A hybrid metaheuristic algorithm for a profit-oriented and energy-efficient disassembly sequencing problem

Qi Lu [a], Yaping Ren [b] *, Hongyue Jin [c], Leilei Meng [d, e], Lei Li [f], Chaoyong Zhang [d] *, and John W. Sutherland [g]

[a] *School of Mechanical Engineering, Xi'an University of Science and Technology, Xi'an, China*

[b] *School of Intelligent Systems Science and Engineering, Jinan University (Zhuhai Campus), Zhuhai, 519070, China*

[c] *Department of Systems & Industrial Engineering, University of Arizona, Tucson, AZ 85721, USA*

[d] *The State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan, 430074, Hubei, China*

[e] *School of Computer Science, Liaocheng University, Liaocheng, 252059, China*

[f] *School of Mechanical Engineering, Hefei University of Technology, Hefei, China*

[g] *Environmental and Ecological Engineering, Purdue University, West Lafayette, IN 47907*

Abstract: Value recovery from end-of-life products plays a key role in sustainability and circular economy, which starts with disassembly of products into components for reuse, remanufacturing, or recycling. As the process is often complex, a disassembly sequencing problem (DSP) studies how to optimally disassemble products considering the physical constraints between subassemblies/disassembly tasks for maximum profit. With a growing attention on energy conservation, this paper addresses a profit-oriented and energy-efficient DSP (PEDSP), whereby not only the profit is maximized, but also energy consumption is accounted as an important decision criterion. In this work, a disassembly AND/OR graph (DAOG) is used to model a disassembly diagram for a product, in which the 'AND' and 'OR' relations illustrate precedence relationships between subassemblies. Based on the DAOG, we propose a hybrid multi-objective metaheuristic that integrates an artificial bee colony algorithm, a non-dominated sorting procedure, and a variable neighborhood search approach to solve the PEDSP for Pareto solutions. The proposed method is applied to real-world cases (i.e., a simple ballpoint pen and a relatively complex radio) and compared with other multi-objective algorithms. The results indicate that our method can quickly produce a Pareto front that outperforms the alternative approaches.

Keywords: Value recovery, disassembly sequencing, energy consumption, AND/OR graph, multi-objective metaheuristic

1. Introduction

In today's modern society, our demand for natural resource is ever increasing. A tremendous amount of new material, water, and energy are being consumed for manufacturing products that in turn get landfilled at end-of-life (EOL). With this cradle-to-

* Corresponding Authors.

E-mail addresses: renyp1@163.com (Y. Ren), zcyhust@hust.edu.cn (C. Zhang).

grave life cycle, the residual value of EOL products are lost as well as the embedded energy so that new resources are constantly extracted for production, raising environmental concerns such as resource depletion and global warming. As products such as electronics, vehicles, and industrial equipment often contain valuable components and materials that can be reused or recycled, value recovery from these products helps reduce virgin material and energy consumption and thus contribute to sustainability [1-4].

Disassembly has received increasing attention in recent years, as it constitutes the first step of value recovery. During the disassembly process, EOL products are separated into subassemblies including components and parts. The selection of disassembly sequence significantly affects the disassembly efficiency and recovered value, so the profit and energy consumption rely heavily on the disassembly plan. Consequently, a disassembly sequencing problem (DSP) has arisen to find the best order of product disassembly for maximum recovery profit and disassembly efficiency [5, 6].

To solve the DSP, a disassembly diagram is used to model the disassembly process mathematically. The disassembly diagram can exhibit the assembly structure of a product and the physical connections (e.g., precedence relationships) between subassemblies. The existing literature adopted metrics such as disassembly tree, directed graph, AND/OR graph, hierarchical tree diagram, and disassembly precedence diagram to describe a disassembly process [7-11]. Among these tools, AND/OR graph is excellent for disassembly modeling [12-14] – A disassembly AND/OR graph (DAOG) not only depicts the complete set of possible sequences, but also clearly shows the assembly relationships between subassemblies. It is especially important for selective/partial disassembly, as the DAOG enables us to determine which components and parts are left for reuse or recycling. Other disassembly diagrams, such as directed graph, disassembly tree, and disassembly precedence diagram, cannot explicitly point out the recoverable components because they are constructed via parts in a product without any component information. Therefore, this paper utilizes AND/OR graph for disassembly planning.

This paper proposes a hybrid multi-objective metaheuristic (HMM) to efficiently solve a profit-oriented and energy-efficient DSP (PEDSP). Although an exact solution approach may be applied to find the optimum disassembly sequence of a DSP, it is limited by the size and complexity of disassembly instances (products). In other words, exact algorithms may become computationally intractable [15, 16] since the DSP is a combinatorial optimization problem with a NP-complete challenge [17, 18]. For this reason, (meta)heuristic methods are good candidates for solving DSPs. However, limited papers study the DSP modeled by the DAOG using an efficient metaheuristic approach, so this paper targets to fill this specific research gap.

HMM is based on the framework of an artificial bee colony (ABC) algorithm, a popular metaheuristic that delivers excellent solutions for many optimization problems [19]. In HMM, we first present a double-phase heuristic to decode the DAOG to rapidly generate feasible disassembly sequences. Second, a non-dominated sorting (NS) procedure is introduced to identify the nondominated (Pareto) solutions for two objectives: maximum recovery profit and minimum energy consumption. Finally, a variable neighborhood descent (VND) approach is adopted to improve the quality of the nondominated solutions.

The fundamental contributions of this work are summarized as follows:

(1) We developed a multi-objective PEDSP to maximize both profit and energy conservation from product disassembly. The existing literature focuses on maximizing recovered value or disassembly efficiency in DSPs without much attention to the energy efficiency. Due to the high carbon footprint of the current energy mix for disassembly [20-24], we included energy efficiency as an important decision criterion.

(2) Based on the DAOG, we established three relationship matrices, i.e., a precedence, exclusive, and incidence matrices, to model the PEDSP.

(3) By combining ABC, NS, and VND, HMM is developed to generate a high-quality Pareto front for Pareto solutions.

(4) Through testing two real-world cases and comparing with two other multi-objective optimization techniques, we demonstrate the efficiency and effectiveness of the proposed method in solving the PEDSP.

The rest of this paper is organized as follows. Section 2 reviews the relevant literature. Section 3 describes modeling of the DSP using DAOG. Section 4 presents the proposed method. Section 5 shows the computational results of two case studies as well as comparisons with alternative solution approaches. Finally, Section 6 concludes the paper and provides future work directions.

2. Literature review

DSPs have been studied by academia and industry over the past decades, and many solution approaches have been developed for DSPs, including exact approaches or (meta)heuristics. The early research focused on the exact algorithms for tackling DSPs with limited problem sizes and complexities. For example, Johnson and Wang [25] constructed a disassembly tree to describe a disassembly process. The authors established an integer linear programming model based on a two-commodity network flow formulation to find an optimal solution with maximum profit [25]. Lambert also proposed an integer linear programming model to maximize profit by applying an AOG for graphical representation of the complete set of possible disassembly operations [26]. Kang et al. presented an integer programming model for disassembly sequencing by modifying the shortest path problem [27].

As such, the exact methods rely on mathematical programming, meaning that the disassembly model must be formally defined. The optimal solutions are found by means of an optimization software, such as LINDO by Johnson and Wang [25], AIMMS by Lambert [26], and CPLEX by Kang et al. [27]. This dependency on an exploratory tool could lead to an unpredictable computation time [28]. Furthermore, the optimization solver could not guarantee the solution optimality for nonlinear models.

