mathematics

MDPI

Article

Multi-Objective Discrete Brainstorming Optimizer to Solve the Stochastic Multiple-Product Robotic Disassembly Line Balancing Problem Subject to Disassembly Failures

Gongdan Xu [1], Zhiwei Zhang [2], Zhiwu Li [1,*], Xiwang Guo [2], Liang Qi [3] and Xianzhao Liu [4]

[1] Institute of Systems Engineering, Macau University of Science and Technology, Taipa, Macau 999078, China
[2] College of Computer and Communication Engineering, Liaoning Petrochemical University, Fushun 113001, China
[3] Computer Science and Technology, Shandong University of Science and Technology, Qingdao 266590, China
[4] Hitachi Building Technology (Guangzhou) Co., Ltd., Guangzhou 510670, China
* Correspondence: zwli@must.edu.mo

Abstract: Robots are now widely used in product disassembly lines, which significantly improves end-of-life (EOL) product disassembly efficiency. Most of the current research on disassembly line balancing problems focuses on decomposing one product. More than one product can be disassembled concurrently, which can further improve the efficiency. Moreover, uncertainty such as the depreciation of EOL products, may result in disassembly failure. In this research, a stochastic multi-product robotic disassembly line balancing model is established using an AND/OR graph. It takes the precedence relationship, cycle constraint, and disassembly failure into consideration to maximize the profit and minimize the energy consumption for disassembling multiple products. A Pareto-improved multi-objective brainstorming optimization algorithm combined with stochastic simulation is proposed to solve the problem. Furthermore, by conducting experiments on some real cases and comparing with four state-of-the-art multi-objective optimization algorithms, i.e., the multi-objective discrete gray wolf optimizer, artificial bee colony algorithm, nondominated sorting genetic algorithm II, and multi-objective evolutionary algorithm based on decomposition, this paper validates its excellent performance in solving the concerned problem.

Keywords: disassembly failure; machine learning; multiple product disassembly; robotic disassembly line balancing problem

MSC: 90-08

Citation: Xu, G.; Zhang, Z.; Li, Z.; Guo, X.; Qi, L.; Liu, X. Multi-Objective Discrete Brainstorming Optimizer to Solve the Stochastic Multiple-Product Robotic Disassembly Line Balancing Problem Subject to Disassembly Failures. *Mathematics* **2023**, *11*, 1557. https://doi.org/10.3390/math11061557

Academic Editor: Ivan Lorencin

Received: 23 February 2023
Revised: 15 March 2023
Accepted: 16 March 2023
Published: 22 March 2023

1. Introduction

With the rapid updating and iteration of industrial technology, end-of-life (EOL) industrial products in urgent need of treatment have been produced, and many environmental problems have resulted. Recycling [1] is a crucial measure to solve the harm caused by EOL products. The value of recycling and reusing EOL products is twofold: (1) they benefit environmental protection; (2) they can save precious resources [2] and reduce production costs. In the process of recycling, disassembly plays a connecting role [3,4]. Disassembly breaks down EOL products into various parts that can then be used for other purposes. Because of human being's operational flexibility, most disassembly operations are performed manually, but the disassembly cost is high [5]. In recent years, with the rapid development of robotic-related technologies, they have been applied to various application fields, including disassembly [5–9]. Robot disassembly technology can not only reduce the risk of disassembly to operators, but also improve the disassembly efficiency and ensure the quality of parts [10]. Therefore, it is of great significance to study the robotic disassembly line balancing (RDLB) problem for recycling EOL. There are extensive studies

on this problem. For instance, Çil et al. [6] investigated the use of RDLB to disassemble an EOL product. Liu et al. [9] solved the RDLB of single product disassembly by taking the disassembly form participated in by robots as the entry point. However, the above studies only focused on disassembling a single product. Meeting the diversified disassembly needs of customers, it is usually necessary to disassemble multiple products at the same time [11]. Hence, it is novel and meaningful to study the robot disassembly problem considering multi-product disassembly forms.

Disassembly line balancing (DLB) is a multi-objective optimization problem whose goal is to shorten the work cycle time as much as possible and quickly obtain the maximum profit under the condition of rapid separation of hazardous parts [6,12–17]. DLB is divided into complete, partial, and selective disassembly [18,19]. Complete disassembly denotes dividing all components of a product into the smallest components. Partial disassembly denotes separating some components of a product from other components. Selective disassembly is the purposeful disassembly of components of a product based on a certain target. In this paper, the complete disassembly of EOL products is considered. In DLB, the balancing rate estimates the workload among workstations, which is one of the main indices for evaluating the performance to solve DLB problems [20]. Therefore, this paper proposes a multi-objective multi-product robotic disassembly line balancing problem (MMRDP) for maximizing total profit and its balancing rate, as well as minimizing total energy consumption. It is crucial to make a good tradeoff among these objectives.

To perform DLB, researchers have explored some efficient methods. Liu et al. [8] and Fang et al. [10] improved the multi-objective discrete bee algorithm and problem-specific bi-criterion evolutionary algorithm, respectively, to solve the RDLB and mixed-model DLB under the condition of robot disassembly form. In this work, the proposed MMRDP is solved by the improved brainstorming optimization (BSO) [21] algorithm. Because of its superior performance, BSO has attracted scholars from various fields to solve various optimization problems. For example, Duan et al. [22,23] successively solved the optimization problem of the direct current brushless motor and Loney's solenoid by proposing a predatory quantum-behaved BSO algorithm and a quantum-behaved BSO, respectively. On the basis of the above research and considering the complexity of MMRDP, this work develops a Pareto-improved multi-objective brainstorming optimization (PIMBO) algorithm based on the Pareto rule [24].

This paper also establishes a corresponding mathematical model of MMRDP where the uncertainty of a disassembly process [25] is considered. Since EOL products are subject to other external factors after being idle for a long time, their quality and shape structure will change. This makes the disassembly more difficult and the disassembly time uncertain, whereby the disassembly operations may not be carried out as planned. Hence, disassembly failure risk [26] should be considered. This paper deals with MMRDP by considering disassembly failure risks. This work intends to make the following unique contributions to the disassembly field:

(1) It formulates a stochastic MMRDP model based on an AND/OR graph considering disassembly failure risk. The objectives are to maximize disassembly profit and balancing rate and minimize energy consumption;

(2) It designs a Pareto-improved multi-objective brainstorming optimization algorithm combined with a stochastic simulation approach to handle the proposed problem. Furthermore, a triple-vector list structure is designed to represent a solution. A Pareto-based clustering algorithm, new individual generation algorithm, selection operator, and external archive evolution approach are developed to improve the performance of the algorithm, while a stochastic simulation method is designed to calculate the objective function value of the obtained solutions.

In addition, this paper uses the proposed algorithm and other four popular algorithms, i.e., multi-objective discrete gray wolf optimizer (MDGWO) [27], multi-objective artificial bee colony (MOABC) algorithm [28], nondominated sorting genetic algorithm II (NSGA-II) [24], and multi-objective evolutionary algorithm based on decomposition

(MOEA/D) [29]. This paper validates the performance of the proposed approach in handling the MMRDP.

2. Problem Description

This section first describes an AND/OR graph of a ballpoint pen [4] and a lamp [30], and then defines two relationship matrices. Next, this paper establishes the corresponding mathematical models.

2.1. Subsection AND/OR Graph

An AND/OR graph is a graph model, which can express not only the precedence and conflict relationships between two disassembly operations but also the relationship between subassemblies/parts and disassembly operations. For example, Figure 1 shows the part structure schematic diagram of a ballpoint pen. Figure 2 shows the part structure schematic diagram of the lamp. Figure 3i,ii show the AND/OR graphs of the ballpoint pen and the lamp, respectively.

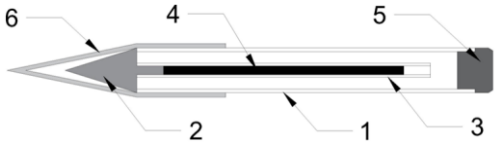

Figure 1. Part structure schematic diagram of a ballpoint pen.

Figure 2. Part structure schematic diagram of a lamp.

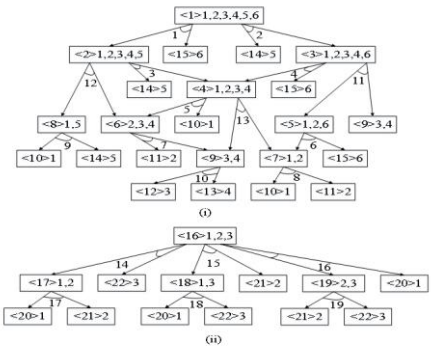

Figure 3. AND/OR graphs of (**i**) a ballpoint pen and (**ii**) a lamp.

In Figure 3, the disassembly operations are represented with hyper-arc indices, $1 - J$, where J is the total number of disassembly operations required to disassembly an EOL product. Nodes indicate subassemblies/parts with the index from <1> to <I>, where I is the number of all subassemblies/parts and components. Let subassembly a be a parent of subassembly b. If b is obtained by disassembling a, then b is called a child of a, where $a, b \in \{1,2, \cdots, I\}$. For example, in Figure 3i, subassembly <1> is the parent of <2> and <15>, while <2> is the child of <1>.

As shown in Figure 3, a disassembly operation disassembles a subassembly into two or more parts whose relationships are described by an "AND" relationship. An "OR" relationship denotes that a part has many ways to disassemble, but these ways cannot be performed simultaneously. For instance, parent subassembly <6> is disassembled into <9> and <11> by operation 7, and <8> is disassembled into <10> and <14> by operation 9. A parent subassembly may have more than one hyper-arc; for example, subassembly

<2> can be disassembled via disassembly operations 3 or 12. Operations 3 and 12 form an OR relationship, which indicates that at most one of the two operations can be executed. According to the definition of an AND/OR graph and the logic of disassembly of a product, two matrices are proposed.

