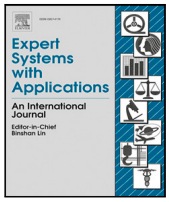
Expert Systems With Applications 236 (2024) 121221

Contents lists available at ScienceDirect

Expert Systems With Applications

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



A multi-objective co-evolutionary algorithm for energy and cost-oriented mixed-model assembly line balancing with multi-skilled workers

Zikai Zhang [a,b], Manuel Chica [d,e], Qiuhua Tang [a,b,*], Zixiang Li [b,c], Liping Zhang [b,c]

[a] Key Laboratory of Metallurgical Equipment and Control Technology, Ministry of Education, Wuhan University of Science and Technology, Wuhan 430081, China
[b] Hubei Key Laboratory of Mechanical Transmission and Manufacturing Engineering, Wuhan University of Science and Technology, Wuhan 430081, China
[c] Precision Manufacturing Institute, Wuhan University of Science and Technology, Wuhan 430081, China
[d] Andalusian Research Institute DaSCI "Data Science and Computational Intelligence", University of Granada, 18071 Granada, Spain
[e] School of Electrical Engineering and Computing, The University of Newcastle, Callaghan, NSW 2308, Australia

ARTICLE INFO

Keywords:
Multi-objective optimization
Co-evolutionary algorithm
Energy requirement
Mixed-model multi-manned assembly line balancing
Multi-skilled workers

ABSTRACT

Energy-saving, one of the most significant strategies for green manufacturing, has become the focus of more and more scholars and enterprise managers. Hence, this work addresses the minimization of energy and cost requirements on mixed-model multi-manned assembly line balancing with multi-skilled workers. A new mixed-integer linear programming model is proposed to define the problem. Additionally, a novel multi-objective co-evolutionary algorithm is designed to achieve the trade-off between energy and cost requirements. This algorithm includes a two-layer solution representation to achieve full coverage of the solution space and a new decoding mechanism with idle time reduction. A collaborative initialization as the first stage of the algorithm is extended to get high-quality and great-diversity initial solutions. A self-learning evolution for each sub-population with four problem-specific evolutionary operators is developed to explore task or worker assignment sequences, and a dual-cooperation strategy is proposed to enhance the interaction between sub-populations. The final experiments, based on 269 instances, demonstrate that the improvement components are effective and the proposed algorithm is superior to seven latest multi-objective evolutionary algorithms from numerical, statistical and differential analyses.

1. Introduction

With the continuous advancement of economic globalization, traditional single-model flow production cannot meet the requirements of mass customization. It requires the enterprises to organize flexible and diversified production systems to process or assemble a variety of products. In this situation, a mixed-model assembly line, one of the flow-oriented flexible production systems, is dedicated to perform the various assembly tasks of different products (Belkharroubi & Yahyaoui, 2022). Its corresponding balancing problem, named mixed-model assembly line balancing problem (MALBP) (Yagmahan, 2011), aims to assign the tasks with precedence graphs and product-dependent processing times to some specific workstations. In MALBPs, some constraints inevitably need to be taken into account in task allocation, such as cycle time constraint, precedence relation constraint and sequence limitation. Compared with single-model assembly line balancing problems (Scholl et al., 2010), MALBP has wider real-world applications due to its high efficiency and flexible attributes. Accordingly, many studies recently focus on MALBP with some real-world scenarios (Boysen et al., 2022; Liu, Yang et al., 2021).

Thanks to the mass customization production, several workers are allowed to work together to assemble different tasks in the same workstations of MALBP to result in lower length, effective space configuration, higher yield and fewer setup times (Chen, 2017). In realistic production horizon, different workers may have different skills. This multi-skilled attribute mainly reflects in two aspects: (1) assembly level where processing times vary from workers and products and (2); energy requirement where workers with different skill levels expend different amounts of energy to assemble a specific task. This multi-skilled information needs considering new variables in the model. First, the model includes workers' assignment to workstations and second, it also considers the different task assignments to each specific worker. Hence, this work decides to study the mixed-model assembly line balancing problem with multi-skilled workers (MALBP-MW).

* Corresponding author at: Key Laboratory of Metallurgical Equipment and Control Technology, Ministry of Education, Wuhan University of Science and Technology, Wuhan 430081, China.
E-mail addresses: zhangzikai@wust.edu.cn (Z. Zhang), manuelchica@ugr.es (M. Chica), tangqiuhua@wust.edu.cn (Q. Tang), lizixiang@wust.edu.cn (Z. Li), zhangliping@wust.edu.cn (L. Zhang).

https://doi.org/10.1016/j.eswa.2023.121221
Received 9 April 2023; Received in revised form 25 July 2023; Accepted 14 August 2023
Available online 19 August 2023

Furthermore, as the global advocates and promotes green manufacturing, it is essential to save energy consumption in modern industries. It is reported that more than 33% energy consumption comes from manufacturing enterprises (Gong et al., 2018). In MALBP-MW, each worker has an energy requirement because of the utilization of robots or electrical machinery. This situation requires the enterprise managers to organize a task and worker assignment with little energy consumption. However, the employment of high-skill workers can reduce energy consumption, but may bring high manual costs. Namely, the minimization of energy and cost requirements often comprises conflicting goals. Meanwhile, both the above-mentioned objectives are the key factors for managers when designing a better assembly line layout. Thus, this work aims to develop a mixed-integer linear programming (MILP) model to minimize the two conflicting objectives including energy and cost requirements simultaneously.

Besides, due to the NP-hardness and multi-criteria properties of this MALBP-MW, a multi-objective evolutionary algorithm (MOEA) is developed to achieve the trade-off between two conflicting objectives and obtain a Pareto set of solutions. The multi-objective co-evolutionary algorithm (MOCEA), one of MOEAs, has better exploration and exploitation abilities and can obtain an approximate Pareto set in an acceptable time. This algorithm involves several populations where each one corresponds to a sub-problem, and lies in a new co-evolution paradigm to achieve the cooperation between different populations. It has been successfully applied in related scheduling or balancing problems, such as flow shop (Zhao et al., 2021), job shop (Li et al., 2022) and assembly line balancing (Nilakantan et al., 2017). Hence, this algorithm is a good alternative for solving the MALBP-MW. This work extends a novel version of MOCEA by including problem-specific knowledge, problem-dependent evolutionary operators and four improved strategies. In a word, the main contributions are:

- Define the new problem MALBP-MW by a mixed-integer linear programming model to minimize the energy and cost requirements.
- Consider the problem-specific knowledge and design a two-layer solution representation and a new decoding mechanism with idle time reduction.
- Develop MOCEA with a collaborative initialization, four problem-specific evolutionary operators, a self-learning selection and dual-cooperation to solve MALBP-MW efficiently.

This work designs four types of experiments based on 269 instances to test the performance of the proposed MOCEA. Different multi-objective performance indicators (i.e., hypervolume ratio, unary epsilon, inverted generational distance and coverage) and analysis tools (i.e., numerical, statistical and differential analyses) are used in the comparison experiments. The final experimental results demonstrate that the proposed MOCEA outperforms its two variants and seven latest algorithms.

The remainder of this work is structured as follows. Section 2 reports the related works. Section 3 introduces the MALBP-MW. Section 4 details the proposed MOCEA. Section 5 shows the experimental study, and Section 6 summarizes the conclusions and future works.

2. Related works

This section reviews the related works from four main fields: worker-related (Section 2.1), energy-oriented (Section 2.2), and cost-oriented assembly line balancing problems (Section 2.3) as well as multi-objective co-evolutionary algorithm (Section 2.4).

2.1. Worker-related assembly line balancing

Since the first work (Akagi et al., 1983) attempted to assign multiple workers to workstations, multi-manned assembly line balancing began to attract scholars' attention. Recently, this problem has been studied in depth from different real-world scenarios and solved by different methods. Regarding the realistic constraints, researchers investigated multi-manned assembly line balancing embedded with resource limitation (Chen et al., 2018), space constraint (Zhang, Tang et al., 2020), sequence-dependent setups (Zhang, Tang, Han et al., 2023), positional constraint (Yang et al., 2022), line feeding mode assignment (Zangaro et al., 2023) and human–robot interaction (Wu, Zhang, Zeng et al., 2023; Wu, Zhang, Zhang et al., 2023) and so on.

Meanwhile, to tackle this NP-hard problem, various methods were designed, such as exact methods, heuristics and meta-heuristics. For example, Michels et al. (2019) proposed a benders' decomposition algorithm with combinatorial cuts. Abidin Çil and Kizilay (2020) developed a constraint programming. Lopes et al. (2020) introduced algorithmic lower bounds and presented a novel model-based heuristic. Andreu-Casas et al. (2022) developed two relax-and-fix procedures and a set of variants of heuristics based on solving a partition problem with constraints. Zhang, Tang, Chica (2021) used a gene expression programming to mine the best rule combination from existing heuristic rules. Şahin and Kellegöz (2019) combined a special heuristic into particle swarm optimization to reduce line costs. Zhang, Tang, Li et al. (2021) proposed an efficient migrating birds optimization algorithm with idle time reduction.

2.2. Energy-oriented assembly line balancing

Although many real-world multi-manned assembly line balancing problems have been tackled by different optimization methods, there is still a lack of studies on energy or cost requirements. In modern manufacturing, the reduction of energy requirements plays an important role in promoting green manufacturing. More and more researchers begin to consider energy-related in different generalized assembly line balancing problems.

Specifically, Battini et al. (2016) introduced the energy expenditure concept in assembly line balancing to evaluate the ergonomics level. Şahin and Kellegöz (2019) aimed to optimize energy efficiency in unpaced synchronous line balancing problems. Li et al. (2016) and Zhang, Tang, Li et al. (2019) considered energy consumption in two-sided and U-shaped robotic assembly line balancing problems, respectively. Further, Zhang, Tang, Zhang et al. (2019) reduced both the carbon and noise emissions in U-shaped robotic assembly line balancing problems via a grey wolf optimization. Zhou and Shen (2018) incorporated energy consumption in the traditional part feeding of assembly lines. Sun et al. (2020) optimized the energy saving and economic criteria of robotic assembly line balancing simultaneously by an estimation of distribution algorithm. Zhang, Xu et al. (2020) focused on the energy-oriented mixed-model assembly line balancing and sequencing problem. Liu, Liu et al. (2021) considered energy consumption optimization and worker assignment in assembly line balancing. Chutima and Khotsaenlee (2022) studied the PM2.5 emission arising from the energy consumption of the robotic workstations and energy expenditure load variation among workers. Rahman et al. (2023) addressed energy aware semi-automatic assembly line balancing problem with ergonomic risk and uncertain processing time.

