mathematics

MDPI

Article

Multi-Objective Optimization for a Partial Disassembly Line Balancing Problem Considering Profit and Carbon Emission

Wanlin Yang [1,2], Zixiang Li [1,2,*], Chenyu Zheng [1,2], Zikai Zhang [1,3], Liping Zhang [1,3] and Qiuhua Tang [1,3]

[1] Key Laboratory of Metallurgical Equipment and Control Technology of Ministry of Education, Wuhan University of Science and Technology, Wuhan 430081, China; yangwanlin1129@163.com (W.Y.); z173570209@163.com (C.Z.); zhangzikai@wust.edu.cn (Z.Z.); zhangliping@wust.edu.cn (L.Z.); tangqiuhua@wust.edu.cn (Q.T.)

[2] Hubei Key Laboratory of Mechanical Transmission and Manufacturing Engineering, Wuhan University of Science and Technology, Wuhan 430081, China

[3] Precision Manufacturing Institute, Wuhan University of Science and Technology, Wuhan 430081, China

* Correspondence: lizixiang@wust.edu.cn

Abstract: Disassembly lines are widely utilized to disassemble end-of-life products. Most of the research focuses on the complete disassembly of obsolete products. However, there is a lack of studies on profit and on carbon emission saved. Hence, this study considers the multi-objective partial disassembly line balancing problem with AND/OR precedence relations to optimize profit, saved carbon emission and line balance simultaneously. Firstly, a multi-objective mixed-integer programming model is formulated, which could optimally solve the small number of instances with a single objective. Meanwhile, an improved multi-objective artificial bee colony algorithm is developed to generate a set of high-quality Pareto solutions. This algorithm utilizes two-layer encoding of the task permutation vector and the number of selected parts, and develops two-phase decoding to handle the precedence relation constraint and cycle time constraint. In addition, the modified employed bee phase utilizes the neighborhood operation, and the onlooker phase utilizes the crossover operator to achieve a diverse population. The modified scout phase selects a solution from the Pareto front to replace the abandoned individual to obtain a new high-quality solution. To test the performance of the proposed algorithm, the algorithm is compared with the multi-objective simulated annealing algorithm, the original multi-objective artificial bee colony algorithm and the well-known fast non-dominated genetic algorithm. The comparative study demonstrates that the proposed improvements enhance the performance of the method presented, and the proposed methodology outperforms all the compared algorithms.

Keywords: disassembly line balancing; partial disassembly line; carbon emission; integer programming; artificial bee colony algorithm; multi-objective optimization

MSC: 90C11; 90C27; 90C29

Citation: Yang, W.; Li, Z.; Zheng, C.; Zhang, Z.; Zhang, L.; Tang, Q. Multi-Objective Optimization for a Partial Disassembly Line Balancing Problem Considering Profit and Carbon Emission. *Mathematics* **2024**, *12*, 1218. https://doi.org/10.3390/math12081218

Academic Editor: Ioannis G. Tsoulos

Received: 7 March 2024
Revised: 15 April 2024
Accepted: 16 April 2024
Published: 18 April 2024

1. Introduction

In recent years, many products have become obsolete due to their shorter lifecycle and rapid technological development [1,2]. These obsolete products might contain many hazardous substances, recyclable parts and recyclable raw materials. If not handled properly, these obsolete products will harm the environment and cause a waste of resources. However, today's society faces increasingly severe environmental pollution and resource shortages. Therefore, implementing recycling and remanufacturing of obsolete products to save resources and effectively reduce environmental hazards is an important measure towards establishing an ecological civilization and green development. As the important part in the recycling and remanufacturing process, disassembly systematically separates and selects high-value parts and components from waste products. To disassemble obsolete

products effectively, disassembly lines have been widely utilized. Nevertheless, disassembly lines might not be effective due to improper assignment of tasks to stations. Hence, the disassembly line balancing problem (DLBP) has drawn increasing attention from the business and academic communities.

In the complete disassembly of obsolete products, the number of stations, parts demand, disassembly cost, idle time equilibrium index, hazard degree and smoothing rate are taken as the main objectives [3–10]. These studies ignore the carbon emission generated in the operation, whereas excess carbon emission is an important factor contributing to the global temperature rise. Reducing carbon emission is crucial in achieving the Paris Agreement's goal of limiting global temperature increase to 2 °C or less, and ideally to 1.5 °C [11]. Therefore, taking carbon emission into consideration in the disassembly line is of practical significance.

In the process of disassembly, a product can be selectively disassembled. Namely, only the harmful and valuable parts need to be disassembled, and this type of disassembly line is referred to as a partial disassembly line [12]. Compared with the complete disassembly line, the partial disassembly line has major applications in industry. For instance, Wang, et al. [13] considered the uncertainty factors such as corrosion and deformation of scrap parts. A multi-objective disassembly line equilibrium model was then established to consider the partial destruction mode on a U-shape line. Liang, et al. [14] constructed a mixed-integer nonlinear programming model and developed a genetic algorithm and a tabu search algorithm to address the multi-parallel partial DLBP. Therefore, it is of practical significance to take partial disassembly lines into account in industrial practice.

For the above reasons, this study investigates the partial disassembly line balancing problem, taking into account profit and carbon emission, using an improved multi-objective artificial bee colony algorithm (IMOABC) to maximize the profit and saved carbon emission. Compared with existing studies, this paper differs from the literature in problem setting and optimization methodology. For instance, this paper differs from Li and Janardhanan [15] in both problem setting and methodology. This study considers the multi-objective partial DLBP with three objectives, whereas Li and Janardhanan [15] consider the U-shaped partial DLBP with only one objective. In addition, this study develops a Pareto algorithm which could obtain a set of Pareto solutions, whereas Li and Janardhanan [15] develop a one-objective algorithm. Meanwhile, compared with the multi-objective artificial bee colony algorithms in published papers, the proposed IMOABC also has some distinguishing features. For instance, compared with the multi-objective artificial bee colony algorithm in Saif, et al. [16], the proposed method advances two main modifications: (1) a new encoding and decoding scheme has been developed for this new problem, and (2) a new modified scout phase selects a solution from the Pareto front to replace the abandoned individual so as to obtain a new high-quality solution. In summary, this paper makes the following contributions: (1) A multi-objective partial disassembly line balancing model is established for optimizing profit, saved carbon emission and line balance. (2) In this paper, both AND and OR predecessors are considered, and the model can address the DLBP with only AND predecessors as well as the DLBP with AND predecessors and OR predecessors. (3) This paper develops a two-layer encoding of the task permutation vector and the number of selected parts, and an IMOABC is utilized and improved to achieve Pareto solutions. This algorithm proposes the following improvements: (1) in the employed bee phase, the neighborhood operation is utilized to obtain a new population to expand the search space of the algorithm; and (2) the algorithm records all the non-dominated solutions in the Pareto front and then replaces the obsolete individuals with the solutions in the Pareto front in order to improve the quality of the scout bees. Here, the Pareto front stores all the Pareto solutions in the search process.

The rest of the paper is organized as follows. Section 2 provides the literature review. Section 3 describes the problem under consideration and its mathematical formulation. Section 4 describes the developed algorithm along with the detailed encoding and decoding

process. Section 5 gives the computational results and a comparative study. Finally, Section 6 concludes this study, and provides future research directions.

2. Literature Review

The literature review consists of two parts: studies on a partial DLBP and studies on a DLBP considering carbon emission.

In practice, partial disassembly is usually carried out to obtain valuable components in order to achieve the goal of maximizing profit. Altekin, et al. [17] first defined and solved the profit-oriented partial DLBP, where different types of precedence relations in disassembly are considered and the mixed-integer programming formulation is developed. Ren, Yu, Zhang, Tian, Meng and Zhou [12] formulated a model for the profit-oriented partial DLBP to maximize profit. Bentaha, et al. [18] investigated a profit-oriented partial DLBP for hazardous parts and uncertain task processing time. Wang, et al. [19] developed a model for a stochastic partial DLBP that takes into consideration uncertainty, environmental protection and economic benefits. The model evaluates the number of stations, workload smoothness, energy consumption and disassembly profit. Wang, et al. [20] proposed a model for balancing a partially destructive disassembly line. This model aims to optimize the number of stations, the smoothness index, energy consumption and disassembly profit. Zhu, et al. [21] developed a multi-objective mathematical model for the partially parallel DLBP. Their study considered four optimization objectives: minimizing cycle time, minimizing the number of stations, minimizing the idleness index and minimizing disassembly resources. Li and Janardhanan [15] recently improved a new discrete cuckoo search algorithm to solve the profit-oriented U-shaped partial DLBP. Zhang, et al. [22] formulated a mixed-integer linear programming model to optimize three objectives: weighted line length, additional profit and hazard evaluation. Two constraint programming models were developed with different methodologies to achieve efficient solutions. Wang, Xi, Guo, Liu, Qin and Han [10] considered the requirement of multi-skilled workers and established a mathematical model to maximize the recovery profit.