(1) Precedence matrix $P = [p_{jk}]$ is employed to describe the precedence and conflict relations of two disassembly operations, which is defined as

$$p_{jk} = \begin{cases} 1, \text{ if operation } k \text{ can be performed after opration } j; \\ -1, \text{ if operations } j \text{ and } k \text{ conflict with each other;} \\ 0, \text{ otherwise.} \end{cases}$$

Precedence matrix P of a multi-product instance for a ballpoint pen and lamp is given as

$$P = \begin{bmatrix}
0 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & -1 & 0 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 1 & -1 & -1 & 1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & -1 & -1 & -1 & 0 & -1 & 1 & -1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 & -1 & -1 & 0 & -1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & -1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & -1 & 1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & -1 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & -1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & -1 & -1 & 0
\end{bmatrix}.$$

(2) Disassembly-incidence matrix $D = [d_{ij}]$ expresses the corresponding relations between subassemblies/parts and disassembly operations, which is defined as

$$d_{ij} = \begin{cases} 1, \text{ if subassembly } i \text{ is obtained by operation } j. \\ -1, \text{ if subassembly } i \text{ is disassembled via operation } j. \\ 0, \text{ otherwise.} \end{cases}$$

The disassembly-incidence matrix D of a multi-product instance for a ballpoint pen and lamp is as follows:

$$D = \begin{bmatrix}
-1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1
\end{bmatrix}.$$

2.2. Robotic Disassembly Line

In Figure 3, 15 subassemblies/parts of a ballpoint pen correspond to 13 disassembly operations, and a lamp includes seven subassemblies/parts and six disassembly operations. In Figure 4, the number in the pentagon represents a disassembly operation, the number in the circle represents a type of robot that performs a disassembly operation, the number in the rectangle represents parts, and the number in the rectangle on the workstation represents the subassembly that is disassembled into subassemblies/parts. The disassembly operations are assigned to the workstation and then executed by the corresponding robot. As shown in Figure 3, disassembly operations 1–13 and subassemblies/parts 1–15 are

related to the ballpoint pen, and disassembly operations 14–19 and subassemblies/parts 16–22 are related to the lamp. A robot in type i is denoted as r_i. A feasible disassembly scheme contains a disassembly operation sequence 1, 3, 14, 5, 17, 7, and 10 and a robot type sequence 1, 3, 2, 3, 2, 2, and 1. Each disassembly operation in the disassembly operation sequence is mapped to the corresponding workstation according to a cycle time constraint. The robot types of each workstation are based on the robot type of the disassembly operation assigned to each workstation. For instance, disassembly operations 1, 3, and 14 are performed on the first workstation 1, and the robot types that perform these disassembly operations are 1, 3, and 2, respectively. Thus, r_1, r_2, and r_3 are assigned to workstation 1. In addition, the parts 15, 14, 22, 10, 21, 20, 11, 12, and 13 are obtained by a feasible disassembly scheme where r_1, r_2, and r_3 are assigned to workstation 1 to perform operations 1, 3, and 14, r_2 and r_3 are assigned to workstation 2 to perform operations 5 and 17, and r_1 and r_3 are assigned to workstation 3 to perform operations 7 and 10.

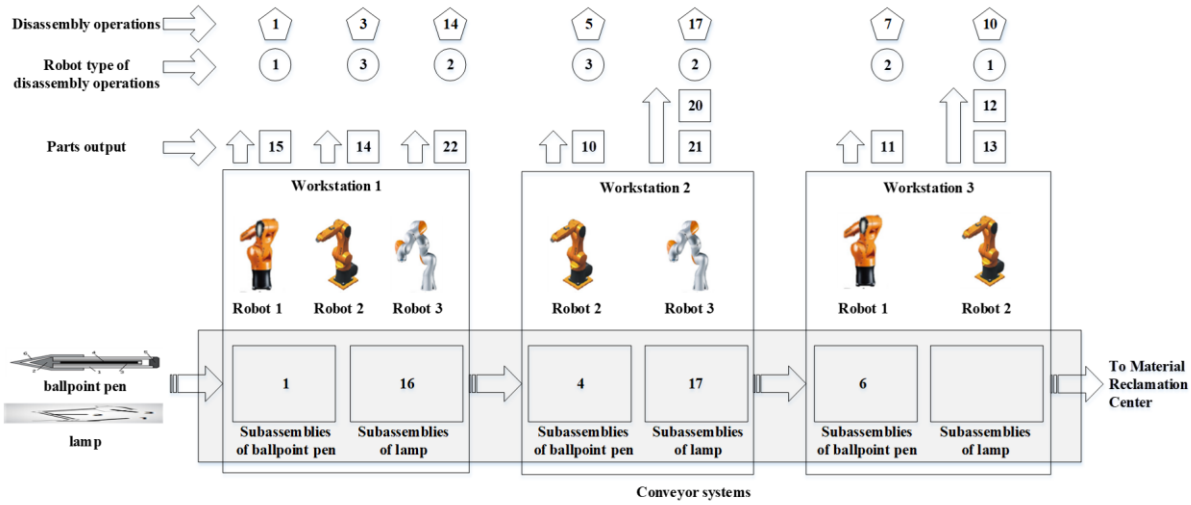

Figure 4. An example of a robotic disassembly line layout.

2.3. Mathematical Model

This paper makes the following assumptions to formulate a mathematical model of a stochastic MMRDP:

(1) The average disassembly time and setup time of each disassembly operation of each type of robot are known.
(2) The disassembly cost, setup cost, disassembly energy consumption, and setup energy consumption per unit time of each type of robot disassembly operation are given.
(3) Only one robot is allowed to perform disassembly operations. Any robot can only handle one disassembly operation assigned to it at a time, and one disassembly operation can only be completed by one robot.
(4) The supply of EOL products is unlimited.
(5) AND/OR graphs of multiple EOL products to be disassembled are known.

Some notations and decision variables are listed below.

2.3.1. Notations

g—EOL product indices, $g \in \{1, 2, \ldots, G\}$, where G represents the number of disassembled products.

i—subassembly/part index, $i \in \{1, 2, \ldots, N\}$, where N denotes the total number of subassemblies/parts of all products.

j, k—operation indices, $j, k \in \{0, 1, \ldots, J\}$, where J means the total number of operations of all products, and 0 is a dummy operation.

l, m—workstation indices, $l, m \in \{1, 2, \ldots, M\}$, where M is the number of workstations.

r—robot type index, $r \in \{1, 2, \ldots, R\}$, where R denotes the number of robot types.

$\mathbb{I}_g^S$—the start index set of subassembly/part of product g, $\mathbb{I}_g^S = \left\{I_1^0, \ldots, I_g^0, \ldots, I_G^0\right\}$, where I_g^0 is the start index of subassembly/part of product g.

$\mathbb{I}_g^E\left\{I_1^1, \ldots, I_g^1, \ldots, I_G^1\right\}$, where I_g^1 is the start index of subassembly/part of product g.

$\mathbb{J}_g^S$—the start index set of tasks of product g, $\mathbb{J}_g^S = \left\{J_1^0, \ldots, J_g^0, \ldots, J_G^0\right\}$, where J_g^0 is the start index of tasks of product g.

$\mathbb{J}_g^E$—the end index set of tasks of product g, $\mathbb{J}_g^E = \left\{J_1^1, \ldots, J_g^1, \ldots, J_G^1\right\}$, where J_g^1 is the end index of tasks of product g.

t_{jr}—disassembly time for operation j by robot r.

t_{jkr}—setup time of performing operation k which is performed immediately after operation j by robot r.

e_{jr}—energy consumption per unit of time of robot r performing operation j.

e_{jkr}—setup energy consumption per unit of time in performing operation k performed immediately after operation j by robot r.

e_l—energy consumption per unit of time during the l-th workstation operation.

c_{jr}—cost per unit of time of robot r performing operation j.

c_{jkr}—setup cost per unit of time in performing operation k performed immediately after operation j by robot r.

c_l—cost per unit of time during the l-th workstation operation.

v_i—reuse value of subassembly/part i.

T_l—cycle time of the l-th workstation.

q_{jkr}—failure probability of operation k performed immediately after operation j in product g by robot r.

θ—preset minimum probability value

$\dot{F}$—maximum failure cost of a disassembly process.

P—multiple products precedence matrix

D—multiple products disassembly-incidence matrix

p_{jk}—an element in the j-th row and k-th column of P.

d_{ij}—an element in the i-th row and j-th column of D.

Notice that t_{jr}^g, t_{jkr}^g, and $t_{jkr}^{g\varphi}$ are random variables with normal distribution.

2.3.2. Decision Variables

x_{jr}—if operation j is performed by robot r, $x_{jr} = 1$; otherwise, $x_{jr} = 0$.

y_{jkr}—if operation k is performed immediately after operation j by robot r, $y_{jkr} = 1$; otherwise, $y_{jkr} = 0$.

z_{jrl}—if operation j is performed by robot r and assigned to the l-th workstation, $z_{jrl} = 1$; otherwise, $z_{jrl} = 0$.

u_l—if the l-th workstation is used, $u_l = 1$; otherwise, $u_l = 1$.

On the basis of AND/OR graphs and relationship matrices, multiple EOL products to be disassembled can be mathematically described. The mathematical model is given below.

The expected maximal disassembly profit is the total profit obtained from disassembly minus total disassembly cost in the disassembly process. As shown in Equation (1), the latter consists of three main parts, i.e., the cost of disassembly operations, the setup cost of adjacent operations, and the cost of workstations.

$$\max f_1 = E\Big(\sum_{g=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{i=I_g^0}^{I_g^1}\sum_{r=1}^{R} d_{ij}v_i x_{jr} - \sum_{g=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{r=1}^{R} t_{jr}c_{jr}x_{jr} - \sum_{g=1}^{G}\sum_{\varphi=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{k=J_\varphi^0}^{J_\varphi^1}\sum_{r=1}^{R} t_{jkr}c_{jkr}y_{jkr} - \sum_{l=1}^{M} T_l c_l u_l\Big). \quad (1)$$

The expected minimal energy consumption mainly takes into account three aspects of energy consumption, i.e., the total energy consumption of disassembly operations, the

total setup energy consumption of adjacent operations, and the energy consumption of workstations, as shown in Equation (2).

$$\min f_2 = E(\sum_{g=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{r=1}^{R} t_{jr}e_{jr}x_{jr} + \sum_{g=1}^{G}\sum_{\varphi=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{k=J_\varphi^0}^{J_\varphi^1}\sum_{r=1}^{R} t_{jkr}e_{jkr}y_{jkr} + \sum_{l=1}^{M} T_l e_l u_l). \quad (2)$$

The expected maximal balancing rate is equal to the average of the total operating time divided by the total cycle time of the switched-on workstations, as shown in Equation (3).

$$\max f_3 = E((\sum_{g=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{r=1}^{R} t_{jr}x_{jr} + \sum_{g=1}^{G}\sum_{\varphi=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{k=J_\varphi^0}^{J_\varphi^1}\sum_{r=1}^{R} t_{jkr}y_{jkr}) / \sum_{l=1}^{M} T_l u_l). \quad (3)$$

The fact that each product is disassembled by at least one disassembly operation excluding operation 0 is constrained by Equation (4).

$$\sum_{j=J_g^0}^{J_g^1}\sum_{r=1}^{R} x_{jr} \geq 1, g = 1, 2, \cdots, G. \quad (4)$$