2.3. Cost-oriented assembly line balancing

The literature in Section 2.2 reviews the energy-oriented assembly line balancing. These researches investigate the energy requirements of assembly line balancing from different aspects, such as assembly, standby and part feeding energy consumption and energy expenditure among robots and workers. However, they ignore that the low energy

consumption may result in high costs. Recently, some researchers have realized the importance of cost optimization in assembly line balancing problems.

For the cost optimization, Li, Janardhanan, Ponnambalam (2021) and Pereira et al. (2018) studied the robot-dependent costs in robotic assembly line balancing problems. Further, Li, Janardhanan, Tang (2021) investigated the purchasing cost of several different collaborative robots. Salehi et al. (2020) considered the costs associated with equipment, labor wage and station establishment. Niroomand (2021) minimized the equipment purchasing cost of the straight assembly line balancing problem.

2.4. Multi-objective co-evolutionary algorithm

For MOCEA, if compared with the traditional MOEAs, it is more suitable for many large-scale optimization problems with several sub-problems. Recently, MOCEA has shown effectiveness on the related scheduling or combinatorial optimization problems.

Nilakantan et al. (2017) proposed a multi-objective co-operative co-evolutionary algorithm to optimize the total carbon footprint and efficiency of robotic assembly line balancing. Meng et al. (2022) developed a new co-operative co-evolutionary algorithm for mixed-model assembly line balancing problem with preventive maintenance scenarios. Tian et al. (2021) designed a co-evolutionary algorithm framework for constrained multi-objective optimization problem. This algorithm optimized two sub-problems via the same optimizer, and used a weak cooperation between different populations. Pan et al. (2022a, 2022b) proposed a two-population evolutionary algorithm to solve fuzzy flexible job shop scheduling and distributed parallel machines scheduling, respectively.

Reviewing the aforementioned literature, it can be observed that there is a lack of literature on energy and cost requirements of MALBP-MW. Due to the important application value of achieving the trade-off between these conflicting objectives, this work decides to tackle this new problem via an efficient MOCEA.

3. Problem formulation

3.1. Definition of the problem and notations

This section explains the MALBP-MW through a MILP model. Table 1 presents the parameters and variables used in the model. The MALBP-MW is dedicated to assigning n tasks and w multi-skilled workers to a set of workstations to finish the assembly of multiple products. The processing time and energy requirement of each task depend on the products, and are denoted as t_{iq} and e_{iq} respectively. There are precedence relationships between different tasks. Namely, each task starts after all its immediate predecessors are completed. Each workstation is only allowed to configure with a maximum of u_{max} workers. Due to the differences in skills among workers, the real processing time and energy requirement of each task also depend on the assigned workers. In this situation, α_k and β_k are used to represent the time and energy skill coefficients of kth worker respectively, and accordingly, $t_{iq} \cdot \alpha_k$ and $e_{iq} \cdot \beta_k$ are the real processing time and energy requirement of task i in product q by worker k respectively. Each task can only be assigned to only one worker of one workstation. The workload of each assigned worker does not exceed the known cycle time.

3.2. Energy and cost-oriented objectives

The main objectives of the MALBP-MW is to minimize the cost and energy requirements. The costs of hiring different workers and opening workstations are considered in the MALBP-MW, as defined in Eq. (1). Once a workstation is assigned with tasks and workers, there is a fixed cost of opening workstation CO. Similarly, Once worker k is allocated to any opening workstation, there is a hiring cost CH_k. Eq. (2) defines the average energy requirement of all workers under all products.

Table 1
Description of the parameters and variables.

Notation	Description
Parameters	
i, h	Task indices.
j	Workstation index.
k	Worker index.
q	Product index.
n	The number of tasks.
m	Maximum number of workstations.
w	Maximum number of workers.
u_{max}	Maximum number of workers for each station.
d	The number of products.
CT	Cycle time.
CO	Cost of opening a workstations.
CH_k	Cost of hiring worker k.
t_{iq}	Processing time of task i in product q.
α_k	Processing time skill coefficient of worker k.
e_{iq}	Energy requirement of task i in product q.
β_k	Energy requirement skill coefficient of worker k.
I	Set of tasks, $\|I\| = n$.
J	Set of workstations, $\|J\| = m$.
K	Set of workers, $\|K\| = w$.
Q	Set of products, $\|Q\| = d$.
P^0	Set of tasks that have no immediate predecessors.
$P(i)$	Set of immediate predecessors of task i
$\hat{M}$	Large positive number.
Variables	
FT_{iq}	The finishing time of task i in model q.
W_{ihk}	A binary value sets to 1 if task i and h are assigned to the same worker k 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 sets to 1 if worker k is hired in workstation j; otherwise sets to 0.
Z_j	A binary value sets to 1 if workstation j is opened; otherwise sets to 0.

$$minf_1 = \sum_{j\in J} CO \cdot Z_j + \sum_{j\in J}\sum_{k\in K} CH_k \cdot Y_{jk} \tag{1}$$

$$minf_2 = \frac{\sum_{i\in I}\sum_{j\in J}\sum_{k\in K}\sum_{q\in Q} e_{iq}\cdot\beta_k \cdot X_{ijk}}{d} \tag{2}$$

3.3. Constraints of the model

Constraints (3)–(15) are developed to limit the variables of the mathematical model. Constraint (3) requires that each task can only be assigned to exactly one worker at one workstation. Constraint (4) confines that the workload of each worker does not exceed the cycle time. Constraint (5) defines the minimum value of the starting times.

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

$$FT_{iq} \le CT, \forall i \in I, q \in Q \tag{4}$$

$$FT_{iq} \ge \sum_{k\in K} X_{ijk} \cdot t_{iq} \cdot \alpha_k, \forall i \in I, q \in Q \tag{5}$$

Constraints (6)–(8) confine the variable Y_{jk}. If some tasks are assigned to worker k at workstation j, constraint (6) becomes active and requires Y_{jk} to equal 1. Constraint (7) controls that each workstation is allowed to configure with a maximum of u_{max} workers. Constraint (8) ensures that each worker is assigned to only one workstation.

$$\sum_{i\in I} X_{ijk} \le n \cdot Y_{jk}, \forall j \in J, k \in K \tag{6}$$

$$\sum_{k\in K} Y_{jk} \le u_{max}, \forall j \in J \tag{7}$$

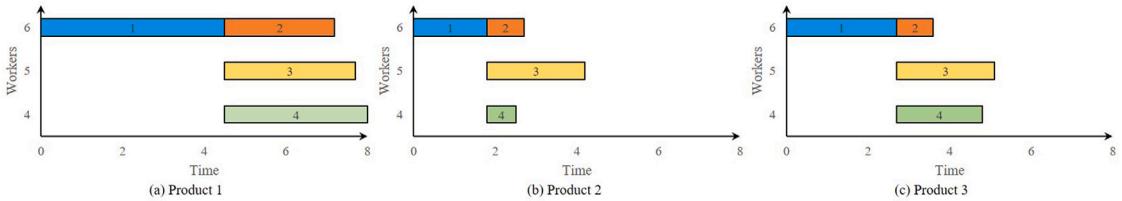


Fig. 1. Gantt charts for workstation 1 assembling different products.

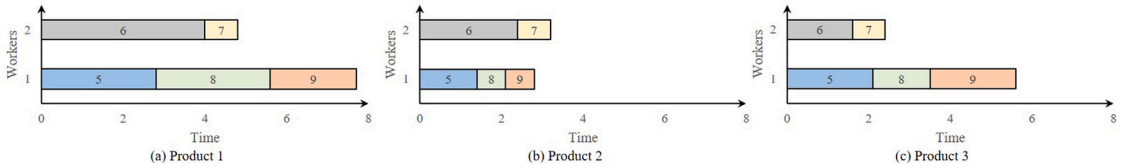


Fig. 2. Gantt charts for workstation 2 assembling different products.

$$\sum_{j\in J} Y_{jk} \le 1, \forall k \in K \tag{8}$$

Constraints (9)–(11) define the variable Z_j. If some tasks are assigned to workstation j, constraint (9) becomes active and requires Z_j to equal 1; otherwise, constraint (10) works and requires Z_j to equal 0. Constraint (11) limits that workstations are opened in an increasing manner.

$$\sum_{i\in I}\sum_{k\in K} X_{ijk} \le n \cdot u_{max} \cdot Z_j, \forall j \in J \tag{9}$$

$$\sum_{i\in I}\sum_{k\in K} X_{ijk} \ge Z_j, \forall j \in J \tag{10}$$

$$Z_{j+1} \le Z_j, \forall j = 1, 2, \dots, m-1 \tag{11}$$

Constraints (12)–(15) define the precedence relation constraints. Constraint (12) defines the immediate precedence relations. Constraint (13) limits the starting times of each pair of tasks with precedence relations. Constraints (14)–(15) control those without precedence relations in adjacent positions of the same worker.

$$\sum_{j\in J}\sum_{k\in K} j \cdot X_{hjk} - \sum_{j\in J}\sum_{k\in K} j \cdot X_{ijk} \le 0, \forall i \in I - P^0, h \in P(i) \tag{12}$$

$$FT_{iq} - FT_{hq} + \hat{M} \cdot (1 - \sum_{k\in K} X_{ijk}) + \hat{M} \cdot (1 - \sum_{k\in K} X_{hjk}) \ge \sum_{k\in K} X_{ijk} \cdot t_{iq} \cdot \alpha_k, \forall i \in I - P^0, h \in P(i), j \in J, q \in Q \tag{13}$$

$$FT_{hq} - FT_{iq} + \hat{M} \cdot (1 - X_{ijk}) + \hat{M} \cdot (1 - X_{hjk}) + \hat{M} \cdot (1 - W_{ihk}) \ge t_{hq} \cdot \alpha_k, \forall (i,h) | i \ne h \wedge i, h \in I, j \in J, k \in K, q \in Q \tag{14}$$

$$FT_{iq} - FT_{hq} + \hat{M} \cdot (1 - X_{ijk}) + \hat{M} \cdot (1 - X_{hjk}) + \hat{M} \cdot W_{ihk} \ge t_{iq} \cdot \alpha_k, \forall (i,h) | i \ne h \wedge i, h \in I, j \in J, k \in K, q \in Q \tag{15}$$

3.4. Numerical example

To give a better understanding of the MALBP-MW, a numerical example with 9 tasks, 3 products and 9 workers is provided. Table 2 presents the precedence graph, processing times and energy requirements of all tasks. For 9 workers, their processing time skill coefficients are {0.7, 0.8, 0.9, 0.7, 0.8, 0.9, 0.9, 0.8, 0.8}; their energy requirement skill coefficients are {1, 0.8, 1, 0.8, 0.7, 0.8, 0.7, 0.8, 0.7}; their hiring costs are {3.9, 6.8, 4, 6.4, 5, 7.3, 6.4, 4.5, 5.6}. Parameters u_{max}, CT and CO are set at 3, 8 and 5.1, respectively.