These limitations have inhibited the wide application of exact solution algorithms and motivated the development of (meta)heuristics to find near optimal solutions within reasonable computational time. Especially in recent years, the studies on metaheuristics and heuristics are rapidly increasing as the problem size and complexity of EOL products increase in the modern industry. For example, Smith et al. [29] utilized expert rules and cost-benefit analysis to reduce the search space of disassembly solutions. Then, an optimized partial disassembly sequence was determined with the goals of maximizing economic benefits and minimizing environmental costs. Sanchez and Haas [30] studied the disassembly process of building components. A selective disassembly sequence planning method with rule-based recursive analyses was proposed to achieve minimal environmental impacts and removal costs for recovery of target components from buildings. These two algorithms utilize heuristic rules to significantly reduce the problem size and complexity. While the optimal solutions cannot be identified, near optimal solutions are obtained with reasonable computational efforts.

In the existing literature on solving DSPs by metaheuristics, intelligent algorithms are the most popular. Kheder et al. [31] used a genetic algorithm (GA) to optimize a disassembly process by various criteria including maintainability of components, part volume, tool changes, and disassembly direction changes. Due to the multiple decision criteria, the DSP was able to obtain

satisfactory results in a short CPU time. Ren et al. [11] studied a complicated asynchronous parallel DSP. Two matrices, i.e. precedence and collision matrices, were defined to determine feasible disassembly solutions. Based on the matrices, a GA was adapted to efficiently tackle the DSP. Ren et al. [7] employed a hierarchical disassembly tree to model a parallel DSP. A heuristic method was presented to rapidly create feasible disassembly solutions, while an artificial bee colony (ABC) approach was developed to find the Pareto solutions for maximum profit and minimum makespan of the disassembly process. Percoco and Diella [32] also adopted an ABC to solve a multi-objective DSP, while they simplified the multiple objectives to a weighted sum equation. Several heuristic rules were applied to reduce the problem complexity such that the optimal or near optimal solutions can be found. Zhong et al. [33] studied a functional component and fastener-based DSP. To reduce the problem complexity, the authors first solved a functional component-based DSP by Dijkstra's algorithm. Then, they considered the combination of components and fasteners and solved the DSP via a particle swarm optimization.

By analyzing the existing literature as above, we conclude that metaheuristics require development of efficient heuristic rules to identify an optimal or near optimal solution within reasonable computational time.

3. Problem statement

This section first describes a DAOG for disassembly of a product with a ballpoint pen as an example. Then, three relationship matrices are defined to model the PEDSP. Lastly, we establish a mathematical programming for the PEDSP based on the DAOG and three relationship matrices.

3.1. DAOG

A DAOG can illustrate the connection and precedence relationships among subassemblies as well as the disassembly tasks. As will be demonstrated in the subsequent example, AND/OR relations are not trivial for a simple product like ballpoint pen, and the number of possible sequences in a DAOG is associated with the product complexity as well as the problem size. It should be noted that subassemblies may represent components or parts that cannot be separated any more, while fasteners such as bolts and screws are not shown in a DAOG to simplify a disassembly diagram. We first use a ballpoint pen [34] to represent a DAOG. Fig. 1 shows the physical structure of a ballpoint, which is then converted into a DAOG as depicted in Fig. 2.

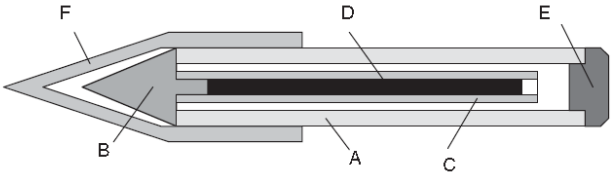


Fig. 1. Assembly drawing of a ballpoint pen [34].

In Fig. 2, the disassembly tasks are represented with integers, 1, 2, …, and *J*, where *J* is the total number of possible tasks. Nodes in the graph represent subassemblies of a product. Each node corresponds to a subassembly indexed from (1) to (*N*) where *N* is the number of all possible subassemblies. Let subassembly *a* be called the parent of subassembly *b,* if *b* is obtained by disassembling *a*; subsequently, *b* is called a child of *a*, for *a*, $b \in \{1, 2, \cdots, N$. For example, subassembly (2) (ABCDE) is the parent of (6) (BCD) and (8) (AE), while (2) (ABCDE) is a child of (1) (ABCDEF).

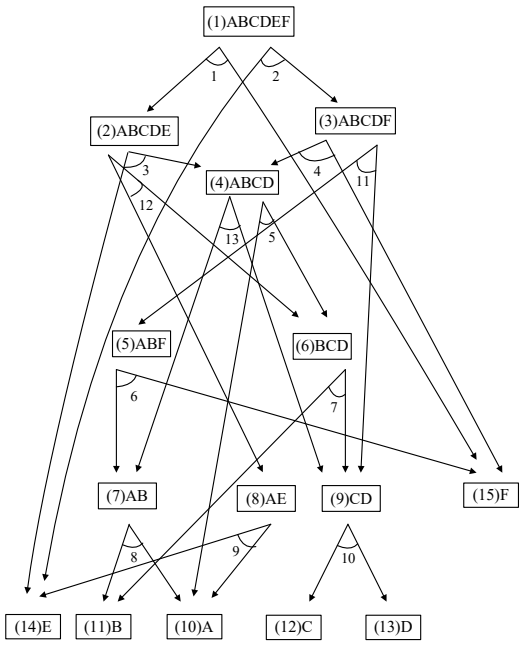


Fig. 2. DAOG of the ballpoint pen.

As shown in the DAOG, each disassembly task is denoted by a hyperarc which connects two child subassemblies (i.e. AND relation) with a parent subassembly. The AND relation guarantees that child subassemblies can be obtained only after completing the disassembly task of their parent subassembly (e.g., subassembly (4) would be released by performing task 3 or 4; subassemblies (6) and (8) would be realized after task 2). A parent subassembly may have more than one hyperarc, such as subassembly (3), which form the OR relation because it is feasible to undertake any of the disassembly tasks. Therefore, the diversity of disassembly options is precisely illustrated by the OR relation. Apparently, different hyperarcs from the same parent are mutually exclusive. For instance, it is impossible to perform tasks 4 and 11 simultaneously in one disassembly sequence. This implies that there is an exclusive OR (EOR) relation in the DAOG. From Fig. 2, it can be observed that there are multiple disassembly alternatives, although the pen has only six parts. As such, the AND and OR relations could be complex even for a simple product.

According to the characteristics of the DAOG, we can derive three matrices P, E, and I for mathematical modeling of the PEDSP as follows.