Each disassembly operation in multiple products should be performed at most once, which is constrained by Equation (5).

$$\sum_{r=1}^{R} x_{kr} = \sum_{j=J_g^0}^{J_g^1}\sum_{r=1}^{R} y_{jkr}, g = 1, 2, \cdots, G, k = 1, 2, \cdots, J. \quad (5)$$

Each disassembly operation of any product can only be assigned to one workstation and can only be performed by one robot, which is constrained by Equation (6).

$$\sum_{r=1}^{R} x_{jr} = \sum_{l=1}^{M}\sum_{r=1}^{R} z_{jrl}, g = 1, 2, \cdots, G, j = 1, 2, \cdots, J. \quad (6)$$

Each switched-on workstation must be assigned at least one operation, which is constrained by Equation (7).

$$\sum_{g=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{r=1}^{R} x_{jr}z_{jrl} \geq 1, l = 1, 2, \cdots, M. \quad (7)$$

To ensure that a feasible disassembly sequence of multiple products satisfies precedence constraint, this paper explicitly defines

$$\sum_{l=1}^{M}\sum_{r=1}^{R} l z_{jrl} \leq \sum_{m=1}^{M}\sum_{r=1}^{R} m z_{krm}, g = 1, 2, \cdots, G, \forall p_{jk} = 1. \quad (8)$$

To ensure that a feasible disassembly sequence of multiple products satisfies conflict constraint, this paper proposes Equation (9).

$$\sum_{l=1}^{M}\sum_{r=1}^{R} z_{jrl} + \sum_{l=1}^{M}\sum_{r=1}^{R} z_{krl} \leq 1, g = 1, 2, \cdots, G, \forall p_{jk} = -1. \quad (9)$$

The operation time of all workstations used is no more than the disassembly line total cycle time, represented by Equation (10).

$$\sum_{g=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{r=1}^{R} t_{jr}x_{jr} + \sum_{g=1}^{G}\sum_{\varphi=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{k=J_\varphi^0}^{J_\varphi^1}\sum_{r=1}^{R} t_{jkr}y_{jkr} \le \sum_{l=1}^{M} T_l u_l,\ l = 1,2,\cdots,M. \tag{10}$$

The probability for the failure cost of a disassembly process being less than or equal to its maximum failure cost is greater than a preset value $\theta > 0$, required by Equation (11).

$$\mathcal{P}\left\{\sum_{g=1}^{G}\sum_{\varphi=1}^{G}\sum_{j=J_g^0}^{J_g^1}\sum_{k=J_\varphi^0}^{J_\varphi^1}\sum_{r=1}^{R}\left(t_{kr}c_{kr}x_{kr} + t_{jkr}c_{jkr}y_{jkr}\right)q_{jkr} < \dot{F}\right\} \ge \theta. \tag{11}$$

The range of decision variables is given by Equation (12).

$$x_{jr}, y_{jkr}, z_{jrl}, u_l \in \{0,1\}, g, \varphi = 1,2,\cdots,G,\ j,k = 1,2,\cdots,J, l = 1,2,\cdots,M,\ r\ =1,\ 2,\ \ldots,\ R. \tag{12}$$

3. Proposed Algorithm

3.1. Base Brainstorming Optimization Algorithm

Brainstorming conference is the process of discussing solutions to difficult problems. The brainstorming optimization (BSO) algorithm is inspired by this feature of group discussion and speaking freely [21]. There are three roles involved in a brainstorming conference, i.e., several owners of problems to be solved, and a group of members with different backgrounds. Since practical experience and ways of thinking are different, for the same problem, they can put forward different solutions. The conference stipulates that the host presides over and urges the team members to generate as many ideas as possible, but the host does not participate in an idea generation process. In addition, four rules are given as follows:

(1) Any idea is meaningful. All ideas, good or bad, cannot be judged until the end of a brainstorming process.

(2) Brainstorming team members should be unreserved, and any ideas generated in their minds should be shared and studied.

(3) Many new ideas are generated by associating with the existing ideas.

(4) Many new ideas are generated as possible and then choose from abundant ideas to get excellent solutions.

In BSO, each solution is represented separately by an individual idea. In a human brainstorming conference, each idea is inspired by other individual ideas and becomes more mature. The same is true in BSO, where one individual is updated through individual evolution and fusion. The algorithm consists of three strategies: solution clustering, new individual generation, and solution selection.

The MMRDP is a multi-objective, stochastic, and discrete optimization problem. However, the basic BSO algorithm was originally proposed to solve single-objective and continuous optimization problems. Therefore, some special strategies need to be designed according to the characteristics of MMRDP. This paper proposes a PIMBO algorithm incorporated with a stochastic simulation approach.

3.2. Encoding

A solution is represented by a triple-vector list designed for this work, i.e., $\pi = (\pi^1, \pi^2, \pi^3)$, where $\pi^1 = (o_1, o_2, \ldots, o_J)$ is an integer string. It denotes a disassembly operation sequence, where o_j indicates an operation index of the j-th disassembly operation, and J is the maximum index of disassembly operations for multiple products to be disassembled. $\pi^2 = (x_1, x_2, \ldots, x_J)$ is a binary vector with J binary values indicating whether the disassembly operation at the corresponding position is performed. If $x_j = 1$, the operation o_j in π^1 is performed; otherwise, o_j is not performed. $\pi^3 = (r_1, r_2, \ldots, r_J)$ represents a robot type

sequence, whose element is an integer between 1 and R and corresponds to the robot type performing a disassembly operation.

3.3. Population Initialization

The key to solve the proposed problem is to design a reasonable population initialization strategy according to the above encoding rules. The proposed algorithm employes the random initialization method of MMRDP solutions to generate diversified initial population in the searching space. Notice that a solution may be infeasible in the case that it disobeys precedence and conflict constraint conditions. In this work, an adjustment approach is developed to initialize individuals so that each solution becomes feasible. The following steps detail its adjustment process:

Step 1: Randomly generate π, i.e., π^1, π^2, and π^3.

Step 2: Adjust the disassembly operation sequence in π^1 according to the rules specified in matrix P to meet the precedence relationship between disassembly operations of multiple products. It is required to traverse the whole disassembly operation sequence π^1. If the disassembly operation o_j is after o_k, but $p_{jk} = 1$ in matrix P, which is a contradiction, then o_j and o_k need to be interchanged in the correct order, and r_j and r_k are swapped at the same time; otherwise, it indicates that operation j and k are in accordance with the disassembly sequence. Lastly, check the next disassembly operation.

Step 3: In order to make the disassembly proceed smoothly, the first disassembly operation must be performed. Hence, if operation o_j in π^1 is the first disassembly operation of a product but $x_j = 0$ in π^2, then adjust x_j to 1; otherwise, go to Step 4.

Step 4: Adjust the binary values in π^2 based on matrix P to meet the precedence relationship between disassembly operations of the same product. It is required to traverse the whole binary vector π^2. If $x_k = 1$ and $p_{jk} = 1$ in matrix P, then x_j is set to 1.

Step 5: Adjust the binary values in π^2 based on P to remove the conflict relationship between operations of the same product to be disassembled. The above process needs to traverse the entire vector π^2. If $x_j = 1$ and $p_{jk} = -1$, then $x_k = 0$.

After completing the strategy steps described above, an initial population set $\mathbb{P}$ which contains Q solutions (individuals) is obtained.

3.4. Decoding

The decoding task is completed by assigning the selected disassembly operations and the robot to each corresponding workstation. Therefore, a disassembly operation and the robot performing it should be assigned to the same workstation at the same time. Moreover, a cycle time constraint needs to be satisfied when assigning disassembly operations to workstations. Figure A1 of Appendix A.1 shows the decoding procedure, the detailed steps of which are as follows:

Step 1: Set disassembly operation $k = 1$.

Step 2: Verify that the disassembly operation k is performed. If $x_k = 1$ and $k = k_f$ where k_f is the first disassembly operation in the whole disassembly sequence, open the l-th workstation; the processing time of the l-th workstation T_{ld} is set to 0, the disassembly operation set C_l in the l-th workstation is set to ζ, where ζ represents an empty set, and the robot set R_l of the l-th workstation is set to r, where r means a type of robot performing disassembly operation k. If $x_k = 1$ and $k \neq k_f$, perform disassembly operation k, followed by Step 3; otherwise, perform Step 8.

Step 3: Set $T_{ld} = T_{ld} + t_{jkr}$, and determine the relationship between the cycle time T_l and the current processing time of the l-th workstation T_{ld}. If $T_{ld} > T_l$, then perform Step 4; otherwise, perform Step 5.

Step 4: Start the next workstation; let $l = l + 1$, $C_l = \zeta$, $R_l = r$, and $T_{ld} = t_{jkr}$.

Step 5: Set $T_{ld} = T_{ld} + t_{kr}$, and determine the relationship between T_l and T_{ld}. If $T_{ld} > T_l$, then perform Step 6; otherwise, perform Step 7.

Step 6: Start the next workstation; let $l = l + 1$, $C_l = \zeta$, $R_l = r$, and $T_{ld} = t_{jkr} + t_{kr}$.

Step 7: Disassembly operation k is assigned to the l-th workstation; then, update C_l and R_l, and let $C_l = C_l \cup k$ and $R_l = R_l \cap r$.

Step 8: If $k \neq J$, then set $k = k + 1$, and repeat Steps 2–7. Otherwise, output a feasible solution.

3.5. Objective Function Evaluation

After performing the relevant operations in Sections 3.2–3.4, many feasible solutions of MMRDP are generated, and the next step is to evaluate them. The MMRDP is different from a deterministic optimization problem. The objective function values of the latter can be straightforwardly calculated according to its objective functions and constraints. In the MMRDP, the disassembly and setup times of the products are randomly generated, and the objective functions in Equations (1)–(3) are expressed as their expected values. They cannot be calculated directly as deterministic optimization problems. Hence, Monte Carlo simulation is used to successfully solve the above problems [31].

Let π be a solution; Γ denotes the number of samples equaling 10. This paper adopts the following steps to evaluate the objective function values and feasibility of π. It should be noted that since the objective functions f_1 and f_3 are to find their maximum values, the values of the objective functions f_1 and f_3 obtained are multiplied by -1 to facilitate the comparison of the quality of the solutions later.

Step 1: $f_1(\pi) := 0$, $f_2(\pi) := 0$, $f_3(\pi) := 0$, $f_c(\pi) := 0$, and $\Gamma := 10$.

Step 2: Generate δ samples; each sample ρ_δ is a sample of disassembly and setup times produced from their normal distributions, $\delta = 1, 2, \cdots, \Gamma$.

