P_s)$ is, the better the Pareto-optimal set is.

The convergence of the Pareto-optimal set ($CP(P_s)$) indicates the difference between the obtained solution set and the approximate true-Pareto set (*TP*). It is computed with Eq. (19) and Eq. (20). dt_l refers the minimum Euclidean distance between *l*th solution (x_l)

```
Restart mechanism based on modified crowding distance operator
% POS means the Pareto-optimal set, Obj and dist means objective and distance,
  SelectCounter means the number of being selected
Setsize:= |POS|;
For all objectives m
    Sort POS using the current objective m
    $f_m^{\max}$ :=maximum value for the objective m
    $f_m^{\min}$ :=minimum value for the objective m
    for all i:=2,⋯,Setsize-1
        $POS[i]_{dist} := POS[i]_{dist} + \left(POS[i+1]_{Obj_i} - POS[i-1]_{Obj_i}\right) / \left(f_m^{\max} - f_m^{\min}\right)$;
    Endfor
    $POS[1]_{dist} := POS[1]_{dist} + 1/m$ ;
    $POS[Setsize]_{dist} := POS[Setsize]_{dist} + 1/m$ ;
Endfor
For all i:=1,⋯,Setsize
    $POS[i]_{dist} := POS[i]_{dist} / POS[i]_{SelectCounter}$ ;
Endfor
Select the solution with largest distance to replace the incumbent one
```

Fig. 5. Restart mechanism procedure.

Table 1
Operation times for 11 tasks by 6 robots.

Task	Robot 1	Robot 2	Robot 3	Robot 4	Robot 5	Robot 6
1	2	2	2	2	2	2
2	2	4	3	3	3	4
3	2	2	2	2	2	2
4	3	3	2	3	3	2
5	1	1	1	1	1	1
6	1	1	1	1	1	1
7	3	3	3	4	4	3
8	2	2	4	3	3	4
9	2	2	2	2	2	2
10	2	2	2	2	2	2
11	2	2	2	2	2	2
12	1	1	1	1	1	1

Table 2
Operation energy consumptions per time unit for 11 tasks by 6 robots.

Task	Robot 1	Robot 2	Robot 3	Robot 4	Robot 5	Robot 6
1	0.5	0.8	0.6	0.7	0.6	0.8
2	0.5	1.6	0.9	1.05	0.9	1.6
3	0.5	0.8	0.6	0.7	0.6	0.8
4	0.75	1.2	0.6	1.05	0.9	0.8
5	0.25	0.4	0.3	0.35	0.3	0.4
6	0.25	0.4	0.3	0.35	0.3	0.4
7	0.75	1.2	0.9	1.4	1.2	1.2
8	0.5	0.8	1.2	1.05	0.9	1.6
9	0.5	0.8	0.6	0.7	0.6	0.8
10	0.5	0.8	0.6	0.7	0.6	0.8
11	0.5	0.8	0.6	0.7	0.6	0.8
12	0.25	0.4	0.3	0.35	0.3	0.4

and solution y in the true-Pareto frontier TP, $f_m^{\max}$ and $f_m^{\min}$ refers the maximum and minimum values of the mth objective in the true-Pareto set. A low value of $CP(P_s)$ indicates a better convergence to the true-Pareto set.

The spread metric ($SP(P_s)$) measures the distribution of the solutions in the Pareto front and it is calculated with Eq. (21) and Eq. (22). d_{f1} and d_{f2} are the Euclidean distance between the extreme solutions in set P_s and the boundary solutions in true-Pareto set.sd_l ($l = 1,2,\cdots,|P_s|-1$) is the Euclidean distance between two consecutive solutions and $\bar{sd}$ the average Euclidean distance. A smaller value of $SP(P_s)$ means a better spread and a bad distribution may exceed 1.0.

$$RP(P_s) = |P_s - \{X \in P_s | \exists Y \in P : Y \succ X\}| / |P_s| \tag{18}$$

$$CP(P_s) = \sum_{l=1}^{|P_s|} dt_l \Big/ |P_s| \tag{19}$$

$$dt_l = \min_{n=1}^{|TP|} \sqrt{\sum_{m=1}^{2} \left[\frac{f_m(x_l) - f_m(y_n)}{f_m^{\max} - f_m^{\min}} \right]^2} \tag{20}$$

$$SP(P_s) = \frac{sd_{f1} + sd_{f2} + \sum_{l=1}^{|P_i|-1} \left| sd_l - \bar{sd} \right|}{sd_{f1} + sd_{f2} + (|P_i| - 1)\bar{sd}} \tag{21}$$

$$sd_l = \sqrt{\sum_{m=1}^{2} \left[\frac{f_m(x_l) - f_m(x_{l+1})}{f_m^{\max} - f_m^{\min}} \right]^2} \tag{22}$$

5.3. Parameter calibration of the proposed algorithm

In this part, calibration experiments for determining the parameters of RSA and NSGA-II are explained in detail. The design of experiment technique is applied, and the calibration process of RSA is represented in detail and four factors in RSA to be tested are as follows:

(1) Initial temperature T_0 is tested at two levels: 0.5 and 1.
(2)The cooling rate is tested at three levels: 0.9, 0.95 and 0.98.
(3)The number of iterations or NS before a temperature change is tested at three levels: 500, 1000 and 2000.
(4) The number of moves or dn before restart is tested at three levels: 100, 200 and 300. Parameter dn is a new parameter for RSA which determines when to perform the restart mechanism. And the restart mechanism is carried out only when the set of Pareto-optimal solutions has not been updated during the last number of dn moves.

