Multi-Objective Optimization of Multi-Factory Remanufacturing Process Considering Worker Fatigue

Liangbo Zhou, Haibin Zhu, Behzad Akbari

***Abstract*—To achieve sustainable manufacturing of large-scale discarded products, disassembly, recycling, and remanufacturing are widely adopted across various industries. Multi-factory remanufacturing is a complex working model that requires coordination among different factories for disassembly, remanufacturing, and resource circulation to achieve optimal resource reuse and reduce environmental impact. Multi-skilled workers play a crucial role in this process, and to fully harness the potential of workers, skill training and task allocation need to be considered. On the other hand, prolonged disassembly leads to an increase in worker fatigue levels. Excessive fatigue can result in reduced work efficiency and quality, even posing a threat to worker health and safety. Considering the impact of worker fatigue during disassembly is essential for achieving healthy and efficient work. This work presents and addresses a multi-factory remanufacturing process optimization problem that considers worker fatigue. This problem is divided into three phases: disassembly factory selection, disassembly scheduling, and manufacturing factory selection. With the objectives of maximizing profit and minimizing fatigue index, a linear programming mathematical model is established. Based on this, a discrete battle royale optimizer is employed to solve the problem, and a novel encoding structure is designed. The superiority and effectiveness of the proposed optimizer are validated through experiments on different scale cases and by comparing the results with the carnivorous plant optimizer, migrating birds optimizer, dingo optimizer, and fruit fly optimizer.**

***Key Words*—Multi-factory remanufacturing process optimization, multi-skilled worker, fatigue index, multi-objective optimization, battle royale optimizer**

I. Introduction

Human society's consumption patterns and technological advancements have made tremendous progress in the past few decades. However, this progress has also brought forth a significant issue: many discarded products and electronic waste [1], [2]. As technology continues to update and the lifespan of products shortens, the quantity of discarded products rapidly increases, causing severe impacts on the environment and resources [3], [4], [5]. Traditionally, discarded products are regarded as waste and simply disposed of, eventually ending up in landfills or incineration facilities [6]. However, this disposal method has significant negative impacts on the environment and wastes resources [7], [8], [9]. In recent years, as the environmental crisis intensifies and the depletion of energy and natural resources becomes more pronounced, many countries are implementing environmental legislation that requires producers to take responsibility for the entire lifecycle of their products to conserve resources and safeguard the environment [10], [11], [12]. To address these issues, the multi-factory remanufacturing model has emerged, and its structure is shown in Fig. 1. As a resource-efficient method, multi-factory remanufacturing transforms waste into regenerated resources, helping to achieve efficient resource use, reduce environmental impact, and enhance corporate competitiveness [13], [14], [15]. As an emerging model, multi-factory remanufacturing faces a series of complex challenges, as follows.

- Multi-factory coordination and collaboration: In a multi-factory setting, it is necessary to coordinate disassembly tasks and component circulation among different factories, ensuring the synchronization of disassembly lines with other remanufacturing processes to maximize resource reuse.
- Optimizing the disassembly process: Different factories may need to handle various types of products and materials, leading to increased complexity in internal disassembly procedures that require effective organization and management to ensure efficient disassembly and maximize resource reuse.
- Training multi-skilled workers: In a multi-factory remanufacturing model, nurturing and maintaining a workforce with a diverse range of skills is a key challenge. This entails providing extensive training to workers and matching them with suitable tasks to ensure they are competent in various responsibilities, thereby enhancing production efficiency and quality.
- Worker fatigue management: Workers may be required to engage in prolonged disassembly tasks, which can lead to worker fatigue. Worker fatigue can impact their health

Manuscript received July 8, 2025; revised July 14 and July 26, 2025; accepted July 27, 2025. This article was recommended for publication by Editor Shujin Qin upon evaluation of the reviewers' comments.

This work was supported in part by the National Local Joint Engineering Laboratory for Optimization of Petrochemical Process Operation and Energy Saving Technology under Grant LJ232410148002, and in part by the Innovation Team Project of the Educational Department of Liaoning Province under Grant LJ222410148036.

L. Zhou is with the Development and Reform Bureau of Congjiang County, Guizhou,557400, China (email: zlb15185602324@163.com).

H. Zhu is with Nipissing University, Canada, North Bay, ON P1B 8L7 Canada (email: haibinz@nipissingu.ca).

Behzad Akbari is with Michigan Technological University, Houghton, MI,49931, USA (email: behzadak@mtu.edu).

Corresponding Author: Haibin Zhu.

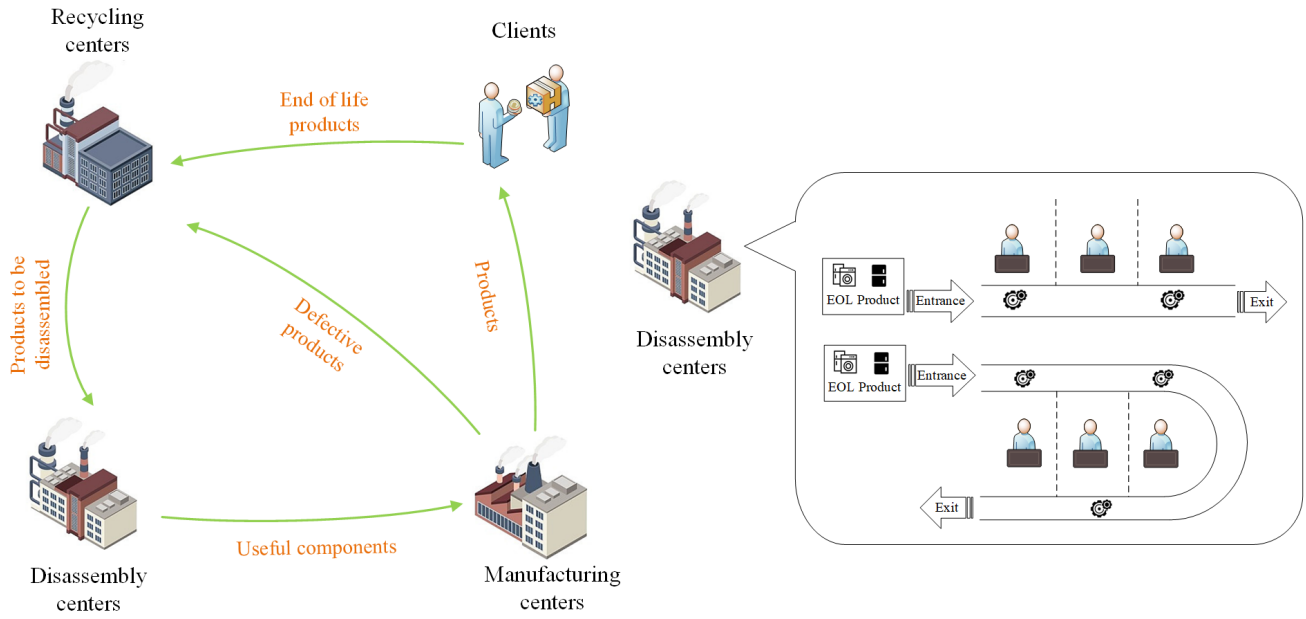

Fig. 1. A multi-factory remanufacturing process optimization problem.

and safety, increasing the risk of accidents and errors. This necessitates the implementation of effective work schedule arrangements and safety measures to ensure worker well-being while maintaining high efficiency and quality.

In modern manufacturing, the challenges of the disassembly line balancing problem (DLBP) [16] and the multi-factory remanufacturing process optimization problem (MRPOP) are two key issues. DLBP involves determining how to efficiently allocate workers and equipment on the disassembly line to achieve optimal production efficiency and resource utilization[17]. On the other hand, MRPOP focuses on effectively scheduling and coordinating disassembly tasks across multiple factories to maximize overall disassembly efficiency and resource utilization [18]. Combining these two issues achieves a more comprehensive and optimized process for disassembling and recycling discarded products. By rationally allocating disassembly tasks across multiple factories and balancing workloads for workers and equipment on the disassembly line, overall production efficiency is enhanced, resource wastage is reduced, and efficient recovery and reuse of discarded products are ensured. Since the inception of DLBP, many scholars have conducted in-depth research on it. Wang *et al.* [19] introduce partial dismantling and uncertain disassembly times into a U-shaped disassembly line, establishing a mathematical model for the U-shaped DLBP with partial dismantling modes, and apply a multi-objective discrete flower pollination algorithm for solving. Cui *et al.* [20] propose a discrete whale optimizer to solve DLBP with the goal of profit maximization. Çil *et al.* [21] propose a hybrid integer linear programming model and an ant colony optimization-based heuristic algorithm to solve the robot DLBP. In recent years, domestic and international scholars conduct relevant research on Multi-factory Production Scheduling Problems (MPSP) [22], yet very few explore MRPOP. For instance, Chung *et al.* [23] introduce an enhanced genetic algorithm (GA) approach to solve distributed scheduling models considering maintenance, aiming to minimize job completion times. Marandi *et al.* [24] propose a novel cloud theory-based Hybrid Cloud Learning Simulated Annealing (HCLSA) for solving multi-factory scheduling problems in supply chains involving batch deliveries and assembly. Gharaei *et al.* [25] study parallel multi-factory scheduling problems with interfering job sets and present an effective decomposition-based multi-objective evolutionary algorithm to resolve them. Chung *et al.* [26] introduce an improved GA for handling MPSP. Research into MPSP yields some achievements. Building upon previous research on MPSP, this work proposes MRPOP by considering worker fatigue, thus filling the gap in previous research on MRPOP.

The actual disassembly process still relies primarily on manual disassembly, and disassembly companies consider human factors engineering as the foundation to ensure production quality, safety, and flexibility in the disassembly line [27]. Workers typically possess multiple skills that enable them to handle the frequent changes in the disassembly system and perform various tasks [28]. Human factors engineering revolves around human factors, thoroughly considering workers' working conditions and physical and mental health, enabling healthy and efficient work. This significantly enhances the disassembly efficiency of the disassembly line while reducing disassembly error rates. Therefore, to achieve healthy and efficient disassembly for workers, considering human factors engineering during the disassembly process and the rational allocation of these multi-skilled workers are crucial. This research has garnered widespread attention from scholars both domestically and internationally. Lian *et al.* [29] propose a metaheuristic algorithm based on NSGA-II to solve the problem of allocating multi-skilled workers, considering worker skill combinations and differences in proficiency levels. Liu *et al.* [30] establish a multi-objective mixed-integer linear programming model and propose a GA to

solve the allocation of multi-skilled workers and assembly line balancing problem considering energy consumption. Carnahan *et al.* [31] argue that the physical energy consumption of workers during the disassembly process is a crucial factor that cannot be overlooked. Baykasoglu *et al.* [32] point out that considering human factors engineering in the production line positively impacts both production efficiency and worker health. Otto *et al.* [33] provide an overview of existing optimization methods for assembly line balancing problems and job rotation scheduling problems considering ergonomic risks. This work integrates DLBP into MRPOP, proposing a multi-factory remanufacturing process optimization problem considering worker fatigue index (MRPWF) and discussing its specific application scenarios.