Step 3: Calculate the disassembly profit $f_{1\delta}(\pi)$, energy consumption $f_{2\delta}(\pi)$, balancing rate $f_{3\delta}(\pi)$, and disassembly failure cost $f_{c\delta}(\pi)$, where $\delta = 1, 2, \cdots, \Gamma$.

Step 4: Sort the disassembly failure cost $f_{c\delta}(\pi)$ of all samples in ascending order; let δ' be equal to $\lfloor \theta \cdot \Gamma \rfloor$. If $f_{c\delta'}(\pi) < \dot{F}$, π is feasible and $f_c(\pi) := f_{c\delta'}(\pi)$, then go to Step 5. Otherwise, π is infeasible, where $f_1(\pi)$, $f_2(\pi)$, and $f_3(\pi)$ are equal to larger integers; then, go to Step 6.

Step 5: Approximate the expected disassembly profit $f_1(\pi)$, energy consumption $f_2(\pi)$, and balancing rate $f_3(\pi)$ as follows:

$$f_1(\pi) := -1 \times \frac{\sum_{\delta=1}^{\Gamma} f_{1\delta}(\pi)}{\Gamma}. \tag{13}$$

$$f_2(\pi) := \frac{\sum_{\delta=1}^{\Gamma} f_{2\delta}(\pi)}{\Gamma}. \tag{14}$$

$$f_3(\pi) := -1 \times \frac{\sum_{\delta=1}^{\Gamma} f_{3\delta}(\pi)}{\Gamma}. \tag{15}$$

Step 6: Return $f_1(\pi)$, $f_2(\pi)$, and $f_3(\pi)$ as the approximated values of Equations (1), (2), and (3).

3.6. Multi-Objective Processing Method

In order to distinguish the advantages and disadvantages of the obtained solutions, this work uses Pareto dominance relationship. The process of comparison is the decision-making process of a choice of solutions.

An external archive $\mathbb{A}$ is constructed to store the obtained nondominated solutions. First, it is updated by using all newly generated solutions based on the Pareto rule. After each iteration is finished, $\mathbb{A}$ is updated by using the current population based on the Pareto rule to ensure that all nondominated solutions in $\mathbb{A}$ are always the current global nondominated solutions.

3.7. Pareto-Based Clustering

During the clustering stage, this work uses the fast nondominated sorting approach proposed in [24]. The individuals are ranked according to the Pareto rule, and then the indi-

viduals of the same rank are classified into the same cluster. The *k*-means clustering constantly calculates the new clustering center after adjusting the sample classification, thus resulting in a large amount of calculation. However, the Pareto-based clustering algorithm reduces the computational complexity, and the number of clusters needs not to be predetermined. The pseudo-code of Pareto-based clustering is shown in Algorithm A1 of Appendix A.2.

3.8. New Individual Generation

The pseudo-code of new individual generation is shown in Algorithm A2 of Appendix A.3, where $\mathbb{X}$ denotes the set of new individuals, p_c is the probability of replacing a cluster center, p_g is the probability that a new individual is generated from one subcluster, p_o is the probability that a new individual is generated concerning a subcluster center, and p_t is the probability that a new individual is generated by combining with two subcluster centers.

In the original BSO, when a random number ϑ_g is less than p_g, only one individual contributes to generate a new individual. This paper combines the center of the first cluster with the current selected individual to generate a new individual, which improves the convergence speed of the algorithm. Moreover, this paper adopts a precedence preserving crossover (PPX) [32] operator and a position-based mutation (PBM) [27] operator to generate a new individual.

3.9. Selection Operator

The selection algorithm is used to retain the solutions with better quality among all the solutions found so far. To accelerate the convergence of the algorithm, the first Q individuals in the combination of $\mathbb{C}$ and $\mathbb{A}$ are chosen as the next population by using rank and crowding distance approaches in [24].

3.10. Procedure of PIMBO

The procedure of PIMBO is summarized as follows:

Firstly, all parameters involved in the algorithm are set initially and then an initial population is generated and evaluated.

Secondly, with the given termination condition, the algorithm enters the following loop: the Pareto-based clustering, new individual generation, and selection are performed repeatedly until a given termination condition is reached.

Lastly, all solutions in $\mathbb{A}$ are output.

In this paper, the iterations of the PIMBO and its comparison algorithm are determined by the fitness value. Before the PIMBO starts to iterate, the fitness value is equal to 0. When evaluating a new individual, the fitness value is increased by 1 until the current fitness value f_v reaches the preset total fitness value f_{tv}; then, the algorithm stops the iteration. Algorithm 1 gives the pseudo-code of PIMBO.

Algorithm 1: PIMBO

Input: p_c, p_g, p_o, p_t, $|\mathbb{P}|$, f_{tv},c_l,T_l,θ, R, crossover probability, and mutation probability.
Output: all solutions in $\mathbb{A}$.
Begin
  Set algorithm parameters.
  Initialize population as shown in Sections 3.2 and 3.3.
  Perform decoding process as shown in Section 3.4.
  Evaluate solutions as shown in Section 3.5.
  while ($f_v < f_{tv}$)
    Perform Pareto-based clustering as shown in Section 3.7.
    Execute new individual generation as shown in Section 3.8.
    Construct next population as shown in Section 3.9.
  end while
    Output all solutions in $\mathbb{A}$.
End.

4. Experiments

To test the performance of PIMBO in handling the considered problem, this work conducts experiments by adopting the set of EOL products given in [4,12,33]. Additionally, MDGWO [27], MOABC [28], NSGAII [24], and MOEAD [29] are compared. They have shown excellent performance in solving various optimization problems and have been used as a benchmark for comparison by many scholars and practitioners [34]. Accordingly, this paper chooses MDGWO, MOABC, NSGAII, and MOEAD as the peer methods. In this work, the position-based crossover and position-based mutation operations are adopted by these algorithms to generate new solutions.

4.1. Test Instance Generation

In order to have sufficient experimental data, this work collects a large amount of product data in the literature as examples: ballpoint pen (BP) [4], hammer drill (HD) [12], washing machine (WM) [35], and radio set (RS) [4]. A BP contains 15 subassemblies/parts, and 13 disassembly operations are needed to disassemble it. An HD includes 62 subassemblies/parts, and 46 disassembly operations should be adopted to disassemble it. A WM contains 15 subassemblies/parts, and 13 disassembly operations are used to disassemble it. There are 29 subassemblies/parts in an RS, and 30 disassembly operations are needed to disassemble it. Their schematic diagrams and AND/OR graphs are shown in Figures A2–A7 of Appendix A.4. By adopting BP, HD, WM, and RS, 10 test instances of small and medium-sized disassembly tasks are generated, and each instance contains multiple disassembled products. Their corresponding relationships are given in Table 1, where $\dot{F}$ represents the maximum failure cost of all disassembly operations performed in each case.

Table 1. Products in instances.

No.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Products	BP, WM	BP, HD	BP, RS	HD, WM	HD, RS	WM, RS	BP, HD, WM	BP, HD, RS	BP, WM, RS	BP, HD, WM, RS
$\dot{F}$	25	95	40	100	120	45	120	135	60	155

4.2. Performance Metrics

This work utilizes three metrics to examine the results obtained by PIMBO and its peers to verify the performance of PIMBO algorithm, i.e., *C*-metric [29], IGD metric [35], and hypervolume metric [35]. The *C*-metric can measure the dominated percentage of two solution sets.

IGD calculates the average distance of the obtained solutions to their nearest solutions in an optimal solution set. A smaller IGD of a solution set denotes a better approximation and distribution. Notice that this work cannot acquire an optimal solution set for the proposed MMRDP beforehand. Thus, an optimal solution set is composed of all solution sets obtained by PIMBO and its peers.

Hypervolume can comprehensively evaluate the convergence and distribution of nondominated solution sets. It focuses on computing the volume as constructed by the obtained solutions and the reference point. A bigger hypervolume of a solution set denotes a better approximation and distribution. It is worth noting that, when calculating the hypervolume of the solutions, this work multiplies the first objective value and the third objective value of all solutions by -1 since the first and third objective functions in the investigated problem are to maximize the profit and balancing rate, respectively.

PIMBO and its comparison algorithms were used to solve each instance 20 times, and then these solutions were used to calculate the average value of the three metrics of each algorithm. Additionally, this paper analyzes the experimental results by employing a one-tailed *t*-test [36] with 38 degrees of freedom at a 0.05 level of significance. In the following analysis of experimental results, we use the symbols "+", "−", or "~" to illustrate the comparison results between P and other algorithms The symbol "+" means that PIMBO is significantly better than its peer methods, The symbol "−" indicates that PIMOB is

significantly worse than its peers, while "~" denotes that PIMBO is statistically equivalent to its peers.

4.3. Parameter Setting

In order to improve the performance of the proposed PIMBO algorithm, the parameters are set as follows: p_c = 0.2, p_g = 0.8, p_o = 0.4, and p_t = 0.5 [37]. The parameters used in all algorithms are set as follows: population size $|\mathbb{P}|$ = 120, crossover probability = 1, mutation probability = 0.3, and total fitness value f_{tv} = 2000. The parameters involved in the objective functions are set as follows: c_l = 0.03, e_l = 0.05, T_l = 50, θ = 0.90, and R = 3 [6,14]. To avoid chance, each algorithm is executed 20 times independently, and their average running time in each instance is shown in Table 2. This work concludes that PIMBO could find a promising solution set in less time.

Table 2. The average running time in each instance on five algorithms.

No.	Average Running Time (s)				
	PIMBO	**MDGWO**	**MOABC**	**NSGAII**	**MOEAD**
1	2.5828	3.5572	3.5451	2.9422	2.7086
2	12.6085	23.5503	13.6601	13.1815	12.5823
3	6.7303	12.5189	9.5496	7.0042	6.7609
4	12.4917	24.7326	16.5742	15.3582	12.5347
5	20.4667	44.7032	22.1236	20.9711	20.4619
6	6.7068	12.2596	8.7577	7.2507	6.4053
7	19.1543	46.0592	23.1308	19.7624	18.5889
8	29.2066	60.1669	36.2157	30.2257	28.7547
9	11.7235	21.7302	14.7726	12.0219	11.4901
10	38.8173	93.6209	51.7922	39.5651	36.6670

4.4. Case Analyzes

Table 3 shows 10 nondominated solutions generated by solving instance 2 through PIMBO, where f_1, f_2, f_3, f_c, and r_s denote profit, energy consumption, balancing rate, failure cost, and robot type set assigned to switched-on workstations, respectively. According to the previous description, BP has 13 disassembly operations, and HD has 46 disassembly operations. Therefore, disassembly operations 1–13 in these solutions represent the disassembly operations of BP, and 14–59 represent the disassembly operations of HD. In the first solution, disassembly operations 2 and 11 are from BP, and disassembly operations 14, 15, 17, 20, 24, 32, 42, 49, and 31 are from HD. A robot in type i is denoted as r_i. Disassembly operations 14, 15, and 17 are performed by r_1 and r_2 in the first workstation, disassembly operations 2, 20, 11, and 24 are performed by r_1 and r_3 in the second workstation, and disassembly operations 32, 42, 49, and 31 are performed by r_3 in the third workstation. The profit, energy consumption, balancing rate, and failure cost are 978.94, 998.13, 0.9459, and 63.03, respectively. Therefore, this solution to the considered problem in this research is satisfactory.

