Disassembly Sequence Planning for Intelligent Manufacturing Using Social Engineering Optimizer

Cheng Zhang 1,2, Amir Mohammad Fathollahi-Fard 3, Jianyong Li 1,*️⃣, Guangdong Tian 1,2,4, *️⃣ and Tongzhu Zhang 4,5

1 School of Mechanical Engineering, Shandong University, Jinan 250061, China; 201934018@mail.sdu.edu.cn
2 State Key Laboratory of Robotics and Systems (HIT), Harbin 150000, China
3 Department of Electrical Engineering, École de Technologie Supérieure, University of Québec, Montréal, QC H3C 3P8, Canada; amirfard@aut.ac.ir
4 Transportation College of Jilin University, Jilin 132000, China; Zhangtongzhu@mail.jlu.edu.cn
5 China Automotive Technology and Research Center Co., Ltd., Tianjin 300300, China
* Correspondence: ljy@sdu.edu.cn (J.L.); tiangd2013@163.com or tiangd2013@sdu.edu.cn (G.T.);
Tel.: +86-138-6914-3670 (J.L.); +86-151-0431-7455 (G.T.)

Abstract: Product disassembly and recycling are important issues in green design. Disassembly sequence planning (DSP) is an important problem in the product disassembly process. The core idea is to generate the best or approximately optimal disassembly sequence to reduce disassembly costs and time. According to the characteristics of the DSP problem, a new algorithm to solve the DSP problem is proposed. Firstly, a disassembly hybrid graph is introduced, and a disassembly constraint matrix is established. Secondly, the disassembling time, replacement frequency of disassembly tool and replacement frequency of disassembly direction are taken as evaluation criteria to establish the product fitness function. Then, an improved social engineering optimizer (SEO) method is proposed. In order to enable the algorithm to solve the problem of disassembly sequence planning, a swap operator and swap sequence are introduced, and steps of the social engineering optimizer are redefined. Finally, taking a worm reducer as an example, the proposed algorithm is used to generate the disassembly sequence, and the influence of the parameters on the optimization results is analyzed. Compared with several heuristic intelligent optimization methods, the effectiveness of the proposed method is verified.

Keywords: disassembly sequence planning; social engineering optimizer; swap operator; swap sequence; intelligent manufacturing

1. Introduction

With the continuous development of the manufacturing industry, the problems of resource reuse and the potential environmental pollution caused by a large number of end-of-life (EOL) products urgently need to be solved. How to deal with EOL products efficiently and economically has become a research interest. The disassembly of EOL products is an important part of recycling or maintenance [1] and can reduce environmental pollution and promote resource recycling. Disassembly Sequence Planning (DSP) aims to generate the disassembly sequence of parts or sub-assemblies [2] to meet various disassembly requirements, such as disassembly costs, benefits, and disassembly methods.

Early research mainly tends to graph theory research. Henrioud et al. [3] describe the direct-connection relationship between product structures through an association graph model. Sanderson and Homem de Mello et al. [4] propose the AND/OR graph model, which represents the disassembly unit of the product as a node. If the connecting line between the nodes is curved, then the two disassembly units have a logical AND relationship; otherwise, the relationship between the two is OR. Li et al. [5] proposed a hybrid diagram that can be used to represent the dynamic changes in geometric constraints and
priority constraints during product disassembly, and simplify the process of determining the disassembly sequence. Huang et al. [6] propose a disassembly matrix model on this basis, which can accurately describe the priority order of disassembly units.