The above papers neglect the carbon emission generated during the disassembly process, but on many occasions carbon emission cannot be neglected and it should be treated seriously to reduce environmental hazards. Igarashi, et al. [23] stated that higher CO_2 savings and lower part selection costs should be pursued during product disassembly. They proposed a multi-objective optimization model to reduce disassembly costs and increase CO_2 savings through environmental and economic part selection. Zhang, et al. [24] stated that if we do not consider carbon emission generated by factories during product disassembly, a lot of wasted resources will be generated. Therefore, a new multi-objective optimization model was proposed for the disassembly line, with the shortest disassembly time and the lowest carbon emission. Yang, et al. [25] concluded that there are more studies on disassembly line balance optimization in terms of both economic efficiency and disassembly time. However, less attention was paid to low carbon emission. Therefore, they proposed a multi-objective Drosophila optimization algorithm for optimizing the disassembly sequence of obsolete agricultural machinery with respect to low carbon. Cui, et al. [26] designed a linear mathematical programming model with profit maximization as the optimization objective. In addition, carbon savings act as a constraint to achieve environmental protection. Therefore, considering carbon emission in the disassembly line is of practical significance.

From the above literature review, it can be seen that there is a lack of research considering profit and carbon emission at the same time. Meanwhile, studies on the multi-objective artificial bee colony algorithm in DLBP are limited. Hence, this study contributes two aspects of research on DLBP: (1) a relatively new problem formulated with a multi-objective model; and (2) an improved multi-objective artificial bee colony algorithm.

3. Problem Description and Formulation

This section first provides the problem description and later formulates the problem being studied.

3.1. Problem Description

In the disassembly line, end-of-life products are disassembled into components on stations. Each station of the disassembly line has one employee or industrial robot to complete the disassembly tasks and collect the recycled components. Meanwhile, each station is equipped with a storage device to store the output (parts of the end-of-life products) of the disassembly line. Disassembly lines are the best option for disassembling complex products with many parts where employees can complete the tasks with simple training. Disassembly lines have higher efficiency, and, hence, they are increasingly applied in industry due to the growing number of end-of-life electromechanical products.

To improve the performance of the disassembly line, the DLBP is studied to determine the best assignment of tasks. The DLBP differs from the assembly line balancing problem (ALBP) in precedence relations, where the precedence relations in DLBP are much more complex. ALBP has only AND precedence relations, whereas DLBP has AND precedence relations, OR precedence relations and complex AND/OR precedence relations [27]. The predecessors in AND precedence relations are called AND predecessors, and the predecessors in OR precedence relations are called OR predecessors. A part could be disassembled only when (1) all of its AND predecessors have been disassembled, and (2) at least one of its OR predecessors is completed.

In the disassembly line, the parts of the obsolete product are divided into stations where they can reasonably be disassembled at each station under the cycle time constraint and precedence relation constraint. Figure 1 illustrates the precedence diagram for one instance with 10 real tasks/parts and one dummy task/part (A1). The dummy part A1 is utilized to transfer the complex AND/OR precedence relations into the simple AND precedence relations and OR precedence relations. Hence, real parts take time to disassemble, whereas dummy parts do not consume time. The disassembly times for part 1 to part 10 are 14, 10, 12, 18, 23, 16, 20, 36, 14 and 10, respectively. In this figure, the numbers within the cycle denote the parts/tasks, and the arrow between tasks denotes the precedence relations. If the parts are linked by an arc, there are OR precedence relations; otherwise, there are AND precedence relations. For instance, part 2 and part 3 are the OR predecessors of the dummy part A1, and part A1 can be operated when either part 2 or part 3 is removed. Part 7 is the AND predecessor of part 5 and part 6, and part 5 and part 6 can be operated only when part 7 has been removed.

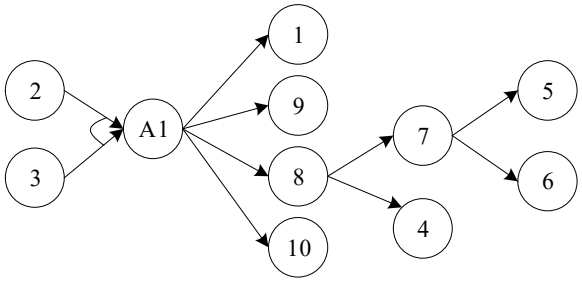

Figure 1. Precedence diagram of one instance.

Figure 2 illustrates optimal task assignment on the disassembly line to minimize the number of stations with a cycle time of 36 units. In the DLBP, there are two constraints that need to be considered: precedence relation constraint and cycle time constraint. The precedence relation constraint requires that (1) all AND predecessors of a task must be completed before this task, and (2) at least one OR predecessor must be completed before this task. For instance, part 8 is a predecessor of part 7, and, hence, part 8 must be removed before part 7. The cycle time constraint requires that the sum of operation time of tasks on

each station is less than or equal to the cycle time. For instance, the sum of operation time of tasks on station 1 is 10 + 14 + 10 = 34, which is less than the given cycle time.

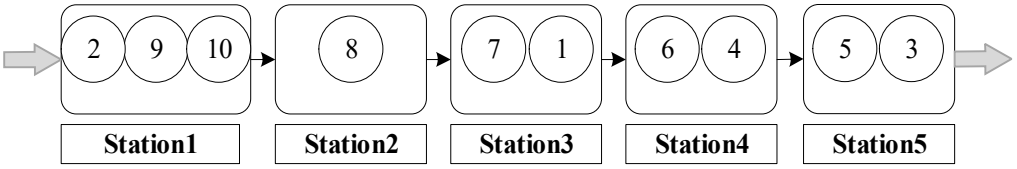

Figure 2. Task assignments on the disassembly line.

The input to an assembly line is the components, and the output is the final complete product. In contrast, the input to a disassembly line is an end-of-life product, and the output is the removed components. As the components of an end-of-life product might have different recycling values, it is necessary to select the valuable components to maximize profit. Hence, in many cases, the industry only disassembles the valuable components for remanufacturing or reuse [15]. At the same time, as the world pays more attention to the environment, it is also important to study the potential for reduced greenhouse gas emission by reusing parts or components of obsolete products. Nevertheless, the disassembly process also causes unavoidable greenhouse gas emission, and, hence, selecting parts that are beneficial for reducing greenhouse gas emission is necessary. Therefore, this study considers the partial DLBP to optimize profit, saved carbon emission and line balance.

3.2. Model Formulation

Before model formulation, there are several basic assumptions, as follows: (1) only one type of product is disassembled in this line; (2) each task could be completed within the cycle time; (3) the total operation time of tasks on one station should be equal to or less than the cycle time; (4) setup time is ignored; (5) the cost of disposing of the discarded parts is negligible. The symbols and decision variables of the model are introduced in Table 1 for clarification.

Table 1. Symbols and decision variables of the model.

Index:	
i, j	Part/task index; i, j = 1, 2, . . ., N, where N is the number of the parts/tasks;
m, n	Station index; m, n = 1, 2, . . ., M, where M is the number of allowed opened stations;
R_i	Recycling value of part i;
c_i	Cost of performing part i;
t_i	Operation time of part i;
ANDP(i)	Set of AND predecessors of task i;
ORP(i)	Set of OR predecessors of task i;
ORPT	Set of tasks which have OR predecessors;
C	Cost of running a station per unit time;
F	Fixed start-up cost of each station;
CHG_i	Saved greenhouse gas for disassembling part i and recycling part i;
$CHGs_i$	Saved greenhouse gas for recycling part i;
$CHGp_i$	Generated greenhouse gas for disassembling part i;
CT	Cycle time;
Decision variables	
x_i	1, if task i is performed; 0, otherwise;
y_{im}	1, if task i is allocated to the mth station; 0, otherwise;
z_m	1, if the mth station is open; 0, otherwise;
ST_m	Total time of station m;

On the basis of the above symbols and decision variables, the multi-objective model is constructed with expressions (1)–(10) for the partial DLBP under consideration. The objective in expression (1) maximizes the total profit, which is the total recovered value of

the removed part (the first term)—the total cost of removing parts (the second term)—the operating cost of the stations used (the third term)—start-up cost of the stations utilized (the fourth term). The objective in expression (2) maximizes the amount of saved carbon emission, and the objective in expression (3) optimizes the line balance of the disassembly line.

