Reinforcement learning-based multi-objective evolutionary algorithm for mixed-model multi-manned assembly line balancing under uncertain demand

Zikai Zhang, Qiuhua Tang, Manuel Chica, and Zixiang Li

***Abstract*—In practical assembly enterprises, customization and rush orders lead to an uncertain demand environment. This situation requires managers and researchers to configure an assembly line that increases production efficiency and robustness. Hence, this work addresses cost-oriented mixed-model multi-manned assembly line balancing under uncertain demand, and presents a new robust mixed integer linear programming model to minimize the production and penalty costs simultaneously. Additionally, a reinforcement learning-based multi-objective evolutionary algorithm is designed to tackle the problem. The algorithm includes a priority-based solution representation and a new task-worker-sequence decoding that considers robustness processing and idle time reductions. Five crossover and three mutation operators are proposed. The Q-learning-based strategy determines the crossover and mutation operator at each iteration to effectively obtain Pareto sets of solutions. Finally, a time-based probability-adaptive strategy is designed to effectively coordinate the crossover and mutation operators. The experimental study, based on 269 benchmark instances, demonstrates that the proposal outperforms 11 competitive multi-objective evolutionary algorithms and a previous single-objective approach to the problem. The managerial insights from the results as well as the limitations of the algorithm are also highlighted.**

***Index Terms*—assembly line balancing, multi-manned, uncertain demand, multi-objective optimization, reinforcement learning, evolutionary algorithms.**

I. Introduction

An assembly line, a typical flow-oriented manufacturing system, is generally dedicated to the assembly of high-volume and complex products such as cars and engines. Its corresponding balancing problem, which has been investigated by many scholars, aims to assign all the tasks with precedence graphs to some specific workstations. Recently, research studies have focused on more practical assembly line balancing (ALB) problems. One of these realistic formulations is the multi-manned assembly line balancing problem (MALBP), a generalization of the ALB [1] that allows more than one worker to perform multiple tasks at the same workstations. Compared with the classical ALB, the MALBP involves finding the optimal number of workers in each workstation and assigning each task to exactly one worker of one workstation. This feature makes the MALBP more complex and difficult to solve but brings four advantages: lower line length, effective space configuration, higher yield, and fewer setup times [2].

In addition, and thanks to the popularity of customization, enterprises aim to organize flexible multi-manned assembly lines to process mixed-model products. In a realistic production horizon, since demand for products is dynamic and uncertain a regular line configuration cannot ensure that all the required products are finished within the cycle time [3]. This situation will greatly affect the production schedule and generate economic losses. However, very few studies have investigated the mixed-model multi-manned assembly line balancing problem under uncertain demand (r-MMALBP).

Therefore, and following previous research [3], this work further studies the cost-oriented r-MMALBP. This r-MMALBP variant has two types of costs involved - production and penalty - which are in conflict. The conflict arises because managers need to reduce the number of workers and workstations in order to lower the production cost but do not want to generate penalty costs due to overdue completion. The minimization of production and penalty costs often comprises conflicting goals. A minimum production cost might lead to a high penalty cost and vice-versa. Thus, this work aims to minimize both the production and penalty costs simultaneously. A mixed integer linear programming (MILP) model is proposed to minimize both objectives while obtaining a robust line configuration that responds to scenarios with uncertain demand.

Additionally, and due to the strong NP-hardness nature of the r-MMALBP, a multi-objective evolutionary algorithm (MOEA) is proposed to obtain trade-off solutions for the above-mentioned two conflicting goals. MOEAs have been widely applied to tackle multi-objective combination optimization problems, including ALB [4], disassembly line balancing [5], job shop scheduling [6], flow shop scheduling [7],

This work is supported by the China Postdoctoral Science Foundation (No. 2021M702536) and the National Natural Science Foundation of China (No. 52275504, 62173260). M. Chica is supported by the Andalusian Government and the European Regional Development Fund under grant SIMARK PY18-4475.

Zikai Zhang, Qiuhua Tang and Zixiang Li are with the Key Laboratory of Metallurgical Equipment and Control Technology, Ministry of Education, Wuhan University of Science and Technology, and Hubei Key Laboratory of Mechanical Transmission and Manufacturing Engineering, Wuhan University of Science and Technology, and Precision Manufacturing Institute, Wuhan University of Science and Technology, Wuhan, Hubei, China (e-mail: zhangzikai@wust.edu.cn; tangqiuhua@wust.edu.cn; lizixiang@wust.edu.cn).

Manuel Chica is with the Department of Software Engineering, Andalusian Research Institute in Data Science and Computational Intelligence, DaSCI, University of Granada, Granada, 18071, Spain, and the School of Electrical Engineering and Computing, University of Newcastle, Callaghan, NSW 2308, Australia (e-mail:manuelchica@ugr.es).

parallel machines scheduling [8] and distributed scheduling [9]. This work extends an MOEA variant by including a reinforcement learning (RL) process [10]. Different problem-dependent heuristics and genetic operators (five crossover and three mutation operators) are included. In this work, this algorithmic approach is named the reinforcement learning-based multi-objective evolutionary algorithm (RL-MOEA). The main contributions of this work are as follows:

- A novel robust multi-objective MILP model with the objectives of minimizing production and penalty costs to formulate a cost-oriented r-MMALBP.
- The inclusion of a Q-learning-based strategy to guide the RL-MOEA to select the operator with the best performance at each iteration in order to improve the adaptive ability of RL-MOEA.
- Five crossover operators and three mutation operators to make the RL-MOEA specific to the problem.
- A time-based probability-adaptive strategy to coordinate the crossover and mutation operators at each iteration and make the RL-MOEA adaptive during its running.

This work validates the proposal on five instances for calibrating the model and 269 test sets. The design of experiments technique, coupled with a multi-factor analysis of variance (ANOVA), is conducted to determine the best RL-MOEA parameter combination. Different multi-objective performance indicators and visual comparison tools (i.e., attainment surfaces) are included to check the convergence, diversity, and dominance of the RL-MOEA and its main components. The experiment is based on comparing the RL-MOEA against 11 recent and competitive MOEAs as well as 10 rules mined by an existing gene expression programming approach for the r-MMALBP [3]. Finally, this study also analyzes the reasons for the good performance of the proposal, the limitations of the algorithm, and its managerial impact.

II. Related works

This section reviews related works in three main fields: multi-manned assembly line balancing (Section II-A), uncertain demand when assembling mixed models (Section II-B), and MOEAs (Section II-C).

A. *Multi-manned assembly line balancing*

The MALBP has been addressed by many scholars. Various methods have been proposed to deal with this problem, including exact methods [11], heuristics [12], and metaheuristics [2]. More specifically, Fattahi et al. [13] proposed ant colony optimization to minimize the total number of workstations and workers, Kellegöz and Toklu [14] designed an efficient branch and bound method to solve the MALBP and Roshani et al. [15] developed a simulated annealing algorithm with different decoding mechanisms to minimize the number of workstations and workers or the cycle time. Kellegöz and Toklu [12] designed an efficient heuristic with several priority rules and a genetic algorithm. Finally, the authors of [16] used a decomposition algorithm, while Çil and Kizilay [11] used a constraint programming method.

The above research mainly focused on the simple version of the MALBP and made a significant number of contributions. However, the simple version does not adequately reflect actual production situations. Recently, various realistic constraints have been considered in the MALBP. Chen et al. [17] considered the resource constraint that different tasks require resources such as machines and tools. They aimed to minimize the total number of workers, workstations and resources. Lopes et al. [18] proposed flexible station frontiers with no limit to the boundaries of worker activities but with the constraint that the workload of each worker cannot exceed the cycle time. Sahin and Kellegoz [19] studied the renewable resource constraint in the MALBP and proposed a particle swarm optimization combined with a special heuristic to minimize the total costs of resources and opened workstations. Zhang et al. [1] introduced a space constraint into the MALBP and developed a memetic ant colony system to minimize the total number of workstations and workers.

B. *Uncertain demand when assembling mixed models*

Although some researchers have studied the practical constraints and optimization methods of the MALBP in the past, there is still a lack of mixed-model studies. Assembly lines are now being required to have the capability of mixed-model production due to the popularity of customization [20]. While recent research studies have considered the mixed-model in different ALB problems, including simple ALB [21], ALB with worker assignment [22], two-sided ALB [23], and human-robot collaboration ALB [24], they all have the common feature of dealing with certain demands.

However, uncertain demands and rush orders frequently appear in practical production. This situation requires enterprises to design a robust line configuration to organize the production. Two works have focused on the uncertain demand of mixed-model ALB. Chica et al. [25] addressed uncertain demand when considering spatial features in ALB and proposed robust-based functions to minimize the number of overloaded workstations and their overload value. An MOEA was also proposed in [20] to solve the problem. Nevertheless, just one study [3] considered the r-MMALBP, even though r-MMALBP can better reflect the actual production environment. Hence, this work aims to propose a new robust multi-objective MILP model for the r-MMALBP to minimize production and penalty costs.

C. *Multi-objective evolutionary algorithms*

Since this work studies multi-objective optimization in the r-MMALBP, this section reports the main MOEAs related to ALB. Recently, different multi-objective methods have been extended to obtain the Pareto front solutions to the problem, including local search and population optimization. More specifically, Nilakantan et al. [4] designed a multi-objective co-operative co-evolutionary algorithm to optimize the carbon footprint and line efficiency of robot ALB. Babazadeh et al. [26] proposed an enhanced fast and elitist multi-objective genetic algorithm (NSGA-II) to minimize the number of workstations and the fuzzy cycle time of simple and U-shaped

ALB. Zhang et al. [27] developed a Pareto artificial bee colony optimization algorithm to minimize the cycle time and energy consumption of U-shaped robotic ALB. In addition, Zhang et al. [28] designed a multi-objective gray wolf optimization to minimize the cycle time, carbon and noise emissions. Li et al. [29] developed the NSGA-II and improved multi-objective artificial bee colony optimization to achieve a trade-off between cycle time and total purchasing cost of robotic ALB.