On this basis, Mitrouchev et al. [7] proposed a disassembly geometric contact diagram based on the lowest level of the disassembly product diagram, which eliminates parts that are not related to the target before the sequence is generated, thereby generating a feasible alternative disassembly sequence. Hu et al. [8] put forward a new disassembly diagram and its related indicators to estimate the time of complete and selective disassembly in the design phase of the disassembly scheme. Edmunds et al. [9] find the best order of component removal by converting an AND/OR graph to a priority graph according to the constraint between parts, thus effectively reducing the complexity of complex mechanism disassembly. In order to increase the revenue when retrieving only some components from a given product, Smith and Chen [10] proposed a rule-based recursive method to find a near-optimal heuristic selective disassembly sequence. Issaoui et al. [11] propose a new disassembly direction matrix as a representative model of the disassembly direction, and updated the movement matrix of the parts to provide information about the movement status of the parts during the disassembly sequencing process. Tian et al. [12] introduce a conflict matrix in the AOG graph to deal with the exclusive OR problem that the existing heuristic disassembly algorithm cannot solve. Wang et al. [13] propose a disassembly design technology, combining regret theory and entropy weight method, and provide a systematic support tool for the evaluation of schemes of disassembly. Feng et al. [14] propose a hybrid disassembly modeling and optimization method for obsolete products based on the reusability of sub-components. Yuan et al. [15] established a multi-objective comprehensive disassembly evaluation model based on Drosophila algorithm, cross-efficiency and extended gray correlation, and evaluated the approximate optimal disassembly scheme from the aspects of time, economy and environment, then obtained the best scheme.

However, as the complexity of the product structure increases, the number of parts involved in disassembly has also increased exponentially [16], and the traditional DSP method has been unable to meet the disassembly requirements of complex products. The heuristic intelligent optimization method has the advantages of a high solving speed and convenient parameter adjustment [17] and has been widely used in optimization problems of various manufacturing sequences, such as automated production lines [18], hot rolling [19], and non-sharp distillation [20]. It also applies to the disassembly sequence. Kongar and Gupta [21] propose a genetic algorithm (GA) for solving disassembly sequence planning. Zhang et al. [22] map the disassembly mixed graph model to the particle swarm model and realized the optimal disassembly sequence planning of complex products through particle swarm optimization. Xing et al. [23] search for feasible solutions through ant colony optimization (ACO) and calculate the dominance relationship between each solution, and obtained the Pareto solution set, thereby realizing the disassembly sequence planning.

With the continuous deepening of research, heuristic intelligent optimization has become an important method to obtain an optimal or nearly optimal disassembly sequence. Many new algorithms have been proposed, and old algorithms have been improved for DSP problems. Wu et al. [24] use the binary tree algorithm to optimize the generation of the initial population and the crossover and mutation process of the population, and realize the solution of disassembly sequence planning through an improved genetic algorithm. Kheder et al. [25] apply GA to DSP by taking the maintainability of the machine, number of components, number of tool changes and directions during disassembly into account. Tseng et al. [26] compare the GA algorithm of Kongar and Gupta with Dijkstra’s algorithms, and propose a new block-based DSP genetic algorithm. Zhang et al. [27] use an artificial bee colony (ABC) algorithm to solve the complicated product disassembly sequence problem under parallel disassembly. Ren et al. [28] improve the multi-objective discrete ABC algorithm for the selective collaborative DSP problem of complex products. Yeh [29] improve the update mechanism of the simplified group optimization method (SSO),
modify the adaptive parameter control program of SSO, and propose a learning-effect DSP. Xia et al. [30] simplify the teaching-learning-based optimization (TLBO) algorithm, develop three new operators and combine them into the STLBO algorithm and apply them to the DSP problem with complex disassembly priority constraints. To solve the degree of freedom in modular product design, Tao et al. [31] design an automatic self-decomposed disassembly precedence matrix and propose a tabu-search-based hyper heuristic algorithm with exponentially decreasing diversity management strategy. As the synchronous parallel disassembly theory cannot accurately reflect the actual situation in the work, Ren et al. [32] propose an asynchronous parallel disassembly plan based on GA. Considering the constraints of disassembly operators and tools, Guo et al. [33] put forward a dual-objective optimization model with a scatter search method, which means that disassembly profit is maximized, and time is minimized. Based on the uncertainty of component quality and operation cost during the disassembly process, Tian et al. [34] propose a hybrid intelligent algorithm combining fuzzy uncertainty theory and an ABC algorithm. According to the requirements of low carbon emission and resource reuse, Yang et al. [35] propose a multi-objective disassembly line balancing fruit-fly optimization algorithm for the DSP of obsolete agricultural machinery.