Constraint (4) requires that the total time of a station cannot be larger than the cycle time. If at least one task is assigned to a station, this station must be opened. Constraint (5) indicates that if one part is removed, this part must be removed on a station. Constraint (6) deals with the AND precedence relation constraint, indicating that a task could be assigned to the mth station only when all of its AND predecessors have been assigned to one station within station 1 to station m. Constraint (7) deals with the OR precedence relation constraint, indicating that a task can be assigned to the mth station only when at least one of its OR predecessors has been assigned to one station within station 1 to station m. Constraint (8) calculates the saved carbon emission for disassembling part i and recycling part i, which is the saved greenhouse gas for recycling part i (the first term)—generated greenhouse gas for disassembling part i (the second term). Constraint (9) calculates the completion time of each station. Constraint (10) restricts the values of decision variables.

$$MaxF_1 = \sum_{i=1}^{N} R_i \cdot X_i - \sum_{i=1}^{N} \sum_{m=1}^{M} c_i \cdot y_{im} - C \cdot \sum_{m=1}^{M} z_m \cdot CT - F \cdot \sum_{m=1}^{M} z_m \quad (1)$$

$$MaxF_2 = \sum_{i=1}^{N} CHG_i \quad (2)$$

$$MinF_3 = \sum_{m=1}^{M} z_m \cdot (CT - ST_m)^2 \quad (3)$$

$$\sum_{i=1}^{N} t_i \cdot y_{im} \leq z_m \cdot CT, \forall m \leq M \quad (4)$$

$$x_i = \sum_{m=1}^{M} y_{im}, \forall i \quad (5)$$

$$y_{im} \leq \sum_{n=1}^{m} y_{jn} \; \forall m, \; i, \; and \; j \in ANDP(i) \quad (6)$$

$$y_{im} \leq \sum_{j \in ORP(i)} \sum_{n=1}^{m} y_{jn} \; \forall m, \; i \in ORPT \quad (7)$$

$$CHG_i = CHGs_i - CHGp_i \quad (8)$$

$$\sum_{i=1}^{N} t_i y_{im} = ST_m \forall m \quad (9)$$

$$x_i, y_{im}, z_m \in \{0, 1\} \forall i, m \quad (10)$$

For the above-formulated model, objective (1) and objective (2) are linear objectives, objective (3) is a nonlinear objective, and all the constraints are linear constraints. Hence, the CPLEX solver could be utilized to optimize objective (1) or objective (2) only, and the mixed-integer nonlinear programming (MINLP) solver could be utilized to optimize objective (3).

4. Multi-Objective Artificial Bee Colony Algorithm

An artificial bee colony algorithm (ABC) is a swarm intelligence algorithm that simulates the honey-harvesting behavior of bees. The algorithm has been widely used in scheduling to find near-optimal solutions and has demonstrated excellent performance [28]. In the original multi-objective artificial bee colony algorithm (MOABC), the global search

ability is achieved by replacing individuals that fall into the local optimum with randomly generated individuals. However, the randomly generated individuals are usually of poor quality and are commonly dominated by incumbent individuals, leading to the poor performance of the algorithm. Therefore, this paper proposes an IMOABC to improve the algorithm's global and local search ability. The IMOABC adopts two improvements: (1) In the employed bee phase, a neighborhood operation is used to obtain new employed bees to expand the search space of the algorithm. (2) All non-dominated solutions in the Pareto frontier are recorded by this algorithm and replace the obsolete individuals to improve the quality of scout bees.

4.1. Artificial Bee Colony Algorithm

In solving one optimization problem using an artificial bee colony algorithm, each solution is a nectar source, and the solution's performance represents the amount of nectar. In a bee colony, there are three types of bees: employed bees, onlooker bees and scout bees. Employed bees explore the nectar source and pass the information about the nectar source to the onlooker bees; onlooker bees choose to explore the source based on the information passed by the employed bees. When the employed bees and the onlooker bees cannot continuously improve the population, one of the employed bees transforms into a scout bee, which randomly searches for nectar sources to ensure the variability of the population [6,29].

Algorithm 1 illustrates the main procedure of the ABC, where Pop_size is the population size. Specifically, with the algorithm's parameters and instance data as input, this algorithm starts with initializing the swam randomly. Afterwards, the algorithm sequentially executes the employed bee phase, the onlooker bee phase, and the scout bee phase to evolve until the algorithm termination conditions are met. In the employed bee phase, a neighborhood operation is used to generate a new neighborhood solution for each individual, and the current solution is replaced when the neighborhood solution performs better. In the onlooker bee phase, a solution is selected based on roulette wheel selection. Then, the neighborhood operation is used to generate a new neighborhood solution, and the current solution is replaced when the neighborhood solution performs better. In the scout bee phase, when a solution has not been improved in limited consecutive iterations, a new solution is randomly generated and replaces the current solution. Hence, ABC generates new neighborhood solutions for exploitation through neighborhood operation in the employed bee phase and the onlooker bee phase and emphasizes the exploration by replacing the current solution with a randomly generated new solution in the scout bee phase.

Algorithm 1: Procedure of artificial bee colony algorithm

Input: Algorithm parameters and instance data

Step 1
% Initializing the swam
Generate a population with a Pop_size number of individuals randomly;
% Employed bee phase
For p:=1 to Pop_size **do**

Step 2
Generate a new neighborhood solution of individual p with neighborhood operation;
Replace current solution when the neighborhood solution performs better;
Endfor
% Onlooker bee phase
For p:=1 to Pop_size **do**

Step 3:
Select one solution based on roulette wheel selection;
Generate a new neighborhood solution with neighborhood operation;
Replace the current solution when the neighborhood solution performs better;
Endfor
% Scout bee phase

Step 4:
Select one individual that has not been improved in limited consecutive iterations;
If exists, replace this individual with a new solution generated randomly;

Step 5: Iteratively execute Step 2, Step 3 and Step 4 until the termination condition is reached

Output: Best solution

4.2. Encoding and Decoding

This paper adopts a two-layer encoding of the task permutation vector and number of selected parts (length). The number of selected parts is a number within [1, N], and it determines the number of selected parts. The task permutation vector is the permutation of a set of tasks, and the task in the front position is assigned first. Length determines the number of selected parts, and the decoding process terminates immediately if the number of allocated tasks (or selected parts) reaches the length.

Based on the two-layer encoding of the task permutation vector and number of selected parts (length), a two-phase decoding procedure is developed in this paper to solve the multi-objective disassembly line balancing problem. The decoding process is divided into two phases. In Phase I, the task permutation vector is transformed into a feasible task permutation. The assignable tasks are first added to the set of assignable tasks, and later one task is selected based on the task permutation vector and added to the feasible task permutation. This process is repeated until the complete feasible task permutation is generated. In Phase II, the tasks to be disassembled are first selected based on the number of selected parts. If the number of selected parts is 3, only the first 3 tasks of the feasible task permutation are selected in this phase. Then, the tasks to be disassembled are assigned stations under the cycle time constraint. In other words, Phase I deals with the precedence relation constraint, and Phase II deals with the cycle time constraint. The main decoding process is illustrated as follows.

Phase I: Converting the task permutation vector into the feasible task permutation
Step 1: Add the assignable task to the set of assignable tasks (a task is assigned if all AND predecessors and at least one OR predecessor of the task have been assigned); Step 2: Select the task that is in the front position of the task permutation vector from the set of assignable tasks and place it in the sequence of feasible tasks; Step 3: Repeat steps 1–2 until the complete feasible task permutation is generated;
Phase II: Select the tasks to be disassembled based on the number of selected parts and obtain detailed task assignments based on cycle time constraint
Step 4: Determine the selected parts based on the number of selected parts (length); Step 5: Open a new station; Step 6: Select the first unassigned part in the selected parts; Step 7: If the part could be completed within the cycle time, assign the part to the current station and execute Step 8; otherwise, execute Step 5. Step 8: If all the selected parts have been removed, terminate the decoding process; otherwise, perform Step 6.

For the instance illustrated in Figure 1, where the task permutation vector in the encoding is [2, 5, 7, 8, 9, 10, 3, 1, 6, 4] and the number of removed parts in the encoding is 3, Figure 3 illustrates the process of the two-phase decoding process. Since the task permutation vector may not satisfy the precedence relation constraint, Phase I first transforms the task permutation vector into the feasible task permutation. Specifically, following the process of Phase I, the procedure of achieving the first three tasks in the feasible task permutation is as follows. For the first task, the set of assignable tasks contains tasks {2, 3}, and task 2 is assigned to the feasible task permutation because of its former position in the task permutation vector. For the second task, the set of assignable tasks contains tasks {1, 3, 8, 9, 10}, and task 8 is assigned to the feasible task permutation because of its former position in the task permutation vector. For the third task, the set of assignable tasks contains tasks {1, 3, 4, 7, 9, 10}, and task 7 is assigned to the feasible task permutation because of its former position in the task permutation vector. The above steps are repeated until a feasible task permutation [2, 8, 7, 5, 9, 10, 3, 1, 6, 4] is formed. In Phase II, since the number of selected parts is 3, the first 3 tasks in the feasible task permutation are selected for execution, which are 2, 8 and 7. Subsequently, these selected tasks are assigned to the

3 stations sequentially while satisfying the cycle time constraint to achieve the final task assignment scheme.