Given that MRPWF is an NP-hard problem, heuristic algorithms typically tackle such problems. In relevant studies, Guo *et al.* [34] present a multi-objective DLBP and utilize a stochastic simulation approach involving simulated annealing and a multi-objective discrete grey wolf optimizer for resolution. Fang *et al.* [35] introduce an evolutionary simulated annealing algorithm for solving the multi-robot hybrid DLBP. This problem involves minimizing cycle time, peak workstation energy consumption, and total energy consumption, among multiple objectives. Zhu *et al.* [36] propose a Pareto Firefly Algorithm for solving a multi-objective DLBP considering hazard assessment. This problem involves minimizing the number of workstations, maximizing smoothness, and minimizing the average maximum risk, solving multiple objectives. These experiences provide us with inspiration and ideas for solving the MRPWF.

The Battle Royale Optimizer (BRO), introduced by Rahkar-Farshi in 2020, is a population-based metaheuristic optimization algorithm inspired by the game PlayerUnknown's Battlegrounds (PUBG) [37]. It achieves the iterative evolution of populations by simulating the search process of trying to defeat neighboring soldiers in the game. We choose BRO to solve MRPWF because, through its unique simulation of competition and cooperation mechanisms, it demonstrates better convergence performance and solution quality when handling large-scale, multi-constrained problems [38], [39]. Compared to traditional multi-objective optimization algorithms, BRO has more significant advantages in exploring the search space and balancing local and global search.

The main contributions this work aims to make are:

1) Introducing DLBP into MRPOP while considering the impact of worker fatigue, we propose MRPWF. A mixed-integer programming model is formulated for this problem, aiming to maximize disassembly profit while minimizing the worker fatigue index.
2) For solving MRPWF, this work introduces a discrete battle royale optimizer (DBRO). In this algorithm, we devise a novel encoding structure to represent solutions, and we design four soldier search strategies, namely sequential variation, task variation, workstation variation, and factory swap, to enhance the algorithm's capability to search for optimal solutions.
3) Different experiments are designed to validate the algorithm's effectiveness. Compare the performance of the DBRO algorithm with the carnivorous plant optimizer (CPA) [40], Migrating Birds optimizer (MBO) [41], dingo optimizer (DOA) [42], and fruit fly optimizer (FOA) [43]. Evaluate the quality of solutions these five algorithms provide for solving MRPWF using three Pareto performance indicators. Experimental results demonstrate that the DBRO more effectively addresses the proposed problem.

The remaining sections of this paper are organized as follows. Section II describes MRPWF and its mathematical model. Section III introduces the encoding and decoding scheme of MRPWF, DBRO, and the soldier's search and battle processes. Section IV analyzes the experimental results. Section V summarizes the work of this paper and discusses future research directions.

II. Problem Description

A. Problem Statement

As shown in Fig.2, MRPWF is divided into three main phases.

1) Disassembly factory selection phase

This phase primarily involves product assignment. Product assignment deals with how to distribute different End-of-Life (EOL) products from the recycling center to the optimal disassembly factories based on constraints.

2) Disassembly scheduling phase

This phase mainly includes disassembly line assignment, task assignment, and workstation assignment. Disassembly line assignment involves selecting a disassembly line in each disassembly factory and assigning a product to it. Task assignment involves selecting a disassembly task sequence of the product. Workstation assignment involves assigning different disassembly tasks of the product to different workstations on the disassembly line.

3) Manufacturing factory selection phase

This phase handles subassembly assignment. Due to the varying prices offered by different manufacturing factories for the subassemblies and the differences in transportation costs due to the distances between disassembly factories and manufacturing factories, the related cost varies. Subassembly assignment involves determining how to allocate the obtained subassemblies to different manufacturing factories.

Therefore, the challenge of this problem lies in optimizing a range of issues, including tasks and workstation assignments within disassembly factories, as well as the selection of suitable disassembly and manufacturing factories. Thus, effectively utilizing existing resources, logically assigning product demands from different customers to various factories, and coordinating the plans of these factories constitute the primary focus of this work.

Faced with increasingly complex products and disassembly tasks, multi-skilled workers are crucial for addressing MRPWF. Each disassembly task may require multiple skills, as shown in Table I. Multi-skilled workers can rapidly adjust work assignments when facing various disassembly task requirements, helping to reduce workstation overload and enhance overall disassembly efficiency.

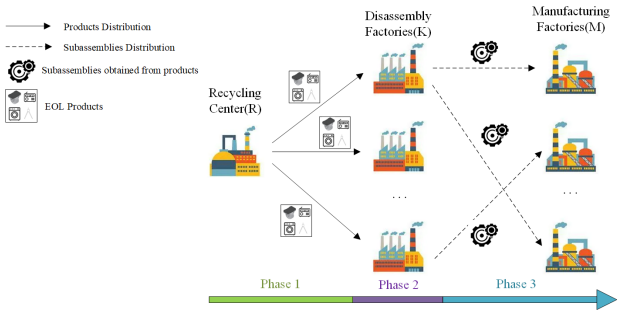

Fig. 2. The workflow of multi-factory remanufacturing process optimization

TABLE I Relationship between tasks and skills.

Tasks	Skill				
	1	2	3	4	5
1	0	1	1	0	1
2	1	0	0	1	0
3	0	1	1	1	0
4	1	0	1	1	0
5	1	0	0	0	1
6	1	0	1	0	1

In the real world, the fatigue level of disassembly workers is not constant, and it varies due to factors such as the fatigue growth parameters associated with different disassembly tasks and the duration of work. Therefore, it is necessary to schedule work hours reasonably to ensure that workers have adequate rest time, mitigating both physical and psychological fatigue during work.

$$F(t) = 1 - e^{-\lambda t} \tag{1}$$

In equation (1), $F(t)$ represents the fatigue level of the worker, and λ denotes the fatigue growth parameter for the disassembly task. The variable t represents the completion time of the disassembly task. Fig. 3 illustrates the relationship between working time and fatigue level. F_{max} represents a specific maximum fatigue threshold. When a worker's fatigue level exceeds this threshold, it increases the risk of injury and reduces work efficiency. Since each workstation is assigned to only one worker, we need to avoid assigning tasks that may lead to health risks due to excessively high fatigue levels.

The objective of MRPWF is to maximize the profit obtained from the disassembly of products while considering worker fatigue levels. The following assumptions are made in this work:

- Matrices D, R, and S are known.
- Not all subassemblies need to be disassembled (called selective disassembly).
- Each workstation is assigned one worker.
- The disassembled products are infinitely supplied.
- Each disassembly task requires at least one disassembly skill. The higher the skill level, the shorter the time required to complete the task.

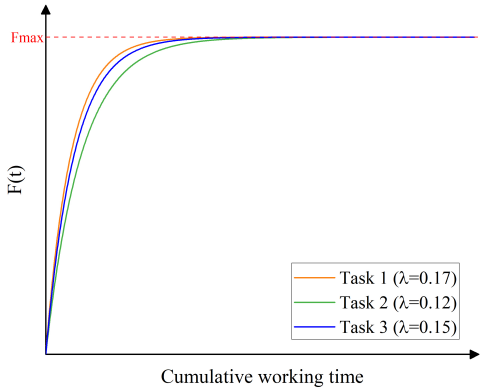

Fig. 3. Relationship between work time and fatigue level.

- The operating time of each workstation should be at most the given system cycle time.
- Each activated workstation has at least one disassembly task.

B. Notations

For clear presentation, all notations used in this work are listed as follows:

Sets:

$\mathbb{K}$ Set of disassembly factories, $\mathbb{K} = \{1, 2, ..., K\}$.
$\mathbb{M}$ Set of Manufacturing factories, $\mathbb{M} = \{1, 2, ..., M\}$.
$\mathbb{P}$ Set of products, $\mathbb{P} = \{1, 2, ..., P\}$.
$\mathbb{I}_p$ Set of all subassemblies in product p, $\mathbb{I}_p = \{1, 2, ..., I_p\}$.
$\mathbb{J}_p$ Set of all tasks in product p, $\mathbb{J}_p = \{1, 2, ..., J_p\}$.
$\mathbb{W}_k^L$ Set of linear workstations for the k-th factory, $\mathbb{W}_k^L = \{1, 2, ..., W_k^L\}$.
$\mathbb{W}_k^U$ Set of U-shaped workstations for the k-th factory, $\mathbb{W}_k^U = \{1, 2, ..., W_k^U\}$.
$\mathbb{E}$ Set of sides of U-shaped disassembly line workstation, $\mathbb{E} = \{1, 2\}$.
$\mathbb{N}$ Set of all additional skills, $\mathbb{N} = \{1, 2, ..., N\}$.

Indexes:

p Product index, $p \in \mathbb{P}$.
i Subassembly index, $i \in \mathbb{I}_p$.
j Disassembly task index, $j \in \mathbb{J}_p$.
e Index of U-shaped workstation side, $e \in \mathbb{E}$.
k Disassembly factory index. $k \in \mathbb{K}$.
m Manufacturing factory index. $m \in \mathbb{M}$.
n Skills index, $n \in \mathbb{N}$.

Parameters:

v_{mpi} The m-th manufacturing factory acquires the price of the i-th subassembly of the p-th product.

c^T_{kmpi} Transportation cost of the i-th subassembly of the p-th product from the k-th disassembly factory to the m-th manufacturing factory.

t_{kwpj} Disassembly time required by workers at the w-th workstation of the kth disassembly factory to complete the j-th task of the p-th product.

c^d_{kpj} The unit time cost of executing the j-th task of the p-th product in the k-th factory.

c_k The unit time cost of activating the k-th disassembly factory.

c^L_{kw} Cost of activating the w-th linear workstation of the k-th disassembly factory.

c^U_{kw} Cost of activating the w-th U-shaped workstation of the k-th disassembly factory.

C Cost of worker skills training.

D Disassembly incidence matrix.

R Disassembly conflict matrix.

S Disassembly precedence matrix.

γ_{kwn} The matrix records the degree of mastery of the n-th skill by workers at the w-th workstation in the k-th factory in the initial state.

β_{pjn} The matrix depicts the disassembly relationship between the j-th task of the p-th product and the n-th skill.

θ_{pj} Fatigue growth parameter of workers completing the j-th task of the p-th product.

Decision variables:

$$z_{pk}=\begin{cases}1, & \text{If the } p\text{-th product is assigned to the } k\text{-th disassembly factory;}\\ 0, & \text{otherwise.}\end{cases}$$

$$x^L_{pjkw}=\begin{cases}1, & \text{If the } j\text{-th task of the } p\text{-th product is assigned for disassembly at the } w\text{-th linear workstation of the } k\text{-th disassembly factory;}\\ 0, & \text{otherwise.}\end{cases}$$

$$x^U_{pjkwe}=\begin{cases}1, & \text{If the } j\text{-th task of the } p\text{-th product is assigned to the } e\text{-side of the } w\text{-th U-shaped workstation at the } k\text{-th disassembly factory for disassembly;}\\ 0, & \text{otherwise.}\end{cases}$$

$$y^L_k=\begin{cases}1, & \text{If the linear disassembly line of the } k\text{-th disassembly factory is activated;}\\ 0, & \text{otherwise.}\end{cases}$$

$$y^U_k=\begin{cases}1, & \text{If the U-shaped disassembly line of the } k\text{-th disassembly factory is activated;}\\ 0, & \text{otherwise.}\end{cases}$$

$$u^L_{kw}=\begin{cases}1, & \text{If the } w\text{-th linear workstation of the } k\text{-th disassembly factory is activated;}\\ 0, & \text{otherwise.}\end{cases}$$

$$u^U_{kw}=\begin{cases}1, & \text{If the } w\text{-th U-shaped workstation of the } k\text{-th disassembly factory is activated;}\\ 0, & \text{otherwise.}\end{cases}$$

$$\alpha_{kmpi}=\begin{cases}1, & \text{If the } i\text{-th subassembly of the } p\text{-th product is transported from the } k\text{-th disassembly factory to the } m\text{-th manufacturing factory;}\\ 0, & \text{otherwise.}\end{cases}$$

$$\xi_{kwn}=\begin{cases}1, & \text{If workers at the } w\text{-th workstation of the } k\text{-th factory utilize the } n\text{-th skill;}\\ 0, & \text{otherwise.}\end{cases}$$

T_k, Cycle time of the k-th disassembly factory.