Stefanini et al. [36] propose a method based on process mining, which combines Time-Driven Activity-Based Costing and process mining approaches to predict the resource consumption of patients in hospitals. Parsa et al. [37] define new optimization parameters based on disassembly and component requirements for the same EOL product in different states. Bentaha et al. [38] regard the quality of EOL products and their parts as random variables, and determine the best process and the depth of disassembly based on the quality of products to be disassembled. Babbitt et al. [39] collect the quality information of the main materials and components contained in 95 kinds of consumer electronic products, and establish a complete bill of materials (BOM) data database to provide assistance in the disassembly and recycling of EOL consumer electronic products. Tian et al. [40] propose a new hybrid multi-criteria decision-making method, which combines fuzzy AHP and fuzzy G-TOPSIS to evaluate the production and remanufacturing methods of automotive component remanufacturing. Tian et al. [41] propose a novel fuzzy Choquet integral-based grey comprehensive evaluation method, use an improved teaching-learning-based optimization algorithm to lambda-fuzzy-measures following the weights given by experts in order to enhance the consistency of weights. Wang et al. [42] improve the artificial bee colony algorithm and propose an optimization method for milling parameters of CNC machine tools, considering energy consumption.

The above research results introduce the application of an intelligent algorithm in disassembly sequence programming solutions. However, according to the free lunch theory [43], none of the meta-heuristics mentioned above can solve all optimization problems. It is always possible that new algorithms based on current or new optimization problems are better than current meta-heuristic algorithms. Therefore, it is still necessary to try to find new optimization algorithms to solve the DSP problem. The social engineering optimizer (SEO) proposed by A. M. Fathollahi-Fard [44], inspired by the concept of social engineering theory, has four main steps and three undetermined parameters, which is simpler and more intelligent than traditional methods. The approach starts with generating two random solutions called attackers and defenders. According to the rules of social engineering technology, the attacker gets the information of the defender to achieve the desired goal (to get the optimal value), while the defender evades it. When the attacker achieves his goal, it searches for the next victim (a new defender) from his current position. At present, there is no research on applying an SEO algorithm to the DSP problem.

Combining the above documents, this article proposes an improved SEO algorithm. Unlike the original SEO and other algorithms, the swap sequence is used as the individual to be optimized instead of the disassembly sequence. Aiming at the shortcomings that the original SEO cannot solve in the discrete optimization problem, the original algorithm flow is improved, and three sequence operation operators are defined to replace numerical
calculations. Finally, through a case of a turbine reducer, the parameter sensitivity and effectiveness of the proposed algorithm are studied.

The rest of this paper is organized as follows. Section 2 describes the disassembly theories. Section 3 introduces the basic concept of Social Engineering Optimizer. The improved SEO method for solving DSP problems is provided in Section 4. In Section 5, an illustrative example to verify the feasibility of the proposed method is utilized, followed by a comparative analysis with other methods. The last section gives some summary remarks and looks forward to furthering the research.

2. Disassembly Information Model

2.1. Disassembly Hybrid Graph

The disassembly hybrid graph [5] is a product topological structure model, which describes the hierarchical information and constraint relations between the product components, and is easy to express by computer programs. The basic product disassembly hybrid diagram is represented by a triple

\[
G = \{V, Z, O\}.
\]

where \(V = [v_1, v_2, \ldots, v_n]\) represents the basic disassembly unit of the product, such as product parts and sub-assemblies, and \(n\) is the number of basic disassembly units; \(Z = [z_1, z_2, \ldots, z_l]\) is the undirected edge set of the product disassembly mixed graph, indicating that there is a direct contact relationship between the two parts; \(O = [o_1, o_2, \ldots, o_m]\) is the product disassembly mixed graph. The directed edge set indicates that there is a priority relationship between the two parts.