Apart from the listed methods, there is a vast set of related MOEAs which includes the decomposition-based multi-objective evolutionary algorithm [30], the restarted iterated Pareto greedy algorithm [31] and enhanced JAYA (EJAYA) [32]. Reinforcement learning mechanisms have also been successfully applied to evolutionary algorithms, as in the research in [33]. However, the referenced study does not use the learning process to select the best evolution operators of the algorithm at each iteration for an industrial problem such as the MALBP. Since an MOEA combines exploration and exploitation abilities, this work proposes to include a reinforcement learning mechanism to find the best operators to apply. Thus, a new algorithm, the RL-MOEA, is proposed to tackle the novel robust multi-objective problem using a Q-learning mechanism as well as different novel operators.

III. Robustness problem formulation

The problem definition and notation (e.g., variables and parameters) are defined in Section III-A. The objectives of the r-MMALBP are established in Section III-B, and the constraints of the problem are defined in Section III-C. A numerical example is given in Section III-D.

A. Definition of the problem and notations

A multi-manned assembly line can assemble multiple products. The MALBP includes n tasks for different mixed products Q to be assembled that have to be assigned to a set of multi-manned workstations. The processing time of each task t_{iq} (different for each product q) and the precedence graph between different tasks are known *a priori*. At each workstation, only u_{max} workers are allowed to work at the same time. Each task i can be assigned to only one worker from one workstation and starts after all its immediate predecessors have been completed. The workload of each worker is not allowed to exceed the cycle time. The setup time, release time and travel time are ignored. Finally, parallel tasks and parallel workstations are not allowed.

Tables I and II present the parameters and variables of the r-MMALBP, respectively. More specifically, in the r-MMALBP, because of its capability to deal with uncertain demand, there is a set of different demand plans defined by E. The number of all considered demand plans is denoted by $|E|$. For each demand plan $e \in E$, there are different types and numbers of products. Let d_{qe} represent the number of product q in plan e. Using the robust optimization scheme proposed in [25], the robust time $\bar{t}_{ie}$ of task i under q is first defined, calculated as $\frac{\sum_{q\in Q} d_{qe}\cdot t_{iq}}{\sum_{q\in Q} d_{qe}}$. The original cycle time is transformed into $CT + \Delta_{CT}$ to allow an additional time.

TABLE I
Description of the parameters

Parameter	Description
i, h	Task indices
j	Workstation index
k	Worker index
e	Demand plan index
q	Product index
n	Number of tasks
m	Maximum number of workstations
CT	Cycle time
u_{max}	Maximum number of workers for each station
d_{qe}	Demand of product q in plan e
Δ_{CT}	Maximum exceeding cycle time per station
t_{iq}	Processing time of task i in product q.
$\bar{t}_{ie}$	Average processing time of task i in plan e.
I	Set of tasks, $\lvert I\rvert = n$
J	Set of workstations, $\lvert J\rvert = m$
K	Set of the assigned workers to each workstation, $\lvert K\rvert = u_{max}$.
E	Set of all the demand plans
Q	Set of products
P^0	Set of tasks that have no immediate predecessors
$P(i)$	Set of immediate predecessors of task i
COO	Cost of hiring a worker
COW	Cost of opening a workstation
CPO	Penalty cost of a worker exceeding the cycle time because of its workload
CPW	The penalty cost for an exceeding workstation
$\hat{M}$	Large positive number

TABLE II
Description of the variables

Variable	Description
FT_{ie}	The finishing time of task i in plan e.
W_{ih}	A binary value sets to 1 if task i and h are assigned to the same worker and task i is performed before task h; otherwise sets to 0.
X_{ijk}	A binary value sets to 1 if task i is assigned to worker k at workstation j; otherwise sets to 0.
Y_{jk}	A binary value set to 1 if worker k is hired in workstation j; otherwise set to 0.
Z_j	A binary value set to 1 if workstation j is opened; otherwise set to 0.
G_{ie}	A binary value set to 1 if the finishing time of task i in plan e exceeds the cycle time; otherwise set to 0.
R_{je}	A binary value set to 1 if the finishing time of workstation j in plan e exceeds the cycle time; otherwise set to 0.
O_{jke}	A binary value set to 1 if the finishing time of worker k at workstation j in plan e exceeds the cycle time; otherwise set to 0.
XG_{ijke}	A binary value set to 1 if the finishing time of task i in plan e exceeds the cycle time and task i is assigned to worker k at workstation j; otherwise set to 0.

B. Objectives including robustness

The main objective of an MALBP is to minimize the total hired worker costs and opened workstation costs. In order to include the robustness capability and higher workload of a solution in the r-MMALBP, the penalty costs of all workers and workstations exceeding the cycle time of all the demand plans are considered as the second objective. This objective permits optimization under an uncertain environment. Therefore, the proposed r-MMALBP model aims to minimize two conflicting objectives at the same time: the costs of all hired workers and opened workstations (Eq. (1)), and the penalty costs of all workers and workstations whose workload exceeds the cycle time under all demand plans (Eq. (2)).

$$minf_1 = \sum_{j\in J}\sum_{k\in K} Y_{jk}\cdot COO + \sum_{j\in J} Z_j \cdot COW \quad (1)$$

$$minf_2 = \sum_{j\in J}\sum_{k\in K}\sum_{e\in E} O_{jke}\cdot CPO + \sum_{j\in J}\sum_{e\in E} R_{je}\cdot CPW \quad (2)$$

C. Constraints of the model

The constraints of the MALBP under single demand are maintained for the r-MMALBP. Eq. (3) is an assignment constraint that each task can only be assigned to one worker of one workstation. Eq. (4) is the precedence constraint that a task starts after all its immediate predecessors have been finished. Eqs. (5)-(6) define the robust cycle time constraints. The finishing time of each task is allowed to exceed CT but not $CT + \Delta_{CT}$. If the finishing time exceeds CT, the corresponding variable G_{ie} is equal to 1.

$$\sum_{j\in J}\sum_{k\in K} X_{ijk} = 1, \forall i \in I \quad (3)$$

$$\sum_{j\in J}\sum_{k\in K} j\cdot X_{hjk} - \sum_{j\in J}\sum_{k\in K} j\cdot X_{ijk} \le 0, \forall i \in I - P^0, h \in P(i) \quad (4)$$

$$FT_{ie} \le CT + \Delta_{CT}\cdot G_{ie}, \forall i \in I, e \in E \quad (5)$$

$$FT_{ie} \ge CT \cdot G_{ie}, \forall i \in I, e \in E \quad (6)$$

Eq. (7) controls the start times of each pair of tasks with precedence relations. That is, if task h belongs to $P(i)$ and tasks i and h are assigned to the same workstation, task i starts after finishing task h. Eqs. (8)-(9) limit the starting time of each pair of tasks with no precedence relationships but assigned to adjacent positions of the same worker. If task i is in front of task h, Eq. (8) becomes active and $FT_{he} - FT_{ie} \ge \bar{t}_{he}$; otherwise Eq. (9) works and $FT_{ie} - FT_{he} \ge \bar{t}_{ie}$.

$$FT_{ie} - FT_{he} + \hat{M}\cdot(1 - \sum_{k\in K} X_{ijk}) + \hat{M}\cdot(1 - \sum_{k\in K} X_{hjk}) \ge \bar{t}_{ie}, \forall i \in I - P^0, h \in P(i), j \in J, e \in E \quad (7)$$

$$FT_{he} - FT_{ie} + \hat{M}\cdot(1 - X_{ijk}) + \hat{M}\cdot(1 - X_{hjk}) + \hat{M}\cdot(1 - W_{ih}) \ge \bar{t}_{he}, \forall (i,h) | i \ne h \wedge i,h \in I, j \in J, k \in K, e \in E \quad (8)$$

$$FT_{ie} - FT_{he} + \hat{M}\cdot(1 - X_{ijk}) + \hat{M}\cdot(1 - X_{hjk}) + \hat{M}\cdot W_{ih} \ge \bar{t}_{ie}, \forall (i,h) | i \ne h \wedge i,h \in I, j \in J, k \in K, e \in E \quad (9)$$

Eq.(10) limits the lower bound of the finishing time. Eqs. (11)-(13) confine the variable Y_{jk}. If some tasks are assigned to worker k at workstation j, Eq. (11) becomes active and Y_{jk} equal to 1; otherwise Eq. (12) works and Y_{jk} is equal to 0. Eq. (13) limits that workers are employed in an increasing manner. Eqs. (14)-(16) control the variable Z_j and are similar to the limitation of variable Y_{jk}.

$$FT_{ie} \ge \bar{t}_{ie}, \forall i \in I, e \in E \quad (10)$$

$$\sum_{i\in I} X_{ijk} \le n \cdot Y_{jk}, \forall j \in J, k \in K \quad (11)$$

$$\sum_{i\in I} X_{ijk} \ge Y_{jk}, \forall j \in J, k \in K \quad (12)$$

$$Y_{j,k+1} \le Y_{jk}, \forall j \in J, k = 1,2,...,u_{max} - 1 \quad (13)$$

$$\sum_{i\in I}\sum_{k\in K} X_{ijk} \le n \cdot u_{max} \cdot Z_j, \forall j \in J \quad (14)$$

$$\sum_{i\in I}\sum_{k\in K} X_{ijk} \ge Z_j, \forall j \in J \quad (15)$$

$$Z_{j+1} \le Z_j, \forall j = 1,2,...,m-1 \quad (16)$$

Eqs. (17)-(19) define the variable XG_{ijke}. Only when both variables G_{ie} and X_{ijk} are equal to 1 can variable XG_{ijke} be equal to 1. Finally, Eqs. (20)-(21) limit the variable O_{jke}. For all tasks assigned to worker k at workstation j, if there is one task whose finishing time in plan e exceeds CT ($XG_{ijke} = 1$), the corresponding variable O_{jke} is equal to 1. Eqs. (22)-(23) confine the variable R_{je} and follow the similar limitation of the variable O_{jke}.

$$XG_{ijke} \ge G_{ie} + X_{ijk} - 1, \forall i \in I, j \in J, k \in K, e \in E \quad (17)$$

$$XG_{ijke} \le G_{ie}, \forall i \in I, j \in J, k \in K, e \in E \quad (18)$$

$$XG_{ijke} \le X_{ijk}, \forall i \in I, j \in J, k \in K, e \in E \quad (19)$$

$$\sum_{i\in I} XG_{ijke} \le n \cdot O_{jke}, \forall j \in J, k \in K, e \in E \quad (20)$$

$$\sum_{i\in I} XG_{ijke} \ge O_{jke}, \forall j \in J, k \in K, e \in E \quad (21)$$

$$\sum_{k\in K} O_{jke} \le u_{max} \cdot R_{je}, \forall j \in J, e \in E \quad (22)$$

$$\sum_{k\in K} O_{jke} \ge R_{je}, \forall j \in J, e \in E \quad (23)$$

D. Numerical example

This section aims to provide a better understanding of the robustness problem through a numerical example with 9 tasks, 10 products and 3 demand plans. Tables III and IV respectively show the processing times and demand plans, and the precedence relation is presented in Fig.1. Parameters u_{max}, CT, Δ_{CT}, COO, COW, CPO and CPW are set at 3, 3.7, 0.6, 9.9, 5.7, 9.2 and 5.4, respectively.