(1) A precedence matrix P is developed to ensure the precedence constraints among subassemblies are satisfied (i.e., parent subassemblies should be removed prior to their children). For example, in Fig. 2, we must complete task 1 before starting task 3. Each element of the matrix P is defined as

$$p_{jk} = \begin{cases} 1, & \text{if task } \ j \ \text{ must be performed immediately prior to task } k \\ 0, & \text{otherwise} \end{cases}$$

Precedence matrix P of the ballpoint pen is given by

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

(2) As aforementioned, we can undertake at most one task (hyperarc) from a parent subassembly due to the EOR relation. Therefore, we use an exclusive matrix $E=[e_{jk}]$ to define the EOR relations as follows:

$$e_{jk}=\begin{cases}-1, & \text{if tasks } j \text{ and } k \text{ are mutually exclusive}\\ 0, & \text{otherwise}\end{cases}$$

The resulting matrix E is symmetric, and that of the ballpoint pen is as follows:

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

(3) To facilitate profit calculation, an incidence matrix $I = [i_{aj}]$ is constructed to model the relationships between subassemblies and tasks as follows.

$$i_{aj}=\begin{cases}1, & \text{if subassebmly } a \text{ is obtained from task } j\\ -1, & \text{if subassebmly } a \text{ is disassembled by task } j\\ 0, & \text{otherwise}\end{cases}$$

For example, the incidence matrix I of the pen is formulated as

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

3.2. Mathematical programming of the PEDSP

Based on the DAOG and three relationship matrices, we formulate a mathematical model for the PEDSP with two objectives: maximizing profit and minimizing energy consumption. The relevant indices are first defined as

(1) a: The index of a subassembly, a =1, 2, …, N, where N is the number of subassemblies in a product;

(2) j, k, m: The indices of tasks, j, k, m = 0, 1, 2,…, J, where J represents the number of tasks and Task 0 refers to a dummy task of creating a product;

Accordingly, the decision variables are expressed as

(1) x_j = 1, if task j is performed, otherwise x_j = 0; notably, $x_0 \equiv 1$;

(2) y_{jk} = 1, if task k is performed immediately after completing task j, otherwise y_{jk} = 0. This notation is developed to account intermediate operations between two adjacent tasks such as change of disassembly tools and transportation of subassemblies, which also consume time and energy.

Other constant variables are defined as follows.

c_j : Cost of completing task j;

C_{jk}: Cost of state transformation y_{jk};

q_{jk}: Power of state transformation y_{jk};

Q: The maximum energy consumption allowed for a disassembly process;

R_a: Recycling/reuse value of subassembly a;

t_j : Consumed time of completing task j;

T_{jk}: Consumed time for state transformation y_{jk};

w_j : Power of completing task j;

With these notations, revenue from disassembly can be calculated iteratively based on a disassembly sequence. Suppose that a disassembly sequence is task 1→3 from Fig. 2, then its revenue is computed by Eq. (1).

$$(i_{11}\cdot R_1 + i_{21}.R_2 + i_{15,1}\cdot R_{15}) + (i_{23}\cdot R_2 + i_{43}\cdot R_4 + i_{14,3}\cdot R_{14}) = (-R_1 + R_2 + R_{15}) + (-R_2 + R_4 + R_{14}) \quad (1)$$

Here, the first and second terms represent revenues from completing tasks 1 and 3, respectively.

The PEDSP is mathematically described by formulations (2)-(9):

$$\text{Max} \quad f_1 = \sum_{j=1}^{J}\sum_{a=1}^{N} i_{aj} R_a x_j - \sum_{j=1}^{J}\sum_{k=1}^{J} C_{jk} y_{jk} - \sum_{j=1}^{J} c_j x_j \tag{2}$$

$$\text{Min} \quad f_2 = \sum_{j=1}^{J}\sum_{k=1}^{J} q_{jk} T_{jk} y_{jk} + \sum_{j=1}^{J} w_j t_j x_j \tag{3}$$

$$\text{s.t.} \quad \sum_{j=1}^{J} x_j \geq 1 \tag{4}$$

$$x_k = \sum_{j=0}^{J} y_{jk}, \quad k = 1, 2, \ldots, J; \tag{5}$$

$$x_k \leq \sum_{j=0}^{J} p_{jk} x_j, \quad k = 1, 2, \ldots, J; \tag{6}$$

$$e_{jk}(x_j + x_k) \geq -1, \quad j, k = 1, 2, \ldots, J; \tag{7}$$

$$\sum_{j=1}^{J}\sum_{k=1}^{J} q_{jk} T_{jk} y_{jk} + \sum_{j=1}^{J} w_j t_j x_j \leq Q; \tag{8}$$

$$\sum_{j=1}^{J} y_{jm} \geq \sum_{k=1}^{J} y_{mk}, \quad m = 1, 2, \ldots, J; \tag{9}$$

$$x_j, y_{kj} \in \{0,1\}, \quad j, k = 1, 2, \ldots, J; \tag{10}$$

Objective function f_1 is to maximize profit from disassembly of a product, which is the difference between total revenue (i.e., the first term of Eq. (2)) and costs associated with intermediate operations and disassembly tasks (i.e., the second and third terms of Eq. (2)). Objective function f_2 is to minimize energy consumption of the disassembly process, where the first term depicts consumed energy for intermediate operations, and second term is energy consumption of disassembly tasks. Constraint (4) guarantees that at least one task, excluding task 0, is performed in a disassembly process. Constraint (5) represents the relationship of two decision variables. Constraint (6) shows the precedence relationships among tasks, i.e., a task can be performed if and only if one of the tasks immediately prior to it is completed. Constraint (7) ensures that any pairwise tasks with EOR cannot be simultaneously performed in a disassembly process. Constraint (8) limits the total energy consumption by the maximum allowed value. Constraint (9) represents the equilibrium relationship of in-degree and out-degree of a subassembly, where $\sum_{j=1}^{J} y_{jm}$ and $\sum_{k.=1}^{J} y_{mk}$ is in-degree and out-degree of task m, respectively. Note that the disassembly process will stop after completing task m if and only if $\sum_{j=1}^{J} y_{jm} = 1$ and $\sum_{k.=1}^{J} y_{mk} = 0$. Constraint (10) defines the decision variables.

4. The proposed approach

This section presents the proposed algorithm-HMM, which is developed based on the disassembly model formulated in

Section 3.2. As DSP is NP-complete, (meta)heuristic methods are suitable for solving such complex problems [7, 35, 36]. Compared to other metaheuristics, our proposed HMM is superior in efficiently producing a variety of feasible disassembly sequences and creating a high-quality Pareto front for the PEDSP.

ABC simulates the behavior of bees searching for food or nectar, which includes employed bee phase, onlooker bee phase, and scout bee phase [37]. In the ABC, a food source refers to a solution or individual. Due to its simplicity and ease of implementation, the ABC algorithm has been used to solve many practical and complex industrial optimization problems [38]. However, the basic ABC algorithm was originally designed for continuous function optimization rather than our proposed integer programming model. Therefore, we first adapt the basic ABC to a discrete version for solving the PEDSP. Then Pareto solutions are found via a NS procedure. In addition, a local search operator, i.e., VND, is employed to further improve the exploitation ability of the ABC, which searches the neighborhood spaces of the current best solution and continually updates local optimum so as to approach the global optimum.

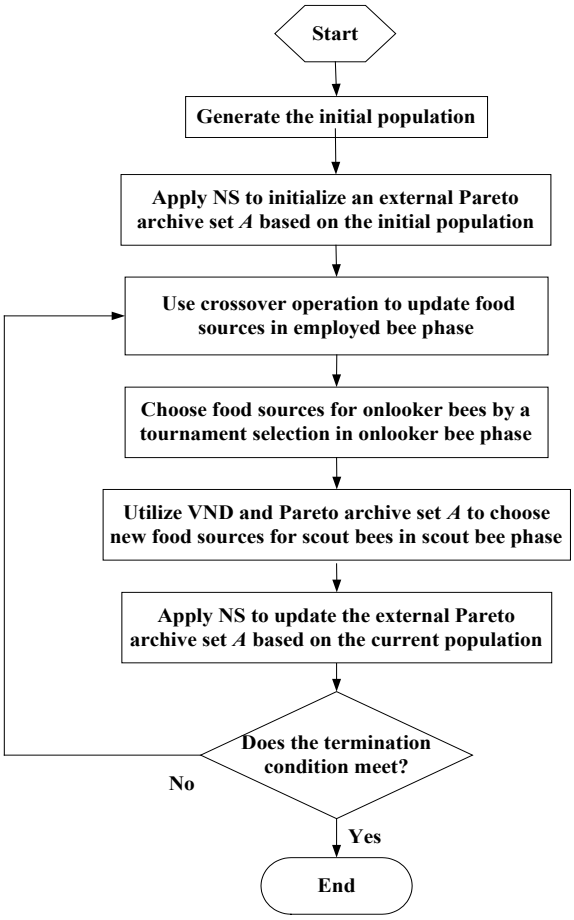


Fig. 3. An overview of HMM.