Table 2
The basic information of all tasks.

i	$P(i)$	t_{iq}			e_{iq}		
		1	2	3	1	2	3
1	0	5	2	3	17	12	11
2	1	3	1	1	15	16	13
3	1	4	3	3	12	14	13
4	1	5	1	3	20	16	14
5	2,3	4	2	3	10	15	20
6	4	5	3	2	18	11	10
7	6	1	1	1	10	19	17
8	5	4	1	2	12	11	10
9	8	3	1	3	19	15	15

Figs. 1–2 depict one task and worker assignment plan. Five workers and two workstations are involved to organize the mixed-model multi-manned assembly line. Specifically, workstation 1 employs workers 4, 5 and 6 to perform tasks 1, 2, 3 and 4. Workstation 2 hires workers 1 and 2 to perform tasks 5, 6, 7, 8 and 9. Accordingly, the cost of opening workstation is 10.2 (5.1×2), and that of hiring workers is 29.4 ($3.9 + 6.8 + 6.4 + 5 + 7.3$), resulting in a total cost of 39.6. The energy requirements of products 1, 2 and 3 are 113.4 ($17\times0.8+15\times0.8+12\times0.7+20\times0.8+10\times1+18\times0.8+10\times0.8+12\times1+19\times1$), 110 and 106.1, respectively.

4. Multi-objective co-evolutionary algorithm

The MOCEA, a typical cooperative co-evolutionary algorithm, is designed by Zhao et al. (2014) for multi-objective problems with multiple sub-problems. This algorithm starts with multiple initial sub-populations where each one aims to optimize one sub-problem. The non-dominated solutions from these initial sub-populations are stored in an external elite archive set (EAS). In the main loop, the individuals in all sub-populations are first updated by some specific evolutionary operators; then the newly generated non-dominated individuals are used to update EAS; finally, a cooperation between different sub-populations is executed. MOCEA has shown greater performance in

many multi-objective optimization problems (Zhao et al., 2014), and also been successfully applied in related scheduling problems (Meng et al., 2022; Nilakantan et al., 2017; Tian et al., 2021).

The MALBP-MW involves two sub-problems such as task and worker assignments, and hence this work designs a novel MOCEA that incorporates some specific knowledge and improvement strategies with the algorithm framework. There are four main differences between the designed MOCEA and the original algorithm: (1) a two-layer solution representation and a new decoding mechanism with idle time reduction are designed in the MOCEA; (2) a collaborative initialization with multiple-heuristic and random approach is developed to generate two initial populations; (3) four problem-specific evolutionary operators and a self-learning selection strategy are proposed to improve two populations and EAS; (4) a dual-cooperation strategy is extended to share superior information among different populations and enable interaction between two populations and an elite archive set. The details of MOCEA are introduced in the following subsections.

4.1. Two-layer solution representation and decoding mechanism

Since the MALBP-MW involves two types of decision variables: task and worker assignments, the proposed MOCEA uses a two-layer vector to express the solutions. Each solution is encoded with two vectors: task priority vector $\{\pi_1, \pi_2, \dots, \pi_i, \dots, \pi_n\}$ and worker priority vector $\{\tau_1, \tau_2, \dots, \tau_k, \dots, \tau_w\}$. The element π_i or τ_j is the priority of task i or worker k. Each value of task priority vector or worker priority vector is generated between [1, n] or [1, w] without repetition, respectively.

Further, to translate the above representation into a feasible task and worker assignment plan, this work extends a new decoding mechanism with idle time reduction. This decoding considers all constraints, objective minimization, sequence-dependent and remaining idle times reduction technology into task and worker assignments. It can effectively balance the workloads of different workers and workstations, and improve line efficiency, as well as minimize the cost and energy requirements. The proposed decoding contains six important steps: determining available worker set, constructing task candidate set, selecting task, constructing worker candidate set, selecting worker and selecting best station configuration. The pseudo-code of the new decoding mechanism is presented in Algorithm 1. In this decoding, three criteria, which are used in task/worker candidate set construction and worker selection, are first defined.

Criterion 1: Construct task candidate set TC according to the precedence relation and task assignment constraints. A task can be stored in TC based on the following criteria: (1) it has not been assigned; (2) all its predecessors have been assigned.

Criterion 2: Construct worker candidate set WC according to the cycle time constraint. A worker can be put into set WC based on the following criteria: (1) it belongs to set WS; (2) it has not been assigned; (3) the workload of this worker does not exceed the cycle time when assembling the task i^1 of all products.

Criterion 3: Select a worker k^1 from WC referring to the following rules: (a) select a worker with the minimum finishing time; (b) if multiple workers exist from (a), select a worker with the minimum sequence-dependent idle time; (c) if multiple worker exists from (b), select a worker with the minimum energy requirement; (d) if multiple workers exist from (c), select a worker with the minimum hiring cost; (f) if multiple workers exist from (d), select a worker with the lowest number.

Algorithm 1: The new decoding mechanism with idle time reduction

Input: Task priority vector $\{\pi_1, \pi_2, ..., \pi_i, ..., \pi_n\}$ and worker priority vector $\{\tau_1, \tau_2, ..., \tau_k, ..., \tau_w\}$.
Output: Workstation configurations and objective values.
Begin:
  For workstation $j = 1$ to m and all tasks have not been assigned **do**
    For $k^1 = 1$ to u_{max} **do** (%% assume k^1 workers in workstation j)
      Put k^1 unassigned workers with the maximum priority values into the available worker set WS;
      Construct task candidate set TC according to **Criterion 1**;
      If the set TC is not empty **then**
        Select the task with the maximum priority value from TC and record it as i^1;
        Construct worker candidate set WC according to **Criterion 2**;
        If the set WC is not empty **then**
          Calculate the finishing times, sequence-dependent idle times and energy requirements for all workers in WC when assembling the task i^1;
          Select a worker k^2 from WC referring to **Criterion 3**;
          Assign task i^1 to worker k^2 and workstation j;
          Return to Step 7;
        Else
          Go to Step 22;
        End If
      Else
        Go to Step 22;
      End If
    Record the current configuration of workstation j with k^1 worker as CC_{j,k^1};
    Calculate the average line efficiency, cost and energy requirements of configuration CC_{j,k^1};
    End For
    Rank different configurations according to average line efficiency, cost and energy requirements;
    Calculate the weighted rank values on the premise of the same proportions of the three indexes;
    Select the configuration with the minimum rank value for workstation j;
  End For
  Calculate the objective values;
Return Workstation configurations and objective values.

4.2. Collaborative initialization

To enhance the diversity and convergence of initial solution set, this work designs a collaborative initialization with multiple-heuristic and random approach to generate the initial dual populations $pop1$ and $pop2$. The multiple-heuristic aims to obtain high-quality code fragments, while the random approach ensures the diversity of populations. Both $pop1$ and $pop2$ have the same size $PS/2$.

(1) Collaborative initialization for pop1. The $pop1$ is mainly used to explore the solution space around task assignments. It is expected to be higher quality during the evolution of task assignments and more dispersion during that of worker assignments. Accordingly, this work decides to use multiple heuristics to generate the task priority vectors and a random approach to obtain the worker priority vector. 28 common heuristic rules are employed here and presented in Table 3. For each individual, one of the heuristic rules is randomly selected to obtain the priority values of all tasks to form the vector $\{\pi_1, \pi_2, \dots, \pi_i, \dots, \pi_n\}$;

Table 3
The multiple heuristic rules related to task and worker assignments.

No.	Heuristic information	*No.*	Heuristic information
(1) Heuristics related to task assignment			
1	The shortest processing time	2	The longest processing time
3	The lowest energy requirement	4	The highest energy requirement
5	The minimum number of predecessors	6	The maximum number of predecessors
7	The minimum number of successors	8	The maximum number of successors
9	The shortest total processing time of all predecessors	10	The longest total processing time of all predecessors
11	The shortest total processing time of all successors	12	The longest total processing time of all successors
13	The lowest total energy requirement of all predecessors	14	The highest total energy requirement of all predecessors
15	The lowest total energy requirement of all successors	16	The highest total energy requirement of all successors
17	The minimum number of immediate predecessors	18	The maximum number of immediate predecessors
19	The minimum number of immediate successors	20	The maximum number of immediate successors
21	The shortest total processing time of all immediate predecessors	22	The longest total processing time of all immediate predecessors
23	The shortest total processing time of all immediate successors	24	The longest total processing time of all immediate successors
25	The lowest total energy requirement of all immediate predecessors	26	The highest total energy requirement of all immediate predecessors
27	The lowest total energy requirement of all immediate successors	28	The highest total energy requirement of all immediate successors
(2) Heuristics related to worker assignment			
1	The lowest hiring worker cost	2	The highest hiring worker cost
3	The minimum processing time skill coefficient	4	The maximum processing time skill coefficient
5	The minimum energy requirement skill coefficient	6	The maximum energy requirement skill coefficient

while the vector $\{\tau_1, \tau_2, \ldots, \tau_k, \ldots, \tau_w\}$ is randomly generated between [1, w] without repetition.

(2) Collaborative initialization for pop2. The effect of $pop2$ is the opposite of $pop1$ and mainly works on the solution space around worker assignments. Six common heuristic rules related to worker assignments are used here and presented in Table 3. For each individual, one of the heuristic rules is randomly selected to form the vector $\{\tau_1, \tau_2, \ldots, \tau_k, \ldots, \tau_w\}$; while the vector $\{\pi_1, \pi_2, \ldots, \pi_i, \ldots, \pi_n\}$ is randomly generated between [1, n] without repetition.

After obtaining the populations $pop1$ and $pop2$, the non-dominated solutions are put into a new elite archive set EAS.

4.3. Self-learning evolution of dual populations

The proposed MOCEA involves two populations $pop1$ and $pop2$ where each one aims to explore different solution spaces around task or worker assignments. This work is dedicated to designing different specific-knowledge evolution operators for each population. Since different evolution operators have different exploitation performances, these operators can guide $pop1$ and $pop2$ to conduct multi-direction local searches around different solution spaces. The details of these operators are introduced below.

(1) Evolution of $pop1$. Two operators, including swap and insert on task priority vector, are developed to work on $pop1$. These operators aim to explore the solution space around task assignment. They generate new solutions by changing the segments of the task priority vector. The procedures are detailed below.