4.5. Results

In this section, the five algorithms are used to calculate the 10 test instances provided, and the C-metric, IGD metric, and hypervolume metric are employed to analyze the previous calculation results. In Tables 4–6, the symbols "t-te", "m", and "v" represent the "t-test", "mean", and "variance", respectively.

Table 3. The disassembly sequence and related data of instance 1 obtained by PIMBO algorithm program.

No.	Disassembly Sequence	f_1	f_2	f_3	f_c	r_s
1	14,15,17---2,20,11,24---32,42,49,31	978.94	998.13	0.9459	63.03	1,2---1,3---3
2	14,15,17,1---21,28,38,47,27—24,12,32	1053.38	1054.99	0.9201	52.12	1,2---2,1,3---3,2
3	14,15,17,2---20,11,24,32---31	1069.02	724.80	0.6404	28.69	1,2,3---1,3---3
4	14,15,17,1---20,24,31,32	803.73	654.17	0.9005	42.03	1,2,3---1,2,3
5	14,15,17,2---21,11,28,38---47,27,24,31---32,37,46	1524.12	1341.82	0.8452	60.02	1,2,3---1,2---1,3---2,3
6	14,15,17,2---20,11,24,32---6,42	978.31	815.96	0.7801	37.20	1,2---1,3---1,3
7	14,15,17,2---11,20,24,32---6	969.73	776.50	0.6595	35.46	1,2---1,3---1
8	14,15,16---2,19,11,23---30,24,32,42	1056.80	1048.14	0.9008	43.61	1,2,3---1,3---2,3
9	14,15,17,2---21,28,38,11,47---24,32,42	1228.71	1043.00	0.8334	44.21	1,2---1,3---3
10	14,15,17,2---11,21,28,38,47---27,24,31,37---46	1392.76	1171.49	0.7497	60.56	1,2---1,2,3---1,3---1

Table 4. Comparison of the experimental results of the five algorithms through the C-metric.

No.	*C*(A, B)	*C*(B, A)	*t*-te	*C*(A, E)	*C*(E, A)	*t*-te	*C*(A, U)	*C*(U, A)	*t*-te	*C*(A, V)	*C*(V, A)	*t*-te
1	0.6129	0.1079	+	0.5392	0.1207	+	0.7189	0.0536	+	0.2555	0.1344	+
2	0.6994	0.1245	+	0.8039	0.0514	+	0.9148	0.0113	+	0.1487	0.3698	-
3	0.7203	0.0958	+	0.7029	0.0953	+	0.8822	0.0271	+	0.2245	0.3140	~
4	0.8151	0.0474	+	0.8533	0.0237	+	0.9354	0.0113	+	0.2200	0.2549	~
5	0.8055	0.0888	+	0.8284	0.0343	+	0.9543	0.0014	+	0.0992	0.4923	-
6	0.5602	0.1932	+	0.6472	0.1226	+	0.8750	0.0230	+	0.3131	0.1614	+
7	0.6930	0.1291	+	0.9017	0.0263	+	0.9188	0.0059	+	0.1370	0.4439	-
8	0.7472	0.1244	+	0.8190	0.0579	+	0.9311	0.0108	+	0.0902	0.5125	-
9	0.8608	0.0389	+	0.8774	0.0266	+	0.9580	0.0061	+	0.3539	0.2229	+
10	0.8172	0.0524	+	0.9055	0.0263	+	0.9685	0.0043	+	0.1134	0.4556	-

Table 5. Comparison of the experimental results of the five algorithms through the IGD metric.

No.	PIMBO			MDGWO			MOABC			NSGAII			MOEAD		
	m	v	*t*-te	m	v	*t*-te	m	v	*t*-te	m	v	*t*-te	m	v	*t*-te
1	0.0549	0.0003		0.0971	0.0031	+	0.0775	0.0003	+	0.1163	0.0008	+	0.1481	0.0010	+
2	0.1360	0.0008		0.1865	0.0013	+	0.2104	0.0006	+	0.2478	0.0006	+	0.1693	0.0016	+
3	0.1059	0.0005		0.1736	0.0030	+	0.1604	0.0008	+	0.2012	0.0009	+	0.1416	0.0011	+
4	0.1157	0.0003		0.2123	0.0041	+	0.1986	0.0003	+	0.2453	0.0009	+	0.1432	0.0006	+
5	0.1693	0.0009		0.2579	0.0034	+	0.2516	0.0008	+	0.2967	0.0010	+	0.1645	0.0015	~
6	0.1115	0.0036		0.1673	0.0063	+	0.1558	0.0016	+	0.2296	0.0009	+	0.1733	0.0044	+
7	0.1581	0.0005		0.2125	0.0031	+	0.2486	0.0008	+	0.2868	0.0006	+	0.1660	0.0013	~
8	0.1850	0.0015		0.2721	0.0053	+	0.2858	0.0011	+	0.3332	0.0011	+	0.1605	0.0016	-
9	0.1056	0.0003		0.2326	0.0053	+	0.2069	0.0013	+	0.2621	0.0010	+	0.1632	0.0005	+
10	0.1959	0.0013		0.2928	0.0048	+	0.3194	0.0011	+	0.3797	0.0018	+	0.1872	0.0007	~

Table 6. Comparison of the experimental results of the five algorithms through the hypervolume metric.

No.	PIMBO			MDGWO			MOABC			NSGAII			MOEAD		
	m	v	*t*-te	m	v	*t*-te	m	v	*t*-te	m	v	*t*-te	m	v	*t*-te
1	0.6803	0.0035		0.6399	0.0121	~	0.6788	0.0034	~	0.6384	0.0093	~	0.6000	0.0129	+
2	0.4583	0.0159		0.3704	0.0088	+	0.3310	0.0105	+	0.2964	0.0089	+	0.4215	0.0289	~
3	0.5560	0.0264		0.4584	0.0265	+	0.5690	0.0276	~	0.4426	0.0097	+	0.5208	0.0210	~
4	0.4168	0.0178		0.3630	0.0101	~	0.3784	0.0094	~	0.3431	0.0056	+	0.3988	0.0136	~
5	0.2980	0.0078		0.3115	0.0064	~	0.2715	0.0040	~	0.2625	0.0040	~	0.3587	0.0282	~
6	0.5102	0.0202		0.4487	0.0258	~	0.4321	0.0142	+	0.3702	0.0072	+	0.4299	0.0158	+
7	0.4282	0.0155		0.4118	0.0076	~	0.3761	0.0053	~	0.3437	0.0063	+	0.4681	0.0261	~
8	0.3414	0.0146		0.3648	0.0172	~	0.3949	0.0122	~	0.2970	0.0060	~	0.3841	0.0329	~
9	0.5434	0.0090		0.4090	0.0213	+	0.4815	0.0139	+	0.3686	0.0085	+	0.5095	0.0209	~
10	0.3803	0.0091		0.4504	0.0053	-	0.4383	0.0055	-	0.3745	0.0067	~	0.3855	0.0125	~

Table 4 gives the comparative experimental results of five algorithms with respect to the *C*-metric, where the symbols A, B, E, U, and V denote PIMBO, MDGWO, MOABC, NSGAII, and MOEAD, respectively. Analyzing the experimental results of this metric, we can conclude that all solutions obtained by PIMBO could dominate those found by MDGWO, MOABC, and NSGAII since *C*(A, B), *C*(A, E), and *C*(A, U) were greater than *C*(B, A), *C*(E, A), and *C*(U, A), respectively. However, there are three instances where more solutions obtained by PIMBO could dominate those acquired by MOEAD. The average values of *C*(A, B), *C*(B, A), *C*(A, E), *C*(E, A), *C*(A, U), *C*(U, A), *C*(A, V), and *C*(V, A) were 0.7332, 0.1002, 0.7879, 0.0585, 0.9057, 0.0155, 0.1956, and 0.3362, respectively. These results confirm that PIMBO outperformed MDGWO, MOABC, and NSGAII, whereas it was inferior to MOEAD for coping with the concerned problem.

Table 5 reveals the comparative experimental results of the five algorithms according to the IGD metric. By analyzing the comparison results of this metric, we can clearly conclude that PIMBO performed better than MDGWO, MOABC, NSGAII, and MOEAD for solving the concerned problem since the IGD values of PIMBO were smaller than those of MDGWO, MOABC, NSGAII, and MOEAD in most instances. The average values of the five algorithms for 10 instances were 0.1338, 0.2105, 0.2115, 0.2599, and 0.1617, respectively. According to the analysis of the above experimental results, the algorithm designed in this paper could obtain a nondominated solution set with fast convergence speed and rich diversity when solving the concerned MMRDP.

In order to reveal the performance of PIMBO and its peers more comprehensively, the hypervolume metric was adopted to analyze the experimental results in Table 6. It can be found that the hypervolume values obtained by PIMBO were bigger than those of NSGAII in all instances. There were seven instances where the hypervolume values obtained by PIMBO were bigger than those of MDGWO and MOABC, and there were six instances where the hypervolume values obtained by PIMBO were bigger than those of MOEAD. Through more detailed calculation, the average values of the five algorithms for 10 instances were 0.4613, 0.4228, 0.4352, 0.3737, and 0.4477, respectively. Hence, this demonstrates that, compared with MDGWO, MOABC NSGAII, and MOEAD, PIMBO provided more uniformly distributed solutions in the solution set when solving MMRDP. In order to further and more visually describe the experimental results, this paper presents the boxplots of experimental results of the five algorithms for 10 instances in terms of the IGD metric and hypervolume metric in Figures A5 and A6 of Appendix A.4, respectively. In Figure A5, "HV" denotes the hypervolume value. From the experimental results and Figures A5 and A6, this work can conclude that PIMBO had better performance than MDGWO, MOABC, NSGAII, and MOEAD in dealing with the proposed problem.