C. Mathematical Model

We formulate the optimization problem of MRPWF as:

$$\begin{aligned}\max f_1=&\Bigg(\sum_{k\in\mathbb{K}}\sum_{m\in\mathbb{M}}\sum_{p\in\mathbb{P}}\sum_{i\in\mathbb{I}_p}\left(v_{mpi}-c^T_{kmpi}\right)\alpha_{kmpi}\\&-\sum_{k\in\mathbb{K}}\sum_{p\in\mathbb{P}}\sum_{j\in\mathbb{J}_p}\sum_{w\in\mathbb{W}^L_k}c^d_{kpj}t_{kwpj}x^L_{pjkw}\\&-\sum_{k\in\mathbb{K}}\sum_{p\in\mathbb{P}}\sum_{j\in\mathbb{J}_p}\sum_{w\in\mathbb{W}^U_k}\sum_{e\in\mathbb{E}}c^d_{kpj}t_{kwpj}x^U_{pjkwe}\\&-\sum_{k\in\mathbb{K}}c_kT_k-\sum_{k\in\mathbb{K}}\sum_{w\in\mathbb{W}^L_k}c^L_{kw}u^L_{kw}-\sum_{k\in\mathbb{K}}\sum_{w\in\mathbb{W}^U_k}c^U_{kw}u^U_{kw}\\&-C\sum_{k\in\mathbb{K}}\sum_{w\in\mathbb{W}^k}\sum_{n\in\mathbb{N}}(\xi_{kwn}-\gamma_{kwn})\Bigg)\end{aligned}\tag{2}$$

$$\min f_2=\sum_{k\in\mathbb{K}}\sum_{w\in\mathbb{W}^L_k}\left(1-e^{-T^L_\alpha}\right)+\sum_{k\in\mathbb{K}}\sum_{w\in\mathbb{W}^U_k}\left(1-e^{-T^U_\alpha}\right)\tag{3}$$

The objective function (2) represents the maximum profit from disassembling the EOL product. The first term represents the total profit of the subassembly minus the transportation cost of the subassembly from the disassembly factory to the manufacturing factory. The second and third terms represent the disassembly cost of performing the disassembly task. The fourth term represents the cost of turning on the disassembly factory. The fifth and sixth terms represent the cost of turning on the associated workstations. The seventh term represents the training cost of the workers.

The objective function (3) represents the total fatigue level of workers. The first term represents the fatigue of workers on the linear disassembly line, and the second term represents the fatigue of workers on the U-shaped disassembly line.

$$T^L_\alpha=\sum_{p\in\mathbb{P}}\sum_{j\in\mathbb{J}_p}\theta_{pj}t_{kwpj}x^L_{pjkw}\tag{4}$$

$$T_{\alpha}^{U} = \sum_{p\in\mathbb{P}}\sum_{j\in\mathbb{J}_p}\sum_{e\in\mathbb{E}} \theta_{pj} t_{kwpj} x_{pjkwe}^{U} \tag{5}$$

As shown in equation 4, T_{α}^{L} represents the accumulated fatigue time for workers completing tasks on the linear disassembly line. As shown in equation 5, T_{α}^{U} represents the accumulated fatigue time for workers completing tasks on the U-shaped disassembly line.

$$\sum_{m\in\mathbb{M}} \alpha_{kmpi} \le \sum_{w\in\mathbb{W}_k^L}\sum_{j\in\mathbb{J}_p} d_{pij} x_{pjkw}^{L} + \sum_{w\in\mathbb{W}_k^U}\sum_{j\in\mathbb{J}_p}\sum_{e\in\mathbb{E}} d_{pij} x_{pjkwe}^{U} \tag{6}$$
$$\forall k\in\mathbb{K},\ \forall p\in\mathbb{P},\ \forall i\in\mathbb{I}_P\setminus\{1\}.$$

$$\sum_{k\in\mathbb{K}} z_{pk} = 1,\ \forall p\in\mathbb{P} \tag{7}$$

$$y_k^L + y_k^U \le 1,\ \forall k\in\mathbb{K} \tag{8}$$

$$z_{pk} \le y_k^L + y_k^U,\ \forall p\in\mathbb{P},\ \forall k\in\mathbb{K} \tag{9}$$

$$u_{kw}^{L} \le y_k^L,\ \forall w\in\mathbb{W}_k^S,\ \forall k\in\mathbb{K} \tag{10}$$

$$u_{kw}^{U} \le y_k^U,\ \forall w\in\mathbb{W}_k^U,\ \forall k\in\mathbb{K} \tag{11}$$

$$\sum_{w\in\mathbb{W}_k^L} x_{pjkw}^{L} + \sum_{w\in\mathbb{W}_k^U}\sum_{e\in\mathbb{E}} x_{pjkwe}^{U} \le z_{pk},\ \forall p\in\mathbb{P},\ \forall k\in\mathbb{K},\ \forall j\in\mathbb{J}_p \tag{12}$$

$$x_{pjkw}^{L} \le u_{kw}^{L},\ \forall p\in\mathbb{P},\ \forall j\in\mathbb{J}_p,\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}_k^L \tag{13}$$

$$x_{pjkwe}^{U} \le u_{kw}^{U},\ \forall p\in\mathbb{P},\ \forall j\in\mathbb{J}_p,\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}_k^U,\ \forall e\in\mathbb{E} \tag{14}$$

$$x_{pjkwe}^{U} \le \sum_{n\in\mathbb{N}} \beta_{pjn}\xi_{kwn},\ \forall p\in\mathbb{P},\ \forall j\in\mathbb{J}_p,\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}_k^U,\ \forall e\in\mathbb{E} \tag{15}$$

$$x_{pjkw}^{L} \le \sum_{n\in\mathbb{N}} \beta_{pjn}\xi_{kwn},\ \forall p\in\mathbb{P},\ \forall j\in\mathbb{J}_p,\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}_k^L \tag{16}$$

$$\xi_{kwn} \ge \gamma_{kwn},\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W},\ \forall n\in\mathbb{N} \tag{17}$$

$$\sum_{k\in\mathbb{K}}\left(\sum_{w\in\mathbb{W}_k^L} x_{pjkw}^{L} + \sum_{w\in\mathbb{W}_k^U}\sum_{e\in\mathbb{E}} x_{pjkwe}^{U}\right) \le 1,\ \forall p\in\mathbb{P},\ \forall j\in\mathbb{J}_p \tag{18}$$

$$\sum_{j\in\mathbb{J}_p} t_{kwpj} x_{pjkw}^{L} \le T_k,\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}_k^L \tag{19}$$

$$\sum_{j\in\mathbb{J}_p}\sum_{e\in\mathbb{E}} t_{kwpj} x_{pjkwe}^{U} \le T_k,\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}_k^U \tag{20}$$

$$\sum_{w\in\mathbb{W}_k^L} w\left(x_{pj_1kw}^{L} - x_{pj_2kw}^{L}\right) + W_k^L\left(\sum_{w\in\mathbb{W}_k^L} x_{pj_2kw}^{L} - 1\right) \le 0 \tag{21}$$
$$\forall k\in\mathbb{K},\ \forall p\in\mathbb{P},\ \forall j_1,j_2\in\mathbb{J}_p,\ s_{pj_1j_2}=1$$

$$\sum_{w\in\mathbb{W}_k^U}\left(w\left(x_{pj_1kw1}^{U} - x_{pj_2kw1}^{U}\right) + \left(2W_k^U - w\right)\left(x_{pj_1kw2}^{U} - x_{pj_2kw2}^{U}\right)\right)$$
$$+ 2W_k^U\left(\sum_{w\in\mathbb{W}_k^U}\sum_{e\in\mathbb{E}} x_{pj_2kwe}^{U} - 1\right) \le 0,\ \forall k\in\mathbb{K},\ \forall p\in\mathbb{P},$$
$$\forall j_1,j_2\in\mathbb{J}_p,\ s_{pj_1j_2}=1 \tag{22}$$

$$\sum_{w\in\mathbb{W}_k^L} x_{pj_2kw}^{L} \le \sum_{j_1\in\mathbb{J}_p}\sum_{w\in\mathbb{W}_k^L} s_{pj_1j_2} x_{pj_1kw}^{L},\ \forall k\in\mathbb{K},\ \forall p\in\mathbb{P}, \tag{23}$$
$$\forall j_2\in\mathbb{J}_p,\ d_{p1j_2}=0$$

$$\sum_{w\in\mathbb{W}_k^U}\sum_{e\in\mathbb{E}} x_{pj_2kwe}^{U} \le \sum_{j_1\in\mathbb{J}_p}\sum_{w\in\mathbb{W}_k^U}\sum_{e\in\mathbb{E}} s_{pj_1j_2} x_{pj_1kwe}^{U},\ \forall k\in\mathbb{K}, \tag{24}$$
$$\forall p\in\mathbb{P},\ \forall j_2\in\mathbb{J}_p,\ d_{p1j_2}=0$$

$$\sum_{w\in\mathbb{W}_k^L}\left(x_{pj_1kw}^{L} + x_{pj_2kw}^{L}\right) \le 1,\ \forall k\in\mathbb{K},\ \forall p\in\mathbb{P}, \tag{25}$$
$$\forall j_1,j_2\in\mathbb{J}_p,\ r_{pj_1j_2}=1$$

$$\sum_{w\in\mathbb{W}_k^U}\sum_{e\in\mathbb{E}}\left(x_{pj_1kwe}^{U} + x_{pj_2kwe}^{U}\right) \le 1,\ \forall k\in\mathbb{K},\ \forall p\in\mathbb{P}, \tag{26}$$
$$\forall j_1,j_2\in\mathbb{J}_p,\ r_{pj_1j_2}=1$$

$$z_{pk}\in\{0,1\},\ \forall p\in\mathbb{P},\ \forall k\in\mathbb{K} \tag{27}$$

$$x_{pj_1kw}^{L}\in\{0,1\},\ \forall p\in\mathbb{P},\ \forall j\in\mathbb{J}_p,\ \forall w\in\mathbb{W}_k^L,\ \forall k\in\mathbb{K} \tag{28}$$

$$x_{pj_1kwe}^{U}\in\{0,1\},\forall p\in\mathbb{P},\forall j\in\mathbb{J}_p,\ \forall w\in\mathbb{W}_k^U,\ \forall k\in\mathbb{K}, \tag{29}$$
$$\forall e\in\mathbb{E}$$