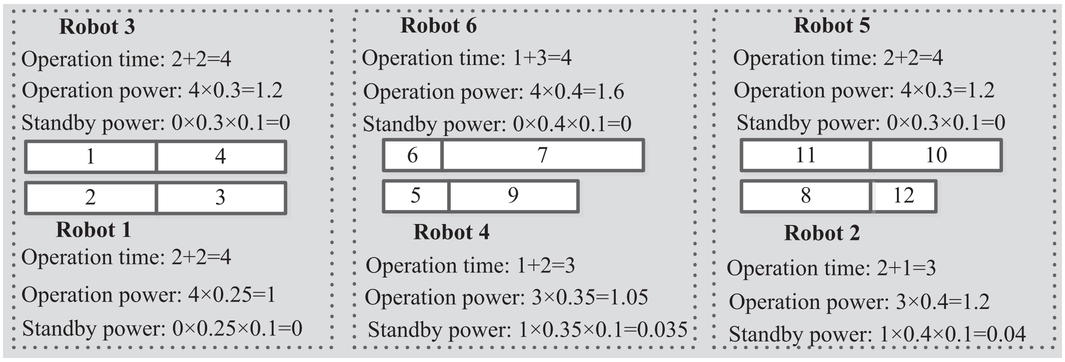
Robot 3
Operation time: 2+2=4
Operation power: 4×0.3=1.2
Standby power: 0×0.3×0.1=0
1
4
2
3
Robot 1
Operation time: 2+2=4
Operation power: 4×0.25=1
Standby power: 0×0.25×0.1=0
Robot 6
Operation time: 1+3=4
Operation power: 4×0.4=1.6
Standby power: 0×0.4×0.1=0
6
7
5
9
Robot 4
Operation time: 1+2=3
Operation power: 3×0.35=1.05
Standby power: 1×0.35×0.1=0.035
Robot 5
Operation time: 2+2=4
Operation power: 4×0.3=1.2
Standby power: 0×0.3×0.1=0
11
10
8
12
Robot 2
Operation time: 2+1=3
Operation power: 3×0.4=1.2
Standby power: 1×0.4×0.1=0.04

Fig. 6. Energy consumption for P12 with six robots.

Table 3
Power consumptions of robots per time unit.

Problem	Nm	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14
P9	2	0.25	0.4	0.3	0.35	–	–	–	–	–	–	–	–	–	–
P9	3	0.3	0.35	0.3	0.4	0.25	0.4	–	–	–	–	–	–	–	–
P12	2	0.25	0.4	0.3	0.35	–	–	–	–	–	–	–	–	–	–
P12	3	0.25	0.4	0.3	0.35	0.3	0.4	–	–	–	–	–	–	–	–
P12	4	0.3	0.35	0.3	0.4	0.35	0.3	0.3	0.4	–	–	–	–	–	–
P12	5	0.3	0.35	0.3	0.4	0.35	0.3	0.3	0.4	0.3	0.4	–	–	–	–
P16	2	0.25	0.4	0.3	0.3	–	–	–	–	–	–	–	–	–	–
P16	3	0.3	0.4	0.4	0.3	0.3	0.35	–	–	–	–	–	–	–	–
P16	4	0.2	0.3	0.25	0.4	0.35	0.4	0.3	0.3	–	–	–	–	–	–
P16	5	0.3	0.2	0.3	0.25	0.25	0.35	0.4	0.4	0.3	0.25				
P24	2	0.5	0.8	0.8	0.9	–	–	–	–	–	–	–	–	–	–
P24	3	0.7	0.5	0.8	0.9	0.5	0.6	–	–	–	–	–	–	–	–
P24	4	0.8	0.9	0.5	0.9	0.7	0.5	0.8	0.6	–	–	–	–	–	–
P24	5	0.9	0.8	0.6	0.8	0.9	0.5	0.5	0.7	0.6	0.9	–	–	–	–
P65	4	1.4	1.7	1.5	1.7	1.3	1.5	1.4	1.6	–	–	–	–	–	–
P65	5	1.2	1.6	1.1	1.2	1.5	1.7	1.4	1.8	1.7	1.6	–	–	–	–
P65	6	1.5	1.7	1.1	1.4	1.1	1.2	1.5	1.7	1.4	1.8	1.7	1.3	–	–
P65	7	1.8	1.4	1.5	1.7	1.1	1.4	1.5	1.8	1.3	1.6	1.1	1.7	1.2	1.4
P65	8	1.5	1.8	1.1	1.7	1.3	1.6	1.2	1.6	1.8	1.3	1.7	1.3	1.6	1.3
P148	4	1.8	2.4	1.6	1.5	1.7	1.5	1.7	2.1	–	–	–	–	–	–
P148	5	2.1	2.2	1.8	2.4	1.6	1.5	2	2.3	2.4	2.5	–	–	–	–
P148	6	2.4	1.5	2.4	1.9	1.6	2	2.1	2	2.2	2.3	2.1	1.5	–	–
P148	7	2.2	2.1	2	2.4	1.9	2.4	1.7	1.5	1.7	2.1	2.2	1	1.7	2.3
P148	8	2.4	2.2	1.8	1.5	2.1	1.9	2.4	1.9	1.6	2	2.1	2	2.2	2.4
P148	9	2.2	1.6	1.7	2.4	1.7	1.7	2.4	2.1	2	2.2	2.3	1.9	2.1	1.8
P148	10	2.4	1.6	1.5	2.4	1.5	2.4	1.9	1.6	2	2.1	2	2.2	2.3	2.1
P148	11	1.6	2	2.1	2	2.2	1.7	2.1	2.1	2	2.2	2.3	1.9	2.1	1.9
P148	12	2.3	2.2	1.6	1.7	2.4	2.3	1.6	2	2.2	1.7	2.1	1.7	1.6	1.7
P205	4	1.8	2	2.9	1.8	2.4	2.1	2.4	2.2	–	–	–	–	–	–
P205	5	2	2.8	2	2.7	2.3	2.5	2.5	2.6	2.1	1.8	–	–	–	–
P205	6	2.1	2.9	2.5	2.8	1.9	1.9	2.4	2.3	2.5	2.2	2	2.5	–	–
P205	7	2.4	2	2.3	2.7	2.3	2.6	2.4	2.5	2.4	2.1	2.3	1.9	2	2.3
P205	8	2.2	2.5	2.1	2.7	1.9	2	2.3	2.9	2.2	2.4	2.2	2.7	1.8	2.9
P205	9	2.8	2.2	2.4	2.3	2.4	2.6	2.4	1.9	2	2.2	2.3	1.9	2.6	3
P205	10	2	2.9	1.8	2.4	2.3	2.9	2.2	2.1	2.2	2.9	2.5	2.8	1.9	1.9
P205	11	1.8	1.8	1.9	2.4	2.9	2.2	2.4	2.2	2.8	2	2.7	2.9	2.7	2.2
P205	12	2.2	2.8	2.7	2.5	2.3	1.9	2	2.3	2.9	2.3	2.6	2.4	2.5	2.4
P205	13	2.7	2.3	2.6	2.4	2.5	1.8	1.9	2.4	2.1	2.2	2	2.1	2.9	2.2
P205	14	2.3	1.9	2.3	2.1	2.2	2	2.1	2.9	2.2	2.8	2.7	2.5	2.1	2.8
Problem	Nm	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26	R27	R28
P65	8	1.1	1.5	–	–	–	–	–	–	–	–	–	–	–	–
P148	8	1.7	1.5	–	–	–	–	–	–	–	–	–	–	–	–
P148	9	1.8	1.5	2.1	1.9	–	–	–	–	–	–	–	–	–	–
P148	10	1.5	1.9	2.1	1.8	2.1	1.7	–	–	–	–	–	–	–	–
P148	11	2.4	1.9	1.6	2.2	2.3	2.1	1.5	1.6	–	–	–	–	–	–
P148	12	1.7	2.4	2.2	1.8	1.5	2.1	1.9	1.8	2.4	1.7	–	–	–	–
P205	8	2.2	1.9	–	–	–	–	–	–	–	–	–	–	–	–
P205	9	2.7	2.5	2.5	2.5					–	–	–	–	–	–
P205	10	1.9	2	2.3	2.9	2.2	2.4	–	–	–	–	–	–	–	–
P205	11	2.8	2.1	2.7	2.6	2.7	2.9	2.7	2.2	–	–	–	–	–	–
P205	12	1.8	1.9	2.4	2.9	2.2	2.8	2.7	2.5	2	2.9	–	–	–	–
P205	13	2.8	2.7	2.5	2.1	2.5	2.6	2.1	1.8	2.1	2.9	2.5	2.5	–	–
P205	14	3	2.1	2.4	2.7	2.9	2.2	2.1	2.2	2.8	2.1	2.7	2.6	2.7	2.9