- **Swap on task priority vector (STPV)**: Two different tasks are randomly selected, and their priority values are swapped.
- **Insert on task priority vector (ITPV)**: One task is randomly selected and its corresponding priority value is extracted. Then this value is inserted into a new position of task priority vector which is different from its original position.

(2) Evolution of $pop2$. This work also uses swap and insert operators to explore the solution space around worker assignment. However, these operators work on worker priority vectors of $pop2$. The specific procedures are introduced below.

- **Swap on worker priority vector (SWPV)**: Two different workers are randomly selected, and their corresponding priority values are exchanged.
- **Insert on worker priority vector (IWPV)**: The priority value of a randomly selected worker is extracted, and reinserted back into a new and different position of worker priority vector.

(3) Self-learning selection strategy. To make full use of four operators on $pop1$ and $pop2$, this work designs a self-learning selection strategy to coordinate them at each iteration. This strategy evaluates all these operators by the weighted cumulative success rates before executing the evolution of $pop1$ and $pop2$, and then selects a superior operator by roulette. First, the success rate of each operator j at iteration g is calculated by Eq. (16). In this equation, $ne_{j,g}$ is the selected number of jth operator during iteration g; $ns_{j,g}$ refers to the number of non-dominated solutions obtained by jth operator. If the jth operator is not selected during iteration g, the corresponding success rate is zero; otherwise, this value depends on the improvements and execution times. Then considering the historical success rates, this work sets different weights for success rates during different iterations. The weighted cumulative success rates are calculated by Eq. (17). Accordingly, the selected probability of jth operator during iteration g is calculated by Eq. (18). Note that the initial success rate $sr_{j,0}$ sets 0 and $fsp_{j,0}$ is 0.5 due to two operators for each population.

$$sr_{j,g} = \begin{cases} 0, \ if \ the \ operator \ j \ during \ iteration \ g \ is \ not \ executed \\ \frac{ns_{j,g}}{ne_{j,g}}, \ otherwise \end{cases} \tag{16}$$

$$wcsr_{j,g} = \frac{\sum_{k=1}^{g-1} sr_{j,k} \cdot k}{\sum_{k=1}^{g-1} k} \tag{17}$$

$$fsp_{j,g} = \frac{wcsr_{j,g}}{\sum_{j=1}^{4} wcsr_{j,g}} \tag{18}$$

At each iteration, one evolution operator is selected for each individual of $pop1$ and $pop2$ according to the self-learning selection. The selected operator works on this individual to generate a new solution. If the new solution is not dominated by the original individual, it replaces the original one.

4.4. Dual-cooperation strategy

In the above procedures, different populations including $pop1$ and $pop2$ evolve independently. There is a lack of cooperation between different populations. To overcome this drawback, this work designs a dual-cooperation strategy to assist different populations to share their superior information and enable population interaction. This strategy contains two cooperation ways. The first way works on $pop1$. For each individual of $pop1$, a cooperative object is randomly selected from $pop2$. The worker priority vector of the selected object replaces that of the individual of $pop1$. The second one works on $pop2$. The task priority vector of a randomly selected object from $pop1$ will replace that of the individual of $pop2$.

Besides, after the evolution of dual populations, the improved populations $pop1$ and $pop2$ need to update the set EAS considering the dominance relations.

4.5. Framework of the proposed MOCEA

The pseudo-code of the MOCEA is presented in Algorithm 2.

Algorithm 2: The proposed MOCEA

Input: Problem data, parameter PS.
Output: EAS.
Begin:
  Generate initial dual populations by collaborative initialization (each population has $PS/2$ solutions);
  Store the non-dominated solutions of dual populations into EAS;
  Set $sr_{j,0} = 0$ and $fsp_{j,0} = 0.5$ for each evolutionary operator;
  Do while *termination criteria is not fulfilled*
    Select STPV or ITPV by self-learning selection;
    For $i = 1$ to $PS/2$ **do**
      Generate new solution by executing the selected operator on i-th solution of $pop1$;
      If the new solution dominates the i-th solution **then**
        Replace the i-th solution with the new solution;
      End If
      Count $ns_{j,g}$ and $ne_{j,g}$;
    End For
    Update $sr_{j,g}$ and $fsp_{j,g}$;
    Select SWPV or IWPV by self-learning selection;
    For $i = 1$ to $PS/2$ **do**
      Generate new solution by executing the selected operator on i-th solution of $pop2$;
      If the new solution dominates the i-th solution **then**
        Replace the i-th solution with the new solution;
      End If
      Count $ns_{j,g}$ and $ne_{j,g}$;
    End For
    Update $sr_{j,g}$ and $fsp_{j,g}$;
    Dual-cooperation on $pop1$, $pop2$ and EAS;
  End While
Return EAS.

5. Experimental study

This section conducts a set of experiments to investigate the effectiveness of the proposed improvements and the performance of MOCEA. Specifically, Section 5.1 introduces the experimental setting, such as the benchmark instances, running environment, multi-objective evaluation indicators and parameter calibration. Section 5.2 verifies the effectiveness of the proposed improvements of MOCEA. Sections 5.3 to 5.5 show the superiority of MOCEA from different aspects, such as the numerical, statistical and differential analyses. Section 5.6 discusses the results.

5.1. Experimental setting

A total of 269 typical benchmark instances in the literature are employed. According to the task number and precedence graph, these instances can be divided into 22 groups which are respectively 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. The basic data of these instances can be downloaded from https://www.assembly-line-balancing.de. Referring to the literature (Zhang, Tang, Chica, 2021; Zhang & Xu, 2020), this work has the parameter setting in Table 4. All the algorithms are programmed in C++ and launched in a computer having an Intel Core i9 12900k with 2.80 GHz and 2 GB of memory.

Table 4
Parameter setting.

Parameter	Value or range	Parameter	Value or range
m	n	α_k	[0.7, 1.0]
w	$2 \cdot n$	β_k	[0.7, 1.0]
U_{max}	[2, 4]	CO	[3, 10]
e_{iq}	[10, 20]	CH_k	[3, 10]

To better evaluate the diversity and coverage of the obtained Pareto sets, three unary (hypervolume ratio, HVR (Zitzler & Thiele, 1999), unary epsilon, I_ϵ (Fonseca et al., 2006) and inverted generational distance, IGD (Yuan & Xu, 2015)) and one binary (coverage, C (Zitzler et al., 2000)) common multi-objective performance indicators are used. The obtained Pareto set with the HVR and I_ϵ values closer to 1 or the IGD value closer to 0 is the better approximation to the true Pareto front. Note that the true Pareto front is not known in advance and thus, we build the pseudo-optimal Pareto set by combining all the non-dominated solutions from the results obtained by all the algorithm of the experimentation. The indicator C compares two obtained Pareto sets P and Q according to the domination relation. A $C(P, Q)$ value closer to 1 means that set Q is strongly dominated by set P.

Before the comparison experiments, this section uses a single-factor ANOVA to calibrate the population size of MOCEA. Four levels are set for the population size: 10, 20, 30 and 40. The proposed MOCEA under each level solves five calibration instances for five times with the CPU time limit of $n \cdot n \cdot 5$ (Zhang, Tang, Chica et al., 2023). A total of $4 \cdot 5 \cdot 5 = 100$ experiments are carried out. Finally, the best population size is 20.

5.2. Effectiveness of the components

This section aims to investigate whether the components including collaborative initialization and dual-cooperation can effectively improve the performance of MOCEA. Two variants of MOCEA are implemented. First, variant MOCEA1 removes collaborative initialization. Second, variant MOCEA2 removes dual-cooperation. These variants independently tackle 269 instances for 30 times under the same CPU time limit of $n \cdot n \cdot 20$ (Zhang, Tang, Chica et al., 2023). The final average indicator values are reported in Tables 5–6.

From Table 5, if compared with those of MOCEA1 and MOCEA2, the average HVR and I_ϵ values of MOCEA are closer to 1 while IGD value is closer to 0. Namely, the collaborative initialization and dual-cooperation effectively improve the diversity and coverage of MOCEA. Regarding the coverage indicator in Table 6, the average C(MOCEA, MOCEA1) is larger than C(MOCEA1, MOCEA), and C(MOCEA, MOCEA2) is larger than C(MOCEA2, MOCEA). These results indicate that most solutions of the Pareto set obtained by MOCEA1 and MOCEA2 are dominated by the Pareto set achieved by MOCEA. Especially, with the instance scale increasing, the dominant performance of MOCEA is more significant.

The means plot of HVR, I_ϵ and IGD with Tukey's honest significant difference 95% confidence intervals for MOCEA and its variants is presented in Fig. 3. Regardless of HVR, I_ϵ or IGD, the interval of MOCEA is significantly better than those of MOCEA1 and MOCEA2. Accordingly, the same conclusion can be drawn that the components including collaborative initialization and dual-cooperation can effectively improve the performance of MOCEA.

5.3. Comparison against other algorithms

The section further verifies the performance of MOCEA by comparing it with 7 multi-objective evolutionary algorithms. These al-

Table 5
Average HVR, I_ϵ and IGD values for MOCEA and its variants.

Instances	HVR			I_ϵ			IGD		
	MOCEA1	MOCEA2	MOCEA	MOCEA1	MOCEA2	MOCEA	MOCEA1	MOCEA2	MOCEA
P7	0.78	0.85	0.85	1.22	1.20	1.19	0.21	0.18	0.19
P8	0.80	0.78	0.79	1.22	1.26	1.24	0.26	0.27	0.27
P9	0.76	0.86	0.85	1.18	1.14	1.15	0.23	0.19	0.20
P11	0.72	0.82	0.82	1.34	1.27	1.29	0.27	0.23	0.25
P21	0.69	0.85	0.86	1.28	1.21	1.20	0.29	0.24	0.23
P25	0.73	0.80	0.83	1.33	1.26	1.23	0.27	0.24	0.22
P28	0.65	0.81	0.85	1.42	1.38	1.32	0.27	0.24	0.21
P29	0.62	0.80	0.83	1.41	1.28	1.23	0.29	0.20	0.18
P30	0.65	0.78	0.84	1.43	1.28	1.21	0.33	0.22	0.17
P32	0.67	0.83	0.84	1.38	1.22	1.20	0.30	0.22	0.21
P35	0.68	0.82	0.87	1.33	1.23	1.18	0.29	0.21	0.18
P45	0.74	0.81	0.86	1.45	1.32	1.24	0.29	0.22	0.17
P53	0.58	0.84	0.84	1.33	1.25	1.27	0.28	0.25	0.25
P58	0.57	0.81	0.85	1.34	1.18	1.14	0.34	0.19	0.16
P70	0.53	0.79	0.86	1.55	1.28	1.20	0.35	0.19	0.15
P75	0.63	0.79	0.85	1.38	1.18	1.13	0.40	0.19	0.16
P83	0.58	0.81	0.85	1.35	1.20	1.15	0.29	0.20	0.17
P89	0.58	0.83	0.86	1.34	1.16	1.13	0.33	0.18	0.15
P94	0.57	0.81	0.87	1.43	1.21	1.14	0.30	0.18	0.14
P111	0.53	0.81	0.85	1.46	1.23	1.17	0.33	0.19	0.16
P148	0.47	0.81	0.86	1.48	1.21	1.17	0.36	0.17	0.15
P297	0.78	0.77	0.87	1.53	1.23	1.14	0.42	0.19	0.13
Avg.	0.60	0.81	**0.85**	1.41	1.22	**1.18**	0.33	0.20	**0.17**

Table 6
Average coverage values for MOCEA and its variants.