The disassembly hybrid diagram can be expressed as the structure shown in Figure 1. The circle represents the parts and sub-assemblies of the product; the solid line represents the direct contact between the disassembled components; the solid arrow represents the mandatory disassembly priority relationship between the two parts.

According to the product disassembly of the vertex set, undirected edge set and directed edge set of the mixed graph model, the contact constraint matrix \(C\) and priority constraint matrix \(P\) can be constructed:

According to the matrices, a generation method of disassembly sequence can be obtained. First, generate a priority constraint matrix and contact constraint matrix according to the disassembly hybrid graph. Next, look for rows in the matrix that meet the detachability, and randomly select one of them as the first part to be disassembled. Then, remove the row from the matrices. Finally, repeat the above steps until the priority constraint matrix and contact constraint matrices are empty to generate a feasible disassembly sequence

\[
C = \begin{bmatrix}
c_{11} & c_{12} & \cdots & c_{1n} \\
c_{21} & c_{22} & \cdots & c_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
c_{n1} & c_{n2} & \cdots & c_{nn}
\end{bmatrix},
\]
\[ P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nn} \end{bmatrix}, \]

where

\[ c_{ij} = \begin{cases} 1, & \text{part } i \text{ and } j \text{ are in direct contact;} \\ 0, & \text{part } i \text{ and } j \text{ are not in direct contact or } i = j; \end{cases} \]

\[ p_{ij} = \begin{cases} 1, & \text{part } j \text{ must be disassembled before } i; \\ 0, & \text{else}; \end{cases} \]

\( n \) is the number of basic disassembly units. If vertex \( i \) is dismountable, it must satisfy [45]:

\[ \sum_{j=1}^{n} c_{ij} = 1 \text{ and } \sum_{j=1}^{n} p_{ij} = 0. \tag{2} \]

According to the matrices, a generation method of disassembly sequence can be obtained. First, generate a priority constraint matrix and contact constraint matrix according to the disassembly hybrid graph. Next, look for rows in the matrix that meet the detachability, and randomly select one of them as the first part to be disassembled. Then, remove the row from the matrices. Finally, repeat the above steps until the priority constraint matrix and contact constraint matrices are empty to generate a feasible disassembly sequence.

### 2.2. Objective Function of DSP

Disassembly cost is an important measure of selective disassembly sequence planning. In previous research, the change in disassembly tool, change in disassembly direction and the disassembly time are usually chosen to measure the cost of disassembly. According to the literature [46], setting the standard time for disassembly tools and direction change can unify the dimensions of the objective function, as shown in Equation (3)

\[ F = \sum \omega_e e_{ij} t_e + \sum \omega_d d_{ij} t_d + \sum \omega_t t_j, \tag{3} \]

where

\[ e_{ij} = \begin{cases} 1, & \text{if tool is not changed;} \\ 0, & \text{if tool is changed}; \end{cases} \]

\[ d_{ij} = \begin{cases} 0, & \text{if direction is not changed} \\ 1, & \text{if direction is changed by } 90^\circ \\ 2, & \text{if direction is changed by } 180^\circ \end{cases} \]

\( F \) represents the comprehensive time cost of product disassembly; \( t_j \) represents the time required to disassemble part \( j \), and the weight is \( \omega_t \); \( t_e \) represents the time to change the disassembly tool during the disassembly process, and the weight is \( \omega_e \); \( t_d \) indicates the time taken to change the direction of disassembly during the process, and the weight is \( \omega_d \).