The full factorial experiment is applied and there are a total of 54 algorithm configurations. In this study, each algorithm configuration is tested on the largest case of P205 for 5 times. The algorithm terminates when the CPU time reaches $t = Nt \times N\,t \times \rho$ milliseconds, where Nt is the number of tasks and ρ is a parameter fixed to 10. Notice that we calibrate the parameters only with one largest case since the performance by different parameters on small-size problems or on different cases of a same problem is similar. We select the convergence of the Pareto-optimal solution as the response variable. Due to the great differential performances of algorithms, the normality is slightly violated after an initial analysis with multi-factor Analysis of Variance (ANOVA) (Montgomery, 2000). In this paper, the parametric ANOVA technique is utilized to analyze the multiple factors together though the normality is slightly violated, and the non-parameter Friedman test (Friedman, 1937) is also employed to strengthen the validity of the parameter calibration.

The ANOVA results are shown in Table 4, and it is observed that the levels of α, NS, dn and the interaction between α and ns are all statistically different with p-values less than 0.01. Since the four P-values are less than 0.05 and very close to zero, focus shift towards the F-ratio. The larger the F-ratio, the more effect this factor has over the convergence of the Pareto-optimal solution. The factor or interaction with largest F-ratio is analyzed at first with a means plot to demonstrate the difference graphically. Then the factor or interaction with second largest F-ratio is followed and this procedure terminates until all factors are determined. As observed in Table 4, the cooling rate has greatest F-ratio and three means of these different levels are plotted in Fig. 7. Clearly, both the AVOVA and Friedman Rank-based test proves that the cooling rate of 0.9 results in a more efficient RSA. For simplification, the remaining analysis is omitted. After calibration, the final configuration of RSA is shown in Table 5.

The parameters of NSGA-II are also determined by the same method. Two-point crossover operator is selected and two individuals are randomly selected. Since three vectors are involved in the encoding and decoding, one of them is randomly chosen and then the crossover operator is operated. And the mutation operator modifies one of the three vectors by utilizing one of the neighbor structures randomly. Apart from the crossover operator and the mutation operator, three more parameters are necessary to be calibrated: (1) population size at five levels: 40, 80, 120, 160 and 200, (2) crossover rate at five levels: 0.5, 0.6, 0.7, 0.8 and 0.9, (3) mutation rate at five levels: 0.5, 0.4, 0.3, 0.2 and 0.1. After calibration, the final parameters of NSGA-II are exhibited in Table 5.

5.4. Performance comparison among algorithms

This section first tests the performance of the proposed multi-objective model by comparing RSA with two other SA algorithms with one objective. Then the proposed RSA is compared with other two multi-objective SA algorithms and the well-known NSGA-II to test the performance of the proposed algorithm. The NSGA-II is modified to solve TRALB problems and NSGA-II is selected since it is one of the most proposed methods for multi-objective optimization.

To test the performance of the proposed model, the RSA with two objectives is compared with two SA algorithms with only one objective: SA-CT with the objective of minimizing the cycle time and SA-TE with the objective of minimizing the total energy consumption. Each case is solved for 10 times and two termination criteria of $nt \times nt \times 10$ milliseconds and $nt \times nt \times 20$ milliseconds are employed. Figs. 8–10 show the Pareto-optimal set by RSA, the ten solutions by SA-CT and ten solutions by SA-TE within ten times' running. Note that RSA can obtain many solutions within ten times' running, and we only show the Pareto-optimal set to have a better visualization. In Fig. 8(a), the best energy consumption by SA-CT is next to 6615, while the best energy consumption by RSA is only 6377. The energy consumption by SA-TE is 6420 and SA-TE obtains the worst results on the cycle time. These results suggest that the proposed model can reduce the energy consumption to a large degree. In Figs. 8–10, the RSA outperforms the SA-CT regarding to the energy consumption for all the cases, and the RSA outperforms the SA-TE regarding to the cycle times for all the cases. And most solutions by SA-CT and SA-TE are dominated by the Pareto front by RSA. The above results prove that the proposed multi-objective model can reduce the energy consumption effectively and utilization of multi-objective optimization is reasonable.

To test the performance of the RSA, the proposed RSA is compared with two other SA algorithms: SA1 and SA2. SA1 does not utilize the restart mechanism while SA2 select a solution randomly from the Pareto-optimal set in the restart mechanism. The performances of the three algorithms on P205 with 8 mated-stations are analyzed carefully. Fig. 11 shows the Pareto fronts with different

Table 4
ANOVA results for the convergence.

Source	Type III sum of squares	df	Mean square	F-ratio	P-value
Corrected Model	1374.336[a]	53	25.931	9.562	0
Intercept	11770.835	1	11770.835	4340.688	0
A:T_0	0.364	1	0.364	0.134	0.715
B: α	357.263	2	178.631	65.873	0.00
C: NS	343.507	2	171.754	63.337	0.00
D:dn	174.366	2	87.183	32.15	0.00
A * B	1.072	2	0.536	0.198	0.821
A * C	11.169	2	5.584	2.059	0.13
A * D	4.453	2	2.227	0.821	0.441
B * C	375.453	4	93.863	34.614	0.00
B * D	11.463	4	2.866	1.057	0.379
C * D	9.537	4	2.384	0.879	0.477
A * B*C	17.512	4	4.378	1.614	0.172
A * B*D	8.802	4	2.2	0.811	0.519
A * C*D	3.358	4	0.839	0.31	0.871
B * C*D	29.212	8	3.651	1.347	0.222
A * B*C * D	26.804	8	3.351	1.236	0.279
Error	585.737	216	2.712		
Total	13730.908	270			
Corrected Total	1960.072	269			

[a] R Squared = 0.701 (Adjusted R Squared = 0.628).

