Order-Driven Disassembly Task Scheduling and Optimization for Multi-Factory Remanufacturing

Chenggang Zheng, Ying Tang, Weitian Wang, and Qi Kang

***Abstract*—Multi-factory remanufacturing systems often operate under practical constraints, including heterogeneous task requirements and limited availability of skilled labor, where factory and workstation operations cannot be assumed to be continuously active. In such environments, worker availability and order demand jointly determine factory activation, workstation utilization, and task allocation, leading to tightly coupled decisions across multiple resource levels. Effectively coordinating these interdependent factors is critical for improving operational productivity in distributed remanufacturing networks. This study investigates a multi-factory remanufacturing problem that jointly optimizes order allocation, worker assignment, and factory activation decisions under labor constraints. A mixed-integer linear programming model is developed to capture the interactions among factories, workstations, and workers while considering disassembly task structures derived from order requirements. To solve the resulting complex optimization problem, the Alpha Evolution algorithm is employed and compared with several representative metaheuristic approaches, including the Improved Beluga Whale Reproductive Optimization, Coati Population Algorithm, Fruit Fly Optimization Algorithm, Dingo Optimization Algorithm, and Modified Dung Beetle Mating Optimization. Experimental results demonstrate that coordinated resource activation can significantly enhance labor utilization and overall system performance. The proposed Alpha Evolution algorithm approach achieves competitive performance in solution quality and stability across multiple test scenarios.**

***Key Words*—Multi-factory remanufacturing, order and worker scheduling, Alpha Evolution algorithm, cost minimization.**

I. Introduction

REMANUFACTURING has emerged as a key approach to sustainable development, allowing end-of-life products to be restored and valuable subassemblies reintroduced into production system. In particular, the optimization of multi-factory remanufacturing processes (MRPOP) [1] has gained increasing attention, as it can substantially reduce resource consumption and operational costs, contributing to environmental sustainability. However, most current studies focus on single-factory scenarios or idealized production conditions, which fail to reflect the real complexity of remanufacturing systems involving multiple factories, diverse orders, and dynamic worker allocation [2]. In practical remanufacturing environments, production efficiency is not only determined by task scheduling but also by the coordination and activation of multiple interdependent resources across factories.

A major challenge in multi-factory manufacturing system lies in the simultaneous consideration of order demands and worker scheduling [3]. In many real-world remanufacturing systems, factories and workstations cannot remain continuously active and are often triggered by worker availability and incoming order requirements [4]. Orders vary in size, priority, and delivery requirements, while workers must be strategically assigned to factories and workstations to maintain balanced workloads and operational efficiency [2, 5]. Ignoring these factors can lead to unbalanced resource utilization, higher costs, and reduced overall system performance [6]. To address these issues, this study develops a mixed-integer linear programming (MILP) model that jointly considers order allocation, worker scheduling, and factory activation decisions within a unified optimization framework.

Over the past five years, multi-factory optimization with order and worker scheduling has attracted growing attention [7]. Unlike traditional single-factor scheduling problems, MFSOP considers inter-factory collaboration, order heterogeneity, and worker scheduling constraints, which significantly increases its complexity [8, 9]. Optimizing MFSOP is therefore critical for improving productivity, reducing operational costs, and maintaining sustainable competitiveness in the global market [10]. However, relatively limited attention has been paid to scenarios where factory activation itself becomes a decision variable influenced by labor availability.

As illustrated in Fig. 1, the proposed multi-factory remanufacturing scheduling framework integrates multi-factory coordination, worker allocation, and order scheduling into a unified decision-making system [11]. However, due to the combinatorial nature of decision variables, the dynamic constraints of real production systems, and the practical complexities of various operational uncertainties and coordination requirements, the multi-factory order-worker scheduling problem remains highly challenging [12]. The interdependence among these decision layers creates tightly coupled relationships that significantly affect system feasibility and operational performance.

Due to the NP-hard nature of the proposed model, exact

Manuscript received February 22, 2026; revised March 1 and March 8, 2026; accepted March 20, 2026. This article was recommended for publication by Associate Editor Shujin Qin upon evaluation of the reviewers' comments.

This work was supported in part by the National Natural Science Foundation of China under Grant 61903229 and in part by the Natural Science Foundation of Shandong Province under Grant ZR2022MF270.

C. Zheng is with the College of Information and Control Engineering, Liaoning Petrochemical University, Fushun 113001, P. R. China (e-mail:2364730326@qq.com)

Y. Tang is with the Department of Electrical and Computer Engineering, Rowan University, Glassboro, NJ 08028, USA (e-mail: tang@rowan.edu).

W. Wang is with the School of Computing, Montclair State University, Montclair, NJ 07043 USA (e-mail: wangw@montclair.edu).

Q. Kang is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA (e-mail: qk22@njit.edu).

Corresponding Author: Chenggang Zheng.

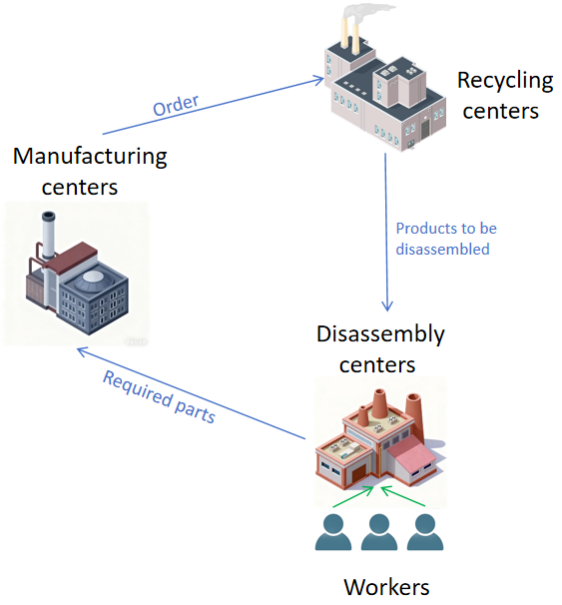


Fig. 1. Multi-factory remanufacturing process.

optimization methods become computationally prohibitive for large-scale scenarios. To address this challenge, this study adopts the Alpha Evolution (AE) algorithm as an effective solution approach [13]. AE incorporates adaptive evolutionary strategies and elite-guided search mechanisms, which enhance global exploration and accelerate convergence while preventing premature stagnation.

Comprehensive comparative experiments are conducted to benchmark AE against several state-of-the-art metaheuristics, including the Improved Beluga Whale Reproductive Optimization (IBRO) [14], Coati Population Algorithm (CPA), Fruit Fly Optimization Algorithm (FOA), Dingo Optimization Algorithm (DOA), and Modified Dung Beetle Mating Optimization (MDMBO). These comparisons help evaluate the effectiveness of the proposed modeling framework under different solution strategies. Results show that AE consistently achieves lower total costs, faster convergence speed, and more stable performance, demonstrating its suitability and advantage in solving large-scale multi-factory scheduling and remanufacturing problems.

This work contributes to the modeling of coordinated decision-making mechanisms in multi-factory remanufacturing systems, with the main contributions summarized as follows:

1) Created a mixed-integer linear programming model that captures the coordination among order allocation, factory activation, and worker scheduling in multi-factory remanufacturing systems.
2) Developed an AE-based solution integrating population evolution and elite strategies to strengthen global search and accelerate convergence for remanufacturing tasks.
3) Implemented extensive experiments on various production scales, validating the proposed model and showing that the developed approach outperforms five state-of-the-art metaheuristics—IBRO [15], CPA [16], FOA [17], DOA [18], and MDMBO [19].

The remainder of this work is organized as follows. Section II formulates the mixed-integer linear programming model for multi-factory remanufacturing considering order allocation and worker scheduling. Section III introduces the AE-based approach designed to solve the proposed optimization problem. Section IV presents the experimental results and compares the proposed solution with several classical algorithms. Section V concludes this study and discusses directions for future research.

II. Related Work

The remanufacturing optimization problem (ROP) has been extensively investigated in the context of multi-factory settings and system-level remanufacturing decision frameworks. Tang *et al.* [20] and Wang *et al.* [21] incorporated uncertainties and multi-objective considerations into multi-factory production. Xu *et al.* [22] explored hybrid heuristics to enhance synchronization across distributed processes. In intelligent disassembly systems, Guo *et al.* [23] introduced a human–robot collaborative disassembly balancing model with shared resources, informing cross-factory coordination strategies. In addition, multiple studies [24–26] advanced the understanding of disassembly systems from task characteristics, human factors, and multi-factory perspectives. Although these efforts extend ROP modeling to more complex environments, research on integrated multi-factory remanufacturing optimization—particularly where disassembly, allocation, and scheduling are coupled—remains insufficient. Most existing studies treat factory coordination, task scheduling, and resource allocation as separate decision layers, while the interaction between order demand, worker availability, and factory activation has rarely been explicitly modeled within a unified decision-making hierarchy.