Instances	C(MOCEA, ...)		C(..., MOCEA)	
	MOCEA1	MOCEA2	MOCEA1	MOCEA2
P7	0.37	0.22	0.48	0.34
P8	0.00	0.50	1.00	0.20
P9	0.49	0.25	0.50	0.48
P11	0.58	0.34	0.32	0.45
P21	0.68	0.39	0.08	0.29
P25	0.85	0.64	0.16	0.16
P28	0.68	0.62	0.43	0.28
P29	0.83	0.51	0.03	0.32
P30	0.86	0.59	0.04	0.26
P32	0.88	0.53	0.10	0.29
P35	0.74	0.54	0.15	0.38
P45	0.66	0.64	0.15	0.19
P53	0.31	0.38	0.41	0.50
P58	0.95	0.73	0.01	0.21
P70	0.95	0.74	0.00	0.14
P75	1.00	0.85	0.00	0.07
P83	0.85	0.65	0.02	0.23
P89	0.91	0.75	0.02	0.20
P94	0.83	0.64	0.06	0.19
P111	0.93	0.74	0.02	0.15
P148	0.95	0.78	0.02	0.13
P297	0.99	0.89	0.01	0.04
Avg.	**0.85**	**0.68**	0.08	0.20

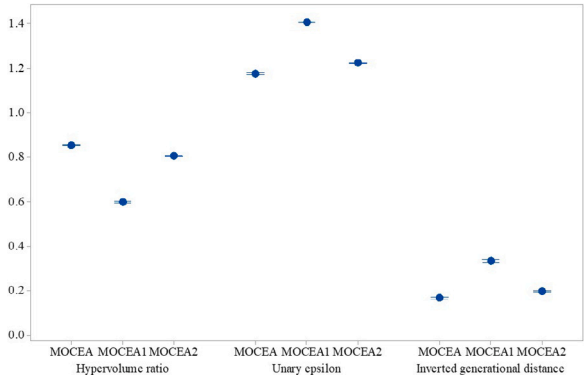


Fig. 3. Means plot of HVR, I_ϵ and IGD with Tukey's honest significant difference 95% confidence intervals for MOCEA and its variants.

gorithms include enhanced JAYA (EJAYA) (Zhang, Tang, Han et al., 2021), hybrid Pareto grey wolf optimization (HPGWO) (Zhang, Tang, Zhang et al., 2019), improved multi-objective artificial bee colony (IMABC) (Li, Janardhanan, Ponnambalam, 2021), multi-objective migrating bird optimization (MMBO) (Li, Janardhanan, Tang, 2021), multi-objective particle swarm optimization (MOPSO) (Rabbani et al., 2016), elitist non-dominated sorting genetic algorithm (NSGA-II) (Li, Janardhanan, Ponnambalam, 2021) and improved strength Pareto evolutionary algorithm (SPEA2) (Zitzler et al., 2001). These algorithms have been successfully applied to solve the related assembly line balancing or combinatorial optimization problems. To adapt these algorithms to this new problem, EJAYA, HPGWO, MOPSO use random-key to encode the task and worker assignments, while the remaining algorithms use the proposed two-layer solution representation. All algorithms use the proposed decoding mechanism to calculate the objective values. The parameters of these algorithms are tuned by the same method in Section 5.2 and presented in Table 7. To make a fair comparison, all algorithms independently tackle 269 instances for 30 times under the same CPU time limit of $n \cdot n \cdot 20$. The final average indicator values are reported in Tables 8–12.

From Tables 8–10, it can be found that in the 22 instance groups, MOCEA is significantly better than the others. If using average HVR, I_ϵ or IGD values to rank these algorithms, MOCEA appears in the first level followed in order by MOPSO, EJAYA, HPGWO, IMABC, NSGA-II, SPEA2 and MMBO. Regarding the coverage value in Tables 11–12, there is no doubt that C(MOCEA, ...) is much larger than C(..., MOCEA) for any compared algorithm. Namely, most of Pareto solutions obtained by MOPSO, EJAYA, HPGWO, IMABC, NSGA-II, SPEA2 and MMBO are dominated by the Pareto set achieved by MOCEA. Besides, the means plots of HVR, I_ϵ and IGD with Tukey's honest significant difference

Table 7
Parameter combinations of the compared algorithms.

Parameter	Value	Parameter	Value
EJAYA		MMBO	
Population size	20	Population size	9
HPGWO		Number of neighbor solutions	10
Population size	20	Number of shared solutions	5
Crossover rate	0.8	Consecutive iteration times	20
IMABC		NSGA-II	
Population size	20	Population size	20
Consecutive times in scout bee	10	Crossover rate	0.8
MOPSO		Mutation rate	0.2
The number of particles	20	SPEA2	
Weighted value	0.4	Population size	20
		Pareto frontier set size	20

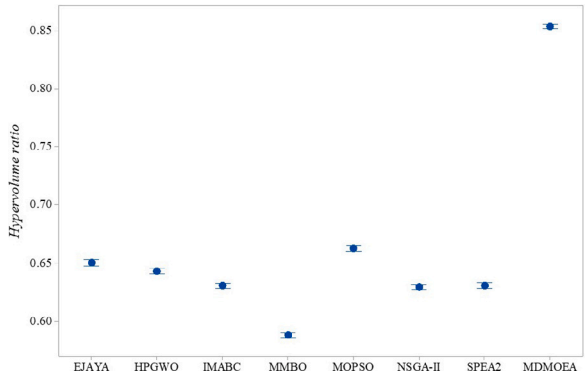


Fig. 4. Means plot of HVR with Tukey's honest significant difference 95% confidence intervals for all algorithms.

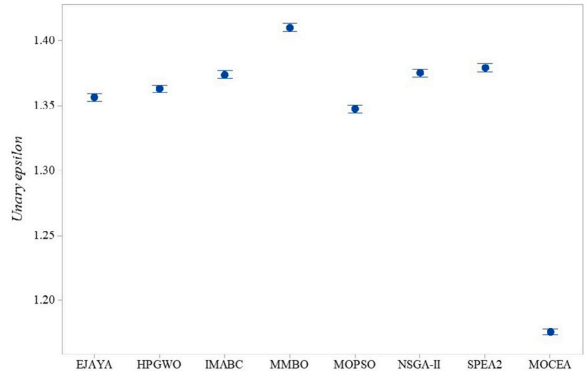


Fig. 5. Means plot of I_ϵ with Tukey's honest significant difference 95% confidence intervals for all algorithms.

95% confidence intervals for all algorithms are presented in Figs. 4–6. These figures further reflect that the proposed MOCEA outperforms the compared algorithms MOPSO, EJAYA, HPGWO, IMABC, NSGA-II, SPEA2 and MMBO.

5.4. Non-parametric tests

This section aims to analyze all HVR and I_ϵ values obtained in Section 5.3 statistically. A total of 129,120 (8 algorithms × 269 instances × 30 running times × 2 indicators) pieces of data are considered.

First, Shapiro–Wilk and Levene's tests are used to analyze normality and heteroscedasticity respectively. The inspection results are reported in Tables 13–14. Table 13 indicates that the HVR and I_ϵ data of 7

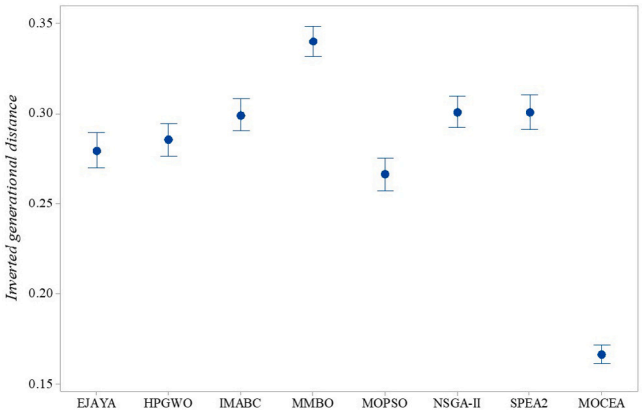


Fig. 6. Means plot of IGD with Tukey's honest significant difference 95% confidence intervals for all algorithms.

compared algorithms are not normally distributed at the 5% level while only the HVR data of MOCEA is normally distributed. Table 14 shows that the variances of the distributions of HVR or I_ϵ values achieved by 8 algorithms are not homogeneous.

The non-normal and non-homogeneous characteristics require this work to use non-parametric tests (such as Wilcoxon's rank-sum test and Friedman ANOVA) to further analyze all HVR and I_ϵ values. Wilcoxon's rank-sum test aims to analyze the difference between any two algorithms, and Friedman ANOVA to statistically rank these algorithms. The final results are reported in Tables 15–16. The results in Table 15 reveal that MOCEA is significantly different from any compared algorithm. Those in Table 16 state that MOCEA ranks the best. Namely, the proposed MOCEA significantly outperforms MOPSO, EJAYA, HPGWO, IMABC, NSGA-II, SPEA2 and MMBO.

5.5. Differential attainment functions

This section aims to analyze the dominance of the obtained Pareto sets over the solution space. The differential empirical attainment function approach (Diff-EAF) (nez et al., 2006) is employed. This section mainly analyses the Pareto sets achieved by MOCEA, EJAYA, HPGWO, IMABC, MOPSO, NSGA-II and SPEA2 on one instance of P58. Each algorithm solves this instance for 100 times to obtain 100 Pareto sets. After calculating the differential dominant probabilities, this work depicts the differences between empirical attainment function for MOCEA and 7 compared algorithms in Fig. 7. It can be found that the solution space around the objective boundaries is colored in red, yellow and green. That is, the solutions in these regions are more likely to be dominated by the Pareto set achieved by MOCEA. Accordingly, the Pareto set obtained by MOCEA has the best diversity and coverage.