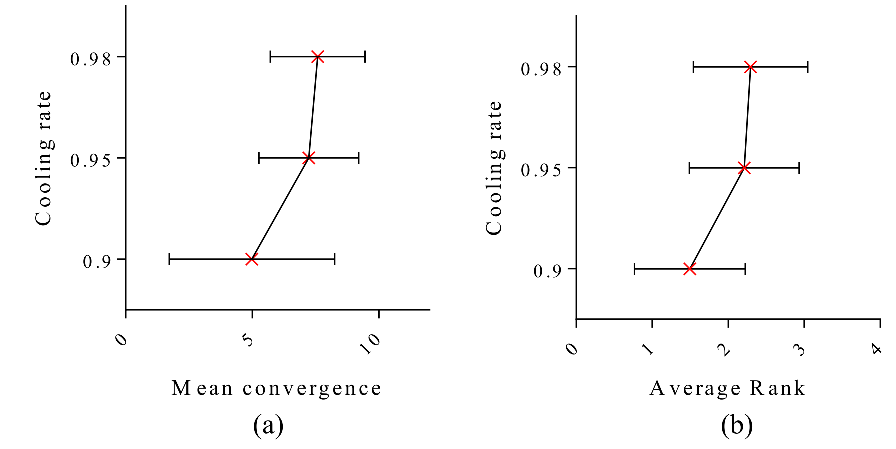

Fig. 7. Means plot and 95% Tukey confidence intervals for the AVOVA (a) and Friedman Rank-based tests (b).

Table 5
Parameters used for RSA and NSGA-II algorithms.

Parameter	Value
RSA algorithm	
Initial temperature	1.0
Cooling rate	0.9
Number of iterations before a temperature change	500
Number of moves before restart	100
NSGA-II algorithm	
Population size	160
Crossover rate	0.8
Mutation rate	0.2

The computational result comparison between NSGA-II and the proposed RSA is presented in Table 6. Each case is tested for ten times, and two termination criteria are also considered ($nt \times nt \times 10$ milliseconds and $nt \times nt \times 20$ milliseconds) resulting in a total of $39 \times 10 \times 2 = 780$ data samples. The average results are summarized in Table 6. Note that the approximate true-Pareto set is the Pareto frontier by running NSGA-II and RSA for twenty times under the termination criteria of $nt \times nt \times 30$ milliseconds.

It is observed that RSA outperforms NSGA-II for 39 cases on the ratio of non-dominated solutions under both two termination criteria. As for the convergence criterion, RSA outperforms NSGA-II

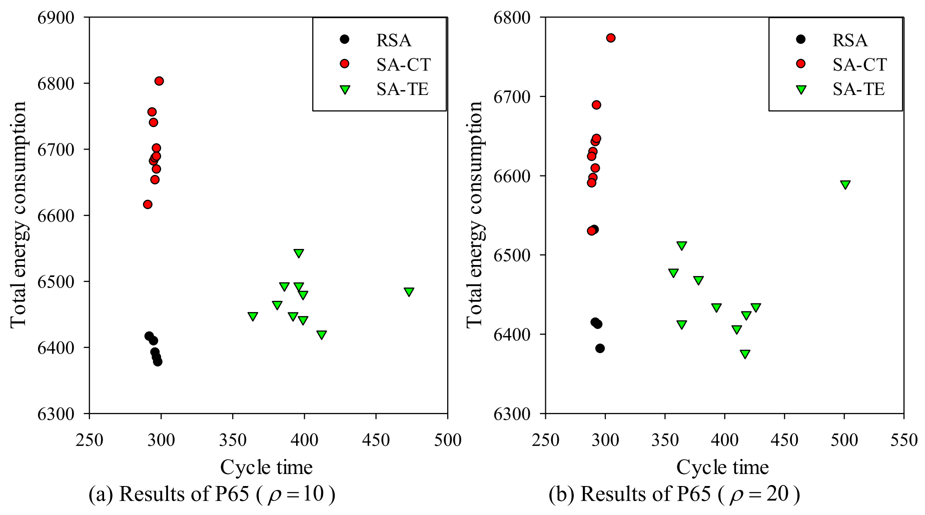

Fig. 8. Results of P65 with eight mated-stations by SA algorithms within ten times' running.

computational time (ρ = 5.0, 10.0, 15.0, 20.0). In Fig. 11(a), it is observed that the Pareto-optimal set by SA1 remains unchanged when the searching time reaching $n\,t \times nt \times 10$ milliseconds. And the Pareto-optimal set achieved whenρ = 15.0 is covered by the Pareto-optimal set achieved whenρ = 20.0. The SA2 and SA3, on the contrary, can further improve the solutions with the increase of computational time. The gap between Pareto-optimal solutions by SA2 is larger than that of RSA which means that the RSA can obtain better-spread Pareto-optimal solutions. The comparison among three SA algorithms demonstrates the effectiveness of the proposed restart mechanism.

for 32 cases and 34 cases under the termination criteria of $nt \times nt \times 10$ milliseconds and $nt \times nt \times 20$ milliseconds respectively. Especially, RSA outperforms NSGA-II for all the large-size cases (P65, P148 and P205) on the convergence criterion. As for the spread metric, the RSA outperforms NSGA-II for 32 cases and 33 cases under the two termination criteria respectively. Furthermore, careful statistical experiments are carried out to check whether there is significant difference between the two algorithms. Since an initial analysis shows a strong deviation from normality, non-parameter Wilcoxon matched-pairs signed rank test is carried out. The data are modified: the best one is ranked with one, and the