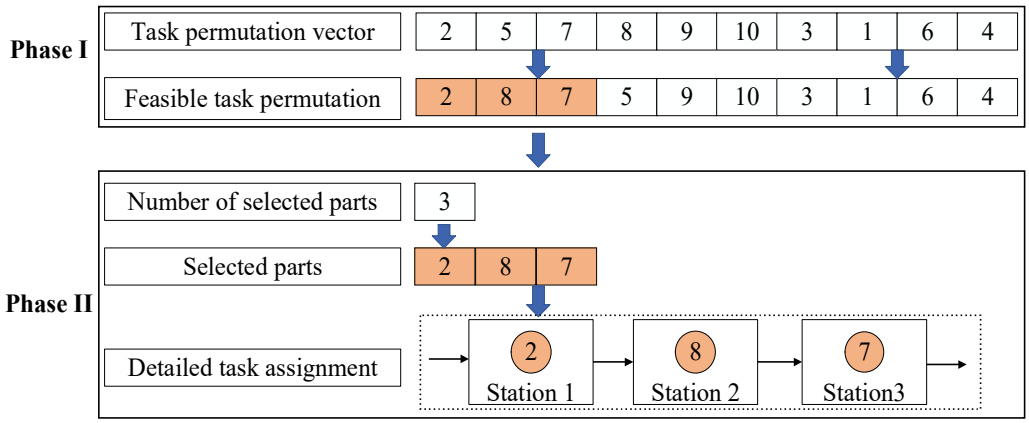

Figure 3. Two-phase decoding method.

It should be noted that the task permutation vector in the encoding does not take account of the precedence relation constraint. Phase I transforms the task permutation vector into feasible task permutations; that is, Phase I deals with the precedence relation constraint, and any task permutation vector can be formed into a feasible task permutation through Phase I. Namely, the two-phase decoding can generate a feasible task allocation scheme based on the two-layer encoding of the task permutation vector and the number of selected parts (length). Therefore, there is no need to consider the precedence relation constraint or worry about avoiding the generation of infeasible solutions during the proposed algorithm's evolution.

4.3. Main Procedure of the Proposed Methodology

Random generation is adopted to obtain the differentiated initial population. Subsequently, the IMOABC executes the employed bee phase, the onlooker bee phase and the scout bee phase sequentially until the algorithm termination condition is satisfied. An update of the Pareto front is attempted in the population initialization, employed bee phase and onlooker bee phase. That is, a comparison is made between any of the newly generated individuals and the solutions in the Pareto front. Supposing that the population is empty or the new solution is not dominated by any solution in the Pareto set, the new solution is added to the Pareto front, and the solutions in the Pareto front that are dominated by the new solution are deleted. The execution process of the employed bee phase, onlooker bee phase and scout bee phase is described in detail below.

In the employed bee phase, the neighborhood operation is employed to perform a local search, and a greedy mechanism is also used as an acceptance criterion to determine whether or not to accept this neighborhood solution. Since the problem is multi-objective, the algorithm utilizes rank and crowding distance to evaluate the individuals. Here, the ranks and crowding distances of individuals are achieved based on the fast non-dominated sorting approach by Deb, et al. [30]. In this case, individuals with lower ranks or with larger crowding distances and the same rank perform better.

In the onlooker bee phase, the onlooker bees will adopt one specific selection strategy to choose nectar sources based on the information provided by the employed bees. The traditional artificial bee colony algorithm usually utilizes the roulette wheel method. In contrast, in the IMOABC, a binary tournament strategy is used to select an individual. Namely, an individual with a lower rank or the larger crowding distance with the same rank is selected. At the same time, the best and non-repeating Pop_size solutions are selected from the existing solutions and neighborhood solutions to form a new population, thus retaining the better ones to accelerate the evolution of the population. Specifically, the algorithm calculates the rank and crowding distance of all individuals based on a fast

non-dominated sorting approach, and then the Pop_size individuals with lower ranks or with larger crowding distances and the same rank are selected to form a new population.

In the scout bee phase, the MOABC uses randomly generated solutions to replace the individuals that do not improve in successive limit iterations. However, the probability of randomly generating high-quality new solutions in such an ample solution space is low. In this paper, a Pareto frontier is adopted to store the Pareto frontier solutions found during the search process, and when a solution does not improve in successive limit iterations, a randomly selected solution from the Pareto frontier is used to replace the current solution.

The specific algorithmic flow of the IMOABC is as follows (Algorithm 2).

Algorithm 2: Main procedure of the IMOABC

Step 1:	% Initialization of the population Generate a population with a Pop_size number of individuals randomly based on the objective functions and constraints in the mathematical model; Update the Pareto front;
Step 2:	% Employed bee phase Generate neighborhood solutions for Pop_size individuals based on neighborhood operation; Calculate rank and crowding distances, and select individuals with low rank or large crowding distances with the same rank; Update the Pareto front;
Step 3:	% Onlooker phase **For** any individual, Select an individual based on a binary tournament; Select another individual randomly and crossover with the current individual to form a new individual; **Endfor** Combine the original population with the new population and select the best Pop_size individuals to form the new population; Update the Pareto front;
Step 4:	% Scout phase When a solution does not improve in limited consecutive iterations, randomly select a solution from the Pareto front to replace the current solution;
Step 5:	Iteratively execute Step 2, Step 3 and Step 4 until the termination condition is reached, and output the final Pareto frontier solutions.

4.4. Proposed Neighborhood Structures and Crossover Operations

In order to obtain a new diverse population, the IMOABC adopts neighborhood operation in the employed bee phase and crossover operation in the onlooker bee phase. The neighborhood operation in the employed bee phase only changes one single individual, while the crossover operation in the onlooker bee phase combines two individuals to form a new one. This paper utilizes the crossover operation to combine two individuals to further expand the search space.

The algorithm adopts the swap and partially mapped crossover operator for the task permutation vector.

Specifically, the swap operation randomly selects two different positions on the operation sort vector and then exchanges the values of these two positions. The partially mapped crossover operator combines two parents to generate two offspring. As the partially mapped crossover operator has been widely utilized, a detailed description is omitted here, and readers can refer to Goldberg and Lingle Jr [31] for a detailed description. Figure 4 illustrates the swap operation and the partial match crossover operation for the task permutation vector.

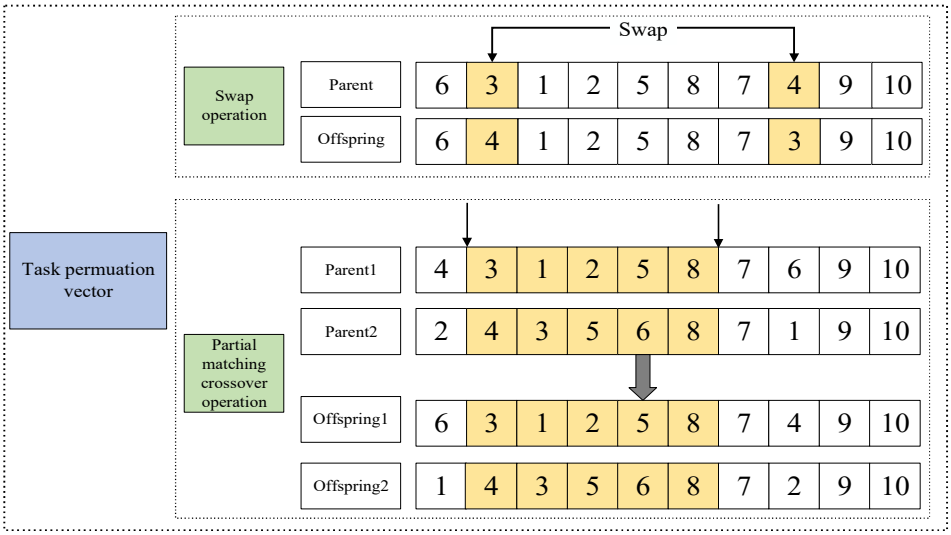

Figure 4. Neighborhood structure and crossover operation for the task permutation vector.

For the number of selected parts, the algorithm adopts a random mutation operation and simulated binary crossover operator. Specifically, the random mutation operation randomly generates a positive integer between 1 and N (the number of parts or operations) and replaces the current number of selected parts with this positive integer. The simulated binary crossover operator combines two parents to generate two offspring. For a detailed description, refer to Deb and Agrawal [32].