5.6. Analysis of the results

This section discusses the above results and analyzes the reasons why the proposed MOCEA outperforms seven MOEAs. The compared algorithms, such as EJAYA, HPGWO, IMABC, MMBO, MOPSO, NSGA-II and SPEA2, first start with one initial population, and then organize different evolution processes with mutation or crossover to guide the population to generate new individuals. These algorithms are sensitive to the scale of problems. For example, in the small-scale instances such as P7, P8, P9 and P11, these algorithms obtain the similar Pareto sets to MOCEA. However, with the increase of instance scale, the performance gap between these algorithms and MOCEA increase.

Table 8
Average HVR values for MOCEA and compared algorithms.

Instances	EJAYA	HPGWO	IMABC	MMBO	MOPSO	NSGA-II	SPEA2	MOCEA
P7	0.90	0.87	0.86	0.78	0.86	0.84	0.87	0.85
P8	0.87	0.85	0.82	0.74	0.83	0.79	0.84	0.79
P9	0.89	0.86	0.84	0.76	0.87	0.84	0.87	0.85
P11	0.87	0.84	0.81	0.73	0.83	0.79	0.84	0.82
P21	0.81	0.79	0.77	0.70	0.80	0.75	0.77	0.86
P25	0.76	0.74	0.72	0.67	0.75	0.73	0.73	0.83
P28	0.80	0.77	0.75	0.69	0.80	0.75	0.78	0.85
P29	0.72	0.70	0.68	0.62	0.72	0.68	0.70	0.83
P30	0.69	0.67	0.65	0.61	0.69	0.65	0.67	0.84
P32	0.73	0.70	0.67	0.62	0.73	0.68	0.71	0.84
P35	0.73	0.72	0.71	0.66	0.75	0.70	0.70	0.87
P45	0.74	0.72	0.71	0.66	0.76	0.71	0.71	0.86
P53	0.80	0.78	0.78	0.73	0.82	0.78	0.80	0.84
P58	0.62	0.62	0.61	0.57	0.64	0.60	0.60	0.85
P70	0.62	0.61	0.61	0.56	0.63	0.60	0.59	0.86
P75	0.57	0.57	0.56	0.54	0.58	0.56	0.56	0.85
P83	0.68	0.68	0.66	0.63	0.71	0.67	0.66	0.85
P89	0.62	0.62	0.61	0.58	0.64	0.61	0.60	0.86
P94	0.62	0.62	0.62	0.57	0.64	0.61	0.60	0.87
P111	0.61	0.61	0.60	0.56	0.63	0.60	0.59	0.85
P148	0.56	0.57	0.56	0.52	0.59	0.56	0.55	0.86
P297	0.50	0.50	0.50	0.47	0.52	0.50	0.49	0.87
Avg.	0.65	0.64	0.63	0.59	0.66	0.63	0.63	**0.85**

Table 9
Average I_ϵ values for MOCEA and compared algorithms.

Instances	EJAYA	HPGWO	IMABC	MMBO	MOPSO	NSGA-II	SPEA2	MOCEA
P7	1.13	1.15	1.16	1.25	1.17	1.18	1.16	1.19
P8	1.15	1.17	1.17	1.27	1.19	1.21	1.18	1.24
P9	1.11	1.13	1.14	1.21	1.13	1.15	1.13	1.15
P11	1.19	1.23	1.26	1.37	1.24	1.28	1.24	1.29
P21	1.20	1.20	1.23	1.31	1.21	1.25	1.23	1.20
P25	1.25	1.27	1.28	1.33	1.27	1.28	1.28	1.23
P28	1.33	1.37	1.39	1.49	1.34	1.39	1.38	1.32
P29	1.34	1.36	1.38	1.45	1.34	1.38	1.38	1.23
P30	1.35	1.38	1.40	1.44	1.36	1.39	1.39	1.21
P32	1.31	1.33	1.36	1.42	1.30	1.35	1.32	1.20
P35	1.28	1.29	1.29	1.34	1.27	1.31	1.32	1.18
P45	1.35	1.39	1.41	1.46	1.34	1.41	1.40	1.24
P53	1.26	1.27	1.28	1.32	1.23	1.28	1.27	1.27
P58	1.31	1.31	1.32	1.34	1.30	1.32	1.33	1.14
P70	1.49	1.49	1.50	1.55	1.47	1.51	1.53	1.20
P75	1.35	1.35	1.35	1.37	1.34	1.35	1.36	1.13
P83	1.31	1.31	1.32	1.34	1.28	1.32	1.33	1.15
P89	1.30	1.30	1.31	1.33	1.28	1.30	1.32	1.13
P94	1.38	1.38	1.38	1.42	1.37	1.39	1.41	1.14
P111	1.41	1.41	1.42	1.45	1.40	1.43	1.44	1.17
P148	1.45	1.44	1.45	1.48	1.43	1.45	1.46	1.17
P297	1.50	1.50	1.50	1.52	1.49	1.50	1.51	1.14
Avg.	1.36	1.36	1.37	1.41	1.35	1.37	1.38	**1.18**

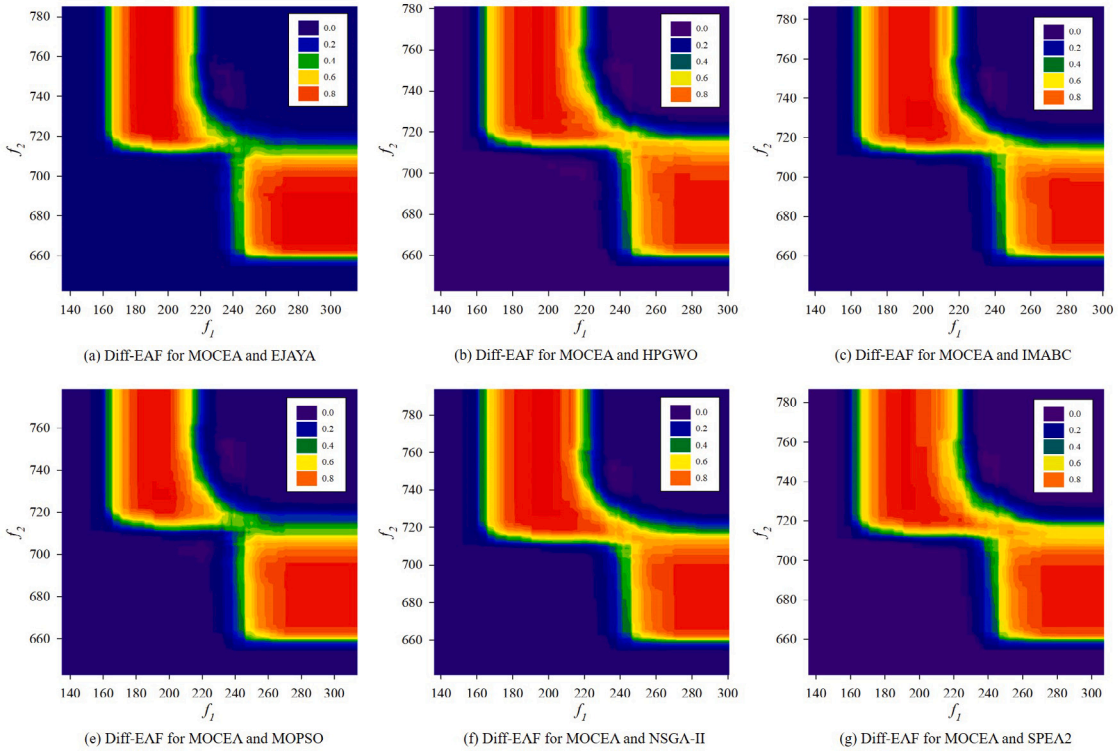
In the proposed MOCEA, to accelerate evolutionary process, a collaborative initialization is designed to generate high-quality initial dual-population. Meanwhile, different populations aim to explore different solution spaces. To ensure full coverage of the solution spaces, this work considers coding and variable knowledge and design problem-specific evolutionary operators for each sub-population. A self-learning selection strategy is embedded into the evolutionary process to dynamically select appropriate operator. This strategy can overcome the sensitivity to instance scale. Finally, this work develops a dual-cooperation to share superior information among different populations to avoid falling into local optimal. Therefore, the designs of the above strategies enhance the performance of MOCEA, resulting in the superiority of MOCEA.

6. Conclusions, managerial insights, and future work

This work studies the energy and cost-oriented mixed-model multi-manned assembly balancing with multi-skilled workers via a new MILP model and a MOCEA. In the MILP model, the skill level of different workers in energy requirement and assembly capacity is considered as well as the multiple-worker collaboration at the same workstations. Two conflicting objectives, including energy and cost requirements, are defined and minimized in this MILP model. In the proposed MOCEA, the problem-specific knowledge is considered to design a two-layer solution representation, a new decoding mechanism with idle time reduction and four evolutionary operators. Three improvement strategies, including collaborative initialization, self-learning selection and dual-cooperation, are developed to enhance the performance of MOCEA. The collaborative initialization help the algorithm obtain high-quality initial populations. The self-learning selection dynamically select an appropriate operator for each sub-population at every iteration to improve the adaptability of different instance scales. The dual-cooperation shares the superior information between different populations to avoid falling into local optimal. Finally, the results of four sets of experiments based on 269 instances demonstrate the effectiveness of the improvement components, as well as the superiority of MOCEA over 7 latest multi-objective evolutionary algorithms.

Table 10
Average IGD values for MOCEA and compared algorithms.