Fig.2 presents the obtained robust solution using the proposed methodology. 3 workstations and 6 workers are involved to organize the robust multi-manned assembly line. Workstations 1-3 employ 3, 2 and 1 workers, respectively. Hence the costs of opened workstations and hired workers are 17.1 (5.7×3) and 59.4 (9.9×6), respectively. Plans 1-2 have 2 workers and 2 workstations whose workloads exceed the cycle time, with the corresponding values for plan 3 being 1 worker and 1 workstation. Therefore, the penalty costs of workstations and workers are 27.0 (5.4×5) and 46.0 (9.2×5), respectively.

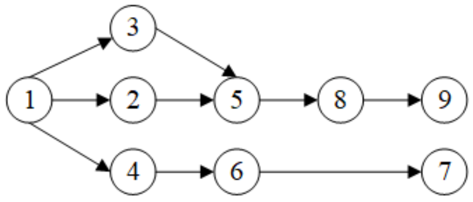


Fig. 1. The precedence relation graph.

TABLE III
THE PROCESSING TIMES OF ALL TASKS

Task (i)	Product (q)									
	1	2	3	4	5	6	7	8	9	10
1	5	1	3	1	1	3	0	1	0	0
2	3	2	1	0	1	1	0	2	2	2
3	4	3	0	2	3	0	3	0	1	3
4	5	1	1	3	0	4	1	3	0	0
5	4	2	3	2	1	2	2	0	0	0
6	5	2	1	1	4	3	0	0	1	2
7	1	0	0	0	0	0	0	0	0	0
8	4	2	3	3	3	2	2	2	3	1
9	6	3	4	4	1	0	4	0	4	1

TABLE IV
NUMERICAL EXAMPLE DEMAND PLANS

Plan (e)	Product (q)									
	1	2	3	4	5	6	7	8	9	10
1	27	8	13	17	19	21	12	22	14	24
2	18	19	12	12	25	14	10	8	11	21
3	18	29	23	6	7	6	8	25	8	8

IV. THE PROPOSED MULTI-OBJECTIVE ALGORITHM

The details of the algorithm are introduced in Sections IV-A to IV-E. The proposed RL-MOEA is based on the framework of NSGA-II and includes reinforcement learning capabilities. However, there are five main differences between the RL-MOEA and the NSGA-II:

- RL-MOEA includes a priority-based solution representation and a new task-worker-sequence solution decoding.
- Five problem-specific crossover operators and three problem-specific mutation operators are designed.
- The RL-MOEA uses similarity-based selection, instead of random selection, to select a solution as the cross object in the crossover phase.
- The algorithm incorporates a Q-learning-based strategy to select an operator with high performance at each iteration.
- Finally, the proposed MOEA uses a time-based probability-adaptive strategy to dynamically update the crossover and mutation probabilities instead of using fixed probability values.

For the initialization of RL-MOEA, a population with PS solutions is randomly generated and decoded according to the task-worker-sequence decoding mechanism. The non-dominated solutions are saved into an external archive called POS. Finally, The pseudo-code of the RL-MOEA is presented in Algorithm 1.

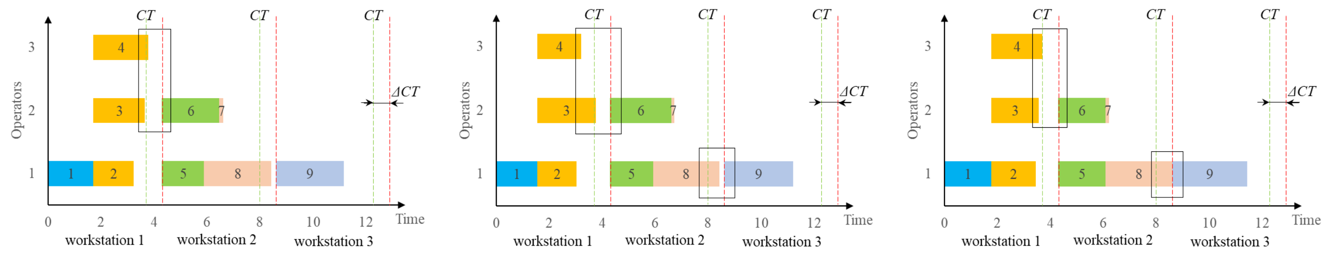
Algorithm 1: RL-MOEA
```
1:  Input: Problem data, parameters $PS$, $\alpha$ and $\gamma$.
2:  Output: $POS$.
3:  Begin:
4:      Randomly generate $PS$ initial solutions $\{\Pi^1, \Pi^2, ..., \Pi^{PS}\}$;
5:      Store initial non-dominated solutions of population into $POS$;
6:      Set initial states $s^{cr} = start$ and $s^{mu} = start$;
7:      $Q^{cr}_{3\times5} = 0$; $Q^{mu}_{3\times3} = 0$;
8:      $CR = 1.0$; $MR = 1.0 - CR$;
9:      Do while termination criteria is not fulfilled
10:         Crossover phase; (include Q-learning-based strategy)
11:         Mutation phase; (include Q-learning-based strategy)
12:         Select initial solutions via hierarchy and crowding distance;
13:         Update the $CR$ and $MR$ via time-based probability-adaptive strategy;
14:     End While
15: Return $POS$.
```

A. Solution representation and decoding mechanism

This algorithm uses an n-dimensional vector, $\Pi = \{\pi_1, \pi_2, ..., \pi_i, ..., \pi_n\}$, to represent a solution of the r-MMALBP. The element π_i means the priority of task i. Each priority value is randomly generated between 1 and n without repetition. For example, a solution Π is encoded as $\{3, 9, 4, 6, 7, 1, 2, 8, 5\}$. That is, the priority of task 1 is 3 and that of task 2 is 9, and those of the remaining tasks 3-9 are 4, 6, 7, 1, 2, 8 and 5, respectively. Based on the above representation, this work further designs a new task-worker-sequence decoding to assign the tasks to one worker and workstation.

The decoding scheme reduces the idle times effectively on the premise of satisfying all the constraints. It first selects a specific task according to the priority-based representation and available candidate set. Then, it uses some rules to determine a worker to carry out the selected task. The procedure is detailed below. At the beginning, the current workstation wc is equal to 0.

- **Step 1**. Open a workstation ($wc = wc + 1$) with u_{max} workers.
- **Step 2**. A set of available tasks is built for being a candidate. A task can be considered as a candidate when satisfying the following criteria: (a) it has not been previously assigned and (b) its predecessors have



(a) The gantt chart for demand plan 1. (b) The gantt chart for demand plan 2. (c) The gantt chart for demand plan 3.

Fig. 2. Gantt charts for three different demand plans. Black boxes highlight differences in the workstations depending on the plan.

already been assigned. When a candidate list is empty the algorithm jumps to **step 6**, otherwise it moves to **step 3**. Note that an empty candidate set means all the tasks have been assigned.

- **Step 3**. A task with the highest priority value is selected from the candidate set.
- **Step 4**. The set of available workers is determined. A worker k is available if the worker can assemble the selected task in all the demand plans. If no worker is available the procedure returns to **step 1**, otherwise it goes to **step 5**.
- **Step 5**. Select a worker. In order to reduce the sequence-dependent and remaining idle times, a worker from the available worker set is selected to carry out the selected task referring to the following criteria: a) select a worker who can carry out the selected task first; b) if there is more than one worker from a), select a worker with the minimum idle time between the previous assigned task and the selected one; and c) if there is more than one worker from the previous rule, select a worker with the lowest number. After finishing the worker selection, return to **step 2**.
- **Step 6**. Calculate the production and penalty costs.

B. Crossover operators

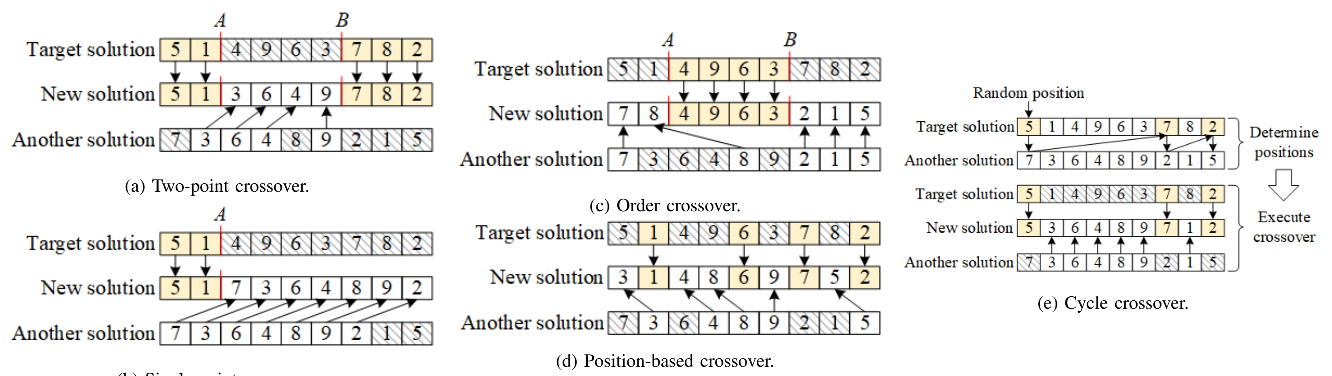
Since different crossover operators can generate different solutions and perform differently, in this work it was decided to adopt five classical crossover operators that operate on the solution representation Π. The adopted crossover operators are: two-point, single-point, order, position-based, and cycle crossovers. Each solution in the population can be a target solution and use the similarity-based selection proposed by Zhang et al. [32] to choose another solution as its crossover object. The Q-learning-based strategy is used to select different crossover operators at each iteration (see Section IV-D). An example of its application is depicted in Fig.3. The details of the crossover operators are specified below:

- **Two-point crossover**: Two crossover points A and B are first randomly selected. The elements between these two points in the target solution are removed, and then the non-repetitive elements of another solution fill these empty positions in turn.
- **Single-point crossover**: One crossover point A is first randomly selected. The elements after this point in the target solution are removed, and then the non-repetitive elements of another solution fill these empty positions in turn.
- **Order crossover**: This has the opposite mechanism to that of the two-point crossover. Elements between two points A and B in the target solution are retained, and then the non-repetitive elements of another solution fill other positions in turn.
- **Position-based crossover**: Several positions are randomly selected, and the elements in these positions of the target solution remain. As with the above crossover operators, the non-repetitive elements of another solution fill other positions in turn.
- **Cycle crossover**: This operator uses a circular mode to determine the positions of reserved elements in the target solution. More specifically, the first position s_1 is randomly selected, and then the element in s_1 of another solution is defined as π_1. Then, the position of element π_1 in the target solution is selected as the second position s_2. These steps are repeated until the position s_i is equal to the first position s_1. After determining the positions, the same method is used to keep the corresponding elements. The non-repetitive elements of another solution fill other positions in turn.