$$y_k^L\in\{0,1\},\ \forall k\in\mathbb{K} \tag{30}$$

$$y_k^U\in\{0,1\},\ \forall k\in\mathbb{K} \tag{31}$$

$$u_{kw}^{L}\in\{0,1\},\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}_k^L \tag{32}$$

$$u_{kw}^{U}\in\{0,1\},\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}_k^U \tag{33}$$

$$\xi_{kwn}\in\{0,1\},\ \forall k\in\mathbb{K},\ \forall w\in\mathbb{W}^k,\ \forall n\in\mathbb{N} \tag{34}$$

$$T_k\in\mathbb{R}_+,\ \forall k\in\mathbb{K} \tag{35}$$

Constraint (6) ensures that subassemblies obtained by disassembling a product can be transported to only one manufacturing factory. Constraint (7) ensures that each product can only be assigned to one disassembly factory. Constraint (8)

ensures that only one type of disassembly line can be turned on at a disassembly factory. Constraint (9) ensures that products are only assigned to the disassembly factory that is currently open. Constraints (10) and (11) ensure that the corresponding workstations are only used after the disassembly line has been activated at the disassembly factory. Constraint (12) ensures that the disassembly task j for product p can only be assigned to the workstation activated by the disassembly factory to which the product assigned. Constraints (13) and (14) ensure that the disassembly task j of product p is assigned to the open workstation. Constraints (15) and (16) ensure that the skills of the workers at the workstation meet the skill constraints required for disassembly task j. Constraints (17) ensure that the skills acquired by all workers are not reduced. Constraint (18) indicates that each disassembly task is performed at most once per product. Constraints (19) and (20) ensure that the working hours of each workstation in each line do not exceed the factory cycle time. Constraint (21) ensures that assigning disassembly tasks for products to a linear disassembly line conforms to the precedence relationship constraint.

Constraint (22) ensures that the assignment of disassembly tasks of a product to a U-shaped disassembly line conforms to the precedence relationship constraint. The first item ensures that the workstation count of task j_2 is greater than or equal to the workstation count of j_1 when the disassembly task is assigned to the inlet side of a U-shaped disassembly line. The second item ensures that the workstation count of task j_2 is less than or equal to the workstation count of j_1 when the disassembly task is assigned to the outlet side of a U-shaped disassembly line. The third item ensures that task j_2 is executed after disassembly task j_1 is executed.

Constraints (23) and (24) ensure that the disassembly sequence of the product can begin from other tasks. Constraints (25) and (26) ensure that the assignment of disassembly tasks causes no conflict among them. Constraints (27)-(35) indicate the range of values of decision variables.

III. Proposed Algorithm

A. Discrete Battle Royale optimizer (DBRO)

BRO is a population-based metaheuristic optimization algorithm proposed by Rahkar-Farshi. The fundamental concept of BRO draws inspiration from the game PUBG. Exploration is a crucial component in PUBG, as players need to search for tools to aid survival while avoiding being eliminated by opponents. In PUBG, a popular game mode is called "deathmatch," which aims to eliminate as many other players as possible until a specified kill count or time limit is reached. Typically, the battles in PUBG take place on specific maps chosen by players [44], [45]. The game maps are treated as optimization problem spaces within BRO. A game begins with players parachuting from an airplane onto the map. Like many other population-based optimization algorithms, the search agents in BRO are randomly initialized within the search space by using uniform random initialization. During the game, if other soldiers eliminate a player, they respawn in a randomly chosen battlefield area. The ultimate winner is the player with the most kills. In BRO, soldiers and players both represent individuals.

Algorithm 1 Discrete Battle Royale Optimization Algorithm

Input: maximum iterations Max, population size n
Output: the best solution X

```
Initialize the soldier population
x_dam = the damage level of each soldier.
r_dam = combat rate
Calculate the fitness of each soldier
while (iter<Max) do
  for each soldier do
    Execute the search process to reach a new location
    Generate random probability r
    if r>r_dam then
      Execute combat process
      x_dam=x_dam+1
    else
      Soldiers continue the search process to reach a new
      location
    end if
    if x_dam>Threshold then
      execute soldier rebirth process
    end if
  end for
  Update X and record the optimal objective value
  Decide whether to initialize the population
  t=t+1
end while
return X
```

BRO is employed for solving continuous optimization problems. However, MRPWF represents a discrete optimization issue. We propose DBRO to solve this problem because it exhibits fast convergence speed and strong global optimization capability. In DBRO, the initial population is randomly distributed throughout the problem space. DBRO introduces a combat rate, where the algorithm generates a random number in each iteration. If the generated random number is greater than the combat rate, the soldier engages in combat after encountering another soldier and gets injured. Conversely, if the random number is lower than the combat rate, the soldier successfully avoids combat and continues searching. Each soldier randomly moves to explore advantageous positions and then attempts to inflict damage on his nearest soldiers. Each soldier's damage level has an initial value of zero. Upon sustaining damage, the soldier's damage level increments and immediately shifts his current position to search for an advantageous location. If the injured soldier can inflict damage on other soldiers in the next iteration, his damage level resets to zero. Once a soldier's damage level surpasses the predetermined threshold, the soldier dies. Then, the soldier is randomly respawned from the feasible problem space and resets his damage level to zero. This process is iterated until an acceptable solution or the maximum iteration count is reached.

We introduce a mutation strategy to prevent DBRO from getting trapped in local optima. Specifically, if the optimal objective value of the population hasn't improved after a certain number of iterations, the population is reinitialized. DBRO is realized by Algorithm 1.

B. Encoding and Decoding

Establishing a disassembly product information model is the first step in researching MRPWF. Common models for disassembly information include AND/OR graphs [46], Petri nets [47], [48], priority graphs [49], and others. The product in this paper adopts the Disassembly AND/OR Graph (DAOG) to represent the precedence relationships among disassembly tasks. Taking a simplified forklift [50] as an example, its DAOG is shown in Fig. 4, where the indices of the subassemblies are denoted by integers within angle brackets, and each disassembly task is represented by a directed edge linking the subassemblies. It is noticeable that this product consists of 19 subassemblies and 18 disassembly tasks.

Effective encoding strategies are crucial for MRPWF. They simplify the problem-solving process, optimize resource utilization, and enhance production efficiency, thus improving the sustainability and competitiveness of the remanufacturing process. Based on the characteristics of MRPWF, we expect the generated solution to correspond accurately to an allocation plan during the actual allocation process. In this problem, the selection of manufacturing factories, disassembly factories along with their internal disassembly lines, the allocation of workstations, and the sequence of disassembly tasks performed on the workstations all affect the quality of the solution. In order to solve the researched problem, we design a five-stage encoding strategy $\sigma(\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ to represent the solutions of MRPWF. σ_1 represents the sequence of disassembly tasks. $\sigma_2 = (a, b)$ signifies the relationship between tasks and disassembly factories, where a signifies a task in σ_1, and b represents a disassembly factory. $\sigma_3 = (b, c)$ denotes the connection between disassembly factories and disassembly lines, where b stands for a disassembly factory, and c represents the type of disassembly line. Here, the value of c being 1 indicates a linear disassembly line, and 2 signifies a U-shaped disassembly line. $\sigma_4 = (a, d)$ indicates the linkage between tasks and workstations, where a corresponds to a task in σ_1, and d denotes a workstation. $\sigma_5 = (e, f)$ represents the relationship between subassemblies and manufacturing factories, where e denotes subassembly and f signifies manufacturing factory.

Taking forklift truck as products, the encoding scheme is shown in Fig. 5. The disassembly task sequence for product 1 is 1, 3, 6, 11, and 16, while for product 2, it is 2, 5, 9, 14. Product 1 is assigned to disassembly factory 1, while product 2 is assigned to disassembly factory 2. Disassembly factory 1 employs a U-shaped disassembly line, while disassembly factory 2 operates a linear disassembly line. Disassembly task 1 for product 1 is assigned to workstation 1 on the U-shaped disassembly line, while tasks 3, 6, and 16 are assigned to workstation 2. Task 11 is assigned to workstation 3. For product 2, tasks 2 and 5 are assigned to workstation 1 on the linear disassembly line, and tasks 9 and 14 are assigned to workstation 3. The subassemblies 3, 13, 15, and 18 of product 1 are assigned to manufacturing factory 1, while subassemblies 4 and 6 are assigned to manufacturing factory 2. For product 2, subassemblies 13 and 19 are assigned to manufacturing factory 1, while subassemblies 4, 6, and 10 are assigned to manufacturing factory 2.

The algorithm first generates a disassembly sequence based on the conflicts and precedence relationships among disassembly tasks during the decoding process. Each task in the disassembly sequence is in a pending allocation state. The following steps are then carried out.

- **Step 1:** Assign the EOL products to the disassembly factories.
- **Step 2:** Select the disassembly lines that factories activate for EOL products.
- **Step 3:** Assign disassembly tasks to workstations in order. If a disassembly task is assigned to a workstation and exceeds its cycle constraint, open the next workstation and assign the task to it for execution.
- **Step 4:** Generate subassembly sequences based on disassembly and select manufacturing factories for each subassembly.

Taking the forklift as an example, the specific assignment process is shown in Fig. 7.

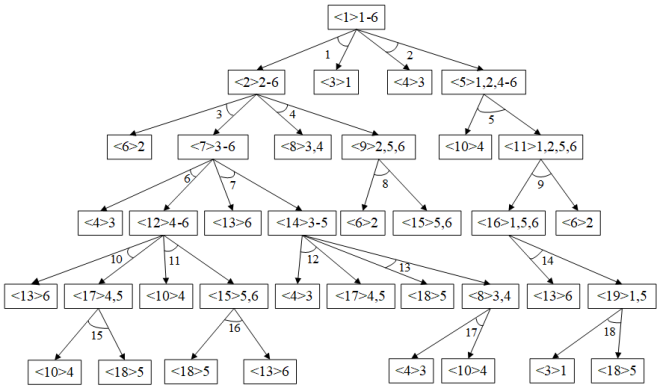

Fig. 4. The DAOG of a forklift truck.

Task sequence (σ_1)	1	3	6	11	16	2	5	9	14	
Disassembly factory	1	1	1	1	1	2	2	2	2	
Disassembly line	2	2	2	2	2	1	1	1	1	
Workstation	1	2	2	3	2	1	1	3	3	
Subassembly sequence	3	4	6	13	15	18	4	6	10	13	19
Manufacturing factory	1	2	2	1	1	1	2	2	2	1	1

Fig. 5. Example of encoding.

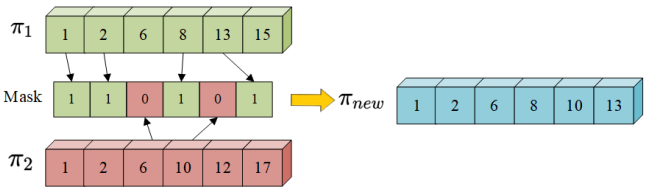

Fig. 6. Process of combat.

C. Population Initialize

A population typically consists of multiple soldiers. During the population initialization, random-length disassembly task sequences are generated based on the disassembly incidence matrix. Then, the generated task sequences are transformed into feasible sequences by considering conflicts and priority relationships among the disassembly tasks. Each task in the disassembly sequence remains unallocated at this stage. Subsequently, these task sequences are allocated to various factories for disassembly. The initialization process is illustrated in Algorithm 2.