5. Conclusions

This work introduced the MMRDP in an uncertain environment with objectives of maximization of profit and balancing rate, as well as minimization of energy consumption. A detailed and complete mathematical model was developed to display the MMRDP. Then, an improved PIMBO algorithm was formulated by incorporating a stochastic simulation approach to solve the MMRDP. Through the comparison of multiple groups of experiments, the proposed algorithm was shown to have excellent phenotype under the measurement of the selected metrics The experimental results can assist producers to make more diversified decision plans.

For future work, we plan to strive in two directions: (1) to consider more practical issues based on this model such as U-type, circular type, and parallel disassembly lines [38,39]; (2) to design much better intelligent algorithms to deal with highly complex optimization problems [40–45].

Author Contributions: Formal analysis, Z.L.; Data curation, X.G. and X.L.; Writing—original draft, G.X.; Writing—review & editing, Z.Z.; Supervision- data accuracy, L.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the Guangzhou Innovation and Entrepreneurship Leading Team Project Funding under Grant No. 202009020008.

Data Availability Statement: Link: https://pan.baidu.com/s/1LdDNgwoiO06uWyZMTbXy3A; extraction code: 556h.

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

Appendix A

Appendix A.1. Decoding Procedure

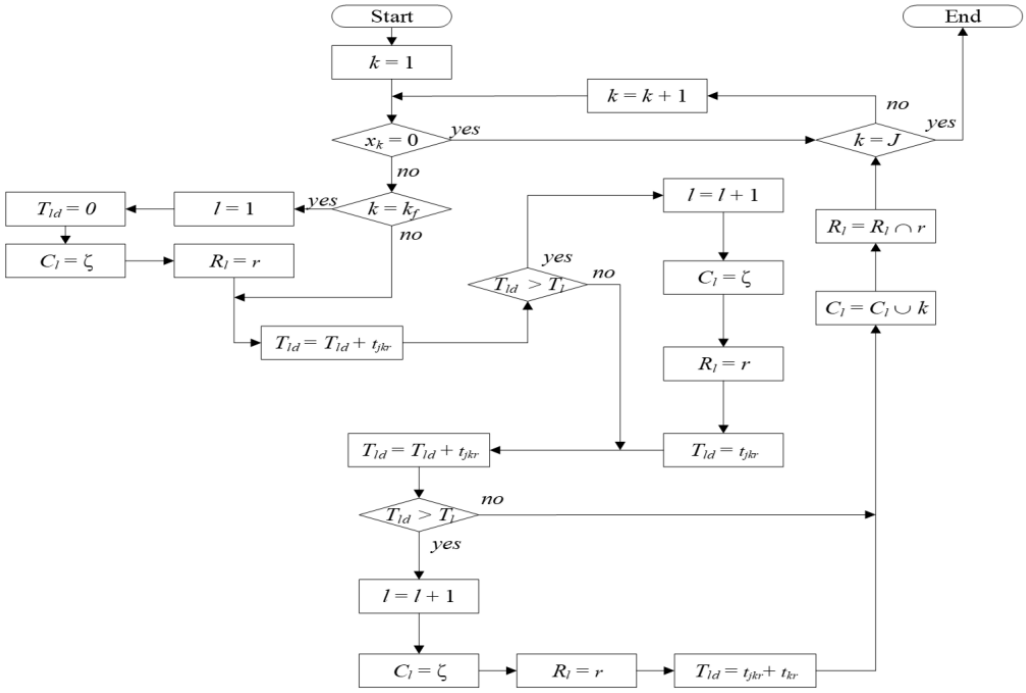

Figure A1. The flowchart of decoding.

Appendix A.2. Clustering

The pseudo-code of Pareto-based clustering is shown in Algorithm A1, where $\mathbb{C}_p$ represents a subcluster in cluster $\mathbb{C}$, and τ denotes the rank level of individuals.

Algorithm A1: Pareto-Based Clustering

Input: all individuals in population $\mathbb{P}$.
Output: the cluster $\mathbb{C}$.
Begin
Initialize two sets δ and $\mathbb{C}_p$. Define an integer r to represent the rank level of an individual. $\delta := \mathbb{P}$, $\mathbb{C}_p := \varnothing$, $\tau := 1$. Rank individuals in δ according to Pareto rule.
  while ($\delta \neq \varnothing$)
   for ($i := Q$ to 1) **do**
    Choose all individuals of the same rank level r from δ.
    Move them to $\mathbb{C}_p$ and delete it from δ.
   end for
  Add $\mathbb{C}_p$ to $\mathbb{C}$ and the first individual in $\mathbb{C}_p$ as center.
  $\mathbb{C}_p := \varnothing$. r++.
  end while
End

Appendix A.3. New Individual Generation

Algorithm A2: New Individual Generation

Input: Q individuals in population $\mathbb{P}$, the cluster $\mathbb{C}$.
Output: $\mathbb{X}$, $\mathbb{A}$.
Begin
Randomly generate a value ϑc in the range [0, 1).
if ($\vartheta_c < p_c$) then
  Select a cluster center randomly.
  Generate an individual to replace the chosen cluster center.
end if
for (i = 1 to Q)
  Randomly generate a value ϑ_g in the range [0, 1).
  if ($\vartheta_g < p_g$) **then**
    Randomly select a cluster and generate a random value ϑ_o in the range [0, 1).
      if ($\vartheta_o < p_o$) **then**
        Select a cluster center in the chosen cluster and a center of the first cluster to execute PPX operation to generate new individual.
        Perform PBM operation on this new individual. Evaluate and store this new individual in $\mathbb{X}$. Update $\mathbb{A}$ using $\mathbb{X}$.
      else
        Select a common individual randomly in the chosen cluster and a center of the first cluster to execute PPX operation to generate new individual. Perform PBM operation on this new individual.
        Evaluate and store this individual in $\mathbb{X}$. Update $\mathbb{A}$ using $\mathbb{X}$.
      end if
  else
    Randomly select two clusters and generate a random value rt in the range [0, 1).
    if ($\vartheta_t < p_t$) **then**
      Select a cluster center in the chosen clusters. Execute PPX operation on these to generate new individual.
      Perform PBM operation on this new individual. Evaluate and store this individual in $\mathbb{X}$. Update $\mathbb{A}$ using $\mathbb{X}$.
    else
      Select a common individual randomly in the chosen clusters, respectively. Execute PPX operation on these to generate new individual. Perform PBM operation on this new individual. Evaluate and store this individual in $\mathbb{X}$. Update $\mathbb{A}$ using $\mathbb{X}$.
    end if
  end if
end for
End

Appendix A.4. Schematic Diagrams and AND/OR Graphs

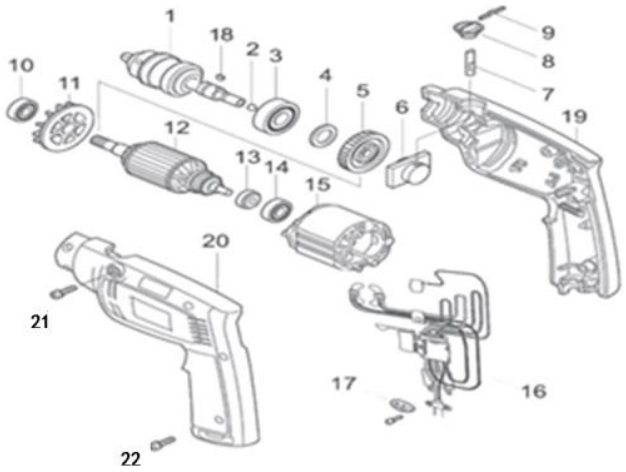

Figure A2. Schematic diagram of a hammer drill [12].

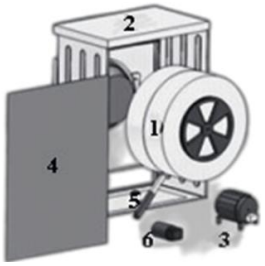

Figure A3. Schematic diagram of a washing machine [35].

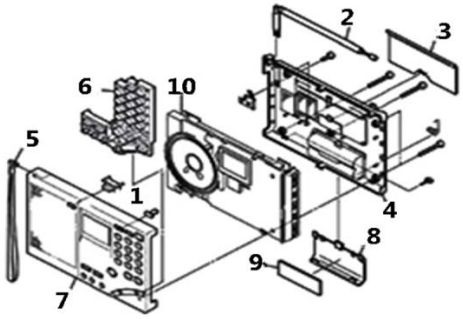

Figure A4. Schematic diagram of a radio set [4].