C. Mutation operators

This section explains the three mutation operators. Only one operator is applied to a solution at each time. The decision of choosing one mutation operator is determined by the Q-learning-based strategy, explained in Section IV-D. The details of the mutation operators are:

- **Swap**: Two different positions s_1 and s_2 in the target solution are randomly selected, and the elements in positions s_1 and s_2 are swapped.
- **Forward insert**: Two different positions s_1 and s_2 in the target solution are randomly selected. The element in position s_2 moves to position s_1, and the elements between positions s_1 and $s_2 - 1$ move one position backwards.
- **Backward insert**: Two different positions s_1 and s_2 in the target solution are randomly selected. The element in position s_1 moves to position s_2, and the elements between positions $s_1 + 1$ and s_2 move one position forwards.



(a) Two-point crossover.

(b) Single-point crossover.

(c) Order crossover.

(d) Position-based crossover.

(e) Cycle crossover.

Fig. 3. Application of the five crossover operators included in the RL-MOEA.

D. Q-learning based strategy

A new Q-learning-based strategy is included to select the best set of crossover and mutation operators to apply to the solutions at each iteration. This strategy dynamically evaluates the improvement of each operator at each iteration. Based on the historical and current evaluation information, the algorithm adaptively selects the operator with the maximum probability of improving the Pareto front solutions.

In brief, before each iteration, the Q-learning strategy selects an action (crossover or mutation operator) with the maximum Q-value based on the current state. Then, after executing the selected operator, this strategy will update the Q-value table and current state for the next iteration. The pseudo-code of the crossover and mutation phases considering this strategy are presented in Algorithms 2 and 3, respectively. The Q-learning strategy involves four elements: state, action, Q-value table, and reward function [34], which are explained in the next paragraphs.

This Q-learning strategy develops three states: start, improvement and no-improvement. Before the first iteration, and since the algorithm does not perform any crossover or mutation operator, the state is at start. At each iteration, if the external archive POS is updated by these newly generated solutions in the corresponding phase (crossover or mutation phase), the state will turn to improvement. Otherwise, the state becomes no-improvement. The action in this context means the crossover or mutation operator. That is, there are 5 actions in the crossover phase and 3 actions in the mutation phase.

The Q-value table is used to store the decision information. It is an $S \times A$ two-dimensional matrix where S and A are the respective numbers of states and actions. This work uses $Q^{cr}_{3\times5}$ and $Q^{mu}_{3\times3}$ to respectively present the Q-value tables in crossover and mutation phases. The initial values of the Q-value tables are 0. After applying the different evolution operator at each iteration, the corresponding Q-value table is updated according to the Behrman equation calculated by Eq. (24).

$$newQ_{s,a} = (1-\alpha)\cdot Q_{s,a} + \alpha \cdot (R_{s,a} + \gamma \cdot \max \tilde{Q}_{s,a}) \quad (24)$$

In the above equation, $newQ_{s,a}$ and $Q_{s,a}$ are the updated and current Q-values of current state s and action a. $\max \tilde{Q}_{s,a}$ is the maximum Q-value in the next current state. Generally, this value is estimated by the corresponding current value. $R_{s,a}$ is the reward value after taking action a under the state s. α and γ are the importance coefficients.

For the reward function, in this work the reward values are determined based on the number of newly generated solutions that improve the external archive POS. If there are v new generated solutions improving the POS, the corresponding reward value adds v. Otherwise, this value is subtracted from the population size (PS).

Algorithm 2: Crossover operator

```
1:  Use the Q-learning-based strategy to determine the operator $a^{cr}$;
2:  For ($i = 1$ to $PS$) do;
3:      If $rand(0,1) < CR$ do
4:          Using similarity-based selection to select a crossover object $\Pi^{ano}$;
5:          Generate new solution $\Pi^i_{new}$ by executing the operator $a^{cr}$ on $\Pi^i$ and $\Pi^{ano}$;
6:          Put $\Pi^i_{new}$ into OffPopulation;
7:      End If
8:  End For
9:  Counter the number $num$ of new non-dominated solutions;
10: If ($num \neq 0$) do
11:     $s^{cr} = improvement$;
12:     Update $Q^{cr}_{3\times5}$;
13:     Update $POS$ with these non-dominated solutions;
14: Else
15:     $s^{cr} = no\text{-}improvement$;
16:     Update $Q^{cr}_{3\times5}$;
17: End If
```

Algorithm 3: Mutation operator

```
1:  Use the Q-learning-based strategy to determine the operator $a^{mu}$;
2:  For ($i = 1$ to $PS$) do;
3:      If $rand(0,1) < MR$ do
4:          Generate new solution $\Pi^i_{new}$ by executing the operator $a^{mu}$ on $\Pi^i$;
5:          Put $\Pi^i_{new}$ into OffPopulation;
6:      End If
7:  End For
8:  Counter the number $num$ of new non-dominated solutions;
9:  If ($num \neq 0$) do
10:     $s^{mu} = improvement$;
11:     Update $Q^{mu}_{3\times3}$;
12:     Update $POS$ with these non-dominated solutions;
13: Else
14:     $s^{mu} = no\text{-}improvement$;
15:     Update $Q^{mu}_{3\times3}$;
16: End If
```

E. Temporal adaptive strategy for operator probability

MOEAs usually define a fixed probability for the crossover and mutation operators, independently of the iterations of the algorithm. However, as the number of iterations increases, the improvement effect of both operators can change. Hence, the decision on how to balance the probabilities of the crossover and mutation operators at every iteration is a challenge. Generally, the crossover operators have a strong global search capability at the beginning, but the improvement effect gradually decreases with the increase of iteration times. The mutation operators have the opposite effect [35]. Hence, the RL-MOEA includes a time-based strategy to effectively adapt crossover and mutation operator probability over time.

Since the crossover operators perform better at the beginning, this strategy sets the crossover probability (CR) to 1 at the first iteration to explore the whole solution space. The mutation probability (MR) is set as 0 at the first iteration. In the subsequent comparison experiments and to ensure fairness, in this work the stopping criterion is set as a CPU time limit of $n \cdot n \cdot p$ milliseconds. Then, CR decreases linearly depending on the running time, as presented in Eq. (25), and MR is updated by $1 - CR$.

$$CR = 1.0 - \frac{1}{n \cdot n \cdot p} \cdot T \tag{25}$$

with T being the current running time. When the running time reaches the stopping criterion ($n \cdot n \cdot p$), CR becomes 0, while MR is 1. Through this strategy, the RL-MOEA adjusts the probabilities of the different operations in an adaptive way and then maximizes the search capacity of different operators at each iteration.

V. Experimental study

In this section, extensive sets of experiments and analyses are conducted to investigate the effectiveness of the RL-MOEA for the r-MMALBP. The experimental setting (Section V-A), parameter calibration (Section V-B) and experimental comparisons (Sections V-C to V-F) are detailed below. Section V-G offers a global analysis and explores any limitations.

A. Experimental setting and evaluation indicators

To analyze the performance of RL-MOEA, a total of 269 instances are first generated and organized into 22 groups according to the task numbers. These groups are marked as P7, P8, P9, P11, P21, P25, P28, P29, P30, P32, P35, P45, P53, P58, P70, P75, P83, P89, P94, P111, P148 and P297. These groups respectively contain 6, 1, 5, 9, 6, 6, 6, 7, 9, 6, 7, 10, 5, 16, 16, 24, 16, 23, 13, 17, 35 and 26 instances. The results of the algorithms shown in the experimental study are averaged for all the instances of each group. Data about the instances can be found at *assembly-line-balancing.de* while parameters d_{qe}, t_{iq}, u_{max}, Δ_{CT}, E, and Q of each instance are in [3]. Since this work focuses on the cost-oriented r-MMALBP, the corresponding cost parameters COO, COW, CPO and CPW are set using a uniform probability distribution in [3, 10]. All the algorithms were programmed in C++. The experiments were launched in a computer with an Intel Core i5 with 2.80 GHz and 2 GB of memory.

Unary and binary multi-objective performance indicators are used to evaluate the obtained Pareto sets: the hypervolume ratio (HVR) [36] and the coverage indicator (C) [37]. The unary indicator HVR is the ratio between the hypervolume of the obtained Pareto set and that of the true Pareto set. The closer to 1 the HVR value of a Pareto set is, the better the approximation to the true frontier. The binary indicator C computes the domination relation between two Pareto sets P and Q. A $C(P, Q)$ value closer to 1 means that Pareto set Q is strongly dominated by Pareto set P.

B. Parameter calibration

The RL-MOEA parameters are calibrated using a multi-factor ANOVA. The RL-MOEA has three main parameters: population size PS, and two coefficients used in the Q-learning-based strategy α and γ. The levels of these parameters are listed below:

- PS (9 levels): 20, 30, 40, 50, 60, 70, 80, 90, 100.
- Q-learning α (5 levels): 0.4, 0.5, 0.6, 0.7, 0.8.
- Q-learning γ (5 levels): 0.1, 0.2, 0.3, 0.4, 0.5.