Algorithm 2 Initialize population

Input: population size n
Output: population P
while $(i<n)$ **do**
  Generate random task sequence σ_1
  Adjust σ_1 to resolve conflict and precedence constraint
  Apply the allocation strategy to generate σ_2, σ_3, σ_4
  Add $\sigma(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$ to P
  $i = i + 1$
end while
return P

D. Combat and Search

Search and combat are critical operations for soldiers to achieve victory. During a search process, r_{dam} determines whether combat occurs. A random number is generated, and if it is greater than r_{dam}, the soldier engages in combat during the search process. Otherwise, the soldier successfully evades the combat and continues the search if the generated random number is less than r_{dam}. The combat process is shown in Fig. 6.

- **Step 1:** Obtain the disassembly task sequences π_1 and π_2 for two soldiers.
- **Step 2:** Generate a new soldier π_{new}.
- **Step 3:** Randomly generate a set of binary masks with the same length as the disassembly sequences. Then, from left to right, examine the values of the masks. A value of 0 indicates that the disassembly task is taken from π_1 and placed into π_{new}, while a value of 1 indicates that the task is taken from π_2 and placed into π_{new}. If the obtained task already exists in π_{new}, we need to skip the current task and retrieve the next task from π_1 and π_2.
- **Step 4:** The newly generated π_{new} may not be a feasible solution, so it needs to be adjusted to satisfy conflict and priority constraints.

Algorithm 3 Search process

Input: $\sigma(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$
Output: $\sigma'(\sigma_1', \sigma_2', \sigma_3', \sigma_4')$
Generate random probability m
if $m<0.2$ **then**
  Execute the sequential variation
else if $m<0.5$ **then**
  Execute the task variation
else if $m<0.9$ **then**
  Execute the workstation variation
else
  Execute the factory swap
end if
return σ'

For soldier search, we design four actions to better search for the optimal solution, namely sequential variation, task variation, workstation variation, and factory swap. The soldier search process is shown in Algorithm 3.

Sequential variation: As shown in Fig. 8, task 10 is randomly selected. Based on the priority relationships between tasks, we identify that task 10's preceding task is task 2, and its subsequent task is task 17. Therefore, we can insert task 10 at any position between task 2 and task 17.

Task variation: As shown in Fig. 9, task 7 is a randomly

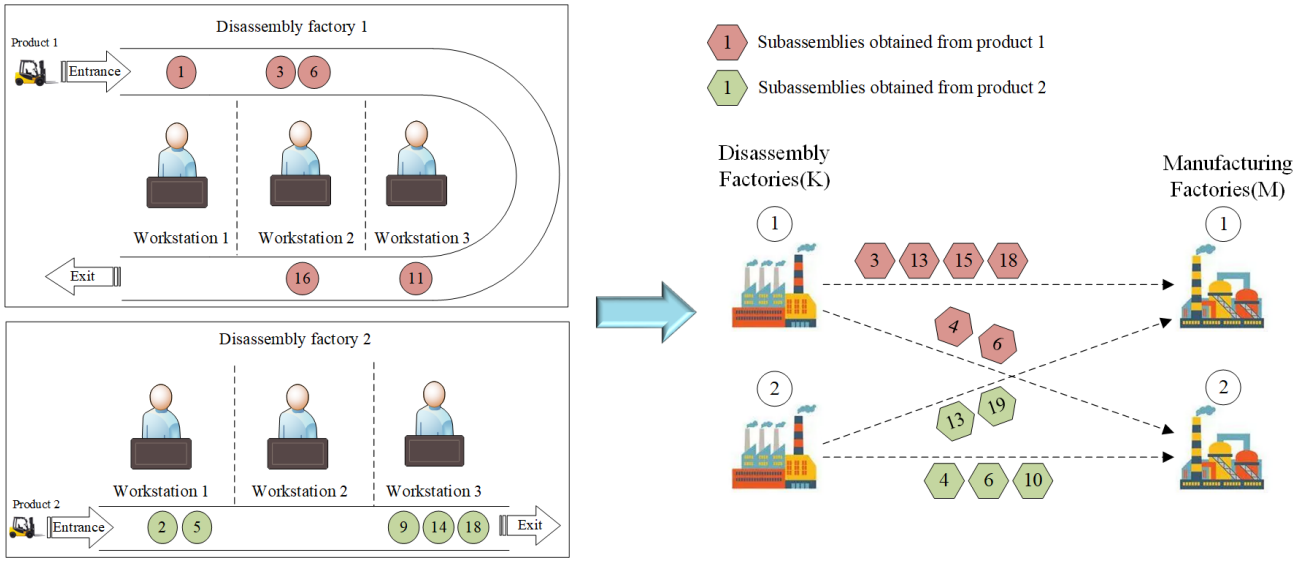

Fig. 7. Example of assignment.

selected task, and we remove it along with its subsequent tasks. Then, based on Fig. 4, we identify the subassemblies obtained from completing task 7 and select for an alternative disassembly strategy for these subassemblies, resulting in tasks 6, 12, and 21. Finally, we incorporate these disassembly tasks into the original disassembly sequence.

Workstation variation: Randomly select a workstation, then choose either the first or the last task of that workstation. If the first task is selected, it is assigned to the previous workstation. If the last task is selected, it is assigned to the next workstation. As shown in Fig. 10, workstation 3 is the randomly chosen workstation and task 7 from that workstation is assigned to workstation 2.

Factory swap: As shown in Fig. 11, we randomly select the disassembly factory to which two products belong and swap them.

E. Rebirth process

In DBRO, if a soldier's damage level exceeds a pre-set threshold, the soldier dies and respawns randomly within the feasible problem space, with their damage level reset to zero. After respawning, the newly generated soldiers are combined with the previous soldiers to form a new population. The population is then re-ranked based on an individual's fitness and proceeds to the next iteration.

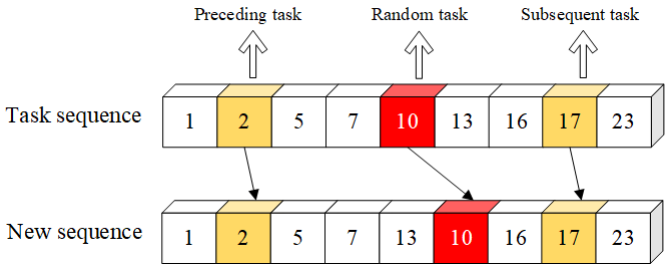

Fig. 8. Process of sequential variation.

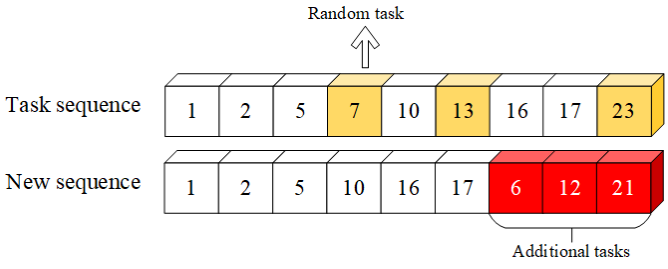

Fig. 9. Process of task variation.

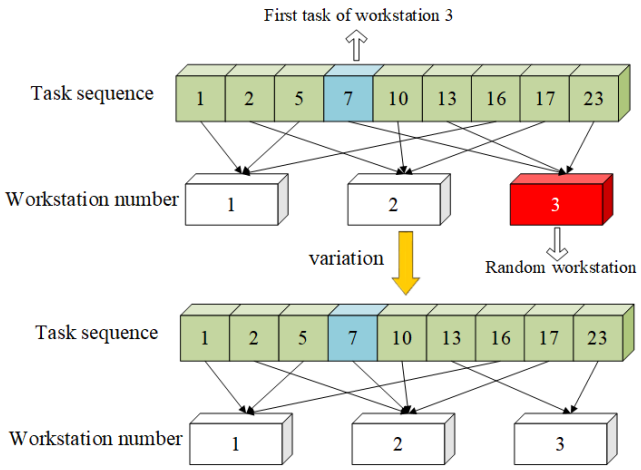

Fig. 10. Process of workstation variation.

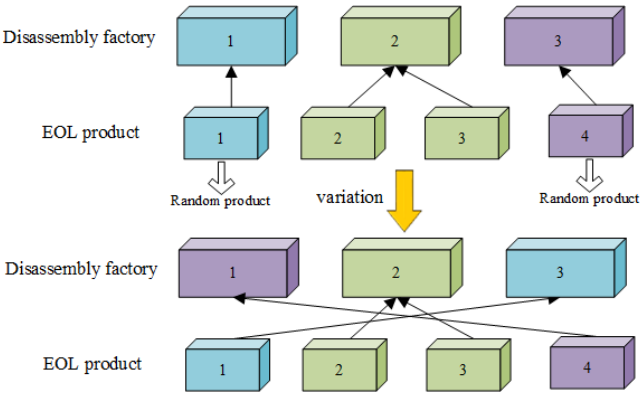

Fig. 11. Process of factory swap.

IV. Experimental Studies

A. Experimental Cases and Parameter Settings

To verify the model's correctness and assess the proposed algorithm's efficiency, we employ the IBM CPLEX optimizer to solve the experimental cases and obtain standard optimal solutions. Simultaneously, we employ DBRO to solve the same-sized instances and compare their experimental outcomes. The computations are conducted on a computer equipped with an Intel(R) Core(TM) i5-8300H (2.30GHz/16.00GB RAM) processor.

To ensure the comprehensiveness of the experimental study, in terms of the case study, we select four distinct product types: forklift truck [50], rigid caster [51], washing machine [52], and radio [53]. These four product types are combined to create various multi-product cases for testing. Table II presents the scale information for the combined cases. Table III outlines the parameter settings for the disassembly factories. Table IV provides detailed information about these products. Taking the forklift truck as an example, it consists of 6 components, involves 18 disassembly tasks, and results in 19 subassemblies upon disassembly. The profit, transportation cost, unit-time disassembly cost, disassembly time, and fatigue growth parameter of subassemblies are all assumed to follow a normal distribution.

B. Performance Indicators

To compare the effectiveness of DBRO with five different algorithms, we use four performance indicators: hypervolume [54], epsilon [55], and inverted generational distance plus (IGD+) [56] to compare the results of different algorithms. The meanings of each indicator are as follows:

- **Hypervolume-indicator:** It evaluates the performance of a solution set by measuring the volume of non-dominated space occupied by the solution set. A larger value indicates better performance of the solution set.
- **Epsilon-indicator:** It measures the distance between the approximate Pareto front and the true Pareto front. A smaller value indicates better overall algorithm performance.
- **IGD+-indicator:** It measures how closely a solution set approximates the true Pareto front, considering the distribution of solutions within the solution set. It provides a more comprehensive assessment compared to the traditional IGD indicator. A smaller value indicates that the solution set is closer to the true Pareto front, indicating a better solution set.

For each case, we perform 20 runs of DBRO and its peer algorithms, calculating the average values of the indicators above. Furthermore, we analyze the experimental results using a t-test [46] with a confidence level of 0.05 and 38 degrees of freedom. In the following experimental results, "+" indicates that DBRO is significantly superior to the compared algorithms, "-" indicates that DBRO is significantly inferior to the compared algorithms, and "∼" signifies that DBRO is equivalent to the compared algorithms.