Extensive efforts have been devoted to optimizing order allocation and related decision-making in multi-factory production environments. Wang *et al.* [27] developed a joint optimization model for order allocation and rack selection in multi-station systems to balance workload and reduce operational costs. Martin *et al.* [28] examined tactical allocation strategies in international manufacturing networks, underscoring the importance of cross-factory coordination and production flexibility. Zhang *et al.* [29] further introduced a reliability-driven dynamic order allocation and inventory management model to enhance decision robustness under production uncertainty. These studies collectively advance intelligent and adaptive order assignment mechanisms, providing conceptual support for the integrated multi-factory remanufacturing optimization framework addressed in this work.However, most of these studies focus primarily on order allocation or production planning, without explicitly incorporating worker-related constraints into the allocation process. As a result, the joint consideration of order demand and worker availability as drivers of factory activation and task allocation remains largely unexplored in existing multi-factory scheduling research.

Given that the MRPOP is an NP-hard problem, its computational complexity increases rapidly with the data scale, making heuristic and metaheuristic algorithms indispensable for practical solution efficiency [30, 31]. Guo *et al.* [32]

proposed a human–machine collaborative DLBP model with stochastic task times and designed a Pareto-improved shuffled frog-leaping algorithm enhanced by a stochastic simulation strategy and elite operators. Wei *et al.* [33] incorporated human factors into disassembly line balancing and developed a multi-objective discrete harmony search optimizer to improve both solution quality and computational speed. Zhao *et al.* [34] addressed time-constrained rolling-process scheduling through Petri-net modeling and iterative greedy heuristics, while Çil *et al.* [35] integrated distributed disassembly line balancing with vehicle routing and solved it using a multi-start simulated annealing approach. Gu *et al.* [36] tackled dynamic job shop scheduling with a self-learning discrete salp swarm algorithm to balance global exploration and local exploitation. Moreover, Zhang *et al.* [37] improved the Fruit Fly Optimization Algorithm by considering worker learning effects for multi-objective disassembly line balancing, which provides inspiration for the adaptive mechanisms incorporated into the AE in this study.

III. Problem Description

A. Problem Statement

Traditional multi-factory remanufacturing systems often assume relatively fixed order assignments and worker schedules. However, with the continuous arrival of new orders and dynamic worker availability, coordinating production activities across multiple factories becomes significantly more complex. In such environments, production efficiency is largely influenced by how orders and workers are jointly allocated rather than by isolated scheduling decisions.

In this study, we focus on the optimization of order allocation and worker scheduling in a multi-factory remanufacturing system with the objective of minimizing total operational costs. Factories are not assumed to be continuously operational; instead, their utilization is implicitly determined by the assigned orders and available workers. As illustrated in Fig. 2, the complexity of the problem arises from the need to balance order processing requirements, worker capacity, and factory resources within a unified decision-making framework.

A typical multi-factory remanufacturing system consists of multiple factories, each equipped with its own workstations. Orders can be allocated to different factories as long as operational capacity and worker availability constraints are satisfied. The integration of order allocation with worker scheduling creates a tightly coupled decision process, where feasible order–worker assignments directly influence factory utilization and overall system performance.

To address this problem, a mixed-integer linear programming model is developed to coordinate order allocation and worker scheduling while ensuring that production activities are carried out only when sufficient workload and qualified workers are available. The goal is to minimize the total cost, including labor costs and resource utilization costs, while guaranteeing timely completion of all orders.

In addition, this research explores the integration of order scheduling with worker scheduling. This combination ensures that orders are assigned to workers based on available skills

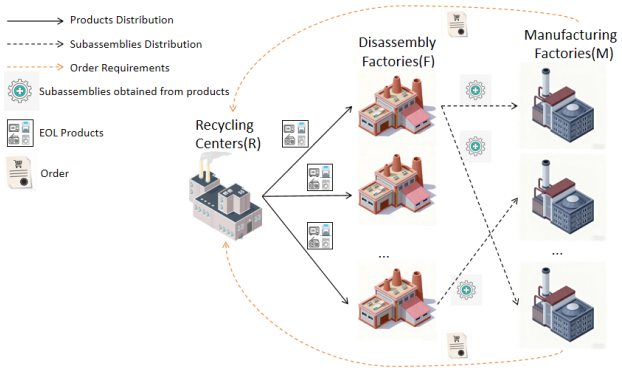


Fig. 2. The workflow of multi-factory remanufacturing process optimization.

and factory resources, optimizing the scheduling of both orders and workers simultaneously. By aligning order needs with worker capabilities, the system enhances flexibility and responsiveness to changing production demands, leading to significant improvements in cost efficiency and overall production output across the entire system. We make the following assumptions:

- The incidence matrix, conflict matrix, and precedence matrix of the EOL products are known.
- Disassembly is selective; not all subassemblies need to be disassembled.
- The operation time of each workstation should not exceed the given system cycle time.
- At least one disassembly task is assigned to each active workstation.
- At least one worker is assigned to each workstation.
- Each worker has a known fixed cycle time.
- The cost and time per time unit of each disassembly operation by a worker is known.

B. AND/OR Graph

Since a disassembly system is a discrete event system, one can use Petri nets or other formal methods to model and analyze the disassembly processes and resource requirements. In this work, we use AND/OR diagrams to describe this relationship among operations for its simplicity. An AND/OR graph, representing all feasible disassembly sequences of an EOL product, specifies the precedence and conflicting relationships among operations. In an AND/OR graph, each node represents a subassembly indexed by an integer i in an angle bracket, i.e., $\langle i \rangle, i = 1, 2, \ldots, I$, where I denotes the number of subassemblies. Nodes are then connected by direct arcs. A disassembly operation is represented by multiple directed edges originated from the same starting node to its child nodes, forming an "AND" relation. All operations are labeled pictorially by an integer number j, $j = 1, 2, \ldots, J$, where J is the total number of disassembly operations of a given EOL product. A node can have more than one operation, forming an "OR" relation.

In our case, the EOL product is a household oven. Fig. 3 shows the schematic diagram of the oven, in which several key subassemblies are indicated. Based on this structure,

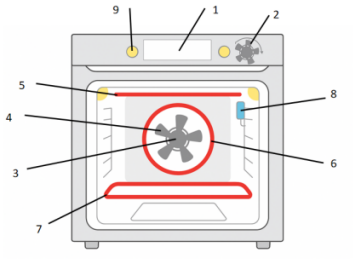


Fig. 3. Partial schematic diagram of oven.

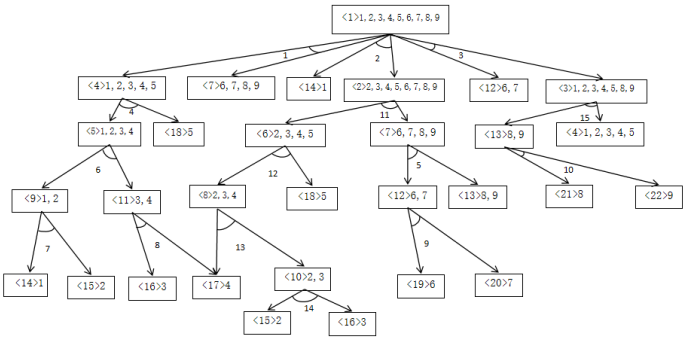


Fig. 4. The AND/OR graph of an oven.

the corresponding AND/OR graph of the oven disassembly process is constructed, as illustrated in Fig. 4. Each node in the graph represents a subassembly of the oven, and the directed edges denote feasible disassembly operations.

C. Notations

Sets:

$\mathbb{F}$ Set of disassembly factories, $\mathbb{F} = \{1, 2, ..., F\}$.

$\mathbb{M}$ Set of manufacturing factories, $\mathbb{M} = \{1, 2, ..., M\}$.

$\mathbb{P}$ Set of products, $\mathbb{P} = \{1, 2, ..., P\}$.

$\mathbb{J}_p$ Set of all subassemblies in product p, $\mathbb{J}_p = \{1, 2, ..., J_p\}$.

$\mathbb{I}_p$ Set of all tasks in product p, $\mathbb{I}_p = \{1, 2, ..., I_p\}$.

$\mathbb{W}$ Set of workstations of the k-th factory, $\mathbb{W} = \{1, 2, ..., W\}$.