Through the full factorial design, a total of $9 \times 5 \times 5 = 225$ experiment configurations are involved. All the configurations are run with 5 calibration instances, and each instance is solved 5 times under a CPU time limit of $n \times n \times 5$. Hence, a total of $225 \times 5 \times 5 = 5,625$ experiments are carried out. The HVR values are regarded as the final response value. The best parameter configuration is $\{PS = 30, \alpha = 0.6, \gamma = 0.3\}$.

C. Effectiveness of the improvement strategies

The RL-MOEA incorporates two main improvement strategies: a Q-learning-based strategy and a time-based probability-adaptive strategy (TBPA), previously explained in Sections IV-D and IV-E. Two strategies are considered for the TBPA. The first (TBPA1) reduces the mutation probability over time, whereas the second (TBPA2) keeps the crossover and mutation probabilities constant. Thus, this section compares 5 variants of the RL-MOEA and evaluates them to demonstrate the effectiveness or otherwise of the proposed improvements. The first variant (MOEAQ) includes a random selection of the operators (no Q-learning strategy) and TBPA1; the second (MOEAS) includes a random selection and TBPA2; the third (MOEAL) includes a random selection and TBPA; the fourth (MOEAM) includes a Q-learning-based strategy and TBPA1; the fifth (MOEAN) includes a Q-learning-based strategy and TBPA2.

The RL-MOEA and its variants are independently run 10 times for the same maximum computing time (i.e., $n \times n \times 20$, being n the number of tasks of the instance). The final results are reported in Tables V-VI. It can be seen in Table V that the HVR value of the RL-MOEA is closest to 1.0, which suggests that the proposed RL-MOEA outperforms its variants. In Table VI, the average value C(RL-MOEA, variants) is larger than C(variants, RL-MOEA), which indicates that more than half of the solutions obtained by these variants are dominated by

TABLE V
AVERAGE HVR FOR RL-MOEA AND VARIANTS

Instances	MOEAL	MOEAM	MOEAN	MOEAQ	MOEAS	RL-MOEA
P7	1	1	1	1	1	1
P8	1	1	1	1	1	1
P9	1	1	1	1	1	1
P11	1	1	1	1	1	1
P21	0.91	0.89	0.90	0.88	0.90	0.90
P25	0.97	0.98	0.97	0.98	0.97	0.98
P28	0.93	0.94	0.94	0.93	0.94	0.93
P29	0.87	0.86	0.88	0.86	0.84	0.89
P30	0.90	0.89	0.89	0.88	0.89	0.90
P32	0.88	0.88	0.89	0.87	0.88	0.89
P35	0.84	0.83	0.84	0.84	0.85	0.83
P45	0.93	0.93	0.94	0.93	0.93	0.94
P53	0.87	0.89	0.87	0.88	0.89	0.88
P58	0.87	0.87	0.87	0.87	0.86	0.88
P70	0.87	0.85	0.87	0.85	0.85	0.88
P75	0.90	0.90	0.89	0.90	0.88	0.91
P83	0.80	0.78	0.79	0.79	0.79	0.81
P89	0.89	0.89	0.89	0.88	0.88	0.91
P94	0.88	0.87	0.88	0.88	0.86	0.88
P111	0.88	0.87	0.88	0.87	0.87	0.89
P148	0.88	0.87	0.87	0.87	0.86	0.89
P297	0.82	0.82	0.82	0.82	0.80	0.84
Avg.	0.88	0.88	0.88	0.88	0.87	**0.89**

the Pareto solutions obtained by the RL-MOEA. In small-scale instances (e.g., P7, P8, P9, P11, P21 and P25), the results of the RL-MOEA and its variants have the same or similar convergence and diversity. However, when the scale increases, especially in middle- and large-scale instances, the RL-MOEA obtains solutions with better diversity and convergence due to the interaction of the proposed strategies.

That is, the solutions obtained by the variants can only dominate a few solutions obtained by the RL-MOEA. The RL-MOEA with all the components obtained better results than all its variants. It can therefore be inferred that all the components are justified to be included in the final version of the MOEA for its comparison with other algorithms, as reported in the following sections.

D. Computational time and complexity

First, this section reports the running times of an iteration for all of the MOEAs for all the instances. The running times and the relative percentage index values between running times are reported in Appendix I. Average running times of EJAYA, ICA, MABC, MOEA1, MOEA2, MOEA3, MOEA4, MOEA5, MOPSO, MOSA, SPEA2 and RL-MOEA were 7123.58, 789.30, 4902.94, 2401.35, 2388.64, 2371.57, 2313.37, 2347.06, 2370.47, 39570.88, 4740.02 and 2398.57 ms, respectively. The corresponding average relative percentage index was 7.69, 0.04, 5.90, 1.84, 1.88, 1.87, 1.90, 2.00, 1.88, 45.69, 4.93 and 2.12, respectively. It can be observed that RL-MOEA is only slightly more complex than ICA, but has a similar computational complexity to that of MOEA1-5 and MOPSO.

Additionally, an approximation for the computational complexity of the proposed RL-MOEA is depicted. The computational complexity of the proposed decoding mechanism is $O(u_{max} \cdot n)$. At each generation, the complexity value of both crossover and mutation phases is $O(PS \cdot u_{max} \cdot n)$. Hence, the overall computational complexity of RL-MOEA is $O(PS \cdot u_{max} \cdot n)$.

E. Comparison with other algorithms

To further test the performance of the RL-MOEA, this section compares the proposal with ten mined rules (TMR) generated by a recent gene expression programming approach [3]. As the gene expression programming approach was not multi-objective, this work generates, for each instance, 10 mined rules to obtain 10 solutions that will conform the non-dominated solutions that form the Pareto set. Additionally, this work compares the RL-MOEA against 11 state-of-the-art MOEAs: EJAYA [32], imperialist competitive algorithm (ICA) [38], multi-objective artificial bee colony (MABC) [39], MOEA1 [40], MOEA2 [40], MOEA3 [40], MOEA4 [40], MOEA5 [40], multi-objective particle swarm optimization (MOPSO) [41], multi-objective simulated algorithm (MOSA) [42] and improved strength Pareto evolutionary algorithm (SPEA2) [43]. These MOEAs were selected due to their successful application in ALB problems. To apply them to this new problem, the proposed decoding was used to determine task assignment and calculate objective values.

To make a fair comparison, all the algorithms were run with the same CPU time limit of $n \times n \times 20$. 269 benchmark instances are solved by these meta-heuristics 10 times. Table VII reports the average HVR for TMR, meta-heuristics and RL-MOEA. When using HVR to rank these MOEAs, RL-MOEA appears in first position followed in order by MABC, MOEA5, EJAYA, SPEA2, MOEA1, MOEA3, MOEA2, MOEA4, ICA, MOSA, MOPSO and TMR. These results indicate that the proposed RL-MOEA outperforms the set of 11 MOEAs and TMR.

Tables VIII and IX reports the average values of C(RL-MOEA, Q) and C(P, RL-MOEA). Regarding the average coverage values of RL-MOEA and MOEAs (or TMR), most of the solutions obtained by these MOEAs and TMR are dominated by the Pareto set obtained by RL-MOEA. Inversely, the solutions obtained by these MOEAs and TMR can only dominate very few solutions obtained by RL-MOEA.

In short, for the small-scale instances such as P7, P8, P9, P11, P21 and P25, RL-MOEA and other MOEAs have the same or similar performance in terms of unary and binary indicators. In the middle- and large-scale instances, RL-MOEA obtains the highest HVR, and C(RL-MOEA, ...) is larger than C(..., RL-MOEA). It is demonstrated that RL-MOEA has better diversity and convergence than other MOEAs. In other words, the non-dominated solutions achieved by RL-MOEA are close to the true Pareto set. Therefore, the proposed RL-MOEA is superior to TMR, EJAYA, ICA, MABC, MOEA1, MOEA2, MOEA3, MOEA4, MOEA5, MOPSO, MOSA and SPEA2 in tackling the r-MMALBP.

The ANOVA test was also applied for the set of results given in this section. Fig.4 presents the means plot of HVR with Tukey's honest significant difference setting 95% confidence intervals. The same conclusion as drawn above can be inferred from Fig.4, namely that the proposed RL-MOEA outperforms

TABLE VI
AVERAGE COVERAGE FOR RL-MOEA AND VARIANTS

Instances	C(RL-MOEA, ...)					C(..., RL-MOEA)				
	MOEAL	MOEAM	MOEAN	MOEAQ	MOEAS	MOEAL	MOEAM	MOEAN	MOEAQ	MOEAS
P7	0.17	0.17	0.17	0.17	0.17	0	0	0	0	0
P8	0	0	0	0	0	0	0	0	0	0
P9	0	0	0	0	0	0	0	0	0	0
P11	0	0	0	0	0	0	0	0	0	0
P21	0.14	0.15	0.18	0.21	0.18	0	0	0	0	0
P25	0.33	0.33	0.34	0.33	0.33	0	0	0	0	0
P28	0.50	0.48	0.50	0.50	0.50	0.02	0.02	0.03	0	0.03
P29	0.62	0.60	0.57	0.59	0.64	0.02	0.02	0	0.01	0.01
P30	0.56	0.61	0.60	0.59	0.60	0.03	0.01	0	0.01	0
P32	0.59	0.54	0.54	0.67	0.54	0.02	0.02	0.03	0	0.01
P35	0.74	0.80	0.77	0.85	0.77	0	0.01	0	0	0.01
P45	0.50	0.47	0.49	0.49	0.51	0	0.02	0	0	0
P53	0.40	0.40	0.40	0.40	0.40	0	0	0	0	0
P58	0.91	0.88	0.90	0.90	0.91	0.05	0.04	0.04	0.04	0.04
P70	0.75	0.75	0.77	0.76	0.77	0.02	0.01	0.03	0.02	0.02
P75	0.93	0.95	0.97	0.95	0.96	0.03	0.02	0.01	0.02	0.01
P83	0.82	0.86	0.87	0.87	0.88	0.04	0.03	0.03	0.04	0.02
P89	0.93	0.95	0.92	0.94	0.98	0.02	0.02	0.02	0.02	0.01
P94	0.86	0.86	0.89	0.90	0.89	0.03	0.03	0.02	0.01	0.02
P111	0.81	0.86	0.83	0.82	0.86	0.04	0.02	0.03	0.04	0.02
P148	0.89	0.91	0.92	0.90	0.92	0.02	0.02	0.02	0.01	0.01
P297	0.93	0.94	0.94	0.95	0.96	0.04	0.03	0.02	0.02	0.02
Avg.	**0.73**	**0.74**	**0.74**	**0.74**	**0.75**	0.02	0.02	0.02	0.02	0.01

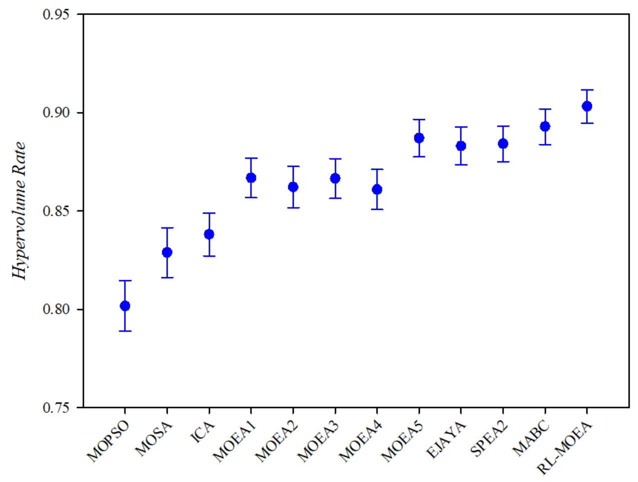


Fig. 4. Means plot of HVR with Tukey's honest significant difference 95% confidence intervals for all the algorithms.