C. Verification of Algorithm With Different Scales of Cases

To validate the superiority of DBRO, we select five algorithms for experimental case testing: DBRO, CPA, MBO, DOA, and FOA. We use four performance indicators to analyze the experimental results. The population sizes for the cases are set at 120, 180, and 240, respectively. We ensure that all algorithms are tested on the same computing platform to avoid biases caused by hardware or platform differences. Additionally, we use the same initial parameters, population size, and maximum number of iterations for each algorithm and case.

TABLE II Case information.

Case ID	Product				Num. of tasks
	Forklift truck	Rigid caster	Washing machine	Radio	
1	1	0	0	0	18
2	0	1	0	0	32
3	1	1	1	0	63
4	1	1	1	1	93
5	2	0	1	2	109
6	2	1	2	2	154

TABLE III Disassembly factory parameter set.

Factory ID	c_k	c_{kw}^L	c_{kw}^U
1	3	6 ∼ 9	7 ∼ 11
2	5	4 ∼ 9	6 ∼ 11
3	4	4 ∼ 7	6 ∼ 12
4	4	4 ∼ 7	8 ∼ 10

Table V provides the experimental results for the five algorithms concerning the hypervolume-indicator. For case 1, when the population size is 180, DBRO outperforms CPA, DOA, and FOA regarding the hypervolume-indicator, which is equivalent to MBO. The experimental results indicate that the solution set generated by DBRO is more diverse, better covering the Pareto front, and contains a greater number of high-quality solutions.

TableVI provides the experimental results for the five algorithms regarding the IGD+-indicator. For case 5, when the population size is 120, DBRO outperforms the other four algorithms regarding the IGD+-indicator. The experimental results indicate that the solution set generated by DBRO is closer to the true Pareto Front, demonstrating good distribution and convergence characteristics.

TableVII provides the experimental results for the five algorithms regarding the epsilon-indicator. For case 4, when the population size is 180, DBRO outperforms the other four algorithms regarding the epsilon-indicator. The experimental results indicate that the solution set generated by DBRO is closer to the true Pareto Front, demonstrating higher quality, diversity, and convergence.

Based on the experimental results shown in Tables IV-VI, we conclude that while DBRO may not outperform other algorithms in all cases, it provides competitive solutions in most situations. Furthermore, as the case scale increases, the performance of DBRO gradually improves, indicating its high scalability. However, it's important to note that due to the inherent randomness of the algorithm, its performance may be constrained when dealing with small-scale cases. As large-scale cases are more common in MRPWF, this makes DBRO more practical. Fig. 12-14 depict the Pareto fronts for case 1, case 3, and case 6, with a population size of 240. By observing the Pareto front, we can see that DBRO outperforms other algorithms significantly. Specifically, DBRO excels in pursuing the objectives of maximizing profit and minimizing fatigue. It not only ensures high profits but also effectively reduces fatigue, demonstrating outstanding performance. Furthermore, taking Case 4 as an example, when the population size is 120, Fig. 15 compares the running times of the five algorithms. It can be observed that DBRO has a faster running time compared to MBO and FOA. In conclusion, DBRO is a better approach for solving MRPWF.

V. Conclusion

The optimization of the multi-factory remanufacturing process is of significant importance for improving resource utilization efficiency, reducing environmental impact, and enhancing supply chain collaboration. In this work, building upon MRPOP, we consider the influence of multi-skilled workers and worker fatigue and, for the first time, introduce and address MRPWF. Additionally, we establish a mixed-integer programming model to maximize profit and minimize fatigue index to formulate this problem. To tackle this problem, we propose DBRO, in which we devise a novel encoding structure

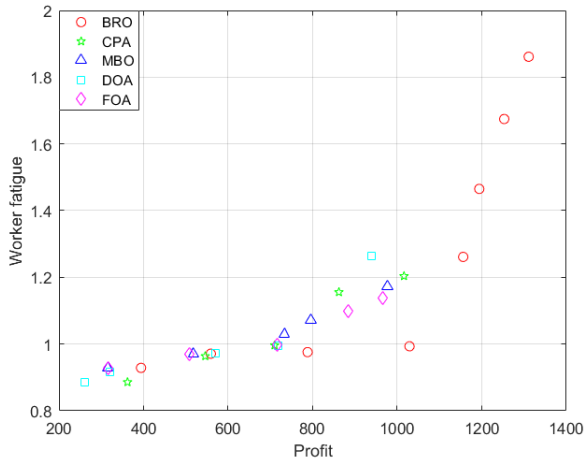

Fig. 12. The Pareto front of the Case 1.

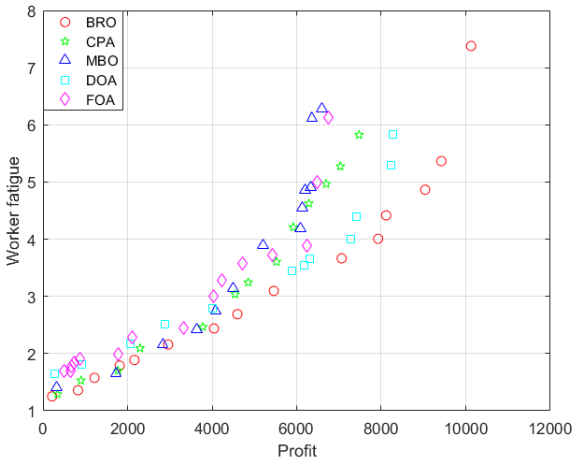

Fig. 14. The Pareto front of the Case 6.

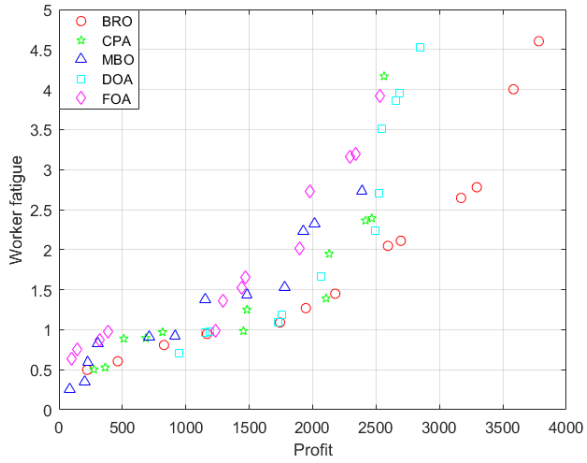

Fig. 13. The Pareto front of the Case 3.

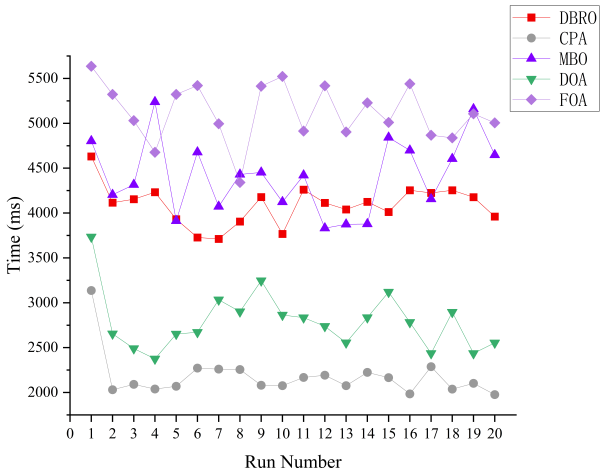

Fig. 15. CPU runtime for the five algorithms in Case 4.

to represent solutions. Additionally, four soldier search methods are designed to more effectively search for the optimal solution to prevent falling into local optima. To validate the effectiveness and superiority of the algorithm, we compare the experimental results of DBRO with those of four other intelligent optimization algorithms. Using the hypervolume-indicator, epsilon-indicator, and IGD+-indicator to analyze the experimental results, we demonstrate that DBRO offers higher efficiency in solving this problem.

Our future plans include:

1) Incorporating more real-world factors, such as worker disassembly postures, etc.

2) Improving DBRO and designing better algorithms to handle complex optimization problems.

3) We will also explore other intelligent optimization algorithms or reinforcement learning and apply them to solving similar problems.

TABLE IV Product parameter set.

Product	Num. of parts	Num. of task	Num of subassembly	v_{mpi}	c^T_{kmpi}	c^d_{kpj}	t_{kwpj}	θ_{pj}
Forklift truck	6	18	19	N(300, 6)	N(20, 3)	N(7, 2)	N(7, 2)	N(0.15, 0.05)
Rigid caster	9	32	25	N(100, 5)	N(10, 2)	N(5, 1)	N(4, 1)	N(0.1, 0.03)
Washing machine	6	13	15	N(230, 4)	N(15, 2)	N(6, 1)	N(5, 1)	N(0.12, 0.04)
Radio	10	30	29	N(150, 4)	N(10, 3)	N(6, 1)	N(4, 1)	N(0.1, 0.03)

TABLE V Comparison of five algorithms via hypervolume-indicator

Case ID	P	DBRO		CPA			MBO			DOA			FOA		
		mean	variance	mean	variance	t-test	mean	variance	t-test	mean	variance	t-test	mean	variance	t-test
1	120	0.3236	0.0233	0.1991	0.0322	+	0.2476	0.0241	~	0.1569	0.0251	+	0.2603	0.0223	~
	180	0.3087	0.0271	0.2006	0.0474	+	0.2483	0.0376	~	0.1091	0.0141	+	0.2113	0.0127	+
	240	**0.1466**	**0.0006**	0.1110	0.0045	**+**	0.1268	0.0011	**+**	0.1075	0.0014	**+**	0.1234	0.0002	**+**
2	120	0.1887	0.0060	0.0499	0.0049	+	0.0469	0.0063	+	0.0944	0.0083	+	0.0043	0.0001	+
	180	0.1185	0.0071	0.0361	0.0129	+	0.0036	0.0001	+	0.0941	0.0048	~	0.0033	0.0002	+
	240	0.0164	0.0016	0.0023	0.0001	~	0.0023	0.0001	~	0.0093	0.0008	~	0.0030	0.0001	~
3	120	0.1302	0.0187	0.0424	0.0027	+	0.0629	0.0034	+	0.0791	0.0072	~	0.0619	0.0116	+
	180	0.3221	0.0122	0.2118	0.0079	+	0.2560	0.0174	+	0.2267	0.0049	+	0.1981	0.0089	+
	240	0.3448	0.0066	0.3337	0.0126	~	0.2505	0.0115	+	0.2851	0.0066	+	0.2017	0.0072	+
4	120	0.1167	0.0097	0.0707	0.0122	~	0.0574	0.0075	+	0.0341	0.0035	+	0.0835	0.0119	~
	180	0.4797	0.0134	0.4433	0.0120	~	0.4163	0.0119	+	0.3649	0.0119	+	0.4404	0.0065	~
	240	0.4536	0.0091	0.4414	0.0153	~	0.4106	0.0119	~	0.3856	0.0108	+	0.3795	0.0108	+
5	120	0.3088	0.0119	0.2311	0.0121	+	0.2345	0.0175	+	0.2229	0.0212	+	0.2244	0.0157	+
	180	0.3339	0.0258	0.2288	0.0068	+	0.2667	0.0232	~	0.2452	0.0152	+	0.3313	0.0175	~
	240	0.1428	0.0217	0.1079	0.0215	~	0.0821	0.0082	~	0.0764	0.0195	~	0.0601	0.0068	+
6	120	0.4203	0.0121	0.3627	0.0137	~	0.3836	0.0215	~	0.3301	0.0316	+	0.3206	0.0117	+
	180	0.3755	0.0109	0.3057	0.0179	+	0.3266	0.0328	~	0.3874	0.0228	~	0.3310	0.0161	~
	240	0.3236	0.0217	0.2490	0.0125	~	0.3284	0.0233	~	0.3230	0.0355	~	0.2668	0.0164	~