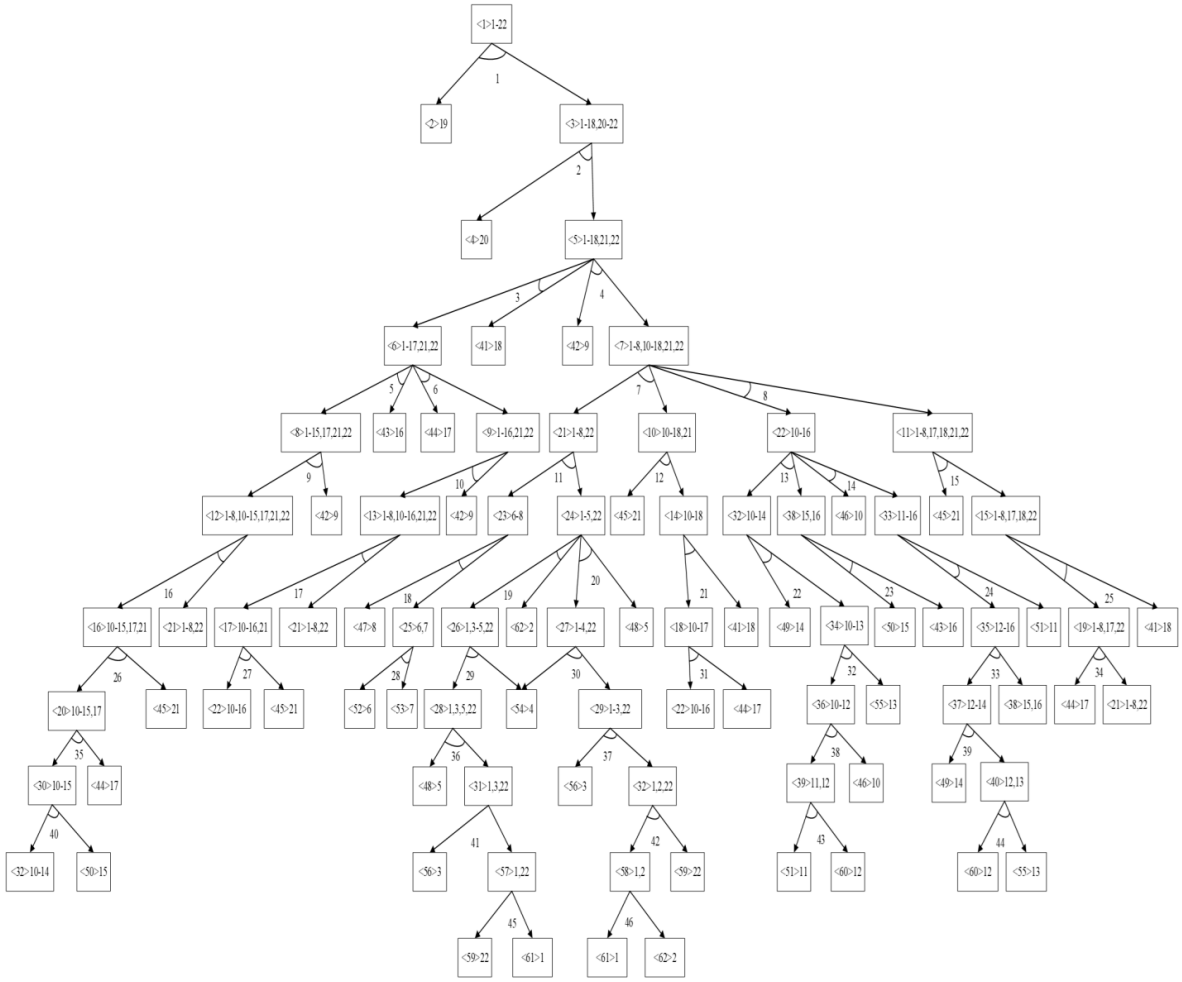

Figure A5. AND/OR graph of a hammer drill.

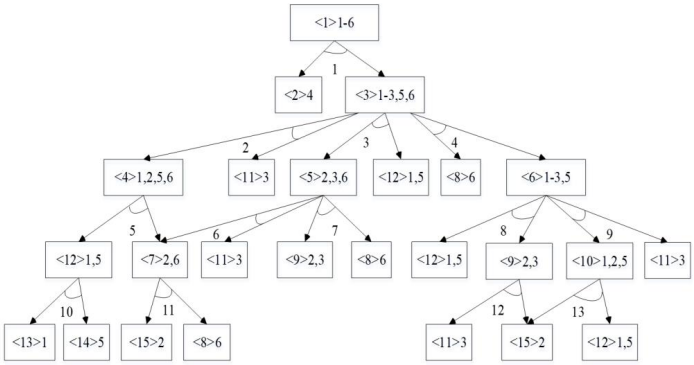

Figure A6. AND/OR graph of a washing machine.

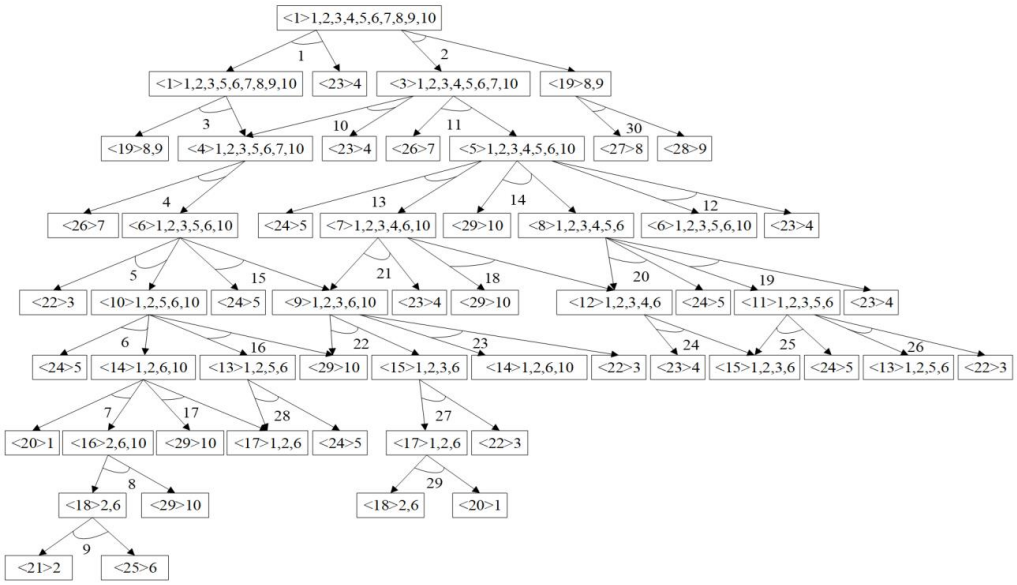

Figure A7. AND/OR graph of a radio set.

Appendix A.5. Two Boxplots

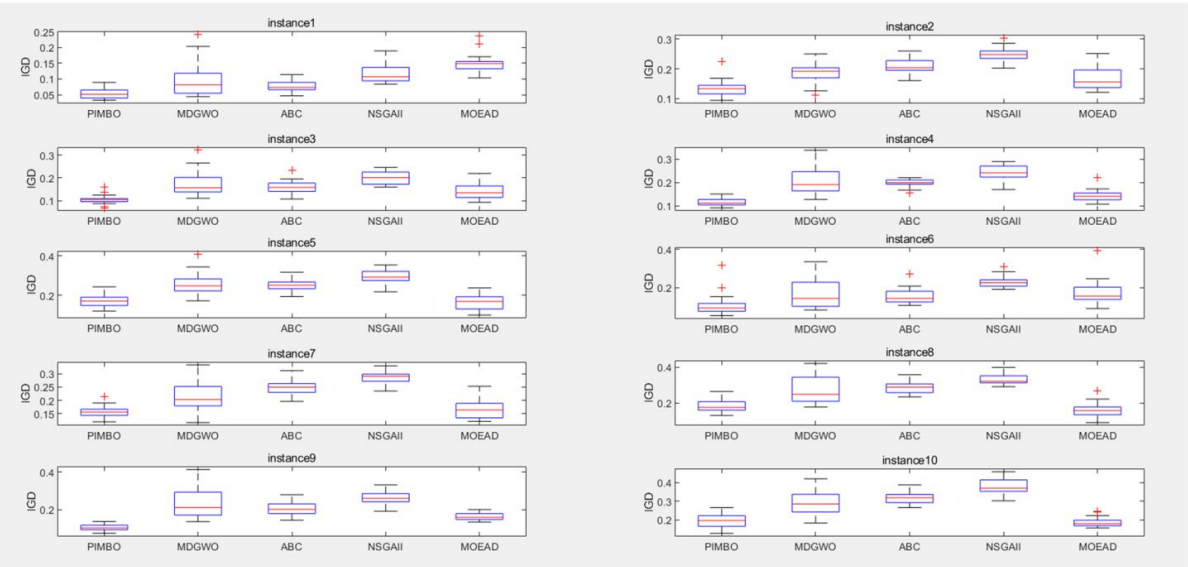

Figure A8. Boxplot of the experimental results of five algorithms via IGD metric.

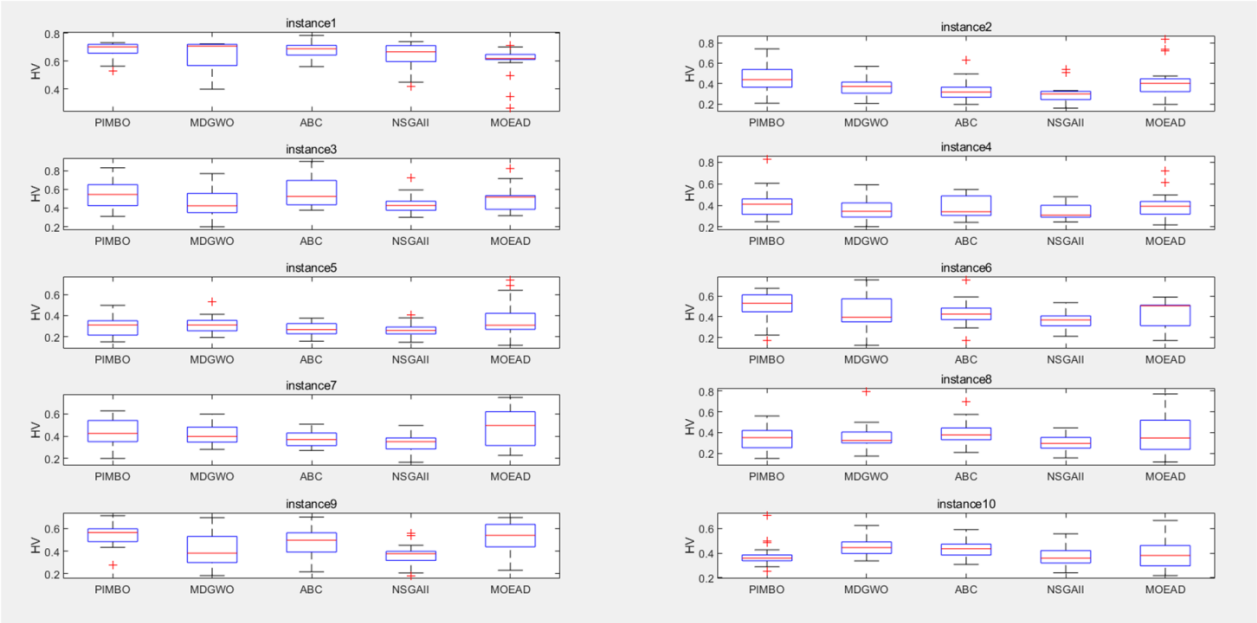

Figure A9. Boxplot of the experimental results of five algorithms via hypervolume metric.

Appendix A.6. Complexity Analysis

This section analyzes the complexity of the algorithms involved in PIMBO.

(1) Crossover operator (PPX): According to the analysis in [32], the time complexity of crossover operator is $O(J)$, where J represents the total number of operations.
(2) Mutation operator (PBM): According to the analysis presented in [27], the time complexity of mutation operator is $O(1)$.
(3) Pareto-based clustering: According to the analysis of Algorithm A1, the time complexity of Pareto-based clustering is $O(Q\log Q)$, where Q represents the number of populations.
(4) New individual generation: According to the analysis of Algorithm A2, the time complexity of new individual generation is $O(J)$, where J represents the total number of operations.