EJAYA, ICA, MABC, MOEA1, MOEA2, MOEA3, MOEA4, MOEA5, MOPSO, MOSA and SPEA2.

F. Differential attainment functions

In this section a comparison is made in qualitative terms of the Pareto sets from all the MOEAs. For this purpose, empirical attainment functions (EAF) and differential empirical attainment functions (Diff-EAF) were employed. This section mainly analyzes the proposed RL-MOEA and the second-best MOEA, the MABC, for benchmark instances 88 and 174. Each algorithm solves each instance 100 times to obtain 100 Pareto sets. Following the approach by Grunert et al. [44], this work depicts the EAFs and the Diff-EAFs for RL-MOEA and MABC for instances 88 and 174 in Figs. 5 and 6. In the EAF plots, areas in intense colors, such as red and orange, correspond to high dominance of the algorithm, while areas in other colors correspond to low or null dominance. In the Diff-EAF plots, different colors indicate different dominant probability by RL-MOEA over MABC.

In Fig.5(c), boundaries around objective f_1 are colored in green, and those around objective f_2 are colored in yellow or red. That is, in the boundaries, most Pareto front solutions are found by RL-MOEA, while only a few solutions are obtained by MABC. Similarly, in Fig.6(c), the borders around objectives f_1 and f_2 are colored in blue, yellow or red, which leads to the same conclusion. Hence, this qualitative analysis verifies the conclusion previously obtained, namely that the proposed RL-MOEA outperforms MABC in both objective spaces.

G. Analysis of the results and limitations

This section analyzes the reasons that lead to the above results. First, EJAYA, ICA and MOPSO use different solutions to guide the evolution of the population. MOSA relies on a neighborhood operator to explore new solution spaces and avoids falling into local optimization through temperature-based acceptance. MOEA1, MOEA2, MOEA3, MOEA4, MOEA5, MABC, and SPEA2 organize crossover and mutation operators through different search modes for global and local search. These algorithms are sensitive to the instance scale of the problem. As the instance scale grows, their performance degrades.

The original RL-MOEA has similar properties to the above MOEAs. To solve this drawback, a Q-learning-based strategy was designed to dynamically select appropriate crossover and mutation operators at each iteration, which explains why RL-MOEA outperformed MOEAS and MOEAQ. In addition, a time-based probability-adaptive strategy was developed to adaptively adjust crossover and mutation probabilities in order

TABLE VII
AVERAGE HVR FOR TMR, META-HEURISTICS AND RL-MOEA

Instances	TMR	EJAYA	ICA	MABC	MOEA1	MOEA2	MOEA3	MOEA4	MOEA5	MOPSO	MOSA	SPEA2	RL-MOEA
P7	0.94	1	1	1	1	1	1	1	1	1	1	1	1
P8	1	1	1	1	1	1	1	1	1	1	1	1	1
P9	0.68	1	1	1	1	1	1	1	1	1	1	1	1
P11	0.80	1	1	1	1	1	1	1	1	1	1	0.99	1
P21	0.53	0.90	0.88	0.92	0.90	0.91	0.90	0.90	0.90	0.87	0.91	0.90	0.91
P25	0.54	0.98	0.96	0.98	0.98	0.98	0.98	0.98	0.98	0.96	0.98	0.98	0.99
P28	0.76	0.93	0.90	0.93	0.93	0.92	0.93	0.90	0.94	0.90	0.92	0.93	0.94
P29	0.43	0.89	0.83	0.93	0.88	0.88	0.88	0.85	0.89	0.78	0.86	0.89	0.93
P30	0.73	0.93	0.89	0.93	0.92	0.92	0.92	0.91	0.92	0.86	0.91	0.92	0.93
P32	0.24	0.83	0.76	0.89	0.89	0.86	0.85	0.86	0.86	0.75	0.84	0.86	0.91
P35	0.40	0.85	0.82	0.88	0.87	0.85	0.88	0.85	0.82	0.81	0.85	0.83	0.86
P45	0.70	0.89	0.89	0.90	0.89	0.89	0.89	0.89	0.89	0.83	0.88	0.88	0.90
P53	0.58	0.96	0.87	0.96	0.96	0.96	0.96	0.96	0.97	0.89	0.94	0.95	0.97
P58	0.67	0.87	0.83	0.87	0.84	0.84	0.84	0.84	0.88	0.79	0.82	0.87	0.89
P70	0.55	0.88	0.82	0.90	0.88	0.87	0.87	0.87	0.88	0.79	0.82	0.90	0.91
P75	0.62	0.88	0.83	0.91	0.85	0.85	0.85	0.85	0.88	0.78	0.82	0.89	0.90
P83	0.35	0.85	0.77	0.83	0.81	0.80	0.81	0.81	0.84	0.74	0.76	0.82	0.87
P89	0.44	0.88	0.79	0.88	0.84	0.83	0.84	0.82	0.89	0.74	0.76	0.87	0.91
P94	0.59	0.86	0.82	0.88	0.85	0.85	0.85	0.86	0.87	0.79	0.81	0.86	0.88
P111	0.55	0.88	0.85	0.89	0.86	0.86	0.87	0.86	0.89	0.80	0.83	0.89	0.90
P148	0.63	0.87	0.84	0.88	0.85	0.85	0.85	0.85	0.88	0.79	0.81	0.87	0.89
P297	0.60	0.80	0.75	0.81	0.77	0.76	0.77	0.77	0.82	0.69	0.68	0.82	0.84
Avg.	0.58	0.88	0.84	0.89	0.87	0.86	0.87	0.86	0.89	0.80	0.83	0.88	**0.90**

TABLE VIII
AVERAGE COVERAGE VALUES FOR C(RL-MOEA, MOEAS/TMR)

Instances	TMR	EJAYA	ICA	MABC	MOEA1	MOEA2	MOEA3	MOEA4	MOEA5	MOPSO	MOSA	SPEA2
P7	0.17	0	0	0	0	0	0	0	0	0	0	0
P8	0	0	0	0	0	0	0	0	0	0	0	0
P9	0.40	0	0	0	0	0	0	0	0	0	0	0
P11	0.22	0	0	0	0	0	0	0	0	0	0	0
P21	0.50	0.24	0.35	0.20	0.23	0.22	0.21	0.27	0.22	0.34	0.26	0.25
P25	1	0.01	0.16	0	0.02	0	0	0.03	0.02	0.13	0.01	0.02
P28	0.33	0.02	0.17	0.10	0.05	0.10	0.07	0.17	0.03	0.21	0.14	0.01
P29	1	0.48	0.78	0.40	0.54	0.61	0.52	0.67	0.37	0.82	0.71	0.47
P30	0.56	0.31	0.57	0.37	0.37	0.44	0.45	0.50	0.32	0.59	0.45	0.34
P32	1	0.39	0.85	0.31	0.38	0.47	0.46	0.45	0.30	0.77	0.58	0.29
P35	1	0.67	0.82	0.58	0.67	0.72	0.64	0.75	0.68	0.73	0.73	0.60
P45	0.50	0.16	0.22	0.13	0.18	0.17	0.20	0.20	0.17	0.25	0.23	0.14
P53	0.70	0.09	0.31	0.06	0.11	0.12	0.10	0.10	0.03	0.33	0.21	0.13
P58	0.66	0.90	0.99	0.83	0.97	0.98	0.96	0.98	0.82	0.98	0.99	0.84
P70	0.84	0.73	0.81	0.61	0.76	0.74	0.74	0.77	0.60	0.81	0.78	0.57
P75	0.84	0.89	0.99	0.63	0.99	0.99	0.98	0.97	0.87	1	1	0.87
P83	1	0.87	0.99	0.90	0.97	0.98	0.98	0.98	0.87	0.99	1	0.84
P89	0.96	0.91	1	0.86	0.98	0.99	0.99	1	0.85	1	1	0.86
P94	1	0.72	0.91	0.63	0.80	0.82	0.83	0.82	0.69	0.93	0.91	0.64
P111	0.63	0.80	0.92	0.74	0.87	0.87	0.88	0.90	0.69	0.90	0.91	0.71
P148	0.78	0.85	0.92	0.63	0.90	0.91	0.91	0.91	0.80	0.95	0.94	0.82
P297	0.30	0.94	0.98	0.68	0.98	0.98	0.98	0.98	0.87	1	0.99	0.85
Avg.	0.71	0.66	0.77	0.55	0.71	0.72	0.71	0.73	0.61	0.77	0.75	0.61

to effectively collaborate the crossover and mutation operators at each iteration. RL-MOEA outperformed MOEAM and MOEAN and, when compared with the 11 evaluated MOEAs, the superiority of RL-MOEA is due to these two strategies.