TABLE VI Comparison of five algorithms via IGD+-indicator

Case ID	P	DBRO		CPA			MBO			DOA			FOA		
		mean	variance	mean	variance	t-test	mean	variance	t-test	mean	variance	t-test	mean	variance	t-test
1	120	0.4229	0.0172	0.6545	0.1567	+	0.5705	0.1084	+	0.5400	0.0173	+	0.5470	0.1033	~
	180	0.4424	0.0214	0.9095	0.4225	+	0.6206	0.1919	+	0.6917	0.1206	+	0.6291	0.1450	+
	240	0.3124	0.0063	0.4334	0.0368	+	0.4628	0.0027	+	0.4272	0.0295	+	0.3946	0.0049	+
2	120	0.4286	0.0117	0.7275	0.0668	+	0.8777	0.1180	+	0.6643	0.1002	+	0.9758	0.1073	+
	180	0.4313	0.0063	1.1770	0.1711	+	1.4123	0.1563	+	0.6320	0.1236	+	1.2132	0.1397	+
	240	0.4379	0.0148	0.8236	0.0377	+	0.9238	0.0277	+	0.4878	0.0204	~	0.8293	0.0235	+
3	120	0.3894	0.0271	0.6213	0.0147	+	0.5566	0.0133	+	0.5145	0.0283	+	0.6206	0.0286	+
	180	0.2497	0.0094	0.3972	0.0121	+	0.3416	0.0132	+	0.3787	0.0057	+	0.4004	0.0113	+
	240	0.2319	0.0038	0.2580	0.0094	~	0.3121	0.0088	+	0.2899	0.0041	+	0.3509	0.0068	+
4	120	0.5296	0.0298	0.7504	0.1180	+	0.7531	0.1369	+	0.9077	0.2390	+	0.7011	0.0976	+
	180	0.2043	0.0057	0.2376	0.0062	~	0.2522	0.0088	+	0.3222	0.0056	+	0.2191	0.0038	~
	240	0.2402	0.0042	0.2882	0.0091	+	0.3115	0.0080	+	0.3496	0.0069	+	0.3336	0.0071	+
5	120	**0.2818**	**0.0042**	0.3891	0.0098	**+**	0.3786	0.0105	**+**	0.3908	0.0158	**+**	0.3690	0.0103	**+**
	180	0.2978	0.0135	0.3804	0.0049	+	0.3668	0.0183	+	0.4238	0.0104	+	0.3119	0.0117	~
	240	0.5393	0.0545	0.6462	0.0798	~	0.5910	0.0487	~	0.7909	0.1290	+	0.7024	0.7024	+
6	120	0.2890	0.0062	0.3477	0.0114	+	0.3198	0.0112	~	0.3638	0.0195	+	0.3736	0.0094	+
	180	0.2798	0.0053	0.3473	0.0136	+	0.3355	0.0275	~	0.2786	0.0145	~	0.3153	0.0088	~
	240	0.3764	0.0241	0.4467	0.0110	~	0.3685	0.0218	~	0.4028	0.0343	~	0.4464	0.0180	~

REFERENCES

[1] X. Guo, Z. Bi, J. Wang, S. Qin, S. Liu, and L. Qi, "Reinforcement learning for disassembly system optimization problems: A survey," *International Journal of Network Dynamics and Intelligence*, vol. 2, no. 1, pp. 1–14, 2023.

[2] S. Qin, J. Li, J. Wang, X. Guo, S. Liu, and L. Qi, "A salp swarm algorithm for parallel disassembly line balancing considering workers with government benefits," *IEEE Transactions on Computational Social Systems*, pp. 1–10, 2023.

[3] X. Guo, T. Wei, J. Wang, S. Liu, S. Qin, and L. Qi, "Multiobjective u-shaped disassembly line balancing problem considering human fatigue index and an efficient solution," *IEEE Transactions on Computational Social Systems*, vol. 10, no. 4, pp. 2061–2073, 2023.

[4] X. Guo, M. Zhou, S. Liu, and L. Qi, "Multiresource-constrained selective disassembly with maximal profit and minimal energy consumption," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 2, pp. 804–816, 2020.

[5] ——, "Lexicographic multiobjective scatter search for the optimization of sequence-dependent selective disassembly subject to multiresource constraints," *IEEE Transactions on Cybernetics*, vol. 50, no. 7, pp. 3307–3317, 2019.

[6] L. Qi, M. Li, X. Guo, and W. Luan, "Multi-objective optimization for robotaxi dispatch with safety-carpooling mode in pandemic era," *IEEE Transactions on Intelligent Transportation Systems*, vol. 26, no. 1, pp. 878–891, 2024.

[7] S. Qin, S. Zhang, J. Wang, S. Liu, X. Guo, and L. Qi, "Multi-objective multi-verse optimizer for multi-robotic u-shaped disassembly

TABLE VII Comparison of five algorithms via epsilon-indicator

Case ID	P	DBRO		CPA			MBO			DOA			FOA		
		mean	variance	mean	variance	t-test	mean	variance	t-test	mean	variance	t-test	mean	variance	t-test
1	120	0.5851	0.0316	0.8842	0.2344	+	0.7601	0.1655	+	0.7704	0.0392	+	0.7314	0.1585	~
	180	0.5953	0.0477	1.1535	0.5639	+	0.8132	0.2794	+	0.9585	0.1551	+	0.8358	0.1976	+
	240	0.5929	0.0072	0.7565	0.0823	+	0.6704	0.0086	+	0.7504	0.0521	+	0.6649	0.0016	+
2	120	0.6115	0.0226	0.7427	0.0118	+	1.2854	0.2188	+	0.9577	0.2259	+	1.4596	0.1328	+
	180	0.7362	0.0179	1.5620	0.2139	+	1.8080	0.1837	+	0.9332	0.1696	+	1.5980	0.1605	+
	240	0.5341	0.0183	1.3102	0.0597	+	1.4441	0.0277	+	0.6131	0.0495	~	1.3029	0.0424	+
3	120	0.6152	0.0333	0.7441	0.0108	+	0.7427	0.0118	+	0.7348	0.0281	+	0.7369	0.0178	+
	180	**0.4191**	**0.0090**	0.5410	0.0201	**+**	0.5394	0.0118	**+**	0.4956	0.0127	**+**	0.5999	0.0183	**+**
	240	0.4119	0.0064	0.4910	0.0214	+	0.5851	0.0248	+	0.5172	0.0156	+	0.6734	0.0179	+
4	120	0.7349	0.0485	0.9928	0.1997	+	0.1997	0.1871	+	1.2029	0.3249	+	0.9437	0.1499	+
	180	0.3775	0.0101	0.0101	0.0072	~	0.4325	0.0105	+	0.4636	0.0127	+	0.4270	0.0089	~
	240	0.4141	0.0083	0.4248	0.0126	~	0.4566	0.0140	~	0.4432	0.0155	~	0.5019	0.0140	+
5	120	0.4826	0.0146	0.6025	0.0247	+	0.5635	0.0241	+	0.6215	0.0439	+	0.6176	0.0284	+
	180	0.4603	0.0202	0.5332	0.0102	+	0.5560	0.0342	+	0.5279	0.0129	~	0.5001	0.0167	~
	240	0.7422	0.0746	0.8331	0.1045	~	0.7989	0.0676	~	1.0197	0.2042	+	0.9355	0.0792	+
6	120	0.4259	0.0111	0.4662	0.0085	~	0.4632	0.0198	~	0.5056	0.0325	+	0.4839	0.0107	+
	180	0.4301	0.0081	0.4798	0.0225	~	0.4673	0.0326	~	0.4286	0.0254	~	0.4622	0.0138	~
	240	0.4854	0.0187	0.5347	0.0129	~	0.4809	0.0274	~	0.5291	0.0472	~	0.5523	0.0155	~

line balancing problems," *IEEE Transactions on Artificial Intelligence*, pp. 1–13, 2023.

[8] C. B. Kalayci, A. Hancilar, A. Gungor, and S. M. Gupta, "Multi-objective fuzzy disassembly line balancing using a hybrid discrete artificial bee colony algorithm," *Journal of Manufacturing Systems*, vol. 37, pp. 672–682, 2015.

[9] M. L. Bentaha, O. Battaïa, and A. Dolgui, "Lagrangian relaxation for stochastic disassembly line balancing problem," *Procedia Cirp*, vol. 17, pp. 56–60, 2014.

[10] X. Guo, M. Zhou, A. Abusorrah, F. Alsokhiry, and K. Sedraoui, "Disassembly sequence planning: a survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 8, no. 7, pp. 1308–1324, 2020.

[11] Z. Li, I. Kucukkoc, and Z. Zhang, "Iterated local search method and mathematical model for sequence-dependent u-shaped disassembly line balancing problem," *Computers & Industrial Engineering*, vol. 137, p. 106056, 2019.

[12] S. Avikal, R. Jain, P. Mishra, and H. Yadav, "A heuristic approach for u-shaped assembly line balancing to improve labor productivity," *Computers & Industrial Engineering*, vol. 64, no. 4, pp. 895–901, 2013.

[13] X. Guo, Z. Bi, J. Wang, S. Qin, S. Liu, and L. Qi, "Reinforcement learning for disassembly system optimization problems: A survey," *International Journal of Network Dynamics and Intelligence*, vol. 2, no. 1, pp. 1–14, 2023.

[14] S. Wang, J. Wan, D. Li, and C. Zhang, "Implementing smart factory of industrie 4.0: an outlook," *International journal of distributed sensor networks*, vol. 12, no. 1, p. 3159805, 2016.

[15] Z.-L. Chen, "Integrated production and outbound distribution scheduling: review and extensions," *Operations research*, vol. 58, no. 1, pp. 130–148, 2010.

[16] A. Güngör and S. M. Gupta, "Disassembly line in product recovery," *International Journal of Production Research*, vol. 40, no. 11, pp. 2569–2589, 2002.

[17] X. Guo, S. Liu, M. Zhou, and G. Tian, "Dual-objective program and scatter search for the optimization of disassembly sequences subject to multiresource constraints," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1091–1103, 2017.

[18] Z. Zhang, X. Guo, M. Zhou, S. Liu, and L. Qi, "Multi-objective discrete grey wolf optimizer for solving stochastic multi-objective disassembly sequencing and line balancing problem," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2020, pp. 682–687.

[19] K. Wang, L. Gao, and X. Li, "A multi-objective algorithm for u-shaped disassembly line balancing with partial destructive mode," *Neural Computing and Applications*, vol. 32, no. 16, pp. 12 715–12 736, 2020.