References

1. Tian, G.; Zhou, M.; Chu, J. A Chance Constrained Programming Approach to Determine the Optimal Disassembly Sequence. *IEEE Trans. Autom. Sci. Eng.* **2013**, *10*, 1004–1013. [CrossRef]
2. Guo, X.; Liu, S.; Zhou, M.; Tian, G. Disassembly Sequence Optimization for Large-Scale Products With Multiresource Constraints Using Scatter Search and Petri Nets. *IEEE Trans. Cybern.* **2015**, *46*, 2435–2446. [CrossRef]
3. Güngör, A.; Gupta, S.M. Disassembly line in product recovery. *Int. J. Prod. Res.* **2002**, *40*, 2569–2589. [CrossRef]
4. Lu, Q.; Ren, Y.; Jin, H.; Meng, L.; Li, L.; Zhang, C.; Sutherland, J.W. A hybrid metaheuristic algorithm for a profit-oriented and energy-efficient disassembly sequencing problem. *Robot. Comput. Manuf.* **2020**, *61*, 101828. [CrossRef]
5. Li, J.; Barwood, M.; Rahimifard, S. A multi-criteria assessment of robotic disassembly to support recycling and recovery. *Resour. Conserv. Recycl.* **2018**, *140*, 158–165. [CrossRef]
6. Çil, Z.A.; Mete, S.; Serin, F. Robotic disassembly line balancing problem: A mathematical model and ant colony optimization approach. *Appl. Math. Model.* **2020**, *86*, 335–348. [CrossRef]
7. Wegener, K.; Chen, W.H.; Dietrich, F.; Dröder, K.; Kara, S. Robot Assisted Disassembly for the Recycling of Electric Vehicle Batteries. *Procedia CIRP* **2015**, *29*, 716–721. [CrossRef]
8. Liu, J.; Zhou, Z.; Pham, D.T.; Xu, W.; Yan, J.; Liu, A.; Ji, C.; Liu, Q. An improved multi-objective discrete bees algorithm for robotic disassembly line balancing problem in remanufacturing. *Int. J. Adv. Manuf. Technol.* **2018**, *97*, 3937–3962. [CrossRef]
9. Liu, J.; Zhou, Z.; Pham, D.T.; Xu, W.; Ji, C.; Liu, Q. Collaborative optimization of robotic disassembly sequence planning and robotic disassembly line balancing problem using improved discrete Bees algorithm in remanufacturing. *Robot. Comput. Manuf.* **2020**, *61*, 101829. [CrossRef]
10. 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]

11. Fang, Y.L.; Liu, Q.; Li, M.Q.; Laili, Y.J.; Pham, D.T. Evolutionary many-objective optimization for mixed-model disassembly line balancing with multi-robotic workstations. *Eur. J. Oper. Res.* **2019**, *276*, 160–174. [CrossRef]
12. Pistolesi, F.; Lazzerini, B.; Mura, M.D.; Dini, G. EMOGA: A Hybrid Genetic Algorithm With Extremal Optimization Core for Multiobjective Disassembly Line Balancing. *IEEE Trans. Ind. Inform.* **2017**, *14*, 1089–1098. [CrossRef]
13. Mete, S.; Çil, Z.A.; Özceylan, E.; Ağpak, K. Resource Constrained Disassembly Line Balancing Problem. *IFAC-PapersOnLine* **2016**, *49*, 921–925. [CrossRef]
14. Wang, K.; Li, X.; Gao, L.; Garg, A. Partial disassembly line balancing for energy consumption and profit under uncertainty. *Robot. Comput. Manuf.* **2019**, *59*, 235–251. [CrossRef]
15. Kalayci, C.B.; Polat, O.; Gupta, S.M. A variable neighbourhood search algorithm for disassembly lines. *J. Manuf. Technol. Manag.* **2015**, *26*, 182–194. [CrossRef]
16. Homem de Mello, L.S.; Sanderson, A.C. AND/OR graph representation of assembly plan. *IEEE Trans. Robot. Autom.* **1990**, *6*, 188–199. [CrossRef]
17. Guo, X.W.; Zhou, M.C.; Liu, S.X.; Qi, L. Lexicographic Multiobjective Scatter Search for the Optimization of Sequence-Dependent Selective Disassembly Subject to Multiresource Constraints. *IEEE Trans. Cyber.* **2019**, *50*, 3307–3317. [CrossRef]
18. Bahubalendruni, M.V.A.R.; Varupala, V.P. Disassembly Sequence Planning for Safe Disposal of End-of-Life Waste Electric and Electronic Equipment. *Natl. Acad. Sci. Lett.* **2020**, *44*, 243–247. [CrossRef]
19. Guo, X.; Zhou, M.; Abusorrah, A.; Alsokhiry, F.; Sedraoui, K. Disassembly Sequence Planning: A Survey. *IEEE/CAA J. Autom. Sin.* **2020**, *8*, 1308–1324. [CrossRef]
20. Zhang, L.; Zhao, X.K.; Ke, Q.D.; Dong, W.F.; Zhong, Y.J. Disassembly line balancing optimization method for high efficiency and low carbon emission. *Int. J. Precis. Eng. Manuf.-Green Technol.* **2021**, *8*, 233–247. [CrossRef]
21. Shi, Y.H. An optimization algorithm based on brainstorming process. *Int. J. Swarm. Intell. Res.* **2011**, *2*, 35–62. [CrossRef]
22. Duan, H.; Li, S.; Shi, Y. Predator–Prey Brain Storm Optimization for DC Brushless Motor. *IEEE Trans. Magn.* **2013**, *49*, 5336–5340. [CrossRef]
23. Duan, H.; Li, C. Quantum-Behaved Brain Storm Optimization Approach to Solving Loney's Solenoid Problem. *IEEE Trans. Magn.* **2014**, *51*, 1–7. [CrossRef]
24. 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]
25. Bentaha, M.L.; Battaïa, O.; Dolgui, A. A sample average approximation method for disassembly line balancing problem under uncertainty. *Comput. Oper. Res.* **2014**, *51*, 111–122. [CrossRef]
26. Gungor, A.; Gupta, S.M. A solution approach to the disassembly line balancing problem in the presence of task failures. *Int. J. Prod. Res.* **2001**, *39*, 1427–1467. [CrossRef]
27. Zhang, Z.; Guo, X.; Zhou, M.; Liu, S.; Qi, L. Multi-objective Discrete Grey Wolf Optimizer for Solving Stochastic Multi-objective Disassembly Sequencing and Line Balancing Problem. In Proceedings of the 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Toronto, ON, Canada, 11–14 October 2020; pp. 682–687. [CrossRef]
28. Xie, J.; Gao, L.; Pan, Q.-K.; Fatih, T.M. An effective multi-objective artificial bee colony algorithm for energy efficient distributed job shop scheduling. *Procedia Manuf.* **2019**, *39*, 1194–1203. [CrossRef]
29. Zhang, Q.; Li, H. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [CrossRef]
30. Zhou, L.; Xiang, D.; Duan, G.H. Disassembly stability planning method in AND-OR graph model. *Microcomput. Inf.* **2008**, *24*, 1–3. [CrossRef]
31. Fu, Y.P.; Wang, H.F.; Huang, M. Integrated scheduling for a distributed manufacturing system: A stochastic multi-objective model. *Enterp. Inform. Syst.* **2019**, *13*, 557–573. [CrossRef]
32. Kongar, E.; Gupta, S.M. Disassembly sequencing using genetic algorithm. *Int. J. Adv. Manuf. Technol.* **2005**, *30*, 497–506. [CrossRef]
33. Nowakowski, P. A novel, cost efficient identification method for disassembly planning of waste electrical and electronic equipment. *J. Clean. Prod.* **2018**, *172*, 2695–2707. [CrossRef]
34. Laili, Y.J.; Li, Y.L.; Fang, Y.L.; Pham, D.T.; Zhang, L. Model review and algorithm comparison on multi-objective disassembly line balancing. *J. Manuf. Syst.* **2020**, *56*, 484–500. [CrossRef]
35. Zitzler, E.; Thiele, L.; Laumanns, M.; Fonseca, C.M.; Da Fonseca, V.G. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Trans. Evolut. Comput.* **2003**, *7*, 117–132. [CrossRef]
36. Fu, Y.; Ding, J.; Wang, H.; Wang, J. Two-objective stochastic flow-shop scheduling with deteriorating and learning effect in Industry 4.0-based manufacturing system. *Appl. Soft Comput.* **2018**, *68*, 847–855. [CrossRef]
37. Jia, Z.X.; Duan, H.B.; Shi, Y.H. Hybrid brain storm optimisation and simulated annealing algorithm for continuous optimisation problems. *Int. J. Bio-Inspir. Com.* **2016**, *8*, 109–121. [CrossRef]
38. Guo, X.; Wei, T.; Wang, J.; Liu, S.; Qin, S.; Qi, L. Multiobjective U-Shaped Disassembly Line Balancing Problem Considering Human Fatigue Index and an Efficient Solution. *IEEE Trans. Comput. Soc. Syst.* **2022**, 1–13. [CrossRef]
39. Deniz, N.; Ozcelik, F. An extended review on disassembly line balancing with bibliometric & social network and future study realization analysis. *J. Clean. Prod.* **2019**, *225*, 697–715.

40. Guo, X.W.; Zhang, Z.W.; Qi, L.; Liu, S.X.; Tang, Y.; Zhao, Z.Y. Stochastic hybrid discrete grey wolf optimizer for multi-objective disassembly sequencing and line balancing planning in disassembling multiple products. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 1744–1756. [CrossRef]
41. Ji, Y.J.; Liu, S.X.; Zhou, M.C.; Zhao, Z.Y.; Guo, X.W.; Qi, L. A machine learning and genetic algorithm-based method for predicting width deviation of hot-rolled strip in steel production systems. *Inf. Sci.* **2022**, *589*, 360–375. [CrossRef]
42. Zhao, Z.; Liu, S.; Zhou, M.; Abusorrah, A. Dual-objective mixed integer linear program and memetic algorithm for an industrial group scheduling problem. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 1199–1209. [CrossRef]
43. Zhao, Z.; Liu, S.; Zhou, M.; Guo, X.; Qi, L. Decomposition method for new single-machine scheduling problems from steel production systems. *IEEE Trans. Autom. Sci. Eng.* **2020**, *17*, 1376–1387. [CrossRef]
44. Wang, W.; Tian, G.; Chen, M.; Tao, F.; Zhang, C.; Ai-Ahmari, A.; Li, Z.; Jiang, Z. Dual-objective program and improved artificial bee colony for the optimization of energy-conscious milling parameters subject to multiple constraints. *J. Clean. Prod.* **2019**, *245*, 118714. [CrossRef]
45. Tian, Z.; Jiang, X.; Liu, W.; Li, Z. Dynamic energy-efficient scheduling of multi-variety and small batch flexible job-shop: A case study for the aerospace industry. *Comput. Ind. Eng.* **2023**, *178*, 109111. [CrossRef]