An overview of HMM is shown in Fig. 3. There are five stages of operations including (1) solution representation and initialization, (2) non-dominated sorting, (3) employed bee phase, (4) onlooker bee phase, and (5) scout bee phase that will be discussed in more details in subsections of 4.1 to 4.5. The algorithm repeats the process (except for the first initialization step)

until it reaches a terminating condition. In this paper, we define a maximum iteration g_{max} so that HMM will stop upon reaching that number of iterations.

4.1. Solution representation and initialization

A good solution representation clearly illustrates a solution and facilitates decoding of a solution. In the DAOG, a disassembly sequence is represented by a set of subassemblies or tasks. To be consistent with the disassembly model of Section 3, the task indices are used to denote disassembly solutions. For example, a complete disassembly sequence of Fig. 2 is {2, 4, 13, 8, 10}, where the tasks are sequentially performed from the leftmost one to the rightmost one. Given such a disassembly solution representation, the profit and energy consumption can be easily calculated by Equations (2) and (3).

The challenge of generating initial population is to ensure the solution feasibility. Unlike other disassembly diagrams such as directed graph [10] and hierarchical tree diagram [7, 10], DAOG has an EOR relation in addition to the precedence relationship, which is not well studied in the existing metaheuristic approaches. To tackle this challenge, we propose a double-phase heuristic to efficiently generate feasible disassembly sequences that satisfy the DAOG requirements. For the double-phase heuristic, a double-vector list structure is defined as $v = \{v^1, v^2\}$, where $v^1 = \{s_1, \ldots, s_j, \ldots, s_J\}$ represents a sequence of disassembly tasks with each element corresponding to a task, and $v^2 = \{q_1, \ldots, q_j, \ldots, q_J\}$ is a binary vector such that q_j is 1 if s_j of v^1 is performed, and otherwise 0. For instance, $v^1 = \{2, 4, 3, 5, 8, 6, 7, 9, 1\}$ and $v^2 = \{1, 1, 0, 0, 1, 0, 0, 1, 0\}$ means that tasks 2 (s_1), 4 (s_2), 8 (s_5) and 9 (s_8) of v^1 are carried out since q_1, q_2, q_5 and q_8, are equal to 1 in v^2. Based on this double-vector list structure, the double-phase heuristic is presented as follows.

Phase 1. Adjust v to v' by eliminating the EOR tasks of v:

As aforementioned in Section 3.1, tasks with EOR relations cannot be simultaneously performed in a disassembly process. Thus, we first eliminate exclusive tasks by the following 9 steps.

Step 1: Start.

Step 2: Randomly generate v, i.e., v^1 and v^2, where v^1 is a disassembly sequence consisting of tasks denoted by integers from s_1 to s_J, and v^2 is a binary array whose elements are all initialized to 1.

Step 3: Set $j = 1$.

Step 4: If q_j is equal 0, go to Step 7. Otherwise, identify the exclusive tasks of s_j based on matrix E by examining tasks from s_{j+1} to s_J in v^1, and store them into a set G.

Step 5: If G is empty, go to Step 7. Otherwise, go to Step 6.

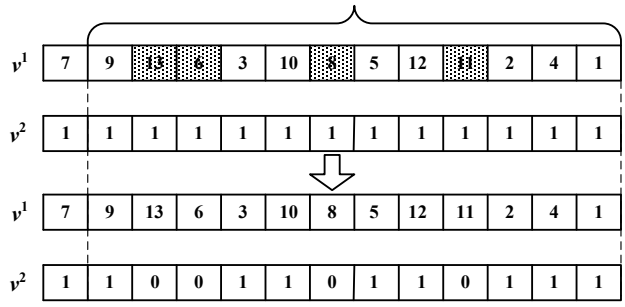
Step 6: If there are tasks in G whose corresponding elements in v^2 are 1, the elements are reset as 0, and let G become an empty set. Otherwise, reset G as an empty set.

Step 7: $j = j + 1$.

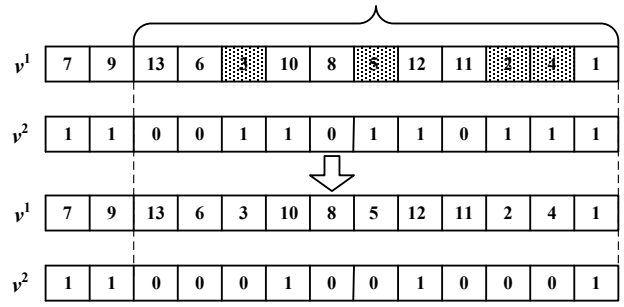
Step 8: If j is equal J, go to Step 9. Otherwise, go to Step 4.

Step 9: Stop the procedure and v' is obtained.

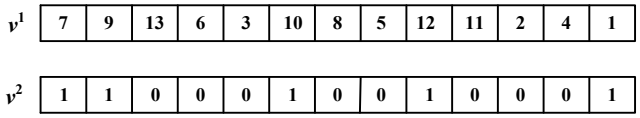
Taking the ballpoint example depicted in Fig. 2, a random solution v is first generated from Steps 1-2. As shown in Fig. 4 (a), v is then adjusted by removing the exclusive tasks of s_7. Thereafter, tasks such as s_6, s_8, s_{11} and s_{13} are marked in shade meaning that they won't be undertaken (i.e., q_6, q_8, q_{11} and q_{13} become 0 in v^2). Repeat this procedure for the rest of the tasks in v^1 so that no exclusive tasks will be performed in v. Fig. 4 (c) shows the final adjusted v, i.e., v'.



(a) The first adjustment of *v*.



(b) The second adjustment of *v*.



(c) *v'*

Fig. 4. The adjustment from a random solution *v* to *v'* for the ballpoint example in **Phase 1**.

Phase 2. Further adjust *v'* to *v''* for satisfying the precedence constraints.

The first phase focuses on meeting the EOR relation of a disassembly plan, but the generated solution *v'* is not justified for the precedence relation. For example, in Fig. 4 (c), *v'* is {7, 9, 10, 12, 1}, but it is infeasible to perform task 7 prior to task 1. To correct the precedence relationship, we further adjust the disassembly sequence *v'* obtained from **Phase 1**.

First, we count the number of immediate predecessors (NIPs) of each task and tasks can be performed when their NIPs equal 0. Here, the NIPs of task *j* is calculated as sum of the cells in column *j* of the precedence matrix *P*. The priorities among tasks vary dynamically as tasks are completed one-by-one, and NIPs of each task is reduced when its predecessors are removed. Taking the final *v'* of Fig. 4(c) as an example, the initial information on the immediate predecessors of all tasks are listed in Table 1. In Tables 1 and 2, the symbol " / " represents the tasks that are not performed in *v*, " ∅ " denotes the tasks that have no immediate predecessors at this time, and " -1 " is assigned to the NIPs of a completed task. Now let task 1 be disassembled first, then Table 1 is updated to Table 2. Because task 1 is the immediate predecessor of task 12, the NIPs of task 12 becomes 0 after completing task 1. When there are multiple tasks with NIPs of 0, one task is randomly selected from the task pool and gets performed until no eligible task is left. Finally, a feasible disassembly sequence is generated for a product. The detailed steps of **Phase 2** are as follows.