$\mathbb{H}$ Set of workers, $\mathbb{H} = \{1, 2, ..., H\}$.

$\mathbb{O}$ Set of order types, $\mathbb{O} = \{1, 2, ..., O\}$.

Indexes:

p Product index, $p \in \mathbb{P}$.

j Subassembly index, $j \in \mathbb{J}_p$.

i Disassembly task index, $i \in \mathbb{I}_p$.

f Disassembly factory index. $f \in \mathbb{F}$.

m Manufacturing factory index. $m \in \mathbb{M}$.

w Workstation index, $w \in \mathbb{W}$.

h Worker index, $h \in \mathbb{H}$.

o order index, $o \in \mathbb{O}$.

Parameters:

v_{mo} The m-th manufacturing factory acquires the price of the o-th order.

c^T_{fmo} Transportation cost of the o-th order of the from the f-th disassembly factory to the m-th manufacturing factory.

t_{hpi} Disassembly time required by h-th workers to complete the i-th task of the p-th product.

c^D_{hpi} The unit time cost of the h-th worker executing the i-th task of the p-th product in the f-th factory.

c^F_f The unit time cost of activating the f-th disassembly factory.

c_{fw} Cost of activating the w-th workstation of the f-th disassembly factory.

α Disassembly incidence matrix.

γ Disassembly conflict matrix.

β Disassembly precedence matrix.

c^H_{hf} Commuting cost of the h-th worker to the f-th disassembly factory.

T The length of each time period.

z_{pjo} Matrix o of the category to which product p and part j belong.

Q_{mo} Manufacturing factory m quantity requirement for order o.

Decision variables:

$$\lambda_{hfw} = \begin{cases} 1, & \text{If the } h\text{-th worker is assigned to the w-th} \\ & \text{workstation in the f-th disassembly factory;} \\ 0, & \text{otherwise.} \end{cases}$$

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

$$x_{hfwpi} = \begin{cases} 1, & \text{If the } h\text{-th worker executes the } i\text{-th task of the} \\ & p\text{-th product at the } w\text{-th workstation} \\ & \text{in the } f\text{-th disassembly factory;} \\ 0, & \text{otherwise.} \end{cases}$$

$$y_f = \begin{cases} 1, & \text{If the } f\text{-th disassembly factory is activated;} \\ 0, & \text{otherwise.} \end{cases}$$

$$u_{fw} = \begin{cases} 1, & \text{If the } w\text{-th workstation of the } f\text{-th disassembly} \\ & \text{factory is activated;} \\ 0, & \text{otherwise.} \end{cases}$$

$$\alpha_{fmo} = \begin{cases} 1, & \text{If the } o\text{-th order is transported from the} \\ & f\text{-th disassembly factory to the} \\ & m\text{-th manufacturing factory;} \\ 0, & \text{otherwise.} \end{cases}$$

D. Mathematical Model

We formulate the optimization problem of MRPWS as:

$$\begin{aligned}\min \psi = &\sum_{f\in\mathbb{F}}\sum_{m\in\mathbb{M}}\sum_{o\in\mathbb{O}} c^{T}_{fmo}\alpha_{fmo} \\ &+ \sum_{h\in\mathbb{H}}\sum_{f\in\mathbb{F}}\sum_{w\in\mathbb{W}}\sum_{p\in\mathbb{P}}\sum_{i\in\mathbb{I}_p} c^{D}_{hpi}t_{hpi}x_{hfwpi} \\ &+ \sum_{f\in\mathbb{F}} c^{F}_{f}y_f + \sum_{f\in\mathbb{F}}\sum_{w\in\mathbb{W}} c_{fw}u_{fw} \\ &+ \sum_{f\in\mathbb{F}}\sum_{w\in\mathbb{W}}\sum_{h\in\mathbb{H}} c^{H}_{hf}\lambda_{hfw}\end{aligned} \tag{1}$$

The objective function (1) represents the minimum cost of disassembly EOL products. The first item represents the transportation cost from the disassembly factory to the manufacturing factory. The second term represents the disassembly cost of performing the disassembly task. The third term represents the cost of turning on the disassembly factory. The fourth term represents the cost of turning on the associated workstations. The fifth term represents the commuting cost of worker scheduling.

$$\sum_{h\in\mathbb{H}}\sum_{p\in\mathbb{P}}\sum_{i\in\mathbb{I}_p} t_{hpi}x_{hfwpi} \le T, \forall f\in\mathbb{F}, \forall w\in\mathbb{W} \tag{2}$$

$$\sum_{f\in\mathbb{F}}\sum_{w\in\mathbb{W}} \lambda_{hfw} \le 1, \forall h\in\mathbb{H} \tag{3}$$

$$\begin{aligned}&\sum_{m\in\mathbb{M}} \alpha_{fmo} \le \sum_{h\in\mathbb{H}}\sum_{w\in\mathbb{W}}\sum_{i\in\mathbb{I}_p} \alpha_{pji}x_{hfwpi}z_{pjo} \\ &\forall f\in\mathbb{F},\ \forall p\in\mathbb{P},\ \forall j\in\mathbb{J}_P\setminus\{1\}, \forall o\in\mathbb{O}\end{aligned} \tag{4}$$

$$\sum_{f\in\mathbb{F}} \mu_{pf} = 1,\ \forall p\in\mathbb{P} \tag{5}$$

$$\mu_{pf} \le y_f,\ \forall p\in\mathbb{P}, \forall f\in\mathbb{F} \tag{6}$$

$$u_{fw} \le y_f, \forall f\in\mathbb{F}, \forall w\in\mathbb{W} \tag{7}$$

$$\sum_{h\in\mathbb{H}} \lambda_{hfw} \le u_{fw}, \forall f\in\mathbb{F}, \forall w\in\mathbb{W} \tag{8}$$

$$x_{hfwpi} \le \mu_{pf}, \forall f\in\mathbb{F}, \forall w\in\mathbb{W}, \forall p\in\mathbb{P}, \forall i\in\mathbb{I}_p \tag{9}$$

$$x_{hfwpi} \le \lambda_{hfw}, \forall f\in\mathbb{F}, \forall w\in\mathbb{W}, \forall p\in\mathbb{P}, \forall i\in\mathbb{I}_p \tag{10}$$

$$\sum_{h\in\mathbb{H}}\sum_{w\in\mathbb{W}} x_{hfwpi} \le \mu_{pf}, \forall f\in\mathbb{F}, \forall p\in\mathbb{P}, \forall I\in\mathbb{i}_p \tag{11}$$

$$\begin{aligned}&\sum_{h\in\mathbb{H}}\sum_{w\in\mathbb{W}} w\left(x_{hfwpi_1} - x_{hfwpi_2}\right) + W\left(\sum_{w\in\mathbb{W}} x_{hfwpi_2 l} - 1\right) \le 0 \\ &\forall f\in\mathbb{F},\ \forall p\in\mathbb{P},\ \forall i_1, i_2\in\mathbb{I}_p,\ \beta_{pi_1i_2} = 1\end{aligned} \tag{12}$$

$$\begin{aligned}&\sum_{h\in\mathbb{H}}\sum_{w\in\mathbb{W}} x_{hfwpi_2} \le \sum_{h\in\mathbb{H}}\sum_{i_1\in\mathbb{I}_p}\sum_{w\in\mathbb{W}} \beta_{pi_1i_2}x_{hfwpi_1}, \forall f\in\mathbb{F},\ \forall p\in\mathbb{P}, \\ &\forall i_2\in\mathbb{I}_p\end{aligned} \tag{13}$$

$$\begin{aligned}&\sum_{h\in\mathbb{H}}\sum_{w\in\mathbb{W}} \left(x_{hfwpi_1} + x_{hfwpi_2}\right) \le 1,\ \forall f\in\mathbb{F},\ \forall p\in\mathbb{P}, \\ &\forall i_1, i_2\in\mathbb{I}_p, \gamma_{pi_1i_2} = 1\end{aligned} \tag{14}$$

$$\sum_{f\in\mathbb{F}} \alpha_{fmo} \ge Q_{mo},\ \forall m\in\mathbb{M},\ \forall O\in\mathbb{O} \tag{15}$$

$$\lambda_{hfw}\in\{0,1\},\ \forall h\in\mathbb{H}, \forall f\in\mathbb{F}, \forall w\in\mathbb{W} \tag{16}$$

$$\mu_{pf}\in\{0,1\},\ \forall p\in\mathbb{P},\ \forall f\in\mathbb{F} \tag{17}$$