Instances	EJAYA	HPGWO	IMABC	MMBO	MOPSO	NSGA-II	SPEA2	MOCEA
P7	0.14	0.16	0.17	0.24	0.18	0.19	0.17	0.19
P8	0.20	0.23	0.24	0.32	0.23	0.27	0.23	0.27
P9	0.13	0.17	0.18	0.26	0.16	0.19	0.16	0.20
P11	0.17	0.20	0.22	0.31	0.21	0.24	0.20	0.25
P21	0.21	0.22	0.25	0.31	0.21	0.26	0.24	0.23
P25	0.22	0.23	0.24	0.28	0.22	0.24	0.24	0.22
P28	0.21	0.24	0.25	0.30	0.21	0.25	0.23	0.21
P29	0.22	0.24	0.26	0.32	0.22	0.26	0.25	0.18
P30	0.26	0.28	0.30	0.34	0.26	0.30	0.28	0.17
P32	0.22	0.25	0.28	0.33	0.23	0.28	0.25	0.21
P35	0.23	0.25	0.26	0.30	0.21	0.27	0.27	0.18
P45	0.23	0.24	0.26	0.30	0.21	0.26	0.25	0.17
P53	0.22	0.23	0.24	0.29	0.20	0.24	0.22	0.25
P58	0.30	0.30	0.31	0.35	0.28	0.32	0.33	0.16
P70	0.30	0.30	0.31	0.36	0.28	0.32	0.32	0.15
P75	0.36	0.36	0.37	0.40	0.34	0.37	0.37	0.16
P83	0.24	0.24	0.26	0.29	0.21	0.25	0.26	0.17
P89	0.28	0.28	0.30	0.33	0.26	0.30	0.30	0.15
P94	0.26	0.25	0.26	0.31	0.24	0.27	0.28	0.14
P111	0.28	0.28	0.29	0.33	0.26	0.30	0.30	0.16
P148	0.32	0.32	0.33	0.36	0.30	0.33	0.34	0.15
P297	0.39	0.39	0.39	0.42	0.36	0.39	0.39	0.13
Avg.	0.28	0.29	0.30	0.34	0.27	0.30	0.30	**0.17**



(a) Diff-EAF for MOCEA and EJAYA (b) Diff-EAF for MOCEA and HPGWO (c) Diff-EAF for MOCEA and IMABC

(e) Diff-EAF for MOCEA and MOPSO (f) Diff-EAF for MOCEA and NSGA-II (g) Diff-EAF for MOCEA and SPEA2

Fig. 7. The Differences between Empirical Attainment Function for MOCEA and compared algorithms in instance91.

Table 11
Average coverage values C(MOCEA, ...).

Instances	EJAYA	HPGWO	IMABC	MMBO	MOPSO	NSGA-II	SPEA2
P7	0.12	0.21	0.11	0.33	0.13	0.19	0.07
P8	0.00	0.00	0.00	0.00	0.00	0.00	0.00
P9	0.13	0.13	0.34	0.40	0.16	0.29	0.22
P11	0.03	0.16	0.25	0.42	0.23	0.26	0.25
P21	0.36	0.43	0.67	0.55	0.40	0.56	0.35
P25	0.46	0.65	0.67	0.67	0.56	0.73	0.66
P28	0.34	0.31	0.48	0.58	0.26	0.58	0.33
P29	0.63	0.70	0.79	0.87	0.65	0.77	0.62
P30	0.78	0.87	0.90	0.86	0.79	0.91	0.75
P32	0.38	0.59	0.76	0.66	0.53	0.68	0.60
P35	0.46	0.45	0.62	0.60	0.44	0.68	0.57
P45	0.57	0.56	0.63	0.67	0.46	0.70	0.64
P53	0.11	0.18	0.21	0.24	0.17	0.28	0.15
P58	0.71	0.64	0.83	0.82	0.72	0.91	0.78
P70	0.74	0.86	0.91	0.86	0.76	0.89	0.87
P75	0.86	0.85	0.94	0.94	0.85	0.97	0.92
P83	0.46	0.45	0.73	0.65	0.48	0.68	0.57
P89	0.65	0.60	0.85	0.80	0.65	0.81	0.79
P94	0.56	0.49	0.61	0.73	0.47	0.69	0.73
P111	0.63	0.71	0.86	0.81	0.60	0.85	0.80
P148	0.81	0.79	0.89	0.89	0.76	0.89	0.84
P297	0.92	0.95	0.97	0.96	0.91	0.96	0.98
Avg.	0.63	0.65	0.77	0.77	0.62	0.77	0.71

Table 12
Average coverage values C(..., MOCEA).

Instances	EJAYA	HPGWO	IMABC	MMBO	MOPSO	NSGA-II	SPEA2
P7	0.74	0.63	0.62	0.42	0.69	0.46	0.71
P8	1.00	1.00	1.00	1.00	1.00	0.80	1.00
P9	0.73	0.59	0.55	0.32	0.70	0.47	0.59
P11	0.63	0.58	0.51	0.25	0.48	0.39	0.49
P21	0.41	0.41	0.27	0.22	0.55	0.07	0.16
P25	0.47	0.37	0.31	0.19	0.42	0.31	0.42
P28	0.67	0.46	0.36	0.34	0.64	0.30	0.60
P29	0.19	0.18	0.19	0.06	0.22	0.08	0.20
P30	0.10	0.06	0.03	0.04	0.14	0.04	0.10
P32	0.39	0.27	0.08	0.20	0.32	0.16	0.21
P35	0.42	0.30	0.31	0.23	0.39	0.18	0.27
P45	0.24	0.32	0.21	0.21	0.39	0.15	0.20
P53	0.53	0.61	0.43	0.55	0.84	0.41	0.75
P58	0.15	0.14	0.10	0.05	0.13	0.06	0.11
P70	0.08	0.05	0.05	0.02	0.11	0.02	0.03
P75	0.02	0.03	0.02	0.01	0.04	0.00	0.02
P83	0.25	0.22	0.12	0.16	0.30	0.12	0.20
P89	0.13	0.15	0.05	0.07	0.14	0.06	0.08
P94	0.17	0.22	0.14	0.08	0.23	0.13	0.09
P111	0.13	0.12	0.06	0.06	0.12	0.08	0.09
P148	0.06	0.08	0.04	0.05	0.08	0.04	0.06
P297	0.02	0.02	0.02	0.01	0.02	0.01	0.01
Avg.	0.20	0.19	0.14	0.11	0.22	0.11	0.16

The managerial relevance of MOCEA is implemented when the multi-skilled workers and mixed-model are considered in multi-manned assembly lines. When using the MOCEA to reduce the energy and cost requirements of real applications, managers need to first quantify the cost factors of opening new stations and hiring different workers, as well as the energy requirements of different workers. Then, they could consider minimizing the energy and cost requirements simultaneously via the MILP model, and employ MOCEA to obtain a set of task and worker assignment plans. Finally, taking into account the importance of different objectives, managers could select one plan to organize the assembly line layout.

Since human–robot cooperation is proposed in Industry 5.0, different operation modes will appear, such as only-human-assembly, only-robot-assembly and human–robot-cooperation-assembly. Therefore, future works can be extended by considering energy-consumption-related human–robot-cooperation assembly line balancing. Besides, the future work could study the application of a MOEA based on differential evolution (Tan et al., 2006; Tian et al., 2021) to compare its performance with respect to the MOEAs studied in this work.

Table 13
The results of Shapiro–Wilk test.

Algorithm	DF	Statistic	*p*-value	Decision at level(5%)
HVR				
EJAYA	269	0.97337	6.47491E−5	Reject normality
HPGWO	269	0.97866	4.65751E−4	Reject
IMABC	269	0.9769	2.36348E−4	Reject
MMBO	269	0.98553	0.00812	Reject
MOPSO	269	0.98105	0.00121	Reject
NSGA-II	269	0.98135	0.00136	Reject
SPEA2	269	0.96823	1.11772E−5	Reject
MOCEA	269	0.99266	0.20669	No reject
I_ϵ				
EJAYA	269	0.98824	0.02744	Reject
HPGWO	269	0.98423	0.0046	Reject
IMABC	269	0.98521	0.00705	Reject
MMBO	269	0.97905	5.43053E−4	Reject
MOPSO	269	0.97895	5.20625E−4	Reject
NSGA-II	269	0.98176	0.00162	Reject
SPEA2	269	0.98807	0.02542	Reject
MOCEA	269	0.88375	1.81188E−13	Reject

Table 14
The results of Levene's test.

	DF	*Seq SS*	*Adj MS*	*F*-value	*p*-value
HVR					
Model	7	0.02818	0.00403	23.64533	3.849E−31
Error	2144	6.9768	0.00325		
I_ϵ					
Model	7	0.03838	0.00548	11.01741	9.6129E−14
Error	2144	1.06702	4.97677E−4		

CRediT authorship contribution statement

Zikai Zhang: Methodology, Writing – original draft, Writing – review & editing. **Manuel Chica:** Methodology, Writing – review & editing. **Qiuhua Tang:** Writing – review & editing. **Zixiang Li:** Writing – review & editing. **Liping Zhang:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the China Postdoctoral Science Foundation (No. 2021M702536) and the National Natural Science Foundation of China (No 62303358, 52275504, 62173260). M. Chica was supported by EMERGIA21_00139, funded by Consejería de Universidad, Investigación e Innovación of the Andalusian Government and by "ERDF A way of making Europe"

Table 15
The results of Wilcoxon's rank-sum test.

Control algorithm	Compared algorithm	p-value on HVR	p-value on I_ϵ	$\alpha = 0.05$	$\alpha = 0.01$
MOCEA	EJAYA	2.56434E−67	0	Yes	Yes
	HPGWO	1.09441E−75	0	Yes	Yes
	IMABC	8.11036E−80	0	Yes	Yes
	MMBO	4.29749E−89	0	Yes	Yes
	MOPSO	1.52159E−74	0	Yes	Yes
	NSGA-II	3.52235E−83	0	Yes	Yes
	SPEA2	4.30529E−75	0	Yes	Yes

Table 16
The results of all Friedman ANOVA.

Algorithm	Rank value on HVR		Rank value on I_ϵ	
	Mean	Sum	Mean	Sum
EJAYA	3.54647	954	3.74721	1008
HPGWO	4.02602	1083	4.1171	1107.5
IMABC	5.47584	1473	5.28253	1421
MMBO	7.9777	2146	7.46468	2008
MOPSO	2.37175	638	2.63197	708
NSGA-II	5.65799	1522	5.30669	1427.5
SPEA2	5.61338	1510	5.94424	1599
MOCEA	1.33086	358	1.50558	405

References

Abidin Çil, Z., & Kizilay, D. (2020). Constraint programming model for multi-manned assembly line balancing problem. *Computers & Operations Research*, *124*, Article 105069.

Akagi, F., Osaki, H., & Kikuchi, S. (1983). A method for assembly line balancing with more than one worker in each station. *International Journal of Production Research*, *21*(5), 755–770.

Andreu-Casas, E., García-Villoria, A., & Pastor, R. (2022). Multi-manned assembly line balancing problem with dependent task times: a heuristic based on solving a partition problem with constraints. *European Journal of Operational Research*, *302*(1), 96–116.

Battini, D., Delorme, X., Dolgui, A., Persona, A., & Sgarbossa, F. (2016). Ergonomics in assembly line balancing based on energy expenditure: a multi-objective model. *International Journal of Production Research*, *54*(3), 824–845.

Belkharroubi, L., & Yahyaoui, K. (2022). Solving the energy-efficient robotic mixed-model assembly line balancing problem using a memory-based cuckoo search algorithm. *Engineering Applications of Artificial Intelligence*, *114*, Article 105112.

Boysen, N., Schulze, P., & Scholl, A. (2022). Assembly line balancing: What happened in the last fifteen years? *European Journal of Operational Research*, *301*(3), 797–814.