It should be acknowledged that the RL-MOEA does present some limitations. All the improvements are related to the specific knowledge of this multi-objective variant. This means the algorithm has not been validated in other ALB and similar industrial problems (e.g., industrial scheduling). When designing the RL-MOEA for other problems, elements of the Q-learning-based strategy such as action, state and reward function may need to be redesigned considering the specific attributes of the problem. Additionally, this work has incorporated specific crossover and mutation operators. Again, these methods use existing knowledge about the problem and are *ad hoc* operators. A mechanism to generate the most appropriate operators for the features of a given problem could be a future research line. Finally, the MOEA used in the proposal is simple. This work did not study combinations of specific operators and RL mechanisms to more sophisticated MOEAs. While such considerations are beyond the scope of the present study, they could constitute a fruitful future research line.

TABLE IX
AVERAGE COVERAGE VALUES FOR C(MOEAS/MR, RL-MOEA)

Instances	TMR	EJAYA	ICA	MABC	MOEA1	MOEA2	MOEA3	MOEA4	MOEA5	MOPSO	MOSA	SPEA2
P7	0	0	0	0	0	0	0	0	0	0	0	0
P8	0	0	0	0	0	0	0	0	0	0	0	0
P9	0	0	0	0	0	0	0	0	0	0	0	0
P11	0	0	0	0	0	0	0	0	0	0	0	0
P21	0	0	0	0	0	0	0	0	0	0	0	0
P25	0	0	0	0	0	0	0	0	0	0	0	0
P28	0	0.02	0	0	0.02	0	0	0	0	0	0	0
P29	0	0.01	0	0.04	0.01	0.01	0.01	0	0.01	0	0	0.03
P30	0.11	0.04	0	0.01	0.02	0.02	0.02	0.01	0.04	0	0.01	0.04
P32	0	0.02	0	0.04	0	0	0	0	0	0	0	0.01
P35	0	0.01	0.01	0.04	0.02	0	0.04	0.01	0.01	0.01	0.02	0.02
P45	0	0	0.01	0.02	0	0	0	0	0	0	0	0.01
P53	0	0.03	0.01	0	0	0	0.03	0	0.01	0	0	0.04
P58	0.11	0.04	0.01	0.07	0.01	0	0.02	0	0.07	0	0	0.05
P70	0	0	0	0.06	0.01	0.01	0	0.01	0.03	0	0	0.04
P75	0.06	0.05	0.01	0.24	0	0	0	0.01	0.07	0	0	0.06
P83	0	0.02	0.01	0.03	0	0	0	0	0.03	0	0	0.02
P89	0	0.02	0	0.06	0.01	0	0	0	0.06	0	0	0.04
P94	0	0.02	0.02	0.09	0.01	0.01	0.01	0.02	0.06	0	0	0.07
P111	0.21	0.03	0.02	0.07	0.01	0.01	0	0	0.05	0	0.01	0.08
P148	0.02	0.03	0.03	0.15	0.01	0.01	0.01	0.01	0.04	0	0.01	0.05
P297	0.19	0.02	0.02	0.13	0.01	0.01	0.01	0.01	0.06	0	0	0.07
Avg.	0.05	0.02	0.01	0.08	0.01	0.01	0.01	0.01	0.04	0.00	0.00	0.04

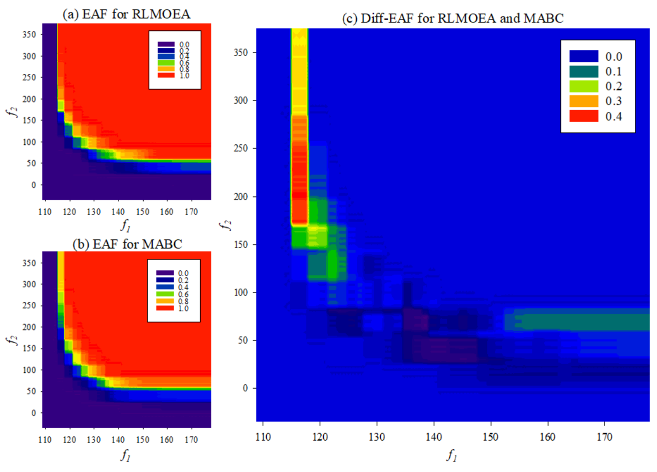


Fig. 5. EAF and Diff-EAF for RL-MOEA and MABC in instance 88.

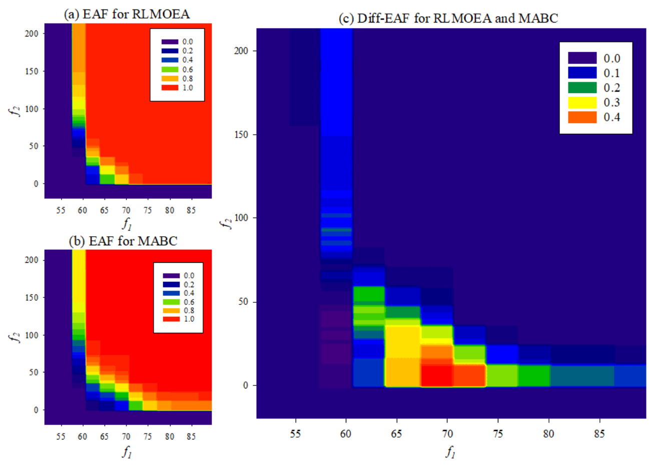


Fig. 6. EAF and Diff-EAF for RL-MOEA and MABC in instance 174.

VI. CONCLUSIONS, MANAGERIAL INSIGHTS AND FUTURE WORKS

This work presented a robust multi-objective r-MMALBP variant via a MILP model and proposed the RL-MOEA to solve it. The MILP model considers total hired worker costs and opened workstation costs as the first objective, and penalty costs of all workers and overloaded workstations under all demand plans as the second objective. The proposed RL-MOEA has a priority-based solution representation and a new task-worker-sequence solution decoding. A Q-learning-based strategy is proposed to learn the most appropriate crossover and mutation operators from problem-specific crossover and mutation operators. Additionally, a temporal adaptive strategy is developed to effectively find the best operator probabilities at each iteration.

The final experimental results demonstrate the effectiveness of the Q-learning-based strategy and the time-based probability-adaptive strategy, as well as the superiority of the RL-MOEA over a multi-objective variant of the only existing approach for the r-MMALBP (a gene programming approach for mined rules) and 11 different MOEAs. The results were consistent over a wide range of experiments using qualitative and quantitative indicators. All the novel components of the RL-MOEA (i.e., the Q-learning strategy and temporal adaptive strategy) are justified and improved the performance of the algorithm. This work also showed that the computational cost of the proposal is competitive with respect to the other MOEAs.

The managerial relevance of the proposed RL-MOEA can be implemented when two requirements are met: (1) the production layout is a mixed-model multi-manned assembly line; and (2) product demand is uncertain. When using the RL-MOEA for real applications, managers could first refer to the historical demand plans to determine the demand plan parameters. Then, the managerial team can develop a robust MILP model, considering actual labor and equipment costs. Finally, they could use the proposed RL-MOEA to obtain a set of non-dominated solutions to their industrial problem. From this Pareto set, managers could select one solution taking into account the importance of the considered objectives, which are normally set by stakeholders and chief officer positions. At the practical implementation stage, decision makers can assign tasks to the workers for all the workstations in order to better organize the assembly line layout.

Future research could be extended in different ways. First, new practical constraints such as setup times and resource limitations should be considered to enrich the robustness model. Second, other MOEAs could be embedded with the Q-learning process to enhance their adaptive and self-learning abilities. Another line could be to take into consideration stakeholder preferences and bias the search for some specific areas of the Pareto fronts. Finally, environmental characteristics could be included to enrich the model and the objectives could be extended by adding a sustainability feature when optimizing the MALBP.

References

[1] Z. Zhang, Q. Tang, and M. Chica, "Multi-manned assembly line balancing with time and space constraints: A MILP model and memetic ant colony system," *Computers & Industrial Engineering*, vol. 150, Dec. 2020.

[2] Y.-Y. Chen, "A hybrid algorithm for allocating tasks, operators, and workstations in multi-manned assembly lines," *Journal of Manufacturing Systems*, vol. 42, pp. 196-209, Jan. 2017.

[3] Z. Zhang, Q. Tang, and M. Chica, "A robust MILP and gene expression programming based on heuristic rules for mixed-model multi-manned assembly line balancing," *Applied Soft Computing*, vol. 109, Sep. 2021.

[4] J. M. Nilakantan, Z. Li, Q. Tang, and P. Nielsen, "Multi-objective co-operative co-evolutionary algorithm for minimizing carbon footprint and maximizing line efficiency in robotic assembly line systems," *Journal of Cleaner Production*, vol. 156, pp. 124-136, Jul. 2017.

[5] K. Wang, X. Li, L. Gao, P. Li, and J. W. Sutherland, "A Discrete Artificial Bee Colony Algorithm for Multiobjective Disassembly Line Balancing of End-of-Life Products," *IEEE Transactions on Cybernetics*, vol. 52, no. 8, pp. 7415-7426, Aug. 2022.

[6] W. Li, L. He, and Y. Cao, "Many-Objective Evolutionary Algorithm With Reference Point-Based Fuzzy Correlation Entropy for Energy-Efficient Job Shop Scheduling With Limited Workers," *IEEE Transactions on Cybernetics*, vol. 52, no. 10, pp. 10721-10734, Oct. 2022.

[7] Q.-K. Pan, L. Gao, and L. Wang, "An Effective Cooperative Co-Evolutionary Algorithm for Distributed Flowshop Group Scheduling Problems," *IEEE Transactions on Cybernetics*, vol. 52, no. 7, pp. 5999-6012, Jul. 2022.

[8] Z. Pan, D. Lei, and L. Wang, "A Knowledge-Based Two-Population Optimization Algorithm for Distributed Energy-Efficient Parallel Machines Scheduling," *IEEE Transactions on Cybernetics*, vol. 52, no. 6, pp. 5051-5063, Jun 2022.

[9] Y. Pan, K. Gao, Z. Li, and N. Wu, "Solving Biobjective Distributed Flow-Shop Scheduling Problems With Lot-Streaming Using an Improved Jaya Algorithm," *IEEE Transactions on Cybernetics* (early access), Apr. 2022.

[10] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep Reinforcement Learning for Multiagent Systems: A Review of Challenges, Solutions, and Applications," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3826-3839, Sep. 2020.