$$\begin{aligned}&x_{hfwpi}\in\{0,1\}, \forall h\in\mathbb{H}, \forall f\in\mathbb{F}, \forall w\in\mathbb{W}, \forall p\in\mathbb{P} \\ &\forall i\in\mathbb{I}_p\end{aligned} \tag{18}$$

$$y_f\in\{0,1\},\ \forall f\in\mathbb{F} \tag{19}$$

$$u_{fw}\in\{0,1\},\ \forall f\in\mathbb{F},\ \forall w\in\mathbb{W} \tag{20}$$

$$\alpha_{fmo}\in\{0,1\}, \forall f\in\mathbb{F}, \forall m\in\mathbb{M}, \forall o\in\mathbb{O} \tag{21}$$

$$T\in\mathbb{R}_+ \tag{22}$$

Constraint (2) ensures that the total working time of the workers can not exceed the length of the time period. Constraint (3) ensures that each worker is assigned to at most one workstation in each time period. Constraint (4) ensures that subassemblies obtained by disassembling a product can be transported to only one manufacturing factory. Constraint (5) ensures that each product is assigned to only one disassembly factory in only one time period. Constraint (6) ensures that products can only be assigned to disassembly factories that have been turned on. Constraint (7) ensures that the workstations in a factory can only be turned on if the disassembly factory is turned on. Constraint (8) ensures that each workstation is assigned at most one worker. Constraint (9) ensures that if a product is not assigned to a disassembly factory, workers cannot perform tasks for that product at any workstation in that factory. Constraint (10) ensures that if a worker is not assigned to a workstation in a disassembly factory, they cannot perform any tasks at that workstation. Constraint (11) ensures that each disassembly task for each product is performed at most once. Constraint (12) ensures that the assignment of product disassembly tasks complies with precedence constraints. Constraint (13) ensures that the disassembly sequence of the product can begin from other tasks. Constraint (14) ensures that the assignment of disassembly tasks causes no conflict among them. Constraint (15) indicates that the decision satisfies the order allocation conditions. Constraints (16)-(22) indicate the range of values of decision variables.

IV. Proposed Algorithm

Alpha Evolution (AE) algorithm is an effective evolutionary algorithm, which has the characteristics of strong global search ability and fast convergence. AE simulates the process of natural evolution, where individuals evolve based on their fitness and environmental factors. The algorithm uses population-based strategies to explore the solution space and optimize complex scheduling problems. This work extends the basic AE to solve the multi-factory remanufacturing scheduling problem, aiming to minimize overall costs and improve production efficiency.

A. Basic Process of AE

In AE, the initial population is randomly distributed throughout the solution space. Each individual represents a possible solution and moves stochastically to find advantageous positions to minimize the total production cost. AE introduces a fusion process, where each individual combines its current solution with the global best solution to enhance its own fitness. This fusion occurs through a crossover process, which integrates the individual's solution with the global optimum to explore better solution spaces.

A mutation strategy is also integrated into the algorithm to introduce random disturbances, preventing the algorithm from getting stuck in local optima. Specifically, if the best solution of the population does not improve after a certain number of iterations, a random mutation applies to the individual solutions, creating new variations in the population. This allows the algorithm to explore new areas of the solution space, enhancing its ability to find global optima. The AE process continues until a satisfactory solution is found or the maximum number of iterations is reached.

The basic idea of AE is described as follows: First, we initialize its basic parameters and population. Then, the population is ranked in descending order according to the fast non-dominated sorting method and crowding distance strategy . Then, a memeplex is constructed by using the memeplex-construction algorithm. A local search is performed for each memeplex to realize memeplex evolution. After the local search is completed, the population is updated by merging all memeplexes. After that, crossover and selective operators are performed on this population, and the Pareto front is updated. Then the next iteration process is executed until a pre-set termination condition is met. The AE process is shown in Algorithm 1 [13].

Algorithm 1 AE

Input: $N, D, ub, lb, FEs = 0, MaxFEs$
Output: Global optimal solution: X_{best}

Initialize the candidate matrix using $\mathbf{X} = lb + (ub - lb) \cdot \text{rand}(0, 1, [N, D])$ and evaluate
$FEs = FEs + N$
while $FEs <= MaxFEs$ **do**
  $\mathbf{E} \xrightarrow{N} \mathbf{X}$
  $ind = \text{sort}(f(\mathbf{X}))$
  $\mathbf{R}_1 = \text{rand}(0, 1, [N, D])$
  $\mathbf{R}_2 = \text{rand}(0, 1, [N, D])$
  $\mathbf{S} = \text{randi}(0, 1, [N, D])$
  $\Delta r = (ub - lb) \cdot (2\mathbf{R}_1 \cdot \mathbf{R}_2 - \mathbf{R}_2) \cdot \mathbf{S}$
  $\alpha = \exp(\ln(1 - FEs/MaxFEs) - (4FEs/MaxFEs)^2)$
  for $i = 1 : N$ **do**
    if $\text{rand}(0, 1) < 0.5$ **then**
      $\mathbf{A} = \mathbf{X}(\text{randi}(1, N, [D, 1]), :)$
      $c_a = 1 - FEs/MaxFEs$
      $p_a^{t+1} = c_a p_a^t + (1 - c_a) \times \text{diagonal}(\mathbf{A})$
      $P = p_a^{t+1}$
    else
      $K = \lfloor N \times \text{rand}(0, 1) \rfloor$
      $I_1 = \text{randperm}(N, K)$
      $\mathbf{B} = \mathbf{X}(I_1, :)$
      $\omega = f(I_1)/\text{sum}(f(I_1))$
      $c_b = 1 - FEs/MaxFEs$
      $p_b^{t+1} = c_b p_b^t + (1 - c_b) \times \omega\mathbf{B}$
      $P = p_b^{t+1}$
    end if
    $W_i = X(\text{ind}(\text{randi}([1, \text{length}(1 : \text{find}(k == ind))])), :)$
    $L_i = X(\text{ind}(\text{randi}([\text{length}(1 : \text{find}(k == ind)), N])), :)$
    $I_2 = \text{round}(\text{rand}(0, 1))$
    $\theta = I_2 \times 2\text{rand}(0, 1) + (1 - I_2) \times \text{rand}(0, 1, [1, D])$
    $E_i^{t+1} = P + \alpha\Delta r_i + \theta \cdot (W_i + E_i^t - P - L_i)$
    Boundary constraints on variables using Eq. (14)
    Select X_k according to Eq. (15)
    Recording the value of X_{best}
    $FEs = FEs + 1$
  end for
end while
return X_{best}

B. Encoding and Decoding Scheme

Through effective encoding and decoding, AE can be applied to optimize order allocation and worker scheduling, thereby improving the efficiency and resource utilization of the remanufacturing process. According to the characteristics of this problem, the randomness of orders, the dynamics of worker scheduling, the selection of disassembly and manufacturing factories, the sequence of disassembly tasks, and the allocation of workstations will all affect the solution. In this study, we design a five-stage encoding method $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ to represent the solution of this problem. Here, σ_1 represents the disassembly task sequence, σ_2 denotes the required parts of the orders, σ_3 refers to the disassembly factory index, σ_4 represents the workstation index, and σ_5 indicates the worker index. The encoding scheme is illustrated in Fig. 5.

For example, the recycling plant requires two units each of parts 11, 12, 13, 14, and 15. Product 1 is assigned to disassembly factory 3, with a disassembly task sequence of $2, 4, 8, 10, 13$. Product 2 is assigned to disassembly factory 1, with a disassembly task sequence of $2, 6, 8, 10, 11$. The disassembly tasks $2, 4, 8, 10, 13$ of Product 1 are assigned to workstation 1, while the tasks $2, 6, 8, 10, 11$ of Product 2 are assigned to workstation 3. During the disassembly of

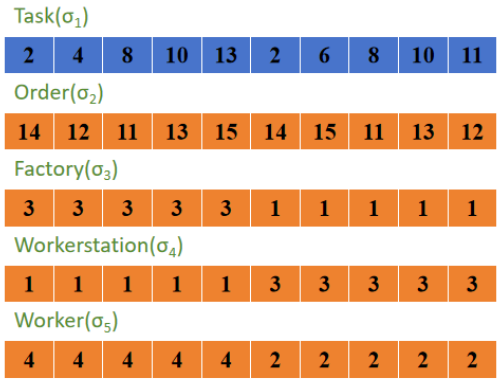


Fig. 5. An encoding scheme of AE.