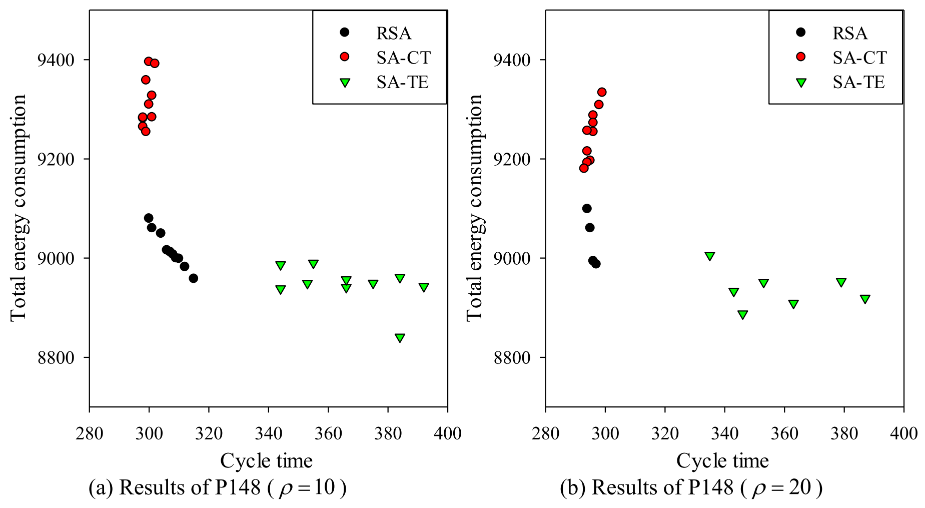

Fig. 9. Results of P148 with eight mated-stations by SA algorithms within ten times' running.

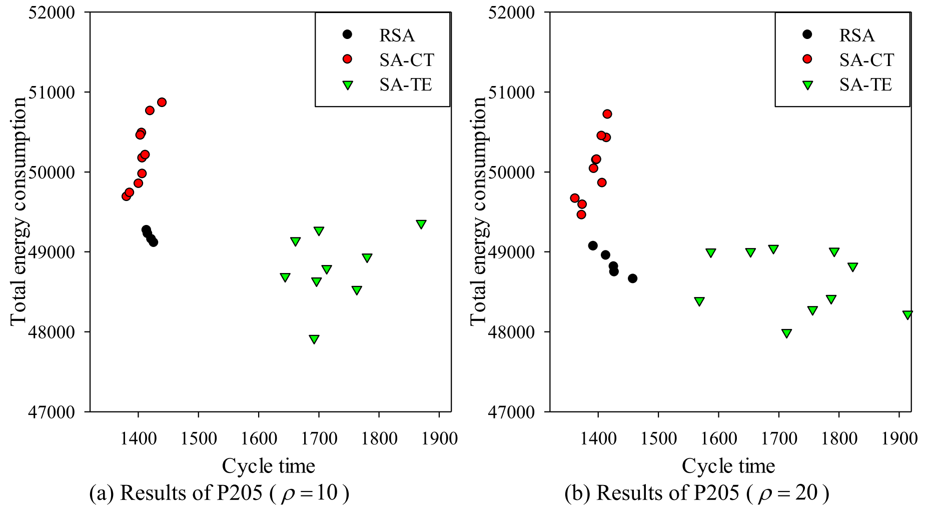

Fig. 10. Results of P205 with eight mated-stations by SA algorithms within ten times' running.

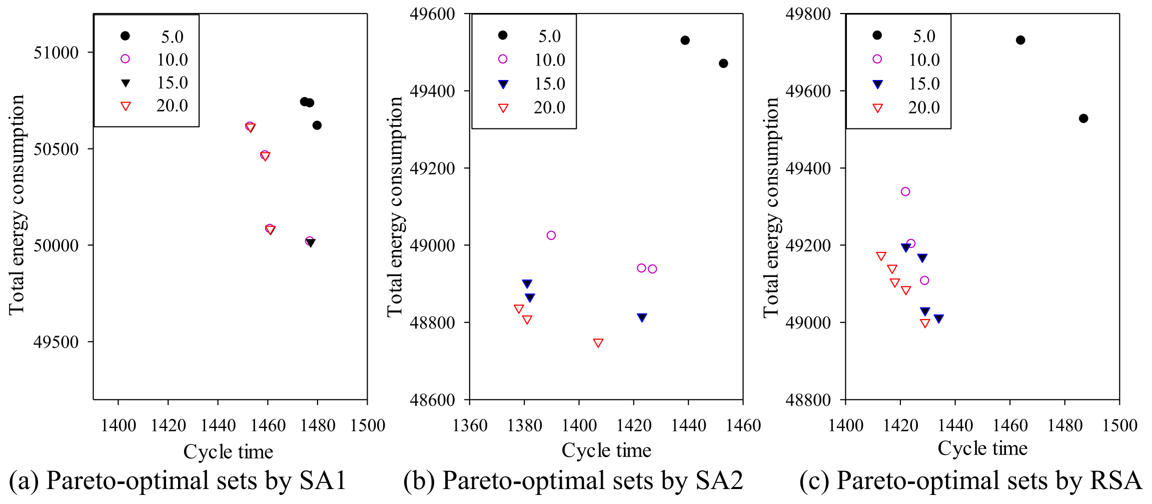

Fig. 11. Pareto fronts by three algorithms with four termination criteria.

Table 6
Results of the computational study for TRALB problems.