5. Experimental Results and Analysis

This section tests the performance of the proposed method. Firstly, Section 5.1 describes the solved instances, the compared algorithms and the evaluation indicators, and provides the selected parameter combination of each implemented algorithm through parameter calibration. Subsequently, Section 5.2 evaluates the performance of the proposed method by comparing it with other multi-objective algorithms. Finally, Section 5.3 provides a real case study.

5.1. Experimental Design

In order to validate the performance of the proposed IMOABC, based on the cases in the literature [15], this paper generates a total number of 21 instances. Since each instance contains a different cycle time, there are a total number of 87 cases. Table 2 illustrates the number of tasks, cycle time, and the number of cases for each instance. Specifically, for the problem under consideration, a different cycle time corresponds to different cases. For example, instance P11 has 2 cases, P11-10 and P11-94, where the number after the dash denotes the cycle time. POR10 means that there is an OR predecessor in this instance. It should be noted that the precedence relation, task operation time, and cost-related data for each case are taken directly from the references and that the carbon emission-related data in this paper are generated randomly. Due to space limitations, this paper cannot show all the data, and the related test cases are available upon request.

To test the performance of the proposed algorithm, it is compared with three other algorithms: the MOABC [16], the multi-objective simulated annealing algorithm (MOSA) [33], and the fast non-dominated genetic algorithm (NSGA-II) [30]. The termination condition for all algorithms is the number of solution evaluations, and the algorithms terminate after conducting the decoding scheme 100,000 times. To better evaluate the performance of the algorithms, each algorithm is solved 10 times independently for all cases, and the results are converted to the hyper-volume ratio (HVR), one-dimensional Epsilon metric (I_ε^1), and inverse generation distance (IGD) to compare the algorithms' performance [34].

Table 2. Tested instances.

Instances	Number of Tasks	Cycle Time	Number of Cases
POR10	10	36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55	20
P25	25	18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37	20
P7	7	7	1
P8	8	20	1
P9	9	7	1
P11	11	10, 94	2
P21	21	15	1
P25	25	16	1
P28	28	216	1
P29	29	30	1
P32	32	2357	1
P35	35	41	1
P45	45	62	1
P53	53	2806	1
P70	70	168, 170, 173, 179, 182	5
P75	75	46, 47, 49, 50, 52	5
P83	83	3985, 5048, 5833, 6842, 7571, 8412, 8898, 10,816	8
P89	89	15, 150	2
P94	94	201, 301	2
P111	111	5755, 7520, 8847, 10,027, 10,743, 11,378, 11,570, 17,067	8
P148	148	85, 89, 91, 95	4

The HVR is calculated with $\mathrm{HVR} = \frac{\mathrm{volume}\left(\mathrm{U}_{i=1}^{\mathrm{size(S)}} x_i\right)}{\mathrm{volume}\left(\mathrm{U}_{j=1}^{\mathrm{size(P)}} x_j\right)}$, where $\mathrm{volume}\left(\mathrm{U}_{i=1}^{\mathrm{size(S)}} x_i\right)$ is the volume of one Pareto front (S) by one algorithm and $\mathrm{volume}\left(\mathrm{U}_{j=1}^{\mathrm{size(P)}} x_j\right)$ is the volume of the true Pareto front (P). The I_ε^1 metric is obtained using $I_\varepsilon^1 = I_\varepsilon(S,P) = \max_{x^2}\min_{x^1}\max_j\left(f_j(x^1) - f_j(x^2)\right)$, and it calculates the minimum distance between one Pareto front (S) and the true Pareto front (P). For these two indicators, a larger value of HVR indicates a better performance of the Pareto front, and a smaller value of I_ε^1 indicates a better performance of the Pareto front. When the HVR value approaches 1.0, the Pareto front S is close to the true Pareto front P. The true or near-true Pareto front is needed to calculate the HVR and I_ε^1. This study utilizes the Pareto front achieved by running all the implemented algorithms 10 times as the near-true Pareto front. Meanwhile, the set of maximum values of the objectives in the near-true Pareto front is regarded as the reference point. For detailed descriptions of the two indicators, please refer to the cited papers [35,36]. The IGD is calculated with $\mathrm{IGD}(S,P) = \frac{\sum_{x\in P}\min_{y\in S}\mathrm{dis}(x,y)}{|P|}$, where the Euclidean distance between a point x in P and a point y in S is denoted by $\mathrm{dis}(x,y)$. In general, a smaller value of IGD denotes a better the performance of the algorithm.

Since the parameters of the algorithm have a significant impact on the algorithm's performance, it is necessary to set the parameters of the algorithm to the appropriate values before testing the algorithm. Therefore, this paper uses a full factorial design of experiments to calibrate the parameters, following [6,35]. Ten different instances are solved 10 times by combining all parameter values. After the experiments are completed, a multi-factor analysis of variance (ANOVA) [37] is performed, in which the parameters are used as control variables, and the mean value of 1-HVR in one run is used as the response variable. For better analysis, this paper replaces the HVR with 1-HVR when calibrating parameters, and the parameter with the smaller 1-HVR value is preferred. Table 3 demonstrates the tested values of parameters and selected parameter values in the tested algorithms. All algorithms were programmed with Python 3.8 programming language and run on a computer configured with an Intel(R) Core (TM) i5-8265U CPU @1.60 GHz.

Table 3. Algorithm parameters and values.

The Name of the Algorithm	Algorithm Parameters	Parameter Values	Selected Parameter Values
IMOABC	Population size	60, 80, 100, 120	100
	Iteration number before replacing solutions	100, 200	200
MOABC	Population size	60, 80, 100, 120	100
	Iteration number before replacing solutions	100, 200	200
MOSA	Initial temperature	0.5, 1.0	1.0
	Cooling rate	0.95, 0.98	0.95
	Iteration number before a temperature change	5, 10	5
NSGA-II	Population size	60, 80, 100, 120	100
	Crossover probability	0.6, 0.8, 1.0	1.0
	Mutation probability	0.6, 0.8, 1.0	1.0

5.2. Comparative Study

To validate the performance of the proposed algorithm, Tables 4–6 show the results obtained by the IMOABC and the other three algorithms for the HVR, I_{ε}^{1}, and IGD metrics. Column 1 of each table represents the instances and columns 2 to 5 represent the values of metrics obtained by the different algorithms solving each instance with several cases. In this table, each metric value is the average result of solving one instance 10 times.

Table 4. Comparison of HVR results for algorithms.

Problem	HVR			
	NSGA-II	IMOABC	MOSA	MOABC
P10	0.950	1.000	0.680	0.948
P25	0.818	0.834	0.569	0.795
P7	1.000	1.000	0.506	1.000
P8	1.000	1.000	0.828	1.000
P9	1.000	1.000	1.000	1.000
P11	1.000	1.000	0.431	1.000
P21	1.000	1.000	0.713	0.992
P25	0.000	0.000	0.000	0.739
P28	0.924	0.925	0.607	0.616
P29	0.985	0.993	0.823	0.973
P32	1.000	0.996	0.873	0.875
P35	0.908	0.939	0.665	0.867
P45	0.440	0.568	0.443	0.150
P53	1.000	1.000	1.000	1.000
P70	0.789	0.793	0.609	0.701
P75	0.817	0.833	0.798	0.784
P83	0.964	0.966	0.913	0.958
P89	0.743	0.766	0.721	0.686
P94	0.926	0.932	0.911	0.890
P111	0.862	0.865	0.857	0.866
P48	0.865	0.879	0.810	0.854
Avg	0.857	**0.871**	0.703	0.843

Best in bold.

Table 4 also provides the average HVR values of the four algorithms in the last row. In decreasing order of the average HVR value, algorithms are ranked as IMOABC, NSGA-II, MOABC and MOSA. Namely, the IMOABC outperforms NSGA-II, MOABC and MOSA as the larger value of HVR indicates superior performance. However, there is a significant difference in the values of I_{ε}^{1} and IGD, and the average values cannot provide meaningful information and are, therefore, not calculated. Nevertheless, the IMOABC outperforms the other algorithms in most instances in terms of the two indicators. For instance, in

terms of I_ε^1, the IMOABC outperforms NSGA-II, MOABC and MOSA in 18 instances and the IMOABC is outperformed by NSGA-II, MOABC and MOSA in only 3 instances. In terms of IGD, the IMOABC outperforms NSGA-II, MOABC and MOSA in 15 instances and is outperformed by NSGA-II, MOABC and MOSA in only 6 instances. In short, the computational results demonstrate that the IMOABC outperforms NSGA-II, MOABC and MOSA in terms of the three evaluation metrics.

Table 5. Comparison of I_ε^1 results for algorithms.