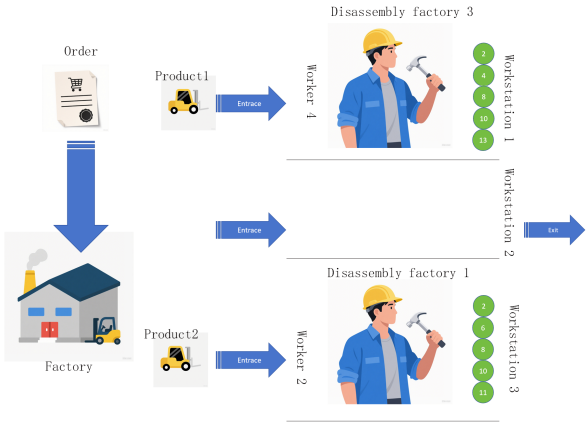


Fig. 6. Decoded assignment process.

the first product, Worker 4 is allocated to workstation 1 in disassembly factory 3, during the disassembly of the second product, Worker 2 is allocated to workstation 3 in disassembly factory 1.

The decoded assignment process is shown in Fig. 6.

C. A Coding Example

- **Step 1:** Generate the disassembly sequence based on the conflicts and precedence relationships between disassembly tasks. Retain tasks that meet the order requirements based on the parts obtained from task decomposition. Each task in the disassembly sequence is in a pending assignment state.
- **Step 2:** Assign the EOL products to the disassembly factories.
- **Step 3:** Sequentially assign the disassembly tasks to the workstations. If assigning a task to a workstation causes the workstation to exceed its cycle time constraint, the next workstation is ”turned on”, and the task is assigned to it.
- **Step 4:** According to the constraints, assign workers to the workstations of the disassembly factories.
- **Step 5:** Deliver the parts to the corresponding manufacturing factories to fulfill their orders.

The adopted encoding scheme directly represents the order allocation and worker scheduling decisions, which preserves the feasibility of candidate solutions and reduces unnecessary search in infeasible regions.

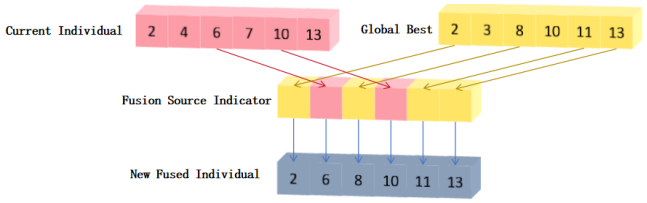


Fig. 7. Process of fusion.

D. Population Initialization

In AE, a population is typically composed of multiple *individuals*, each encoding a potential feasible solution. During population initialization, a disassembly task sequence of random length is first generated according to the disassembly incidence matrix. This sequence is then adjusted to satisfy conflict constraints and precedence relationships among disassembly tasks, guaranteeing that no conflicts or order violations occur in actual execution. Finally, several feasible scheduling schemes are derived from the validated task sequence. The initialization procedure is illustrated in Algorithm 2.

Algorithm 2 Initialize population

Input: population size n
Output: population P
1: **while** (i<n) **do**
2: Generate random task sequence σ_1
3: Select the task sequence that meets the order requirements based on σ_2
4: Adjust σ_1 to resolve conflict and precedence constraint
5: Apply the assignment strategy to generate σ_3, σ_4, σ_5
6: Add $\sigma(\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5)$ to P
7: $i = i + 1$
8: **end while**
9: **return** P

- **Step 1:** Obtain the disassembly sequence of the current individual and the global best individual.
- **Step 2:** Generate a new individual to store the fused solution.
- **Step 3:** Randomly generate a binary mask of the same length as the disassembly sequence, ensuring that a fixed proportion p of the values are 1. Then, from left to right, determine the value of each mask element. A value of 0 means the task is selected from the current individual and inserted into the new individual, while a value of 1 means the task is selected from the global best individual and inserted into the new individual. If the selected task already exists in the new individual, skip this task and continue until the new individual is filled.
- **Step 4:** The resulting disassembly sequence may not be directly feasible, so the new sequence must be repaired to satisfy conflict and precedence constraints before being added to the population.

E. Information Fusion and Random Disturbance

In the optimization process, the core mechanism of the AE lies in information fusion between individuals. During the fusion process, the system determines whether to perform information fusion based on a preset fusion probability r_{fusion}. Specifically, the system generates a random number, and if this number is greater than r_{fusion}, the individual undergoes information fusion with the global best solution. Conversely, if the generated random number is less than or equal to r_{fusion}, the individual retains its current state and proceeds to the next generation without modification. During the fusion, the decision variables of the current individual and the global best are combined according to a predefined crossover strategy to produce a new solution candidate. This procedure balances exploration and exploitation by incorporating high-quality information from the global best while preserving diversity in the population. The specific fusion process is illustrated in Fig. 7, and the detailed steps are as follows:

- **Step 1:** Acquire the feature sets of the current individual and the global optimal individual.
- **Step 2:** Construct a new individual as a carrier after information fusion.
- **Step 3:** Randomly generate a proportion threshold. If a random number is greater than this threshold, select a feature from the current individual. otherwise, select one from the global optimal individual. If the selected feature already exists in the new individual, skip it and select the next feature from the corresponding source until a unique feature is obtained and added to the new individual.
- **Step 4:** The newly generated individual may not meet the requirements, so its feature combination needs to be adjusted to satisfy the criteria of rationality and practicability.

During information fusion, based on preset fusion probability and a random number, an individual either fuses with the global best or retains its state. The Fusion process is illustrated in Algorithm 3.

Algorithm 3 Fusion process

Input: Parent solutions $\sigma^p = (\sigma_1^p, \sigma_2^p, \sigma_3^p, \sigma_4^p, \sigma_5^p)$
Output: Offspring solutions $\sigma^c = (\sigma_1^c, \sigma_2^c, \sigma_3^c, \sigma_4^c, \sigma_5^c)$
1: Select two parent individuals P_1, P_2
2: Randomly generate a crossover point k
3: Inherit the first k dimensions from P_1 into child C
4: Copy the remaining dimensions from P_2 into child C
5: Repair conflicts to ensure feasibility of solution
6: **return** σ^c

Load-balancing perturbation: To address resource idleness caused by excessively low task loads on some workstations, the algorithm first traverses all workstations to accurately locate the "light-load workstation" with the fewest tasks. Through the random disturbance mechanism, one workstation with a reasonable load range is randomly selected from the remaining workstations as the task receiver, and all tasks from the light-load workstation are migrated to this receiver. Subsequently,as shown in Fig. 8, the task allocation record of the original light-load workstation is deleted, and the numbers

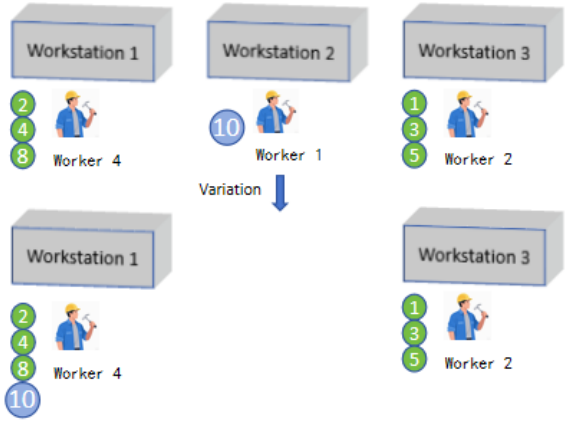


Fig. 8. Process of load balancing.

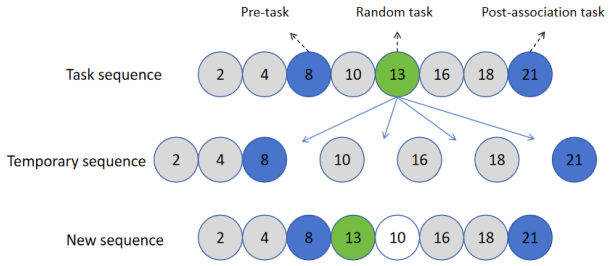


Fig. 9. Process of sequence constraint.

of the remaining workstations are calibrated for continuity to eliminate mapping confusion caused by number gaps. Finally, the disturbance is completed with the goal of load balancing, improving the overall resource utilization efficiency.

Sequence-constraint perturbation: As shown in Fig. 9, when random disturbance triggers sequence optimization, the algorithm randomly selects one target task from tasks with a "valid range length greater than 2", temporarily removes it from the current execution sequence, then randomly generates a new insertion position within the valid range, and reinserts the target task into the sequence. The entire process strictly adheres to task constraint relationships, breaking the limitation of local optimal solutions through sequence disturbance and exploring a better task execution sequence scheme.