Problem	Nm	$\rho = 10$						$\rho = 20$					
		NSGA-II			RSA			NSGA-II			RSA		
		RP	CP	SP	RP	CP	SP	RP	CP	SP	RP	CP	SP
P9	2	0.40	1.19	0.82	0.97	0.03	0.34	0.55	0.64	0.80	1.00	0.03	0.34
P9	3	0.60	0.61	1.00	0.75	0.81	0.56	0.45	0.59	0.95	0.85	0.38	0.37
P12	2	0.35	0.54	0.37	1.00	0.00	0.00	0.27	0.65	0.61	1.00	0.00	0.00
P12	3	0.00	1.24	0.97	1.00	0.08	0.14	0.00	1.05	0.99	1.00	0.03	0.05
P12	4	0.20	[a]	[a]	1.00	[a]	[a]	0.10	[a]	[a]	1.00	[a]	[a]
P12	5	0.00	[a]	[a]	1.00	[a]	[a]	0.00	[a]	[a]	1.00	[a]	[a]
P16	2	0.20	1.04	0.91	1.00	0.20	0.04	0.15	1.31	0.87	1.00	0.00	0.00
P16	3	0.00	3.37	0.91	1.00	0.36	0.64	0.00	3.15	0.95	1.00	0.25	0.40
P16	4	0.00	[a]	[a]	1.00	[a]	[a]	0.00	[a]	[a]	1.00	[a]	[a]
P16	5	0.03	2.03	0.82	1.00	0.52	0.63	0.05	1.94	0.89	1.00	0.54	0.71
P24	2	0.00	1.43	0.65	1.00	0.17	0.52	0.05	1.02	0.82	1.00	0.14	0.54
P24	3	0.03	0.35	0.77	1.00	0.13	0.67	0.12	0.32	0.79	0.92	0.16	0.68
P24	4	0.00	3.12	0.86	1.00	0.82	0.73	0.05	2.73	0.90	0.97	0.78	0.67
P24	5	0.00	1.09	0.93	1.00	0.28	0.81	0.00	0.98	0.92	1.00	0.19	0.82
P65	4	0.00	7.15	0.97	1.00	1.19	0.92	0.00	6.36	0.96	1.00	1.09	0.88
P65	5	0.00	7.48	0.91	1.00	1.02	0.87	0.00	6.23	0.91	1.00	1.25	0.91
P65	6	0.00	14.60	0.92	1.00	2.32	0.89	0.00	11.42	0.92	1.00	1.68	0.80
P65	7	0.00	11.77	0.94	1.00	1.87	0.89	0.00	10.10	0.96	1.00	1.06	0.84
P65	8	0.00	7.54	0.97	1.00	1.32	0.85	0.00	7.07	0.95	1.00	0.73	0.85
P148	4	0.13	29.46	0.90	0.93	14.34	0.89	0.04	21.42	0.93	1.00	11.64	0.85
P148	5	0.00	5.47	0.95	1.00	2.07	0.88	0.00	4.98	0.95	1.00	1.25	0.87
P148	6	0.00	57.91	0.96	1.00	17.03	0.87	0.00	60.41	0.93	1.00	12.88	0.91
P148	7	0.06	6.02	0.94	1.00	2.17	0.88	0.00	5.21	0.95	1.00	1.15	0.89
P148	8	0.00	7.57	0.94	1.00	2.62	0.93	0.00	6.49	0.96	1.00	1.51	0.90
P148	9	0.00	[a]	[a]	1.00	[a]	[a]	0.00	[a]	[a]	1.00	[a]	[a]
P148	10	0.00	8.74	0.97	1.00	3.09	0.93	0.00	7.70	0.97	1.00	1.49	0.94
P148	11	0.00	21.55	0.98	1.00	6.22	0.94	0.00	20.76	0.97	1.00	3.42	0.90
P148	12	0.00	10.41	0.97	1.00	3.62	0.95	0.00	9.91	0.98	1.00	1.50	0.91
P205	4	0.00	60.61	0.94	1.00	28.69	0.96	0.00	57.65	0.95	1.00	21.24	0.92
P205	5	0.00	4.07	0.95	1.00	1.33	0.94	0.05	3.65	0.94	1.00	0.74	0.87
P205	6	0.00	12.36	0.96	1.00	4.64	0.96	0.00	11.36	0.95	1.00	3.59	0.95
P205	7	0.10	68.73	0.96	0.97	30.50	0.95	0.10	60.50	0.98	1.00	18.85	0.94
P205	8	0.00	6.84	0.97	1.00	1.99	0.91	0.00	7.04	0.98	1.00	1.18	0.94
P205	9	0.00	52.59	0.96	1.00	20.09	0.97	0.00	47.91	0.98	1.00	14.31	0.95
P205	10	0.00	30.39	0.96	1.00	11.08	0.93	0.00	25.63	0.96	1.00	7.79	0.92
P205	11	0.15	29.19	0.98	0.89	11.31	0.94	0.00	29.13	0.97	1.00	4.45	0.90
P205	12	0.05	46.12	0.98	1.00	19.92	0.95	0.00	42.29	0.98	1.00	13.71	0.93
P205	13	0.00	7.70	0.97	1.00	2.97	0.93	0.00	7.36	0.98	1.00	1.34	0.95
P205	14	0.03	[a]	[a]	1.00	[a]	[a]	0.00	[a]	[a]	1.00	[a]	[a]

[a] Only one solution in the approximate true-Pareto set.

worse one is ranked with two. Analytical results show that the *p*-values for all the three evaluation metrics under two termination criteria are smaller than 0.01 and RSA algorithm outperforms NSGA-II on all the three evaluation metrics. These computational results suggest that RSA outperforms NSGA-II statistically.

To exhibit the performances of the two algorithms, Figs. 12–14 depicts the Pareto fronts of some examples from the tested benchmark problems. Each case is solved for ten times, and the final Pareto front within ten times' running is shown in the figures. As depicted in the figures, the RSA obtains better Pareto front than NSGA-II for all the tested cases. To be specific, the best cycle time by RSA is less than 320 while the best cycle time by NSGA-II is bigger than 340 in Fig. 12(a). By comparing the two figures (Fig. 12(a) and (b)) with different computational times, it is observed that two algorithms can obtain better Pareto-optimal set with increasing computational time. The NSGA-II appears to be able to improve the Pareto fronts to a large degree while the RSA can improve the Pareto fronts a little. This situation is caused due to that the finding of new solutions gets more and more difficult while the Pareto front reaching the true Pareto front. The RSA can find better solutions with much less computational time and thus updating of the Pareto front by RSA becomes much more difficult. The above results further demonstrate the superiority of the RSA over the NSGA-II.

6. Conclusion and future research

The robotic assembly line is widely used in developed countries and it can be proposed for flexible production to deal with a great variety of products. Due to the consequence of the serious environmental impacts and the increased cost of energy consumption, reducing the total energy consumption becomes more and more important in two-sided assembly lines. This paper focuses on minimizing the energy consumption and cycle time in two-sided robotic assembly line, and it is the first one to consider the energy consumption in two-sided robotic assembly line. A mathematical model for type II two-sided robotic assembly line balancing (TRALB) is provided. This model considers the line balance and the cost of the energy consumption simultaneously, and multiple constraints are also presented.