Problem	I_ε^1			
	NSGA-II	**IMOABC**	**MOSA**	**MOABC**
P10	0.216	0.378	31.366	3.351
P25	0.779	0.620	20.450	3.404
P7	0.000	0.000	6.350	0.000
P8	0.000	0.000	11.300	0.000
P9	0.000	0.000	0.000	0.000
P11	0.000	0.000	11.540	0.000
P21	0.000	0.000	20.200	1.600
P25	0.000	0.000	9.800	2.900
P28	4.310	4.300	31.590	12.950
P29	8.300	6.530	27.890	10.550
P32	82.080	82.080	1339.565	464.645
P35	35.500	32.380	128.300	47.670
P45	11.540	11.560	28.100	16.280
P53	1.080	1.080	777.570	32.740
P70	317.078	292.463	410.531	363.448
P75	135.658	136.29	135.042	114.728
P83	4334.826	3908.571	6767.489	5137.114
P89	26.560	21.105	87.615	42.355
P94	159.838	137.033	181.767	168.090
P111	14,156.024	12,849.111	14,197.295	13,595.020
P148	471.565	469.195	497.128	487.740

Table 6. Comparison of IGD results for algorithms.

Problem	IGD			
	NSGA-II	**IMOABC**	**MOSA**	**MOABC**
P10	0.010	0.080	20.955	0.756
P25	1.143	1.131	18.769	2.752
P7	0.000	0.000	6.530	0.000
P8	0.000	0.000	8.095	0.000
P9	0.000	0.000	0.000	0.000
P11	0.000	0.000	7.522	0.000
P21	0.100	0.300	27.915	4.032
P25	0.000	0.000	31.608	1.297
P28	9.076	5.507	31.531	14.984
P29	4.712	3.977	17.115	7.706
P32	392.914	631.466	27,831.97	698.093
P35	18.593	19.555	63.520	25.234
P45	17.795	15.327	36.860	21.175
P53	1.000	1.000	825.845	49.335
P70	154.882	154.021	216.173	208.005
P75	53.235	60.993	63.558	49.686
P83	10,415.780	7809.136	22,097.650	11,297.190
P89	13.847	13.326	65.475	21.178
P94	117.243	115.881	130.240	132.335
P111	222,699.600	180,080.400	195,807.900	210,688.700
P148	217.848	217.719	191.943	208.449

In order to further compare the performance of the algorithms statistically, this section describes a Friedman test performed on the algorithms. In the Friedman test, the algorithm type is selected as the control variable, and the average 1-HVR, I_{ε}^{1} and IGD values in one run are selected as the response variables. Figure 5 demonstrates the average ranking and 95% confidence intervals of the algorithm's metric values. Table 7 shows the Friedman table evaluating these four algorithms using a pairwise multiple comparison test. As shown in Figure 5, the IMOABC performs the best in terms of 1-HVR. The results show that there is a statistically significant difference between IMOABC vs. MOSA and IMOABC vs. MOABC, and there is no statistically significant difference between NSGA-II vs. IMOABC. For I_{ε}^{1}, IMOABC and NSGA-II perform similarly, but IMOABC is slightly better. The results also show that there is a statistically significant difference between IMOABC vs. MOSA and IMOABC vs. MOABC, and that there is no statistically significant difference between NSGA-II vs. IMOABC. Regarding IGD, IMOABC and NSGA-II perform similarly, but IMOABC is slightly better. Furthermore, the results show that there is a statistically significant difference between IMOABC vs. MOSA and IMOABC vs. MOABC, and that there is no statistically significant difference between NSGA-II vs. IMOABC. In summary, the computational study demonstrates that the proposed IMOABC statistically outperforms MOSA and MOABC, and produces a competitive performance in comparison with NSGA-II. As a consequence, it can be concluded that the proposed method is effective for the multi-objective partial DLBP under consideration.

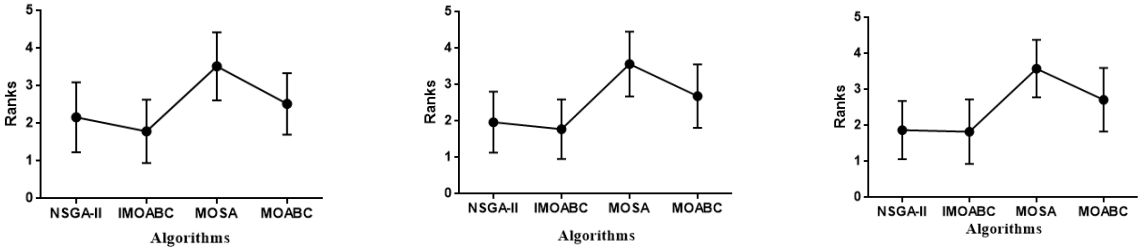

Figure 5. Mean rankings and 95% confidence intervals for algorithmic indicator values. (**a**) Ranks of algorithms in terms 1-HVR; (**b**) Ranks of algorithms in terms I_{ε}^{1}; (**c**) Ranks of algorithms in terms IGD.

Table 7. Friedman table of four algorithms.

Multiple Comparisons Test	1-HVR		I_{ε}^{1}		IGD	
	Rank Sum Diff	Significant	Rank Sum Diff	Significant	Rank Sum Diff	Significant
NSGA-II vs. IMOABC	25.00	No	13.00	No	3.000	No
NSGA-II vs. MOSA	−91.00	Yes	−107.0	Yes	−114.5	Yes
NSGA-II vs. MOABC	−24.00	No	−48.00	Yes	−56.50	Yes
IMOABC vs. MOSA	−116.0	Yes	−120.0	Yes	−117.5	Yes
IMOABC vs. MOABC	−49.00	Yes	−61.00	Yes	−59.50	Yes
MOSA vs. MOABC	67.00	Yes	59.00	Yes	58.00	Yes

5.3. A Real Case Study

This section optimizes a disassembly line of used drum washing machines for an electronic disassembly company in the northeast region of China. This drum washer disassembly line contains 28 tasks with a cycle time of 30, where the cost of running a station per unit of time is 0.05 and the fixed start-up cost of each station is 1. The operation time of the tasks, the recycling value of parts, the costs of performing tasks, as well as the value of saved carbon emission by part and the value of carbon emission generated by the disassembled parts are shown in Table 8. Meanwhile, Figure 6 demonstrates the precedence relations between the tasks in this disassembly line.

Table 8. Information for the drum washing machine.

Part Number	Part Name	t	R	C	CHG_s	CHG_p
1	Countertop assembly	7	9	5.1	3.3	0.8
2	Bottom decorative panel	19	8	2.8	13.2	0.9
3	Housing front plate assembly	15	4	3.0	17.1	0.5
4	Back cover	5	16	9.6	13.5	0.2
5	Shell assembly	12	14	5.6	6.3	0.7
6	Distribution box assembly	10	10	4.6	1.3	0.0
7	Main control board	8	15	6.4	39.0	0.5
8	Main control board control assembly	16	9	3.5	13.8	0.5
9	Electromagnetic door locks	2	10	7.9	8.4	0.3
10	Pressure switch	6	18	3.3	8.2	1.0
11	Power supply	21	6	5.8	8.2	0.5
12	Seal assembly	10	16	2.3	7.5	0.1
13	Inner cylinder	9	4	8.2	8.8	0.4
14	Outer cylinder assembly	4	17	3.7	1.4	0.1
15	Pulley	14	13	6.6	7.3	0.6
16	Belt	7	5	8.2	6.2	0.1
17	Electrical machinery	14	8	8.8	3.2	0.4
18	Front weight	17	7	6.6	24.3	0.7
19	Counterweight	10	2	7.7	33.3	0.2
20	Suspension spring shock absorption	16	9	2.8	7.1	0.7
21	Water inlet solenoid valve	1	10	9.6	5.6	0.7
22	Inlet pipe assembly	9	18	8.2	8.0	0.5
23	Water storage tank	25	3	6.7	4.2	0.8
24	Reservoir—outer tube	14	14	6.1	16.8	0.5
25	Outer cylinder—drainage pump pipe	14	12	4.0	5.2	0.4
26	Collector valve pressure switch tube	2	8	6.6	37.7	0.3
27	Drainage pump	10	5	8.4	35.6	0.7
28	Drainage pipe assemblies	7	13	4.2	7.2	0.4

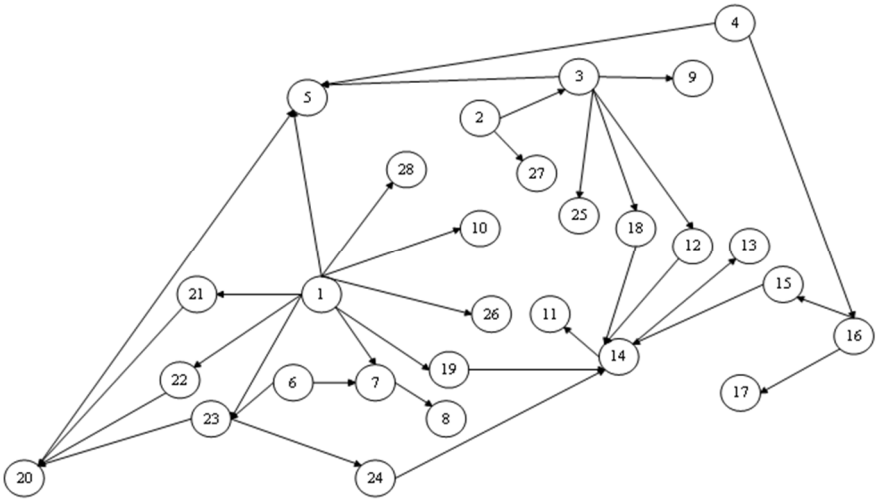

Figure 6. Disassembly precedence relations of the parts.