Core-refactoring perturbation: To address redundancy or conflict issues in local task sequences, the random disturbance mechanism of the AE first locks one core task as the disturbance starting point through the incidence matrix, and simultaneously identifies and deletes all associated tasks that have direct conflicts with this core task to clear obstacles for sequence optimization.As shown in Fig. 10, based on the associated nodes corresponding to the core task in the incidence matrix, a stack-based search strategy is adopted to screen candidate tasks that meet the constraint conditions, and a new associated task subsequence is randomly generated. Finally, the resource allocation module is called to reallocate disassembly factories, manufacturing resources, and workstations for the newly generated subsequence. Through the full-process disturbance of "conflict deletion - sequence regeneration - resource reallocation", the collaborative optimization of task sequences and resource allocation is achieved, enhancing

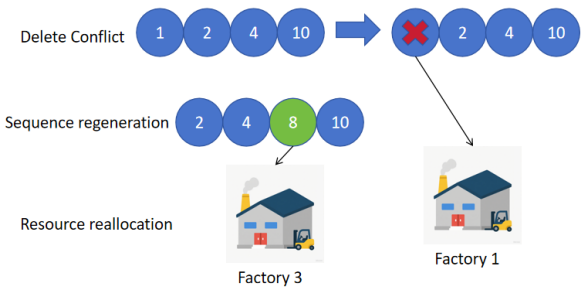


Fig. 10. Process of core refactoring.

the algorithm's ability for in-depth exploration of the solution space.

Furthermore, the information fusion mechanism integrates guidance from elite individuals with population-level information to balance exploitation and exploration, while the perturbation strategy adaptively adjusts solution structures to help the algorithm escape local optima and maintain solution feasibility under problem constraints.

V. Experimental Results and Analysis

This section explores the significance of order allocation and worker scheduling in multi-factory optimization and evaluates the performance of the proposed AE in solving the formulated model. We utilize IBM CPLEX to obtain the optimal solutions of the mathematical programming model and compare the performance of the AE against these solutions. All codes are implemented in IntelliJ IDEA2020.x64 and compiled using Java JDK21. The experiments are conducted on a personal computer equipped with an AMD Ryzen 5 4500U with Radeon Graphics (2.38 GHz) processor, 16.0 GB RAM (15.4 GB usable), and running Windows 11 operating system.

TABLE I Case information.

Case ID	Product				Num of tasks	Part quantity demand
	Washing machine	Oven	Treadmill	Radio		
1	0	1	0	0	15	9
2	1	1	0	1	58	25
3	1	1	1	1	75	32
4	2	2	0	1	86	40
5	2	2	1	1	103	47
6	3	2	1	1	116	53
7	3	2	1	2	146	63
8	3	2	2	2	163	70
9	3	3	2	3	208	89
10	4	4	4	3	270	118

TABLE II Disassembly factory parameters.

Factory ID	c_f^F	c_{fw}^L	c_{hf}^H
1	80	16 ~ 19	18 ~ 22
2	90	16 ~ 18	14 ~ 23
3	85	16 ~ 18	16 ~ 21
4	88	16 ~ 19	17 ~ 24

A. Test Instances

To ensure the comprehensiveness of the experimental study, four types of products are selected: washing machine [38], oven [39], treadmill [40] and radio [41]. These four products are combined into multiple product cases for evaluation. Table I shows the scale information of the combined cases. Table II displays the parameter settings of the disassembly factories, while Table III provides detailed information about the products and their parameter settings. For example, the oven consists of 22 subassemblies and incorporates 15 disassembly tasks. The profits and transportation costs of its subassemblies follow a normal distribution.

TABLE III Product parameters.

Product	Num of task	Num of subassembly	v_{mo}	c_{kmo}^T
Washing machine	13	15	$N(230, 4)$	$N(14, 2)$
Oven	15	22	$N(200, 6)$	$N(13, 5)$
Treadmill	17	18	$N(270, 5)$	$N(20, 4)$
Radio	30	29	$N(130, 4)$	$N(10, 2)$

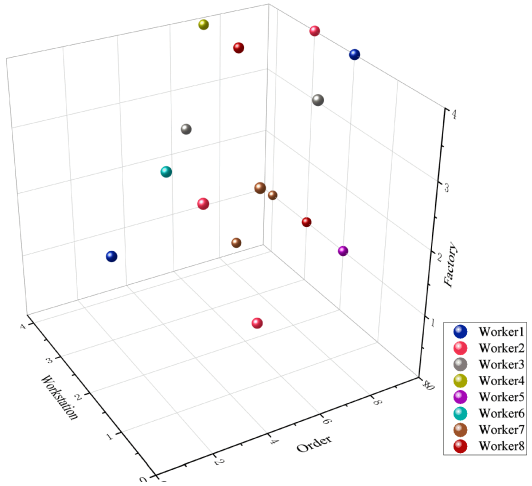


Fig. 11. Factory-workstation distribution diagram for order allocation and worker scheduling.

B. Performance Metrics

We use CPLEX to test the experimental cases, with parameters configured for 4 disassembly factories, 8 workers, and 32 orders. The factory numbers are $F = F_1, \ldots, F_4$, worker numbers are $w = w_1, \ldots, w_8$, and order numbers are $o = o_1, \ldots, o_{32}$. The "Objective" column represents the objective value corresponding to this disassembly and scheduling plan, and the "Best bound" indicates the upper bound of the current known solution. When the number of tasks is small, CPLEX can obtain good-quality solutions within the time limit. However, as the number of tasks increases, the efficiency of CPLEX decreases significantly, and it often fails to obtain the optimal solution within the runtime, providing only feasible solutions. The "Gap" column indicates the relative difference between the current best integer solution and the best linear relaxation solution, which is used to measure the quality of the solution and its potential for improvement. The "Computation time" column represents the time required to solve the problem. Within the 10800-second calculation time limit, CPLEX can still obtain near-optimal solutions for most

TABLE IV Comparison of Results Between CPLEX and AE.

Case ID	CPLEX			AE		
	Objective (Best bound)	Gap	Calculation time	Iterations / Population	Objective	Calculation time
1	361	0.00%	1.65s	600 / 500	361	7.9s
2	864	0.50%	20.49s	600 / 500	869	21.1s
3	1043	0.62%	43.21s	600 / 500	1047	32.1s
4	1384	0.29%	586.88s	600 / 500	1385	37.7s
5	1559	0.17%	6326.73s	600 / 500	1598	56.1s
6	1891 (1837)	2.94%	10800s	600 / 500	1856	70.7s
7	2324 (2221)	4.64%	10800s	600 / 500	2244	94.5s
8	2549 (2371)	7.51%	10800s	600 / 500	2408	114.1s
9	2915 (2653)	9.90%	10800s	600 / 500	2907	188.1s
10	4206 (3935)	6.91%	10800s	600 / 500	3967	297.3s

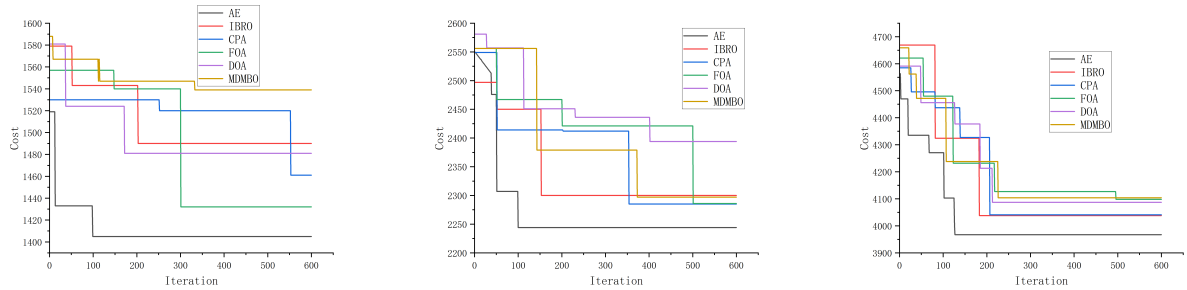


Fig. 12. Comparison of iteration costs between AE and benchmark algorithms under different problem scales.