Step 1: Start.

Step 2: Extract the tasks to be performed from *v'* and aggregate them into an updated disassembly sequence *v'* (e.g., *v'* = {7, 9, 10, 12, 1}, shown in Fig. 5, is obtained from Fig. 4(c)).

Step 3: According to matrix *P*, initialize NIPs of *v'* (e.g., row 3 of Table 1).

Step 4: Based on NIPs, choose one task with the highest priority and perform the task. If there is no task left, go to Step 6. Otherwise, go to Step 5.

Step 5: Update matrix *P* and NIPs. Then, go back to Step 4.

Step 6: Stop the procedure, as a feasible disassembly sequence *v"* is derived (e.g., Fig. 5).

Let *PS* denote the population size and initialize the *PS* number of food sources according to the double-phase heuristic. The double-phase heuristic is employed to adjust new food sources and ensure the feasibility of the crossover (Section 4.3) and neighborhood search operations (Section 4.5).

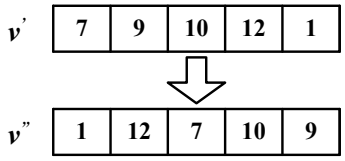


Fig. 5. The adjustment from *v'* to *v"* in **Phase 2**.

Table 1

The initial disassembly information of NIPs.

Task index (*j*)	1	2	3	4	5	6	7	8	9	10	11	12	13
Immediate predecessors of task *j*	∅	/	/	/	/	/	12	/	12	7	/	1	/
NIPs	0	/	/	/	/	/	1	/	1	1	/	1	/

Table 2

The updated information of NIPs.

Task index (*j*)	1	2	3	4	5	6	7	8	9	10	11	12	13
Immediate predecessors of task *j*	∅	/	/	/	/	/	12	/	12	7	/	∅	/
NIPs	-1	/	/	/	/	/	1	/	1	1	/	0	/

4.2. Multi-objective optimizer-NS

This paper uses a NS algorithm proposed by Deb et al. [39] to divide the population solutions into several levels according to their dominated solution number. In addition, an external archive set *A* is set to store the top solutions that cannot be dominated by other solutions up to the current population. In each generation, all nondominated (Pareto) solutions are regarded as candidate solutions to update *A*. In addition, all dominated solutions are removed from *A*.

4.3. Employed bee phase

The employed bees start from the current food source (solution) and try to search new food sources. If a new food source dominates the old one, the employed bee will abandon the old food source and choose the new one; otherwise, it will stay on

the old food source. Differing from the basic ABC, a crossover operation is applied to search new food sources for employed bees due to the discrete feature of our problem. In this work, we implement a double-point crossover operation [35], in which a crossover section is obtained from another randomly chosen food source of an employed bee and inserted into the same position of the food source of the current employed bee.

From the double-phase heuristic, we can conclude that a feasible disassembly solution only depends on v^1 in v and both v' and v'' carries less disassembly information of a product. Thus, v^1 is crossed in the crossover process. Similarly, the neighborhood structures of v^1 is constructed in a VND process of Section 4.5.

4.4. Onlooker bee phase

Each onlooker bee selects a food source based on the percent of the nectar amount of each food source among the total nectar amounts in the basic ABC algorithm. This strategy is a fitness-based selection technique, such as the roulette wheel selection method. However, a fitness-based selection strategy does not seem applicable to the Pareto solutions. For this reason, we introduce a tournament selection with the size of 5, in which five food sources are picked randomly from the population, and then the nondominated solution will be selected by the onlooker bee. If there is more than one nondominated solution, one of them is randomly selected. Similar to the employed bee phase, the selected food source will be assigned to the onlooker bee if and only if it can dominate the current solution, and otherwise the onlooker bee still stays on the current food source.

4.5. Scout bee phase

In this phase, a parameter l_{max} is predetermined to limit the iteration number l of a food source without any improvement in ABC. l is initialized to be 0 for each food source. If a food source has not been improved after l_{max} cycles, the food source is abandoned and replaced with a new food source by the scout bees. Like the practical situation, the nectar amount of a food source reduces with the increase of visits by bees, when the nectar shortage occurs in the food source, bees will abandon it and search a new food source.

The scout bee phase is devoted to escaping from local optima, but there is a possibility that abandoned food sources are not fully exploited only via employed and onlooker bees. This may result in missing the valuable information, i.e., high-quality subsequences in v^1 carried by abandoned food sources. To solve this, we draw on a VND process to further search the neighborhood space of abandoned food sources.

VND is a metaheuristic that starts with an initial solution and continually searches better solutions to escape the local optimum trap using various neighborhood structures. Suppose that $X(v^1)$ is the set of all neighbors of sequence v^1 and $X_h(v^1)$ is all neighbors in the h^{th} neighborhood structure of v^1, where $h = 1, \ldots, h_{max}$ and h_{max} represents the number of the neighborhood structures, namely, $X(v^1) = \{X_1(v^1), \ldots, X_h(v^1), \ldots, X_{h_{max}}(v^1)\}$. The pseudocode of VND is presented in **Algorithm 1**. It can be observed from **Algorithm 1** that the neighborhood space of v^1 is searched in a deterministic way from $X_1(v^1)$ to $X_{h_{max}}(v^1)$ and the best v^1 is obtained after exploring h neighborhood structures. Furthermore, the construction of neighborhood structures directly affects the performance of VND. Five neighborhood structures are created by Ren et al [9] in a disassembly line balancing problem, i.e., *left exchange*, *right exchange*, *left insert*, *right insert*, and *2-opt* operator. Left and right exchanges are same operations but exchanging positions in different directions, so do left and right inserts. We combine these neighborhood

structures into *exchange* and *insert* by ignoring the direction. Hence, *exchange*, *insertion*, and *2-opt* are adopted to construct the neighborhood structures.

After the completion of **Algorithm 1**, if the resulting neighbor solution cannot be dominated by the abandoned solution, the scout bee will adopt the resulting neighbor solution as a new food source. Otherwise, the scout bee randomly selects a solution from A, where the Pareto solutions often carry better information than others and the search space around these Pareto solutions could be the most promising region. Finally, l is updated as 0 if $l = l_{max}$ or $l + 1$ if $l < l_{max}$.

Algorithm 1 VND.

Input: v^1, h_{max}
Output: v^1
$h = 1$
While ($h \leq h_{max}$) **Do**
  /* Local search */
  $v^1{}_h \leftarrow$ Find the best neighbor in $X_h(v^1)$
  If $v^1{}_h$ is better than v^1 **then**
    $v^1 \leftarrow v^1{}_h$
    $h \leftarrow 1$
  Else
    $h \leftarrow h + 1$
  End if
End While

5. Experimental Results and Analysis

This section presents experimental results from solving PEDSP using HMM. First, two products of different sizes are applied to demonstrate the feasibility and effectiveness of the proposed algorithm in Section 5.1. The first product is the ballpoint pen shown in Fig. 2, and the second one is a larger product – radio set used by Lambert [26]. The DAOG is depicted in Fig. 6. Second, the control parameters of HMM such as g_{max}, PS, and l_{max} are tuned in Section 5.2. Finally, two other multi-objective metaheuristics, i.e., a multi-objective discrete artificial bee colony (MODABC) algorithm [7] and a non-dominated sorting genetic algorithm (NSGA-II) [39], are applied to compare the performance with our method in Section 5.3. MODABC was recently proposed by Ren et al. [7] to tackle a parallel DSP based on a hierarchical disassembly tree. NSGA-II, also described in Section 4.2, is a popular metaheuristics proposed by Deb et al. [39] to solve a multi-objective optimization problem. Because both algorithms are not initially developed to solve the DAOG-based PEDSP, we first adapt them by applying the same solution representation and objective functions of HMM. All tests were run on Intel(R) Xeon CPU and 3.50 GHz 16 GB RAM, and all algorithms are coded in MATLAB R2017a.