Based on on-site research, the disassembly line mainly adopts manual scheduling to make decisions on the allocation of tasks, resulting in the irrational allocation of dismantling tasks. Meanwhile, the decision-making for the disassembly line did not consider the profit of the disassembly line or the carbon emission saved. In order to reduce the disassembly cost and improve the efficiency of this disassembly line, this section solves the DLBP of the drum washing machine and optimizes the disassembly line using the proposed method. Meanwhile, the performance of the proposed method is verified by comparing it with NSGA-II. Specifically, the algorithm generates the task permutation vector and number of selected parts based on the two-layer encoding method proposed in this paper. Afterwards,

based on the decoding procedure, the objective values are calculated utilizing expressions (1), (2) and (3). Here, expression (1) maximizes the value of the profit, expression (2) maximizes the value of saved carbon emission and expression (3) minimizes the value of the line balance. In order to facilitate the observation of the spatial distribution, F_3 is changed to $-F_3$ and all the objectives now prefer the larger values.

Figure 7 shows the multi-perspective spatial distribution of the Pareto front on the objective function in a single run of the two algorithms. From the figure, it can be seen that the Pareto solutions achieved by the IMOABC dominate the Pareto solutions by NSGA-II. Meanwhile, it is also observed that the IMOABC obtains more Pareto solutions and that the Pareto solutions obtained by the IMOABC have the better distribution. Hence, it could be concluded that the proposed IMOABC outperforms NSGA-II in this real case.

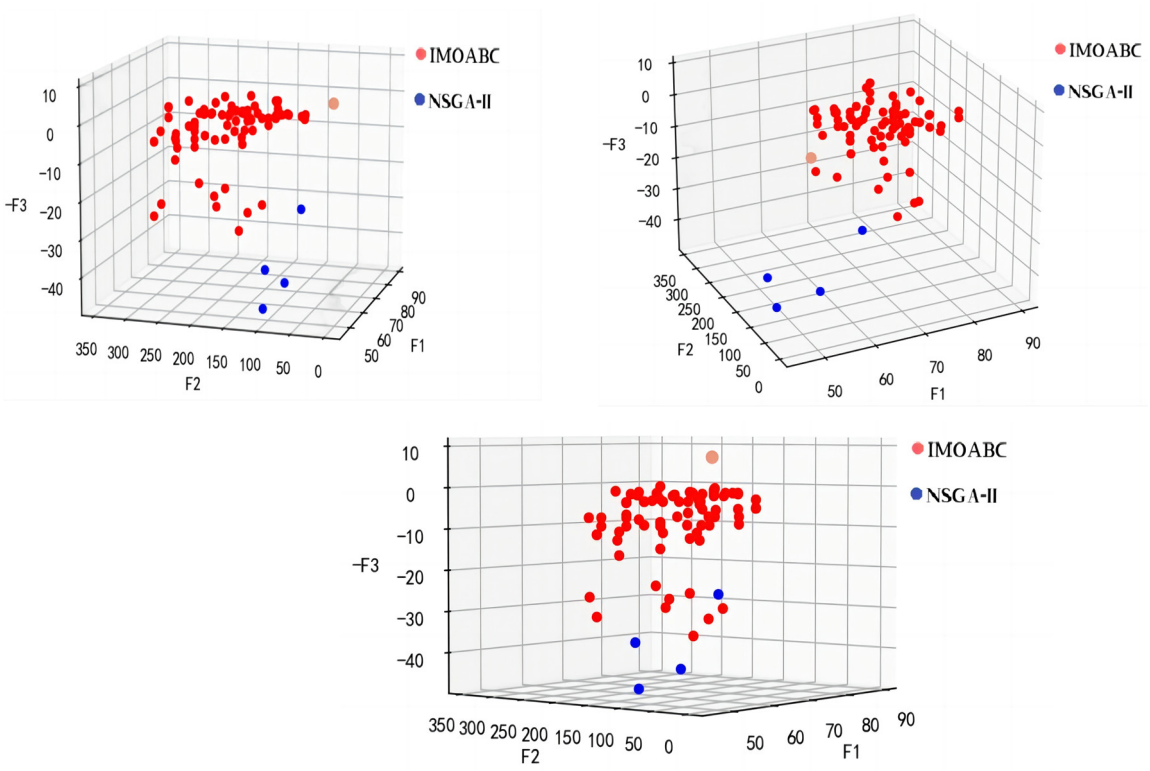

Figure 7. Multi-perspective spatial distribution of the results on the objective function in a single run.

6. Conclusions and Future Research

Disassembly is an essential process when recycling end-of-life products. As there are limited studies on profit and saved carbon emission in disassembly lines, this study considers the multi-objective partial disassembly line balancing problem with AND/OR precedence relations to optimize profit, saved carbon emission and line balance simultaneously. Firstly, a single multi-objective mixed-integer programming model is formulated. This model can solve small-size instances to optimize total profit or saved carbon emission solely with the CPLEX solver. It could also solve the small-size instance to optimize the line balance with the mixed-integer nonlinear programming (MINLP) solver. Meanwhile, the IMOABC is developed to solve the identified multi-objective problem effectively by achieving a set of high-quality Pareto solutions. In this algorithm, two-layer encoding of the task permutation vector and the number of selected parts is developed, where the task permutation vector determines the priority of the disassembled parts. Two-phase decoding based on the two-layer encoding is developed, where the first layer deals with the precedence relation constraint to achieve feasible task permutation and the second layer determines the disassembled parts and deals with the cycle time constraint to achieve one feasible solution. Meanwhile, in the proposed method, a modified employed bee phase utilizes neighborhood operation and the onlooker phase utilizes a crossover operator to

achieve a diverse population. The modified scout phase replaces the abandoned solution with a solution selected from the Pareto front to bring a new high-quality solution into the population. All these improvements favor the proposed method achieving a proper balance between exploration and exploitation, and, hence, a single algorithm with a strong global search capacity and local search capability is realized.

In order to validate the performance of the proposed algorithm, it is compared with MOSA, MOABC and NSGA-II. The comparative study demonstrates that the improvements enhance the performance of the proposed multi-objective algorithm due to its outperforming the MOABC. Meanwhile, the proposed algorithm outperforms the compared algorithms, and, as a consequence, the proposed algorithm could effectively solve the problem under consideration. Additionally, the proposed algorithm solves a real case (disassembly line of a used drum washing machine) and it is compared with NSGA-II. The case study demonstrates that the proposed multi-objective algorithm can obtain a set of high-quality Pareto solutions for the decision-makers to select, and the proposed algorithm demonstrates superior performance in comparison with NSGA-II.

The findings of this study help the enterprise to achieve higher profits and sustainable development, and this proposed algorithm could be embedded into a decision system for the disassembly line manager to determine the disassembled parts. Future studies might apply the developed algorithm to other related disassembly line balancing problems. For instance, different layouts of disassembly lines could be considered, such as U-shaped disassembly lines and two-sided disassembly lines. Other realistic optimization objectives could be included, such as human factors.