[11] Z. Abidin Çil and D. Kizilay, "Constraint programming model for multi-manned assembly line balancing problem," *Computers & Operations Research*, vol. 124, Dec. 2020.

[12] T. Kellegöz and B. Toklu, "A priority rule-based constructive heuristic and an improvement method for balancing assembly lines with parallel multi-manned workstations," *International Journal of Production Research*, vol. 53, no. 3, pp. 736-756, Feb. 2015.

[13] P. Fattahi, A. Roshani, and A. Roshani, "A mathematical model and ant colony algorithm for multi-manned assembly line balancing problem," *The International Journal of Advanced Manufacturing Technology*, vol. 53, no. 1-4, pp. 363-378, Mar. 2011.

[14] T. Kellegöz and B. Toklu, "An efficient branch and bound algorithm for assembly line balancing problems with parallel multi-manned workstations," *Computers & Operations Research*, vol. 39, no. 12, pp. 3344-3360, Dec. 2012.

[15] A. Roshani, A. Roshani, A. Roshani, M. Salehi, and A. Esfandyari, "A simulated annealing algorithm for multi-manned assembly line balancing problem," *Journal of Manufacturing Systems*, vol. 32, no. 1, pp. 238-247, Jan. 2013.

[16] A. S. Michels, T. C. Lopes, C. G. S. Sikora, and L. Magatão, "A Benders' decomposition algorithm with combinatorial cuts for the multi-manned assembly line balancing problem," *European Journal of Operational Research*, vol. 278, no. 3, pp. 796-808, Nov. 2019.

[17] Y.-Y. Chen, C.-Y. Cheng, and J.-Y. Li, "Resource-constrained assembly line balancing problems with multi-manned workstations," *Journal of Manufacturing Systems*, vol. 48, pp. 107-119, Jul. 2018.

[18] T. C. Lopes, G. V. Pastre, A. S. Michels, and L. Magatao, "Flexible multi-manned assembly line balancing problem: Model, heuristic procedure, and lower bounds for line length minimization," *Omega*, vol. 95, Sep. 2020.

[19] M. Sahin and T. Kellegoz, "A new mixed-integer linear programming formulation and particle swarm optimization based hybrid heuristic for the problem of resource investment and balancing of the assembly line with multi-manned workstations," *Computers & Industrial Engineering*, vol. 133, pp. 107-120, Jul. 2019.

[20] M. Chica, J. Bautista, Ó. Cordón, and S. Damas, "A multiobjective model and evolutionary algorithms for robust time and space assembly line balancing under uncertain demand," *Omega*, vol. 58, pp. 55-68, Jan. 2016.

[21] T. Monch, A. Huchzermeier, and P. Bebersdorf, "Variable takt times in mixed-model assembly line balancing with random customisation," *International Journal of Production Research*, vol. 59, no. 15, pp. 4670-4689, Aug. 2021.

[22] R. Ramezanian and A. Ezzatpanah, "Modeling and solving multi-objective mixed-model assembly line balancing and worker assignment problem," *Computers & Industrial Engineering*, vol. 87, pp. 74-80, Sep. 2015.

[23] W. C. Yang and W. M. Cheng, "Modelling and solving mixed-model two-sided assembly line balancing problem with sequence-dependent setup time," *International Journal of Production Research*, vol. 58, no. 21, pp. 6638-6659, Nov. 2020.

[24] Z. A. Çil, Z. Li, S. Mete, and E. Özceylan, "Mathematical model and bee algorithms for mixed-model assembly line balancing problem with physical human–robot collaboration," *Applied Soft Computing*, vol. 93, Aug. 2020.

[25] M. Chica, Ó. Cordón, S. Damas, and J. Bautista, "A robustness information and visualization model for time and space assembly line balancing under uncertain demand," *International Journal of Production Economics*, vol. 145, no. 2, pp. 761-772, Oct. 2013.

[26] H. Babazadeh, M. H. Alavidoost, M. H. F. Zarandi, and S. T. Sayyari, "An enhanced NSGA-II algorithm for fuzzy bi-objective assembly line balancing problems," *Computers & Industrial Engineering*, vol. 123, pp. 189-208, Sep. 2018.

[27] Z. Zhang, Q. Tang, Z. Li, and L. Zhang, "Modelling and optimisation of energy-efficient U-shaped robotic assembly line balancing problems," *International Journal of Production Research*, vol. 57, no. 17, pp. 5520-5537, Sep. 2019.

[28] Z. K. Zhang, Q. H. Tang, and L. P. Zhang, "Mathematical model and grey wolf optimization for low-carbon and low-noise U-shaped robotic assembly line balancing problem," *Journal of Cleaner Production*, vol. 215, pp. 744-756, Apr. 2019.

[29] Z. X. Li, M. N. Janardhanan, and S. G. Ponnambalam, "Cost-oriented robotic assembly line balancing problem with setup times: multi-objective algorithms," *Journal of Intelligent Manufacturing*, vol. 32, no. 4, pp. 989-1007, Apr. 2021.

[30] B. Zhou and Q. Wu, "Decomposition-based bi-objective optimization for sustainable robotic assembly line balancing problems," *Journal of Manufacturing Systems*, vol. 55, pp. 30-43, Apr. 2020.

[31] Z. K. Zhang, Q. H. Tang, R. Ruiz, and L. P. Zhang, "Ergonomic risk and cycle time minimization for the U-shaped worker assignment assembly line balancing problem: A multi-objective approach," *Computers & Operations Research*, vol. 118, Jun. 2020.

[32] Z. K. Zhang, Q. H. Tang, D. Y. Han, and X. B. Qian, “An enhanced multi-objective JAYA algorithm for U-shaped assembly line balancing considering preventive maintenance scenarios,” *International Journal of Production Research*, vol. 59, no. 20, pp. 6146-6165, Oct. 2021.
[33] F. Zhao, S. Di, and L. Wang, “A Hyperheuristic With Q-Learning for the Multiobjective Energy-Efficient Distributed Blocking Flow Shop Scheduling Problem,” *IEEE Transactions on Cybernetics* (early access), Aug. 2022.
[34] J. J. Ji, Y. N. Guo, X. Z. Gao, D. W. Gong, and Y. P. Wang, “Q-Learning-Based Hyperheuristic Evolutionary Algorithm for Dynamic Task Allocation of Crowdsensing,” *IEEE Transactions on Cybernetics* (early access), Oct. 2021.
[35] M. Abdel-Basset, R. Mohamed, and M. Abouhawwash, “Balanced multi-objective optimization algorithm using improvement based reference points approach,” *Swarm and Evolutionary Computation*, vol. 60, Feb. 2021.
[36] E. Zitzler and L. Thiele, “Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, Nov. 1999.
[37] E. Zitzler, K. Deb, and L. Thiele, “Comparison of Multiobjective Evolutionary Algorithms: Empirical Results,” *Evolutionary Computation*, vol. 8, no. 2, pp. 173-195, Jun. 2000.
[38] A. Nourmohammadi, M. Zandieh, and R. Tavakkoli-Moghaddam, “An imperialist competitive algorithm for multi-objective U-type assembly line design,” *Journal of Computational Science*, vol. 4, no. 5, pp. 393-400, Sep. 2013.
[39] U. Saif, Z. Guan, L. Zhang, F. Zhang, B. Wang, and J. Mirza, “Multi-objective artificial bee colony algorithm for order oriented simultaneous sequencing and balancing of multi-mixed model assembly line,” *Journal of Intelligent Manufacturing*, vol. 30, no. 3, pp. 1195-1220, Mar. 2019.
[40] P. T. Zacharia and A. C. Nearchou, “A population-based algorithm for the bi-objective assembly line worker assignment and balancing problem,” *Engineering Applications of Artificial Intelligence*, vol. 49, pp. 1-9, Mar. 2016.
[41] M. Rabbani, Z. Mousavi, and H. Farrokhiasl, “Multi-objective metaheuristics for solving a type II robotic mixed-model assembly line balancing problem,” *Journal of Industrial & Production Engineering*, vol. 33, no. 7, pp. 472-484, Feb. 2016.
[42] M. Fathi, M. J. Álvarez, and V. Rodríguez, “A new heuristic-based bi-objective simulated annealing method for U-shaped assembly line balancing,” *European Journal of Industrial Engineering*, vol. 10, no. 2, p. 145-169, Jan. 2016.
[43] E. Zitzler, M. Laumanns, and L. Thiele, ”SPEA2: Improving the strength Pareto evolutionary algorithm,” Zürich, Switzerland, TIK-Rep. 103, 2001.
[44] V. Grunert da Fonseca, C. M. Fonseca, and A. O. Hall, “Inferential performance assessment of stochastic optimizers and the attainment function,” in *Evolutionary Multi-Criterion Optimization (EMO2001)*, E. Zitzler et al., Eds. Berlin, Germany: Springer, 2001, pp. 213–225.

Zikai Zhang received the B.Sc. and Ph.D. degrees from the Wuhan University of Science and Technology, Wuhan, China, in 2016 and 2021, respectively.

He is currently a lecturer at the Wuhan University of Science and Technology, China. He has authored one academic book and more than 40 refereed papers (over 30 JCR-ranked papers). His research interests include assembly line balancing, assembly scheduling and intelligent optimization.

Qiuhua Tang received the B.S. degree in Process and Equipment of Machinery Manufacturing from Northeastern University, in 1992, and the master’s degree in Mechanical Design and Theory from Wuhan University of Technology, as well as a Ph.D. in the same field from Wuhan University of Science and Technology.

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

Manuel Chica has a Ph.D. degree in Computer Science and Artificial Intelligence (2011), and he is Senior Researcher at the University of Granada. His research interests include agent-based modeling, evolutionary game theory, machine learning, and evolutionary optimization. He has published more than 100 scientific papers (more than 55 JCR-ranked papers up to date), co-inventor of an international patent, and has participated in more than 20 R&D projects, where he played the PI role in more than 10. According to the ranking of the Stanford World´s top 2% Scientists 2022, he is among the top 2% most cited researchers in AI.

Zixiang Li received his Ph.D. degree in industrial engineering from Wuhan University of Science and Technology in 2018.

He had authored more than 40 SCI papers. He is currently an associate professor at Wuhan University of Science and Technology, China. His research interests include assembly line balancing, scheduling and metaheuristics.