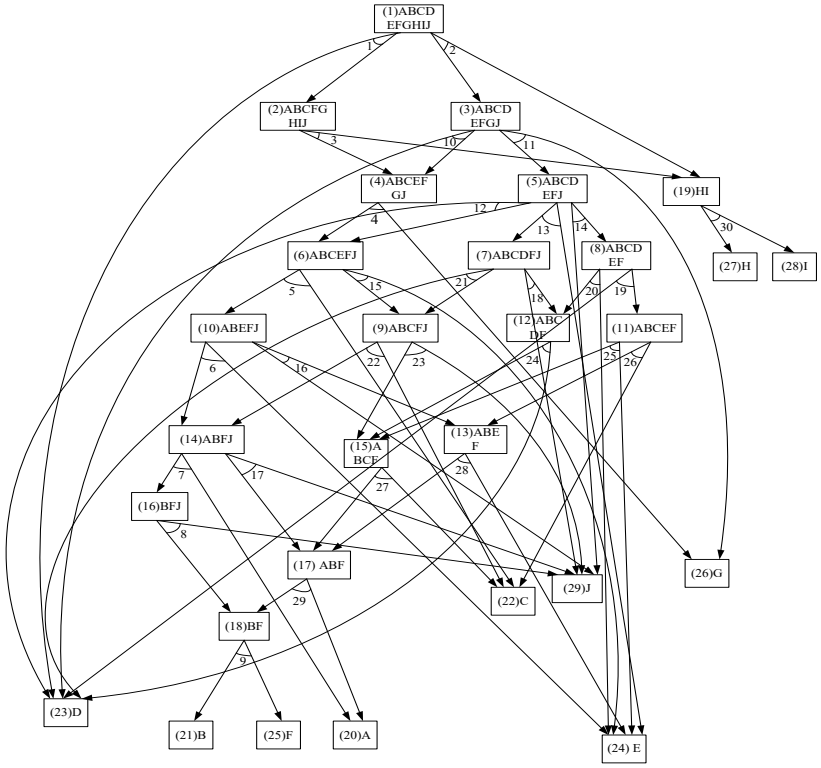


Fig. 6. The DAOG of a radio.

5.1. Two case studies

As depicted in Fig. 2, the ballpoint has 15 subassemblies and 13 tasks. The maximum energy consumption for disassembling the ballpoint is limited to Q = 3 J. The second case, i.e., the radio, is more complex as seen in Fig. 6 and consists of 29 subassemblies and 30 tasks. When disassembling the radio set, we do not allow Q to exceed 240 J. In addition, the required data of instances are evaluated and randomly generated, which comprises the disassembly time, cost, and power of each task and intermediate operation as well as the recycling revenue of each subassembly. The detailed data is available in Supporting Information of **supplementary file**. The related parameters of HMM are initially specified as:

1) The size of the population *PS* is equal to the number of employed bee and the number of onlookers, which is set to 100.

2) The maximum iteration number g_{max} of the proposed algorithm is set to 200.

3) The limit number l_{max} of cycles, through which no improvement occurs on the food source, is equal to *PS* in the scout phase.

Five replicated runs are executed for the two disassembled products, respectively. Table 3 presents the Pareto solutions and the computational time in each run. In terms of the ballpoint, the set of Pareto solutions are the same in each test, i.e., five identical nondominated solutions are obtained from each test. On the other hand, the sets of the Pareto solutions are not consistent with each other in the radio case; however, a high similarity can be found among these Pareto solution sets. Specifically, from the fifth and sixth columns of Table 3, it can be observed that six same Pareto solutions as marked in bold are identified every execution of HMM while the majority of Pareto solutions are obtained more than once over five runs. The fourth and last columns of Table 3 reveal that the CPU time increases with the problem size which is associated with the

numbers of subassemblies and tasks in a product. Fortunately, the computational time is acceptable even for the complex radio set that approximately uses 40 seconds. In summary, the proposed algorithm has an excellent stability to identify Pareto solutions while the computational time is satisfactory.

Table 3

The Pareto solutions of two case studies

Runs	The ballpoint			The radio		
	f_1	f_2	CPU time (s)	f_1	f_2	CPU time (s)
1	0.0440 0.0550 0.1260 0.1730 0.1830	0.6660 0.8820 1.7020 1.8710 2.7290	7.30	6.1230 **11.2010** **9.2360** **11.6810** **7.8090** **8.9780** 5.0850 **7.6070** 0.6870 2.8340	152.6430 **208.9143** **207.5087** **220.8766** **185.2156** **206.1321** 143.0477 **157.1342** 24.8647 120.9737	39.63
2	0.0440 0.0550 0.1260 0.1830 0.1730	0.6660 0.8820 1.7020 2.7290 1.8710	7.21	**11.6810** **11.2010** **8.9780** **9.2360** 6.4750 **7.8090** **7.6070** 5.0850 6.4760 9.0100	**220.8766** **208.9143** **206.1321** **207.5087** 145.9100 **185.2156** **157.1342** 143.0477 147.9502 206.8803	41.35
3	0.0550 0.1260 0.1730 0.0440 0.1830	0.8820 1.7020 1.8710 0.6660 2.7290	7.34	**7.6070** **11.2010** **9.2360** **7.8090** **11.6810** **8.9780** 5.1360 9.0100	**157.1342** **208.9143** **207.5087** **185.2156** **220.8766** **206.1321** 97.0408 206.8803	40.94
4	0.0440 0.1730 0.0550 0.1260 0.1830	0.6660 1.8710 0.8820 1.7020 2.7290	7.11	**11.2010** **7.6070** **9.2360** **7.8090** **11.6810** **8.9780** 5.0850 6.4750 2.8340	**208.9143** **157.1342** **207.5087** **185.2156** **220.8766** **206.1321** 143.0477 145.9100 120.9737	40.75
5	0.1730 0.0550 0.0440 0.1260 0.1830	1.8710 0.8820 0.6660 1.7020 2.7290	7.26	**9.2360** **11.2010** **8.9780** **11.6810** **7.8090** 5.0850 **7.6070** 6.1230 9.0100	**207.5087** **208.9143** **206.1321** **220.8766** **185.2156** 143.0477 **157.1342** 152.6430 206.8803	41.47

5.2. Tuning configurations of HMM

There are three parameters, i.e., g_{max}, PS, and l_{max}, that require to be initialized before running HMM. In particular, l_{max} is a special parameter defined in the ABC and we focus on the sensitivity analysis of l_{max}. Let l_{max} change linearly with PS. In this segment, only the complex product is applied because the variation of g_{max}, PS, and l_{max} has a slightly influence on a small-scale product. Each parameter combination is repeatedly run five times. Furthermore, we define two indicators to measure the quality of Pareto solutions:

(1) α: The average number of nondominated solutions found by HMM over the five runs;

(2) β: The number of dominated solutions over the five runs, i.e., the sum of abandoned solutions from the Pareto solution set after five runs;

Table 4 shows the computational results with different parameter combinations. From Table 4, it can be observed that the number of Pareto solutions increases as g_{max} and PS enlarge, while more than 50% same Pareto solutions marked in bold are shared in each instance, which demonstrates that the proposed method has a good robustness.