Author Contributions: Conceptualization, L.Z.; Methodology, W.Y., Z.L. and C.Z.; Software, W.Y.; Validation, W.Y. and Z.L.; Formal analysis, Z.Z.; Investigation, L.Z.; Resources, Q.T.; Data curation, Z.Z. and Q.T.; Writing—original draft, W.Y.; Writing—review & editing, W.Y. and Z.L.; Visualization, C.Z.; Project administration, C.Z.; Funding acquisition, Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This project is partially supported by the National Natural Science Foundation of China under grants 62173260 and 62303358.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, Z.; Kucukkoc, I.; Zhang, Z. Iterated local search method and mathematical model for sequence-dependent U-shaped disassembly line balancing problem. *Comput. Ind. Eng.* **2019**, *137*, 106056. [CrossRef]
2. Wang, K.; Li, X.; Gao, L. Modeling and optimization of multi-objective partial disassembly line balancing problem considering hazard and profit. *J. Clean. Prod.* **2019**, *211*, 115–133. [CrossRef]
3. Koc, A.; Sabuncuoglu, I.; Erel, E. Two exact formulations for disassembly line balancing problems with task precedence diagram construction using an AND/OR graph. *IIE Trans.* **2009**, *41*, 866–881. [CrossRef]
4. McGovern, S.M.; Gupta, S.M. Ant colony optimization for disassembly sequencing with multiple objectives. *Int. J. Adv. Manuf. Technol.* **2005**, *30*, 481–496. [CrossRef]
5. Kalayci, C.B.; Sabry Shaaban, D.; Abdul Salam Darwish, D.; Polat, O.; Gupta, S.M. A variable neighbourhood search algorithm for disassembly lines. *J. Manuf. Technol. Manag.* **2015**, *26*, 182–194. [CrossRef]
6. Kalayci, C.B.; Gupta, S.M. Artificial bee colony algorithm for solving sequence-dependent disassembly line balancing problem. *Expert Syst. Appl.* **2013**, *40*, 7231–7241. [CrossRef]
7. McGovern, S.M.; Gupta, S.M. A balancing method and genetic algorithm for disassembly line balancing. *Eur. J. Oper. Res.* **2007**, *179*, 692–708. [CrossRef]
8. Tuo, Y.; Zhang, Z.; Wu, T.; Zeng, Y.; Zhang, Y.; Junqi, L. Multimanned disassembly line balancing optimization considering walking workers and task evaluation indicators. *J. Manuf. Syst.* **2024**, *72*, 263–286. [CrossRef]
9. Tuncel, E.; Zeid, A.; Kamarthi, S. Solving large scale disassembly line balancing problem with uncertainty using reinforcement learning. *J. Intell. Manuf.* **2012**, *25*, 647–659. [CrossRef]
10. Wang, J.; Xi, G.; Guo, X.; Liu, S.; Qin, S.; Han, H. Reinforcement learning for Hybrid Disassembly Line Balancing Problems. *Neurocomputing* **2024**, *569*, 127145. [CrossRef]

11. Wu, F.; Huang, N.; Zhang, F.; Niu, L.; Zhang, Y. Analysis of the carbon emission reduction potential of China's key industries under the IPCC 2 °C and 1.5 °C limits. *Technol. Forecast. Soc. Chang.* **2020**, *159*, 120198. [CrossRef]
12. Ren, Y.; Yu, D.; Zhang, C.; Tian, G.; Meng, L.; Zhou, X. An improved gravitational search algorithm for profit-oriented partial disassembly line balancing problem. *Int. J. Prod. Res.* **2017**, *55*, 7302–7316. [CrossRef]
13. Wang, K.; Gao, L.; Li, X. A multi-objective algorithm for U-shaped disassembly line balancing with partial destructive mode. *Neural Comput. Appl.* **2020**, *32*, 12715–12736. [CrossRef]
14. Liang, W.; Zhang, Z.; Yin, T.; Zhang, Y.; Wu, T. Modelling and optimisation of energy consumption and profit-oriented multi-parallel partial disassembly line balancing problem. *Int. J. Prod. Econ.* **2023**, *262*, 108928. [CrossRef]
15. Li, Z.; Janardhanan, M.N. Modelling and solving profit-oriented U-shaped partial disassembly line balancing problem. *Expert Syst. Appl.* **2021**, *183*, 115431. [CrossRef]
16. Saif, U.; Guan, Z.; Liu, W.; Wang, B.; Zhang, C. Multi-objective artificial bee colony algorithm for simultaneous sequencing and balancing of mixed model assembly line. *Int. J. Adv. Manuf. Technol.* **2014**, *75*, 1809–1827. [CrossRef]
17. Altekin, F.T.; Kandiller, L.; Ozdemirel, N.E. Profit-oriented disassembly-line balancing. *Int. J. Prod. Res.* **2008**, *46*, 2675–2693. [CrossRef]
18. Bentaha, M.L.; Dolgui, A.; Battaïa, O.; Riggs, R.J.; Hu, J. Profit-oriented partial disassembly line design: Dealing with hazardous parts and task processing times uncertainty. *Int. J. Prod. Res.* **2018**, *56*, 7220–7242. [CrossRef]
19. Wang, K.; Li, X.; Gao, L.; Garg, A. Partial disassembly line balancing for energy consumption and profit under uncertainty. *Robot. Comput. Integr. Manuf.* **2019**, *59*, 235–251. [CrossRef]
20. Wang, K.; Li, X.; Gao, L.; Li, P. Energy consumption and profit-oriented disassembly line balancing for waste electrical and electronic equipment. *J. Clean. Prod.* **2020**, *265*, 121829. [CrossRef]
21. Zhu, L.; Zhang, Z.; Guan, C. Multi-objective partial parallel disassembly line balancing problem using hybrid group neighbourhood search algorithm. *J. Manuf. Syst.* **2020**, *56*, 252–269. [CrossRef]
22. Zhang, Y.; Zhang, Z.; Zeng, Y.; Wu, T. Constraint programming for multi-line parallel partial disassembly line balancing problem with optional common stations. *Appl. Math. Model.* **2023**, *122*, 435–455. [CrossRef]
23. Igarashi, K.; Yamada, T.; Gupta, S.M.; Inoue, M.; Itsubo, N. Disassembly system modeling and design with parts selection for cost, recycling and CO_2 saving rates using multi criteria optimization. *J. Manuf. Syst.* **2016**, *38*, 151–164. [CrossRef]
24. Zhang, L.; Zhao, X.; Ke, Q.; Dong, W.; Zhong, Y. Disassembly Line Balancing Optimization Method for High Efficiency and Low Carbon Emission. *Int. J. Precis. Eng. Manuf. Green Technol.* **2019**, *8*, 233–247. [CrossRef]
25. Yang, Y.; Yuan, G.; Zhuang, Q.; Tian, G. Multi-objective low-carbon disassembly line balancing for agricultural machinery using MDFOA and fuzzy AHP. *J. Clean. Prod.* **2019**, *233*, 1465–1474. [CrossRef]
26. Cui, X.; Guo, X.; Zhou, M.; Wang, J.; Qin, S.; Qi, L. Discrete Whale Optimization Algorithm for Disassembly Line Balancing with Carbon Emission Constraint. *IEEE Robot. Autom. Lett.* **2023**, *8*, 3055–3061. [CrossRef]
27. Güngör, A.; Gupta, S.M. Disassembly line in product recovery. *Int. J. Prod. Res.* **2002**, *40*, 2569–2589. [CrossRef]
28. Çil, Z.A.; Li, Z.; Mete, S.; Özceylan, E. Mathematical model and bee algorithms for mixed-model assembly line balancing problem with physical human–robot collaboration. *Appl. Soft Comput.* **2020**, *93*, 106394. [CrossRef]
29. Kalayci, C.B.; Hancilar, A.; Gungor, A.; Gupta, S.M. Multi-objective fuzzy disassembly line balancing using a hybrid discrete artificial bee colony algorithm. *J. Manuf. Syst.* **2015**, *37*, 672–682. [CrossRef]
30. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
31. Goldberg, D.E.; Lingle, R., Jr. Alleles, loci, and the traveling salesman problem. In Proceedings of the 1st International Conference on Genetic Algorithms, Pittsburgh, PA, USA, 24–26 July 1985; pp. 154–159.
32. Deb, K.; Agrawal, R.B. Simulated binary crossover for continuous search space. *Complex Syst.* **1995**, *9*, 115–148.
33. Liang, J.; Guo, S.; Du, B.; Li, Y.; Guo, J.; Yang, Z.; Pang, S. Minimizing energy consumption in multi-objective two-sided disassembly line balancing problem with complex execution constraints using dual-individual simulated annealing algorithm. *J. Clean. Prod.* **2021**, *284*, 125418. [CrossRef]
34. Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C.M.; da Fonseca, V.G. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Trans. Evol. Comput.* **2003**, *7*, 117–132. [CrossRef]
35. Nilakantan, J.M.; Li, Z.; Tang, Q.; Nielsen, P. Multi-objective co-operative co-evolutionary algorithm for minimizing carbon footprint and maximizing line efficiency in robotic assembly line systems. *J. Clean. Prod.* **2017**, *156*, 124–136. [CrossRef]
36. Ciavotta, M.; Minella, G.; Ruiz, R. Multi-objective sequence dependent setup times permutation flowshop: A new algorithm and a comprehensive study. *Eur. J. Oper. Res.* **2013**, *227*, 301–313. [CrossRef]
37. Montgomery, D.C. *Design and Analysis of Experiments*, 10th ed.; J. Wiley: Hoboken, NJ, USA, 2020.