Chen, Y.-Y. (2017). A hybrid algorithm for allocating tasks, operators, and workstations in multi-manned assembly lines. *Journal of Manufacturing Systems*, *42*, 196–209.

Chen, Y.-Y., Cheng, C.-Y., & Li, J.-Y. (2018). Resource-constrained assembly line balancing problems with multi-manned workstations. *Journal of Manufacturing Systems*, *48*, 107–119.

Chutima, P., & Khotsaenlee, A. (2022). Multi-objective parallel adjacent U-shaped assembly line balancing collaborated by robots and normal and disabled workers. *Computers & Operations Research*, *143*, Article 105775.

Fonseca, C. M., Knowles, J. D., Thiele, L., & Zitzler, E. (2006). A tutorial on tbe performance assessment of stochastic multiobjective optimizers. *TIK-report*.

Gong, G., Deng, Q., Gong, X., Liu, W., & Ren, Q. (2018). A new double flexible job-shop scheduling problem integrating processing time, green production, and human factor indicators. *Journal of Cleaner Production*, *174*, 560–576.

Li, R., Gong, W., Wang, L., Lu, C., & Jiang, S. (2022). Two-stage knowledge-driven evolutionary algorithm for distributed green flexible job shop scheduling with type-2 fuzzy processing time. *Swarm and Evolutionary Computation*, *74*, Article 101139.

Li, Z., Janardhanan, M. N., & Ponnambalam, S. G. (2021). Cost-oriented robotic assembly line balancing problem with setup times: multi-objective algorithms. *Journal of Intelligent Manufacturing*, *32*(4), 989–1007.

Li, Z., Janardhanan, M. N., & Tang, Q. (2021). Multi-objective migrating bird optimization algorithm for cost-oriented assembly line balancing problem with collaborative robots. *Neural Computing & Applications*, *33*(14, SI), 8575–8596.

Li, Z., Tang, Q., & Zhang, L. (2016). Minimizing energy consumption and cycle time in two-sided robotic assembly line systems using restarted simulated annealing algorithm. *Journal of Cleaner Production*, *135*, 508–522.

Liu, R., Liu, M., Chu, F., Zheng, F., & Chu, C. (2021). Eco-friendly multi-skilled worker assignment and assembly line balancing problem. *Computers & Industrial Engineering*, *151*, Article 106944.

Liu, X., Yang, X., & Lei, M. (2021). Optimisation of mixed-model assembly line balancing problem under uncertain demand. *Journal of Manufacturing Systems*, *59*, 214–227.

Lopes, T. C., Pastre, G. V., Michels, A. S., & Magatão, L. (2020). Flexible multi-manned assembly line balancing problem: Model, heuristic procedure, and lower bounds for line length minimization. *Omega*, *95*, Article 102063.

Meng, K., Tang, Q., Cheng, L., & Zhang, Z. (2022). Mixed-model assembly line balancing problem considering preventive maintenance scenarios: MILP model and cooperative co-evolutionary algorithm. *Applied Soft Computing*, *127*, Article 109341.

Michels, A. S., Lopes, T. C., Sikora, C. G. S., & Magatão, L. (2019). A benders' decomposition algorithm with combinatorial cuts for the multi-manned assembly line balancing problem. *European Journal of Operational Research*, *278*(3), 796–808.

nez, M. L.-I., Paquete, L., & Stützle, T. (2006). Hybrid population-based algorithms for the bi-objective quadratic assignment problem. *Journal of Mathematical Modelling & Algorithms*, *5*(1), 111–137.

Nilakantan, J. M., Li, Z., Tang, Q., & Nielsen, P. (2017). Multi-objective co-operative co-evolutionary algorithm for minimizing carbon footprint and maximizing line efficiency in robotic assembly line systems. *Journal of Cleaner Production*, *156*, 124–136.

Niroomand, S. (2021). Hybrid artificial electric field algorithm for assembly line balancing problem with equipment model selection possibility. *Knowledge-Based Systems*, *219*, Article 106905.

Pan, Z., Lei, D., & Wang, L. (2022a). A bi-population evolutionary algorithm with feedback for energy-efficient fuzzy flexible job shop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *52*(8), 5295–5307.

Pan, Z., Lei, D., & Wang, L. (2022b). A knowledge-based two-population optimization algorithm for distributed energy-efficient parallel machines scheduling. *IEEE Transactions on Cybernetics*, *52*(6), 5051–5063.

Pereira, J., Ritt, M., Óscar, C., & Vásquez (2018). A memetic algorithm for the cost-oriented robotic assembly line balancing problem. *Computers & Operations Research*, *99*, 249–261.

Rabbani, M., Mousavi, Z., & Farrokhi-Asl, H. (2016). Multi-objective metaheuristics for solving a type II robotic mixed-model assembly line balancing problem. *Journal of Industrial and Production Engineering*, *33*(7), 472–484.

Rahman, H. F., Janardhanan, M. N., & Ponnambalam, S. (2023). Energy aware semi-automatic assembly line balancing problem considering ergonomic risk and uncertain processing time. *Expert Systems with Applications*, *231*, Article 120737.

Şahin, M., & Kellegöz, T. (2019). 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*, *133*, 107–120.

Salehi, M., Maleki, H. R., & Niroomand, S. (2020). Solving a new cost-oriented assembly line balancing problem by classical and hybrid meta-heuristic algorithms. *Neural Computing & Applications*, *32*(12, SI), 8217–8243.

Scholl, A., Fliedner, M., & Boysen, N. (2010). Absalom: Balancing assembly lines with assignment restrictions. *European Journal of Operational Research*, *200*(3), 688–701.

Sun, B., Wang, L., & Peng, Z. (2020). Bound-guided hybrid estimation of distribution algorithm for energy-efficient robotic assembly line balancing. *Computers & Industrial Engineering*, *146*, Article 106604.

Tan, K., Yang, Y., & Goh, C. (2006). A distributed cooperative coevolutionary algorithm for multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, *10*(5), 527–549.

Tian, Y., Zhang, T., Xiao, J., Zhang, X., & Jin, Y. (2021). A coevolutionary framework for constrained multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, *25*(1), 102–116.

Wu, T., Zhang, Z., Zeng, Y., & Zhang, Y. (2023). Mixed-integer programming model and hybrid local search genetic algorithm for human–robot collaborative disassembly line balancing problem. *International Journal of Production Research*, 1–25.

Wu, T., Zhang, Z., Zhang, Y., & Zeng, Y. (2023). Modelling and optimisation of two-sided disassembly line balancing problem with human–robot interaction constraints. *Expert Systems with Applications*, *230*, Article 120589.

Yagmahan, B. (2011). Mixed-model assembly line balancing using a multi-objective ant colony optimization approach. *Expert Systems with Applications*, *38*(10), 12453–12461.

Yang, H., Lee, J.-H., Lee, S. H., Lee, S. G., Kim, H. R., & Kim, H.-J. (2022). A multi-manned assembly line worker assignment and balancing problem with positional constraints. *IEEE Robotics and Automation Letters*, *7*(3), 7786–7793.

Yuan, Y., & Xu, H. (2015). Multiobjective flexible job shop scheduling using memetic algorithms. *IEEE Transactions on Automation Science and Engineering*, *12*(1), 336–353.

Zangaro, F., Minner, S., & Battini, D. (2023). The multi-manned joint assembly line balancing and feeding problem. *International Journal of Production Research*, *61*(16), 5543–5565.

Zhang, Z., Tang, Q., & Chica, M. (2020). Multi-manned assembly line balancing with time and space constraints: A MILP model and memetic ant colony system. *Computers & Industrial Engineering*, *150*, Article 106862.

Zhang, Z., Tang, Q., & Chica, M. (2021). A robust MILP and gene expression programming based on heuristic rules for mixed-model multi-manned assembly line balancing. *Applied Soft Computing*, *109*, Article 107513.

Zhang, Z., Tang, Q., Chica, M., & Li, Z. (2023). Reinforcement learning-based multiobjective evolutionary algorithm for mixed-model multimanned assembly line balancing under uncertain demand. *IEEE Transactions on Cybernetics*, 1–14.

Zhang, Z., Tang, Q., Han, D., & Li, Z. (2023). Multi-manned assembly line balancing with sequence-dependent set-up times using an enhanced migrating birds optimization algorithm. *Engineering Optimization*, *55*(7), 1243–1262.

Zhang, Z., Tang, Q., Han, D., & Qian, X. (2021). An enhanced multi-objective JAYA algorithm for U-shaped assembly line balancing considering preventive maintenance scenarios. *International Journal of Production Research*, *59*(20), 6146–6165.

Zhang, Z., Tang, Q., Li, Z., & Han, D. (2021). An efficient migrating birds optimization algorithm with idle time reduction for type-i multi-manned assembly line balancing problem. *Journal of Systems Engineering and Electronics*, *32*(2), 286–296.

Zhang, Z., Tang, Q., Li, Z., & Zhang, L. (2019). Modelling and optimisation of energy-efficient U-shaped robotic assembly line balancing problems. *International Journal of Production Research*, *57*(17), 5520–5537.

Zhang, Z., Tang, Q., & Zhang, L. (2019). Mathematical model and grey wolf optimization for low-carbon and low-noise U-shaped robotic assembly line balancing problem. *Journal of Cleaner Production*, *215*, 744–756.

Zhang, B., & Xu, L. (2020). An improved flower pollination algorithm for solving a type-II U-shaped assembly line balancing problem with energy consideration. *Assembly Automation*, *40*(6), 847–856.

Zhang, B., Xu, L., & Zhang, J. (2020). A multi-objective cellular genetic algorithm for energy-oriented balancing and sequencing problem of mixed-model assembly line. *Journal of Cleaner Production*, *244*, Article 118845.

Zhao, W., Alam, S., & Abbass, H. A. (2014). MOCCA-II: A multi-objective co-operative co-evolutionary algorithm. *Applied Soft Computing*, *23*, 407–416.

Zhao, F., He, X., & Wang, L. (2021). A two-stage cooperative evolutionary algorithm with problem-specific knowledge for energy-efficient scheduling of no-wait flow-shop problem. *IEEE Transactions on Cybernetics*, *51*(11), 5291–5303.

Zhou, B.-H., & Shen, C.-Y. (2018). Multi-objective optimization of material delivery for mixed model assembly lines with energy consideration. *Journal of Cleaner Production*, *192*, 293–305.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, *8*(2), 173–195.

Zitzler, E., Laumanns, M., & Thiele, L. (2001). SPEA2: Improving the strength pareto evolutionary algorithm. *Technical Report Gloriastrasse*.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, *3*(4), 257–271.