As seen in the second last column of Table 4, the number of nondominated solutions is the least when $l_{max} = 2PS$. There seems no significant difference between $l_{max} = PS/2$ and $l_{max} = PS$ both the quality and number of Pareto solutions. However, in terms of α and β, $l_{max} = PS/2$ significantly outperforms $l_{max} = PS$ when $g_{max} = 50$ and $PS = 20$. Specifically, HMM with $l_{max} = PS/2$ averagely finds more nondominated solutions and removes less dominated solutions from the Pareto solution set over five runs. Moreover, the computational times with different l_{max} are close when g_{max} and PS are the same. As a result, $l_{max} = PS/2$ is preferred and adopted in the following experiments.

Table 4

The computational results with different parameters for the radio case

No.	g_{max}	PS	l_{max}	f_1	f_2	α	β
1	50	20	$PS/2$	**8.9780** **9.2360** **11.6810** 5.0850 **7.8090** **11.2010** **7.6070** 9.0100 **6.1230**	**206.1321** **207.5087** **220.8766** 143.0477 **185.2156** **208.9143** **157.1342** 206.8803 **152.6430**	6.20	1
2	50	20	PS	**8.9780** **9.2360** **11.6810** 5.0850 **7.8090** **11.2010** **7.6070** **6.1230** 1.4940	**206.1321** **207.5087** **220.8766** 143.0477 **185.2156** **208.9143** **157.1342** **152.6430** 72.3360	5.40	3

3	50	20	2*PS*	**11.2010** **8.9780** **11.6810** **9.2360** **7.8090** **7.6070** **6.1230**	**208.9143** **206.1321** **220.8766** **207.5087** **185.2156** **157.1342** **152.6430**	4.80	1
4	100	50	*PS*/2	**8.9780** **9.2360** **11.6810** 5.1360 **7.8090** **11.2010** **7.6070** 9.0100 **6.1230** 1.4940 2.8340	**206.1321** **207.5087** **220.8766** 97.0408 **185.2156** **208.9143** **157.1342** 206.8803 **152.6430** 72.3360 120.9737	8.20	1
5	100	50	*PS*	**11.2010** **8.9780** **11.6810** **9.2360** **7.8090** **7.6070** **6.1230** 5.1360 0.6870 1.4940 2.8340	**208.9143** **206.1321** **220.8766** **207.5087** **185.2156** **157.1342** **152.6430** 97.0408 24.8647 72.3360 120.9737	8.40	1
6	100	50	2*PS*	**11.2010** **8.9780** **11.6810** **9.2360** **7.8090** **7.6070** **6.1230** 5.0850 1.4940	**208.9143** **206.1321** **220.8766** **207.5087** **185.2156** **157.1342** **152.6430** 143.0477 72.3360	7.60	0

5.3. Comparisons

In this comparison section, the modified MODABC and NSGA-II for solving the PEDSP will adopt the same algorithm parameters including g_{max} and *PS* with HMM.

To evaluate the quality of Pareto solutions, two quantitative performance indicators, i.e., inverted generational distance (*IGD*) and spacing (*SP*), are calculated for the Pareto fronts of three multi-objective algorithms. *IGD* and *SP* are two common measurements used to estimate the closeness to the true Pareto front and uniform distribution of the known Pareto solutions [40, 41]. In this work, we generate the true Pareto front of our problem by enumeration using parallel processing techniques, and the smaller the two metrics, the better the multi-objective algorithm performance [7]. To guarantee the quality of the Pareto front within an acceptable computational time, g_{max} and *PS* are set to 500 and 100, respectively. Also, let l_{max} equal *PS*/2 according to the sensitivity analysis shown in Section 5.2. The nondominated solutions are selected from five runs, which form the known Pareto front for each approach. The comparison results are presented in Table 5.

As shown in Table 5, the rank of *IGD* from the best to the worst is HMM, MODABC, and NSGA-II, respectively. Exactly, the Pareto front of the proposed algorithm is much closer to the true Pareto front than that of MODABC and NSGA-II. In terms of *SP*, NSGA-II performs best, the second one is HMM, and the worst one is MODABC. Although the distribution of nondominated solutions of NSGA-II appears the most uniform, it finds the least Pareto solutions as seen in the fourth row of Table 5. Further, we find that MODABC and HMM share nine same Pareto solutions, but an exception is that a Pareto solution (f_1 = 5.0850 and f_2 = 143.0477) of MODABC is dominated by that (f_1 = 5.1360 and f_2 = 97.0408) of HMM. In addition, the proposed approach uses the shortest CPU time, NSGA-II is inferior, and MODABC has the worst computational efficiency. These comparisons indicate that the proposed method performs excellently both the Pareto front and the computational performance in solving the PEDSP.

Table 5

Comparison results among MODABC, NSGA-II, and HMM in the radio case

Indicator	MODABC	NSGA-II	HMM
IGD	0.32	0.66	0.15
SP	17.27	11.85	14.89
The number of Pareto solutions	10	7	11
Average CPU time (s)	221.87	193.33	125.42

6. Conclusion

This work studies a profit-oriented and energy-efficient disassembly sequencing problem (PEDSP), in which recovered profit and energy consumption are both optimized for a disassembly process. The AND/OR graph is adapted to represent a disassembly diagram, which clearly illustrates the relationships among subassemblies and the connections between subassemblies and disassembly tasks. Two relations, i.e., the AND relation and the exclusive OR relation, are defined in the disassembly AND/OR graph (DAOG). Based on the DAOG, we modeled the PEDSP and proposed a hybrid multi-objective metaheuristic (HMM) approach for solving such a problem. HMM integrates an artificial bee colony algorithm, a nondominated sorting procedure, and a variable neighborhood descent process to identify the Pareto solutions of the PEDSP. By testing the HMM with two different case studies, we demonstrate the feasibility and effectiveness of our approach. The further comparisons with two other multi-objective metaheuristic algorithms indicate that the proposed method performs excellently in solving our problem.

Due to limited availability of primary data, the case studies adopted some data estimated by the authors (e.g., the power of each disassembly task). However, it should not affect the application of the proposed algorithm to solve a real-world PEDSP as long as all the required data are given. In reality, it is often difficult to collect accurate and reliable information for parameters such as disassembly time and power consumption due to the inherent uncertainties of disassembly processes. Therefore, new optimization techniques should be developed to tackle this challenge. As such, we suggest several future work directions herein. First, a disassembly operation may be performed by human and/or robots, so both manpower and electricity consumption should be taken into consideration and evaluated properly. Second, the subassembly quality varies across EOL products, which greatly affects the recovered value as well as the completion time of a disassembly task. A predictive model on the subassembly quality could improve the solutions of PEDSP for industrial practice, which requires further research.

Acknowledgement

This research is supported by the Funds for National Natural Science Foundation of China (nos. 51575211), the International Cooperation and Exchange of the National Natural Science Foundation of China, (no.51861165202), the U.S. National Science Foundation (Grant No. 1512217), and the China Scholarship Council (Grant No. 201706160025). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Natural Science Foundation of China and the U.S. National Science Foundation. We are grateful to the editor and anonymous referees for their constructive comments to improve this paper.