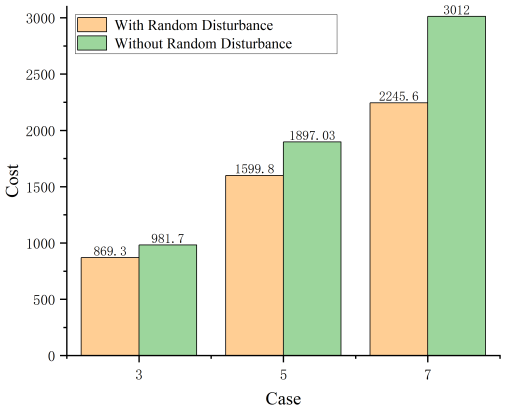


Fig. 13. Comparison of the impact of random perturbations on cost under different Cases.

small- and medium-sized cases, but larger gaps in some large-scale instances (e.g., Case 3) indicate that there is still room for improvement when dealing with more complex problems.

C. Case Study

Fig. 11 presents the decision logic of order-worker-factory allocation under the constraints of labor skill matching, workstation cycle time and cost minimization. Taking Order 10 as an example, its core attributes involve a high delivery priority, a multi-part demand for both oven and treadmill, and a strict time limit for disassembly. Against the backdrop of

TABLE V Comparison of results of AE with peers.

Case ID	AE	IBRO[15]	CPA[16]	FOA[17]	DOA[18]	MDMBO[19]
1	361	365	365	365	376	365
2	869	908	908	887	991	961
3	1047	1084	1071	1079	1096	1067
4	1385	1490	1461	1432	1481	1539
5	1598	1677	1690	1688	1816	1728
6	1856	1887	1881	1881	2021	1880
7	2244	2300	2285	2286	2394	2297
8	2408	2470	2425	2475	2582	2462
9	2907	2930	2937	2968	2964	2992
10	3967	4038	4041	4098	4087	4104

these attributes, factory 1 and factory 4 are first screened out as alternative disassembly factories for the reason that their activated workstations satisfy the skill matching requirements of multi-product disassembly and comply with the cycle time constraint (T≤500s), whereas factory 2 is ruled out due to the overload of its only active workstation. A subsequent cost-benefit trade-off is then conducted, with workstations 2, 3 and 4 of factory 1 and workstations 2 and 3 of factory 4 allocated to Order 10, and Workers 5, 7, 8 and Workers 1, 2 assigned to the corresponding workstations respectively. This allocation is determined by considering the low commuting cost of the selected workers and the need to maintain a balanced workstation load with a load rate ranging from 75% to 85%. In terms of optimization effect, this allocation scheme not only ensures the on-time delivery of Order 10 but also avoids the idleness of activated factories and workstations. Compared with the single-factory allocation scheme, the total cost of this integrated allocation scheme is reduced by 12%. For idle workers such as Worker 3 and Worker 4, the fundamental cause

lies in the mismatch between their professional disassembly skills, which are only applicable to radio disassembly, and the current order demand for oven and treadmill. This skill-demand mismatch further provides a reliable decision basis for the dynamic adjustment of the subsequent workforce in the multi-factory remanufacturing system.

D. Algorithm Performance Analysis

Table IV reports the comparative results of CPLEX and AE under different problem scales. While CPLEX is able to obtain zero optimality gap for small instances, its performance deteriorates noticeably as the problem size increases. For example, in Case 10, the optimality gap rises to 0.0691 and the computation time reaches the preset limit of 10,800 seconds. In contrast, AE is able to generate high-quality solutions for all instances within 300 seconds, indicating its suitability for larger-scale multi-factory scheduling scenarios.

Table V and Fig. 12 present the comparison between AE and five peer metaheuristic algorithms. These algorithms represent a variety of recently widely used meta-heuristic optimization methods with different search mechanisms. Some of them are newly proposed algorithms that excel in complex optimization problems, while others are widely adopted general-purpose population optimization algorithms. Using these representative algorithms as benchmarks, we can comprehensively evaluate the exploration-exploitation capabilities and robustness of the proposed AE algorithm. As a representative case, AE achieves an objective value of 2244 in Case 7, whereas the competing algorithms converge to higher-cost solutions. The convergence curves further show that AE reaches high-quality solutions earlier and maintains smoother convergence behavior, suggesting a more stable search process when order allocation and worker scheduling decisions are closely coupled. All comparative algorithms are run for 30 independent repetitions per test case to eliminate heuristic randomness, with their core parameters set according to the original literatures [15–19] and optimized for disassembly-scheduling scenarios. A five-level grid search is adopted for parameter tuning of all algorithms within the literature-recommended range, with the minimum average total cost of 10 preliminary experiments as the optimal evaluation index. Moreover, consistent basic settings (population size=500, max iterations=600) and the same termination criterion are applied to all algorithms to ensure comparative fairness.

Fig. 13 compares the results obtained with and without random perturbation under different order scales. In all considered cases, the solutions with random perturbation exhibit slightly lower total costs, and this difference becomes more apparent as the problem size increases. This observation suggests that random perturbation contributes to maintaining solution quality under varying order distributions.

In summary, the results indicate that jointly considering order allocation and worker scheduling leads to more consistent and efficient solutions across different production scales. The integrated decision-making framework enables effective coordination of labor and production resources, particularly in larger and more complex multi-factory settings.The perturbation strategies (load balancing, sequence constraint, core reconstruction) are aligned with the inherent characteristics of disassembly scheduling (e.g., task precedence, resource load balancing), enhancing the algorithm's adaptability to complex problem structures.

VI. Conclusions and Future Work

The results indicate that effective coordination of order allocation and worker scheduling is critical for improving cost efficiency and resource utilization in labor-constrained multi-factory remanufacturing systems. When factory operations are driven by actual order demand and available labor, integrated decision-making leads to more balanced production and more reliable order fulfillment. The AE enables efficient exploration of the resulting solution space. Comparative experiments with five other intelligent optimization algorithms validate the effectiveness and superiority of AE in solving this problem.

In future research, we plan to consider more real-world factors, such as dynamic order urgency and variable processing times, further improve the AE to handle more complex multi-factory remanufacturing scenarios, and explore intelligent optimization or reinforcement learning approaches for large-scale scheduling problems.

References

[1] H. Kazemi, M. Nourelfath *et al.*, "The multi-factory two-stage assembly scheduling problem," *Journal of Industrial Information Integration*, vol. 38, p. 100574, 2024.

[2] F. Güner, A. Görür *et al.*, "A constraint programming approach to a real-world workforce scheduling problem for multi-manned assembly lines with sequence-dependent setup times," *International Journal of Production Research*, vol. 62, no. 9, pp. 3212–3229, 2024.

[3] J. Wang, M. Zhou *et al.*, "Multiperiod asset allocation considering dynamic loss aversion behavior of investors," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 1, pp. 73–81, 2019.

[4] K. Zhang, R. Zhou *et al.*, "Transmission line component defect detection based on uav patrol images: A self-supervised hc-vit method," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 54, no. 11, pp. 6510–6521, 2024.

[5] W. Han, Q. Deng *et al.*, "Multi-objective evolutionary algorithms with heuristic decoding for hybrid flow shop scheduling problem with worker constraint," *Expert Systems with Applications*, vol. 168, p. 114282, 2021.

[6] Z. Zhang, X. Guo *et al.*, "Multi-objective discrete grey wolf optimizer for solving stochastic multi-objective disassembly sequencing and line balancing problem," in *Proc. 2020 IEEE Int. Conf. Systems, Man, and Cybernetics (SMC)*, 2020, pp. 682–687.

[7] X. Wang, M. Zhou *et al.*, "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, 2021.

[8] D. Hertel, G. Bräunig *et al.*, "Towards a green electromobility transition: a systematic review of the state of the art on electric vehicle battery systems disassembly," *Journal of Manufacturing Systems*, vol. 74, pp. 387–396, 2024.

[9] N. Rad and J. Behnamian, "Real-time multi-factory scheduling in industry 4.0 with virtual alliances," *Engineering Applications of Artificial Intelligence*, vol. 125, p. 106636, 2023.

[10] Y. Kose, E. Cevikcan *et al.*, "Game theory-oriented approach for disassembly line worker assignment and balancing problem with multi-manned workstations," *Computers & Industrial Engineering*, vol. 181, p. 109294, 2023.

[11] S. Qin, S. Zhang *et al.*, "Multiobjective multiverse optimizer for multi-robotic u-shaped disassembly line balancing problems," *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 2, pp. 882–894, 2024.

[12] L. Zhu, Z. Zhang *et al.*, "Multi-objective partial parallel disassembly line balancing problem using hybrid group neighbourhood search algorithm," *Journal of Manufacturing Systems*, vol. 56, pp. 252–269, 2020.

[13] H. Gao and Q. Zhang, "Alpha evolution: An efficient evolutionary algorithm with evolution path adaptation and matrix generation," *Engineering Applications of Artificial Intelligence*, vol. 137, p. 109202, 2024.