[20] X. Cui, X. Guo, M. Zhou, J. Wang, S. Qin, and L. Qi, "Discrete whale optimization algorithm for disassembly line balancing with carbon emission constraint," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 3055–3061, 2023.

[21] Z. A. Çil, S. Mete, and F. Serin, "Robotic disassembly line balancing problem: A mathematical model and ant colony optimization approach," *Applied Mathematical Modelling*, vol. 86, pp. 335–348, 2020.

[22] J. Behnamian and S. Fatemi Ghomi, "A survey of multi-factory scheduling," *Journal of Intelligent Manufacturing*, vol. 27, no. 1, pp. 231–249, 2016.

[23] S. H. Chung, F. T. Chan, and H. K. Chan, "A modified genetic algorithm approach for scheduling of perfect maintenance in distributed production scheduling," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 7, pp. 1005–1014, 2009.

[24] F. Marandi and S. F. Ghomi, "Network configuration multi-factory scheduling with batch delivery: a learning-oriented simulated annealing approach," *Computers & Industrial Engineering*, vol. 132, pp. 293–310, 2019.

[25] A. Gharaei and F. Jolai, "A multi-agent approach to the integrated production scheduling and distribution problem in multi-factory supply chain," *Applied Soft Computing*, vol. 65, pp. 577–589, 2018.

[26] S. H. Chung, H. C. Lau, K. Choy, G. T. Ho, and Y. Tse, "Application of genetic approach for advanced planning in multi-factory environment," *International Journal of Production Economics*, vol. 127, no. 2, pp. 300–308, 2010.

[27] J. Petronijevic, A. Etienne, and J.-Y. Dantan, "Human factors under uncertainty: A manufacturing systems design using simulation-optimisation approach," *Computers & Industrial Engineering*, vol. 127, pp. 665–676, 2019.

[28] X. Wang, M. Zhou, Q. Zhao, S. Liu, X. Guo, and L. Qi, "A branch and price algorithm for crane assignment and scheduling in slab yard," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 3, pp. 1122–1133, 2020.

[29] J. Lian, C. Liu, W. Li, and Y. Yin, "A multi-skilled worker assignment problem in seru production systems considering the worker heterogeneity," *Computers & Industrial Engineering*, vol. 118, pp. 366–382, 2018.

[30] R. Liu, M. Liu, F. Chu, F. Zheng, and C. Chu, "Eco-friendly multi-skilled worker assignment and assembly line balancing problem," *Computers & Industrial Engineering*, vol. 151, p. 106944, 2021.

[31] B. J. Carnahan, B. A. Norman, and M. S. Redfern, "Incorporating physical demand criteria into assembly line balancing," *Iie Transactions*, vol. 33, no. 10, pp. 875–887, 2001.

[32] A. Baykasoglu, S. O. Tasan, A. S. Tasan, and S. D. Akyol, "Modeling and solving assembly line design problems by considering human

factors with a real-life application," *Human Factors and Ergonomics in Manufacturing & Service Industries*, vol. 27, no. 2, pp. 96–115, 2017.
[33] A. Otto and O. Battaïa, "Reducing physical ergonomic risks at assembly lines by line balancing and job rotation: A survey," *Computers & Industrial Engineering*, vol. 111, pp. 467–480, 2017.
[34] X. Guo, Z. Zhang, L. Qi, S. Liu, Y. Tang, and Z. Zhao, "Stochastic hybrid discrete grey wolf optimizer for multi-objective disassembly sequencing and line balancing planning in disassembling multiple products," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 1744–1756, 2021.
[35] Y. Fang, H. Ming, M. Li, Q. Liu, and D. T. Pham, "Multi-objective evolutionary simulated annealing optimisation for mixed-model multi-robotic disassembly line balancing with interval processing time," *International Journal of Production Research*, vol. 58, no. 3, pp. 846–862, 2020.
[36] L. Zhu, Z. Zhang, and Y. Wang, "A pareto firefly algorithm for multi-objective disassembly line balancing problems with hazard evaluation," *International Journal of Production Research*, vol. 56, no. 24, pp. 7354–7374, 2018.
[37] T. Rahkar Farshi, "Battle royale optimization algorithm," *Neural Computing and Applications*, vol. 33, no. 4, pp. 1139–1157, 2021.
[38] S. Alp, R. Dehkharghani, T. Akan, and M. A. Bhuiyan, "Mobro: multi-objective battle royale optimizer," *The Journal of Supercomputing*, vol. 80, no. 5, pp. 5979–6016, 2024.
[39] S. Agahian and T. Akan, "Battle royale optimizer for training multi-layer perceptron," *Evolving Systems*, vol. 13, no. 4, pp. 563–575, 2022.
[40] K. M. Ong, P. Ong, and C. K. Sia, "A carnivorous plant algorithm for solving global optimization problems," *Applied Soft Computing*, vol. 98, p. 106833, 2021.
[41] E. Duman, M. Uysal, and A. F. Alkaya, "Migrating birds optimization: a new metaheuristic approach and its performance on quadratic assignment problem," *Information Sciences*, vol. 217, pp. 65–77, 2012.
[42] H. Peraza-Vázquez, A. F. Peña-Delgado, G. Echavarría-Castillo, A. B. Morales-Cepeda, J. Velasco-Álvarez, and F. Ruiz-Perez, "A bio-inspired method for engineering design optimization inspired by dingoes hunting strategies," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–19, 2021.
[43] Q.-K. Pan, H.-Y. Sang, J.-H. Duan, and L. Gao, "An improved fruit fly optimization algorithm for continuous function optimization problems," *Knowledge-Based Systems*, vol. 62, pp. 69–83, 2014.
[44] T. Akan, S. Agahian, and R. Dehkharghani, "Binbro: Binary battle royale optimizer algorithm," *Expert Systems with Applications*, vol. 195, p. 116599, 2022.
[45] ——, "Battle royale optimizer for solving binary optimization problems," *Software Impacts*, vol. 12, p. 100274, 2022.
[46] X. Guo, C. Fan, M. Zhou, S. Liu, J. Wang, S. Qin, and Y. Tang, "Human–robot collaborative disassembly line balancing problem with stochastic operation time and a solution via multi-objective shuffled frog leaping algorithm," *IEEE Transactions on Automation Science and Engineering*, 2023.
[47] Z. Zhao, S. Liu, M. Zhou, D. You, and X. Guo, "Heuristic scheduling of batch production processes based on petri nets and iterated greedy algorithms," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 1, pp. 251–261, 2020.
[48] X. Guo, S. Liu, M. Zhou, and G. Tian, "Disassembly sequence optimization for large-scale products with multiresource constraints using scatter search and petri nets," *IEEE transactions on cybernetics*, vol. 46, no. 11, pp. 2435–2446, 2015.
[49] R. J. Riggs, O. Battaïa, and S. J. Hu, "Disassembly line balancing under high variety of end of life states using a joint precedence graph approach," *Journal of Manufacturing Systems*, vol. 37, pp. 638–648, 2015.
[50] C. Ullerich and U. Buscher, "Flexible disassembly planning considering product conditions," *International Journal of Production Research*, vol. 51, no. 20, pp. 6209–6228, 2013.
[51] M. L. Bentaha, A. Dolgui, O. Battaïa, R. J. Riggs, and J. Hu, "Profit-oriented partial disassembly line design: dealing with hazardous parts and task processing times uncertainty," *International Journal of Production Research*, vol. 56, no. 24, pp. 7220–7242, 2018.
[52] P. Nowakowski, "A novel, cost efficient identification method for disassembly planning of waste electrical and electronic equipment," *Journal of Cleaner Production*, vol. 172, pp. 2695–2707, 2018.
[53] Q. Lu, Y. Ren, H. Jin, L. Meng, L. Li, C. Zhang, and J. W. Sutherland, "A hybrid metaheuristic algorithm for a profit-oriented and energy-efficient disassembly sequencing problem," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101828, 2020.
[54] A. P. Guerreiro, C. M. Fonseca, and L. Paquete, "The hypervolume indicator: Computational problems and algorithms," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–42, 2021.
[55] J. Luo, X. Huang, Y. Yang, X. Li, Z. Wang, and J. Feng, "A many-objective particle swarm optimizer based on indicator and direction vectors for many-objective optimization," *Information Sciences*, vol. 514, pp. 166–202, 2020.
[56] J. A. Rangel-González, H. Fraire, J. F. Solís, L. Cruz-Reyes, C. Gomez-Santillan, N. Rangel-Valdez, and J. M. Carpio-Valadez, "Fuzzy multi-objective particle swarm optimization solving the three-objective portfolio optimization problem," *International Journal of Fuzzy Systems*, vol. 22, no. 8, pp. 2760–2768, 2020.

LiangBo Zhou Graduated from Jiangsu University of Science and Technology in 2022 with a bachelor's degree in Internet of Things Engineering. Graduated from Liaoning Petrochemical University in 2025 with a master's degree in Artificial Intelligence. Currently working in the Development and Reform Bureau of Congjiang County, Congjiang, China.

Haibin Zhu is a Full Professor at Nipissing University, Canada. He is also an affiliate full professor of Concordia Univ. and an adjunct professor of Laurentian Univ., Canada. He has accomplished over 300+ research publications, including 60+ IEEE Transactions articles. He is a fellow of IEEE, AAIA (Asia-Pacific Artificial Intelligence Association) and I2CICC (International Institute of Cognitive Informatics and Cognitive Computing), a senior member of ACM. He is Vice President, Systems Science and Engineering (SSE) (2023-), a co-chair (2006-) of the technical committee of Distributed Intelligent Systems, and a Distinguished Lecturer of IEEE Systems, Man and Cybernetics Society (SMCS), Associate Editor (AE) of IEEE Transactions on SMC: Systems (2018-), IEEE Transactions on Computational Social Systems (2018-), IEEE Systems Journal (2024-), Frontiers of Computer Science (2021-), IEEE Canada Review (2017-), and Deputy Editor-in-Chief of Artificial Intelligence Science and Engineering (2025-). He was General Chair: E-CARGO 2025, China, 2024, China, 2023, online, ScalCom 2023, UK; ISEEIE 2023, Singapore, SPCS 2022, China, ICCSIT 2021, France, and Program Chair: ICFTIC 2025, 2024, China, CSCWD 2020, CSCWD 2018, China, ICNSC 2019, and CSCWD 2013, Canada. He is the founding researcher of Role-Based Collaboration and the creator of the E-CARGO model. He has offered 38 keynote speeches for international conferences and 93 invited talks internationally.

Behzad Akbari is an IEEE member who earned his bachelor's and master's degrees in Computer Software and Computer Architecture from Tehran University and Knowledge and Research University, Iran, in 1996 and 1999, respectively. He continued his academic journey in Canada, completing a second master's in Computer Science and a Ph.D. in Electrical and Computer Engineering (ECE) at McMaster University in 2015 and 2021. Dr. Akbari is currently an Assistant Professor at Michigan Technological University in Houghton, Michigan, USA. He has served as a reviewer for prominent journals including IEEE Access, IEEE Transactions on Robotics, and IEEE Transactions on Systems, Man, and Cybernetics. His research focuses on state estimation algorithms, collaborative multi-agent systems, multi-target tracking, multi-output Gaussian processes, iterative localization and mapping, factor graph optimization, and reinforcement learning.