References

[1] V.K. Gonnuru, Disassembly planning and sequencing for end-of-life products with RFID enriched information, Robotics and Computer-Integrated Manufacturing, 29 (2013) 112-118.
[2] L. Li, C. Li, Y. Tang, Y. Du, An integrated approach of reverse engineering aided remanufacturing process for worn components, Robotics and Computer-Integrated Manufacturing, 48 (2017) 39-50.
[3] J. Li, M. Barwood, S. Rahimifard, Robotic disassembly for increased recovery of strategically important materials from electrical vehicles, Robotics and Computer-Integrated Manufacturing, (2017).
[4] P. Jiang, J. Leng, K. Ding, P. Gu, Y. Koren, Social manufacturing as a sustainable paradigm for mass individualization, Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 230 (2016) 1961-1968.
[5] C. Li, Y. Tang, C. Li, L. Li, A modeling approach to analyze variability of remanufacturing process routing, IEEE Transactions on Automation Science and Engineering, 10 (2013) 86-98.
[6] A. Gungor, S.M. Gupta, An evaluation methodology for disassembly processes, Computers & Industrial Engineering, 33 (1997) 329-332.
[7] Y. Ren, G. Tian, F. Zhao, D. Yu, C. Zhang, Selective cooperative disassembly planning based on multi-objective discrete artificial bee colony algorithm, Engineering Applications of Artificial Intelligence, 64 (2017) 415-431.
[8] Y. Ren, D. Yu, C. Zhang, G. Tian, L. Meng, X. Zhou, An improved gravitational search algorithm for profit-oriented partial disassembly line balancing problem, International Journal of Production Research, 55 (2017) 7302-7316.
[9] Y. Ren, C. Zhang, F. Zhao, M.J. Triebe, L. Meng, An MCDM-Based Multiobjective General Variable Neighborhood Search Approach for Disassembly Line Balancing Problem, IEEE Transactions on Systems, Man, and Cybernetics: Systems, (2018) 1-14.
[10] Y. Ren, C. Zhang, F. Zhao, G. Tian, W. Lin, L. Meng, H. Li, Disassembly line balancing problem using interdependent weights-based multi-criteria decision making and 2-Optimal algorithm, Journal of Cleaner Production, 174 (2018) 1475-1486.
[11] Y. Ren, C. Zhang, F. Zhao, H. Xiao, G. Tian, An asynchronous parallel disassembly planning based on genetic algorithm, European Journal of Operational Research, 269 (2018) 647-660.
[12] A. Koc, I. Sabuncuoglu, E. Erel, Two exact formulations for disassembly line balancing problems with task precedence diagram construction using an AND/OR graph, Iie Transactions, 41 (2009) 866-881.
[13] A.J. Lambert, Optimizing disassembly processes subjected to sequence-dependent cost, Computers & Operations Research, 34 (2007) 536-551.
[14] K.E. Moore, A. Güngör, S.M. Gupta, Petri net approach to disassembly process planning for products with complex AND/OR precedence relationships, European Journal of Operational Research, 135 (2001) 428-449.
[15] L. Meng, C. Zhang, X. Shao, Y. Ren, C. Ren, Mathematical modelling and optimisation of energy-conscious hybrid flow shop scheduling problem with unrelated parallel machines, International Journal of Production Research, (2018) 1-27.
[16] L. Meng, C. Zhang, X. Shao, Y. Ren, MILP models for energy-aware flexible job shop scheduling problem, Journal of Cleaner Production, (2018).
[17] S.M. McGovern, S.M. Gupta, A balancing method and genetic algorithm for disassembly line balancing, European journal of operational research, 179 (2007) 692-708.
[18] C. Zhang, P. Li, Z. Guan, Y. Rao, A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem, Computers & Operations Research, 34 (2007) 3229-3242.
[19] D. Karaboga, B. Akay, A comparative study of artificial bee colony algorithm, Applied mathematics and computation, 214 (2009) 108-132.
[20] Q. Lu, G. Zhou, F. Zhao, L. Li, Y. Ren, Determination of the shape and distribution parameters for abrasive grains to reduce carbon emissions, Journal of Manufacturing Science and Engineering.

[21] Q. Yi, C. Li, Y. Tang, X. Chen, Multi-objective parameter optimization of CNC machining for low carbon manufacturing, Journal of Cleaner Production, 95 (2015) 256-264.
[22] J. Leng, P. Jiang, Mining and matching relationships from interaction contexts in a social manufacturing paradigm, IEEE Transactions on Systems, Man, and Cybernetics: Systems, 47 (2017) 276-288.
[23] Q. Lu, G.H. Zhou, C. Zhou, Z.D. Xiao, A carbon emissions allocation method based on temperature field for products in the usage stage, International Journal of Advanced Manufacturing Technology, 91 (2017) 917-929.
[24] Q. Lu, G.H. Zhou, Z.D. Xiao, F.T. Chang, C.L. Tian, A selection methodology of key parts based on the characteristic of carbon emissions for low-carbon design, International Journal of Advanced Manufacturing Technology, (2017) 1-15.
[25] M. Johnson, M.H. Wang, Economical evaluation of disassembly operations for recycling, remanufacturing and reuse, International Journal of Production Research, 36 (1998) 3227-3252.
[26] A. Lambert, Linear programming in disassembly/clustering sequence generation, Computers & Industrial Engineering, 36 (1999) 723-738.
[27] J.G. Kang, D.H. Lee, P. Xirouchakis, J.G. Persson, Parallel Disassembly Sequencing with Sequence-Dependent Operation Times, CIRP Annals - Manufacturing Technology, 50 (2001) 343-346.
[28] B. Adenso-Díaz, S. García-Carbajal, S. Lozano, An efficient GRASP algorithm for disassembly sequence planning, OR spectrum, 29 (2007) 535-549.
[29] S. Smith, L.-Y. Hsu, G.C. Smith, Partial disassembly sequence planning based on cost-benefit analysis, Journal of cleaner production, 139 (2016) 729-739.
[30] B. Sanchez, C. Haas, A novel selective disassembly sequence planning method for adaptive reuse of buildings, Journal of Cleaner Production, 183 (2018) 998-1010.
[31] M. Kheder, M. Trigui, N. Aifaoui, Disassembly sequence planning based on a genetic algorithm, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 229 (2015) 2281-2290.
[32] G. Percoco, M. Diella, Preliminary evaluation of artificial bee colony algorithm when applied to multi objective partial disassembly planning, Research Journal of Applied Sciences, Engineering and Technology, 6 (2013) 3234-3243.
[33] L. Zhong, S. Youchao, O. Ekene Gabriel, W. Haiqiao, Disassembly sequence planning for maintenance based on metaheuristic method, Aircraft Engineering and Aerospace Technology, 83 (2011) 138-145.
[34] X. Guo, S. Liu, M. Zhou, G. Tian, Disassembly sequence optimization for large-scale products with multiresource constraints using scatter search and Petri nets, IEEE transactions on cybernetics, 46 (2016) 2435-2446.
[35] G. Tian, Y. Ren, M. Zhou, Dual-objective scheduling of rescue vehicles to distinguish forest fires via differential evolution and particle swarm optimization combined algorithm, IEEE Transactions on Intelligent Transportation Systems, 17 (2016) 3009-3021.
[36] Y. Ren, G. Tian, Emergency scheduling for forest fires subject to limited rescue team resources and priority disaster areas, IEEJ Transactions on Electrical and Electronic Engineering, 11 (2016) 753-759.
[37] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, Journal of global optimization, 39 (2007) 459-471.
[38] D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, A comprehensive survey: artificial bee colony (ABC) algorithm and applications, Artificial Intelligence Review, 42 (2014) 21-57.
[39] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE transactions on evolutionary computation, 6 (2002) 182-197.
[40] C.A.C. Coello, N.C. Cortés, Solving multiobjective optimization problems using an artificial immune system, Genetic Programming and Evolvable Machines, 6 (2005) 163-190.
[41] D.A. Van Veldhuizen, G.B. Lamont, On measuring multiobjective evolutionary algorithm performance, in: Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, IEEE, 2000, pp. 204-211.