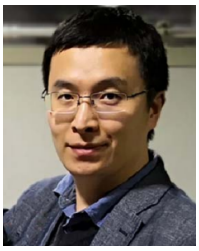
[14] X. Guo, L. Zhou *et al.*, "Multi-objective optimization of multi-product parallel disassembly line balancing problem considering multi-skilled workers using a discrete chemical reaction optimization algorithm," *Computers, Materials & Continua*, vol. 80, no. 3, pp. 4475–4496, 2024.
[15] N. Shukla, F. Shajin *et al.*, "Speech enhancement system using deep neural network optimized with battle royale optimization," *Biomedical Signal Processing and Control*, vol. 92, p. 105991, 2024.
[16] K. Ong, P. Ong *et al.*, "A carnivorous plant algorithm for solving global optimization problems," *Applied Soft Computing*, vol. 98, p. 106833, 2021.
[17] W. Pan, "A new fruit fly optimization algorithm: Taking the financial distress model as an example," *Knowledge-Based Systems*, vol. 26, pp. 69–74, 2012.
[18] H. Peraza-Vázquez, A. Peña-Delgado *et al.*, "A bio-inspired method for engineering design optimization inspired by dingoes hunting strategies," *Mathematical Problems in Engineering*, vol. 2021, pp. 1–19, 2021.
[19] G. Qin, X. Guo *et al.*, "Multi-objective discrete migrating birds optimizer solving multiple-product partial u-shaped disassembly line balancing problem," in *Proc. 29th Mediterranean Conference on Control and Automation (MED)*, 2021, pp. 822–827.
[20] J. Liang, S. Guo *et al.*, "Restart genetic flatworm algorithm for two-sided disassembly line balancing problem considering negative impact of destructive disassembly," *Journal of Cleaner Production*, vol. 355, p. 131708, 2022.
[21] C. Liu, Q. Zhu *et al.*, "An integrated optimization control method for remanufacturing assembly system," *Journal of Cleaner Production*, vol. 248, p. 119261, 2020.
[22] Y. Zhang, Z. Zhang *et al.*, "Improved whale optimisation algorithm for two-sided disassembly line balancing problems considering part characteristic indexes," *International Journal of Production Research*, vol. 60, no. 8, pp. 2553–2571, 2022.
[23] X. Guo, F. Guo *et al.*, "Modeling and optimization of multi-product human-robot collaborative hybrid disassembly line balancing with resource sharing," *IEEE Transactions on Computational Social Systems*, vol. 12, no. 1, pp. 1–15, 2025.
[24] E. Güler, C. Kalayci *et al.*, "Advances in partial disassembly line balancing: A state-of-the-art review," *Computers & Industrial Engineering*, vol. 196, p. 109898, 2024.
[25] Y. Tuo, Z. Zhang *et al.*, "Multimanned disassembly line balancing optimization considering walking workers and task evaluation indicators," *Journal of Manufacturing Systems*, vol. 72, pp. 263–286, 2024.
[26] X. Guo, L. Chen *et al.*, "Multi-factory disassembly process optimization considering worker posture," *IEEE Transactions on Computational Social Systems*, vol. 12, no. 1, pp. 16–30, 2025.
[27] F. Wang, Y. Wang *et al.*, "Joint optimization of order allocation and rack selection in the "parts-to-picker" picking system considering multiple stations workload balance," *Systems*, vol. 11, no. 4, p. 179, 2023.
[28] M. Martin, M. Hörger *et al.*, "Tactical order allocation in international manufacturing networks," *ZWF Zeitschrift für wirtschaftlichen Fabrikbetrieb*, vol. 119, no. 1-2, pp. 17–22, 2024.
[29] Q. Zhang, D. Lu *et al.*, "Design and optimization of dynamic reliability-driven order allocation and inventory management decision model," *PeerJ Computer Science*, vol. 10, p. e2294, 2024.
[30] Q. Liu, X. Li *et al.*, "A modified genetic algorithm with new encoding and decoding methods for integrated process planning and scheduling problem," *IEEE Transactions on Cybernetics*, vol. 51, no. 9, pp. 4429–4438, 2020.
[31] Y. Ren, C. Zhang *et al.*, "An asynchronous parallel disassembly planning based on genetic algorithm," *European Journal of Operational Research*, vol. 269, no. 2, pp. 647–660, 2018.
[32] X. Guo, C. Fan *et al.*, "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*, vol. 21, no. 2, pp. 1344–1356, 2023.
[33] T. Wei, X. Guo *et al.*, "A multi-objective discrete harmony search optimizer for disassembly line balancing problems considering human factors," *IEEE Transactions on Human-Machine Systems*, vol. 15, no. 2, pp. 20–35, 2025.
[34] Z. Zhao, S. Liu *et al.*, "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.
[35] Z. Cil, H. Öztop *et al.*, "Integrating distributed disassembly line balancing and vehicle routing problem in supply chain: Integer programming, constraint programming, and heuristic algorithms," *International Journal of Production Economics*, vol. 265, p. 109014, 2023.
[36] Y. Gu, M. Chen *et al.*, "A self-learning discrete salp swarm algorithm based on deep reinforcement learning for dynamic job shop scheduling problem," *Applied Intelligence*, vol. 53, no. 15, pp. 18 925–18 958, 2023.
[37] H. Zhang, D. Pham *et al.*, "Improved fruit fly algorithm for multi-objective disassembly line balancing problem considering learning effect," *International Journal of Artificial Intelligence and Green Manufacturing*, vol. 1, no. 2, pp. 51–62, 2025.
[38] 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.
[39] N. Boix and C. Favi, "Eco-design guidelines takeaways from the analysis of product repairability and ease of disassembly: a case study for electric ovens," *Procedia CIRP*, vol. 105, pp. 595–600, 2022.
[40] X. Niu, X. Guo *et al.*, "Hybrid disassembly line balancing of multi-factory remanufacturing process considering workers with government benefits," *Mathematics*, vol. 13, no. 5, p. 880, 2025.
[41] Q. Lu, Y. Ren *et al.*, "A hybrid metaheuristic algorithm for a profit-oriented and energy-efficient disassembly sequencing problem," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101828, 2020.

ChengGang Zheng received his B.S. degree in Communication Engineering from Changchun University of Science and Technology in 2024. He is currently a graduate student with the School of Information and Control Engineering at Liaoning Petrochemical University, China.

Ying Tang received the B.S. degree in electrical engineering and the M.S. degree in computer engineering from Northeastern University, Shenyang, China, in 1996 and 1998, respectively, and the Ph.D. degree in electrical engineering from New Jersey Institute of Technology, Newark, NJ, USA, in 2001.

She is currently a Full Professor and the Undergraduate Program Chair of Electrical and Computer Engineering at Rowan University, Glassboro, NJ, USA. Her work has been continuously supported by NSF, EPA, U.S. Army, FAA, DOT, private foundations, and industry. She has three U.S. patents and over 250 peer-reviewed publications, including 88 journal articles, two edited books, and six book/encyclopedia chapters. Her current research interests lie in the area of cyber-physical social systems, extended reality, adaptive and personalized systems, modeling and adaptive control for computer-integrated systems, and sustainable production automation.

Dr. Tang is the Founding Chair of the Technical Committee on Intelligent Solutions to Human-Aware Sustainability for IEEE Systems, Man, and Cybernetic, and the Technical Committee on Sustainable Production Automation for IEEE Robotic and Automation. She is currently an Associate Editor of IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, IEEE TRANSACTIONS ON INTELLIGENT VEHICLES, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, and *Springer's Discover Artificial Intelligence*.



Weitian Wang received the Ph. D. degree from the University of Science and Technology of China, Hefei, China, in 2016. He worked as a Post-Doctoral Fellow at Clemson University. He is currently an Associate Professor in the School of Computing and the Founder Director of the Collaborative Robotics and Smart Systems Laboratory (CRoSS Lab) at Montclair State University. His research focuses on collaborative robotics, artificial intelligence, smart systems, and their synergistic CRoSS-disciplinary applications in smart manufacturing and remanufacturing, healthcare, smart ecosystem, intelligent transportation, smart agriculture, daily assistance, data science, cybersecurity, and interactive learning. He is the recipient of IEEE R1 Technological Innovation Award.

Qi Kang is a Ph.D. candidate in Electrical and Computer Engineering department of New Jersey Institute of Technology. His research focuses the neural modulation with transcranial electrical stimulation and focused ultrasound stimulation. Prior to arriving at NJIT, he holds a Master of Science degree in Electrical and Computer Engineering from New York Institute of Technology in Old Westbury.