To deal with the multi-objective type II TRALB problem, a restarted simulated annealing (RSA) algorithm is developed to obtain Pareto-optimal solutions. New encoding and decoding procedures based on robot assignment vector, task assignment to mated-station vector and task sequence vector are implemented to explore the search space. A multinomial probability mass function is utilized to activate one objective to decide the probability of accepting a new dominated solution. And a new restart mechanism is developed based on a modified crowding distance assignment

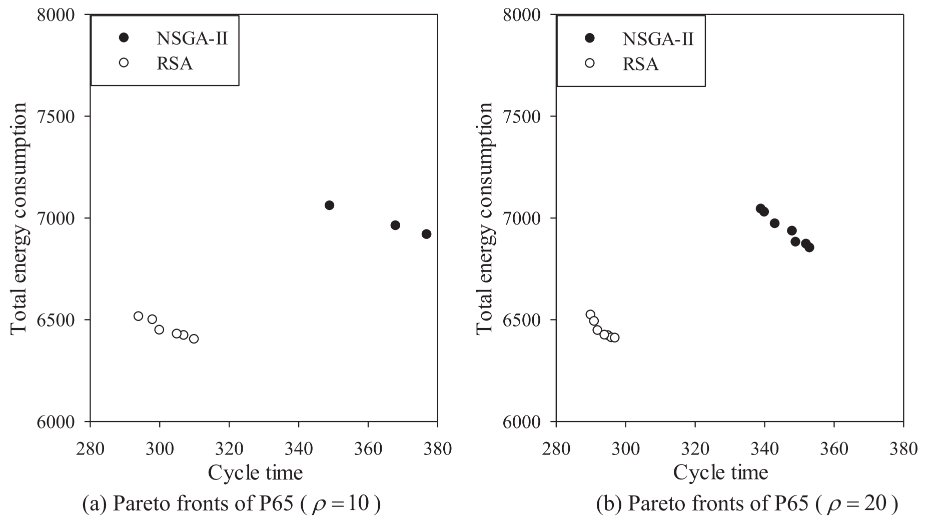

Fig. 12. Pareto fronts for P65 with eight mated-stations by RSA and NSGA-II.

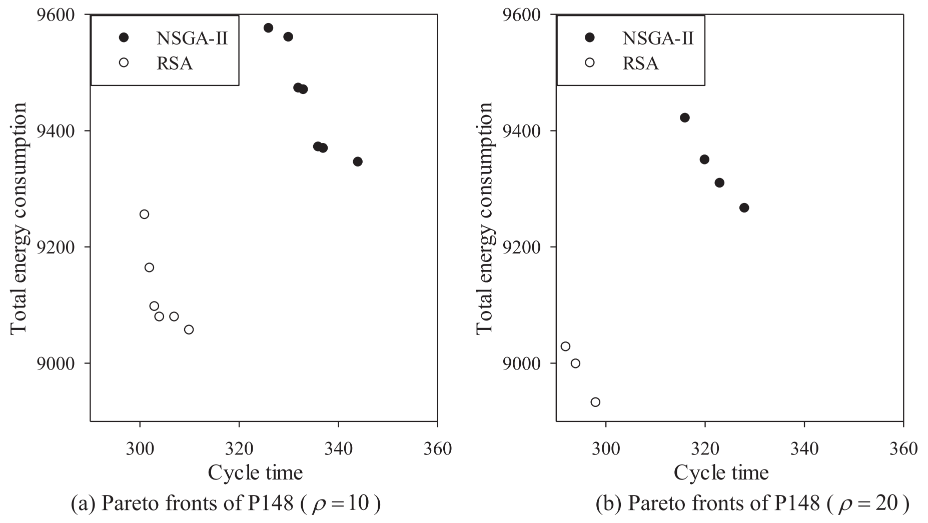

Fig. 13. Pareto fronts for P148 with eight mated-stations by RSA and NSGA-II.

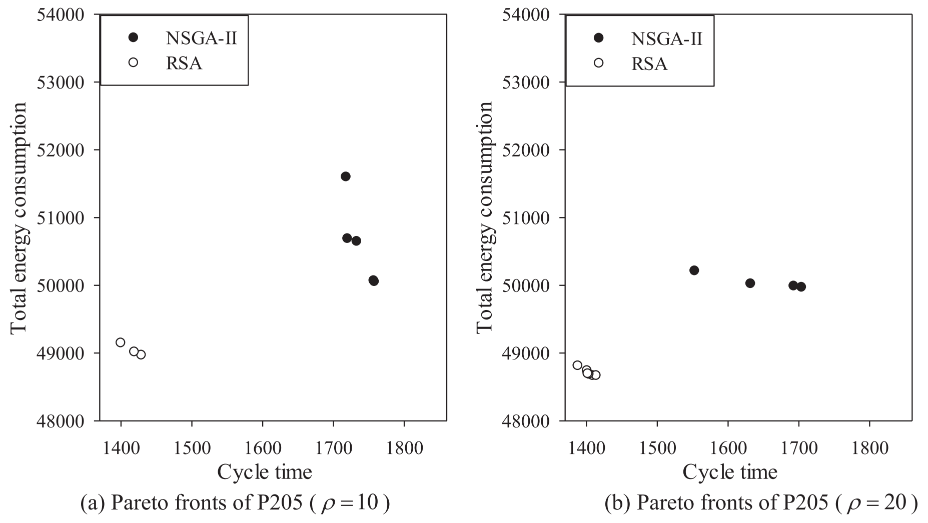

Fig. 14. Pareto fronts for P205 with eight mated-stations by RSA and NSGA-II.

procedure in order to obtain a better spread of the Pareto-optimal set. In order to evaluate the proposed multi-objective model, the RSA is compared with two SA algorithms with only one objective. The computational results show that the SA with the objective of minimizing the cycle time cannot reduce the energy consumption effectively. The RSA, on the contrary, can obtain solutions with much less energy consumption. These results show that the multi-objective model is helpful to reduce the energy consumption and the RSA can obtain a set of diverse high-quality solutions. In order to evaluate the proposed RSA, a set of benchmark problems composed by seven benchmark problems are generated and RSA is compared with the well-known elitist non-dominated sorting genetic algorithm (NSGA-II). The computational results are evaluated with three metrics, including the ratio of non-dominated solutions, the convergence of the Pareto-optimal solution and the spread metric, and comparison results demonstrate the superiority of the modified RSA over NSGA-II in both convergence and spread criteria.

This paper addresses only single model TRALB problem and multi and mixed model TRALB problems could be addressed in the future. And new metaheuristics, such as co-evolutionary algorithms and scatter search algorithm, can be employed for better performance.

Acknowledgment

This research work is funded by the National Natural Science Foundation of China (Grant No. 51275366 and 51305311).

References

Aghajani, M., Ghodsi, R., Javadi, B., 2014. Balancing of robotic mixed-model two-sided assembly line with robot setup times. Int. J. Adv. Manuf. Technol. 74 (5), 1005–1016.

Arkat, J., Saidi, M., Abbasi, B., 2007. Applying simulated annealing to cellular manufacturing system design. Int. J. Adv. Manuf. Technol. 32 (5), 531–536.

Bartholdi, J.J., 1993. Balancing two-sided assembly lines: a case study. Int. J. Prod. Res. 31 (10), 2447–2461.

Chutima, P., Chimklai, P., 2012. Multi-objective two-sided mixed-model assembly line balancing using particle swarm optimization with negative knowledge. Comput. Ind. Eng. 62 (1), 39–55.

Dai, M., Tang, D., Giret, A., Salido, M.A., Li, W.D., 2013. Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. Rob. Comput. Integr. Manuf. 29 (5), 418–429.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evolut. Computation 6 (2), 182–197.

Friedman, M., 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J. Am. Stat. Assoc. 32, 675–701.

Fysikopoulos, A., Anagnostakis, D., Salonitis, K., Chryssolouris, G., 2012. An empirical study of the energy consumption in automotive assembly. Procedia CIRP 3, 477–482.

Gao, J., Sun, L., Wang, L., Gen, M., 2009. An efficient approach for type II robotic assembly line balancing problems. Comput. Ind Eng. 56 (3), 1065–1080.

He, Y., Liu, B., Zhang, X., Gao, H., Liu, X., 2012. A modeling method of task-oriented energy consumption for machining manufacturing system. J. Cleaner Prod. 23 (1), 167–174.

Hu, X.-F., Wu, E.-F., Ye, Jin, 2008. A station-oriented enumerative algorithm for two-sided assembly line balancing. Eur. J. Op. Res. 186 (1), 435–440.

Hu, X.-F., Wu, E.-F., Bao, J.-S., Ye, Jin, 2010. A branch-and-bound algorithm to minimize the line length of a two-sided assembly line. Eur. J. Op. Res. 206 (3), 703–707.

Kim, Y.K., Kim, Y., Kim, Y.J., 2000. Two-sided assembly line balancing: a genetic algorithm approach. Prod. Plan. Control 11 (1), 44–53.

Kim, Y.K., Song, W.S., Kim, J.H., 2009. A mathematical model and a genetic algorithm for two-sided assembly line balancing. Comput. Op. Res. 36 (3), 853–865.

Kim, H., Park, S., 1995. Strong cutting plane algorithm for the robotic assembly line balancing. Int. Journal of Prod. Res. 33 (8), 2311–2323.

Kirkpatrick, S., Gelatt, C.D., Veechi, M.P., 1983. Optimization by simulated annealing. Science 220 (4598), 671–679.

Khorasanian, D., Hejazi, S.R., Moslehi, G., 2013. Two-sided assembly line balancing considering the relationships between tasks. Comput. Ind Eng. 66 (4), 1096–1105.

Kulturel-Konak, S., Smith, A.E., Norman, B.A., 2006. Multi-objective tabu search using a multinomial probability mass function. Eur. J. Op. Res. 169 (3), 918–931.

Lee, T.O., Kim, Y., Kim, Y.K., 2001. Two-sided assembly line balancing to maximize work relatedness and slackness. Comput. Ind. Eng. 40 (3), 273–292.

Levitin, G., Rubinovitz, J., Shnits, B., 2006. A genetic algorithm for robotic assembly line balancing. Eur. J. Op. Res. 168 (3), 811–825.

Montgomery, D.C., 2000. Design and analysis of experiments, 5th ed. Wiley, , New York.

Nilakantan, J.M., Huang, G.Q., Ponnambalam, S.G., 2015a. An investigation on minimizing cycle time and total energy consumption in robotic assembly line systems. J. Clean. Prod. 90, 311–325.

Nilakantan, J.M., Ponnambalam, S.G., Huang, G.Q., 2015b. Minimizing energy consumption in a U-shaped robotic assembly line. In: 2015 International Conference on Advanced Mechatronic Systems, Beijing, pp. 119–124.

Nilakantan, J.M., Ponnambalam, S.G., Jawahar, N., 2016. Design of energy efficient RAL system using evolutionary algorithms. Eng. Comput. 33 (2), 580–602.

Özbakir, L., Tapkan, P., 2011. Bee colony intelligence in zone constrained two-sided assembly line balancing problem. Expert Syst. Appl. 38 (9), 11947–11957.

Özcan, U., 2010. Balancing stochastic two-sided assembly lines: a chance-constrained, piecewise-linear, mixed integer program and a simulated annealing algorithm. Eur. J. Op. Res. 205 (1), 81–97.

Özcan, U., Toklu, B., 2010. Balancing two-sided assembly lines with sequence-dependent setup times. Int. J. Prod. Res. 48 (18), 5363–5383.

Özcan, U., Toklu, B., 2009a. A tabu search algorithm for two-sided assembly line balancing. Int. J. Adv. Manuf. Technol. 43 (7), 822–829.

Özcan, U., Toklu, B., 2009b. Balancing of mixed-model two-sided assembly lines. Comput. Ind Eng. 57, 217–227.

Purnomo, H.D., Wee, H.-M., 2014. Maximizing production rate and workload balancing in a two-sided assembly line using harmony search. Comput. Ind Eng. 76, 222–230.

Purnomo, H.D., Wee, H.-M., Rau, H., 2013. Two-sided assembly lines balancing with assignment restrictions. Math. Comput. Model. 57 (1–2), 189–199.

Rubinovitz, J., Bukchin, J., 1991. Design and balancing of robotic assembly lines. In: Proceedings of the Fourth World Conference on Robotics Research, Pittsburgh, PA.

Rubinovitz, J., Bukchin, J., Lenz, E., 1993. RALB—a heuristic algorithm for design and balancing of robotic assembly line. CIRP Ann. 42 (1), 497–500.

Tang, Q.-H., Li, Z.-X., Zhang, L.-P., Floudas, C.A., Cao, X.-J., 2015. Effective hybrid teaching-learning-based optimization algorithm for balancing two-sided assembly lines with multiple constraints. Chin. J. Mech. Eng. 28 (5), 1067–1079.

Tsai, D.-M., Yao, M.-J., 1993. A line-balanced-base capacity planning procedure for series-type robotic assembly line. Int. J. Prod. Res. 31 (8), 1901–1920.

Wu, E.-F., Ye, Jin, Bao, J.-S., Hu, X.-F., 2008. A branch-and-bound algorithm for two-sided assembly line balancing. Int. J. Adv. Manuf. Technol. 39 (9), 1009–1015.

Yoosefelahi, A., Aminnayeri, M., Mosadegh, H., Ardakani, H.D., 2012. Type II robotic assembly line balancing problem: an evolution strategies algorithm for a multi-objective model. J. Manuf. Syst. 31 (2), 139–151.

Yuan, B., Zhang, C.-Y., Shao, X.-Y., 2015. A late acceptance hill-climbing algorithm for balancing two-sided assembly lines with multiple restrictions. J. Intelligent Manuf. 26 (1), 159–168.