### 3. The Improved Social Engineering Optimizer

#### 3.1. Social Engineering

The advent of social engineering optimizer has been inspired by social engineering theory. Social engineering (SE) is defined as the act of obtaining people’s personal information through certain technologies and inducing or forcing them to meet the requirements of social engineers [47]. Social engineers often use disguise and deception to achieve their goals. In social engineering, the party collecting information is called the attacker, and the target of the information stolen is called the defender. A social engineering attack is mainly divided into four steps: First, the attacker obtains part of the defender’s information from various channels and specifies an attack plan against the defender. Then, the attacker
Symmetry 2021, 13, 663

harasses the defender to force him to reveal his weaknesses. Next, the defender analyzes the attacker’s thinking and takes measures to get rid of the attack. Finally, the above steps are repeated until the attack is successful or the defense is successful.

According to SE, the steps of social engineering optimizer are established according to the above process.

3.2. Swap Operator and Swap Sequence

DSP is a discrete combinatorial optimization problem. A social engineering optimizer is widely used in continuous optimization problems, but is rarely used in discrete combinatorial optimization problems. In order to solve this problem, the concepts of swap operator and swap sequence [48] are introduced in SEO.

Suppose a disassembly sequence of a mechanism consisting of \( n \) parts is \( S = (a_i) \), \( i = 1, 2, \ldots, n \). Let \( a_j \) and \( a_k \) be two points in \( S \), \( SO(a_j, a_k) \) is defined as a swap operator, means exchanging points \( a_j \) and \( a_k \). “\( \oplus \)“ is defined as the calculation of two swap operators, \( S' = S \oplus SO(a_j, a_k) \) means using swap operator \( SO(a_j, a_k) \) to \( S \) and obtain a new solution sequence \( S' \).

Swap sequence means an ordered queue of multiple swap operators. A swap sequence formed by \( l \) swap operators is expressed as \( SS = (SO_1, SO_2, \ldots, SO_l) \). When the swap sequence acts on a disassembly sequence, all swap operators included in the swap sequence must act in turn, which is

\[
S' = S \oplus SS = [(S \oplus SO_1) \oplus SO_2] \oplus \ldots \oplus SO_l. \tag{4}
\]

Several swap sequences and swap operators can be combined to form a new swap sequence, so the above equation can also be expressed as

\[
S' = S \oplus SS = S \oplus (SO_1 \oplus SO_2 \oplus \ldots \oplus SO_l). \tag{5}
\]

The set of all swap sequences that produce same solutions is called equivalent set. In equivalent set, the swap sequence that contains the fewest swap operators is called a basic swap sequence. Taking the structure shown in Figure 1 as an example, the generation method of a basic swap sequence is as follows:

Let \( S_1 = (1, 4, 2, 3, 5, 6) \), \( S_2 = (3, 1, 4, 2, 5, 6) \) be two disassembly sequences of the structure shown in Figure 1, search from the first bit of sequence \( S_1 \), it can be found that \( S_1(1) = S_2(2) = 1 \), so the first swap operator is \( SO_1 = SO(1, 2) \), \( S_21 = (1, 3, 4, 2, 5, 6) \). By analogy, \( S_1(2) = S_22(3) = 4 \), \( SO_2 = SO(2, 3) \), \( S_22 = (1, 4, 3, 2, 5, 6) \); \( S_1(3) = S_22(4) = 2 \), \( SO_3 = SO(3, 4) \), \( S_23 = (1, 4, 2, 3, 5, 6) \).

Since \( S_1(4) = S_23(5) \), \( S_1(5) = S_23(5) \), \( S_1(6) = S_23(6) \), \( SO_4 = SO(4, 5) \), \( SO_5 = SO(5, 5) \), \( SO_6 = SO(6, 6) \), the sequence does not change, which is equivalent to a do-nothing operation. Therefore, the basic swap sequence from \( S_2 \) to \( S_1 \) is

\[
SS = [SO(1, 2), SO(2, 3), SO(3, 4)]
\]

3.3. Swap Sequence Based SEO

A standard SEO consists of the following steps: initialize the attacker and the defender; train and retrain, spot an attack; respond to attack and create a new defender. An SEO that introduces swap operators and swapping sequences needs to redefine several steps. The steps for improvement are as follows.

(1) Initialize disassembly sequence

Unlike the continuous optimization problem, the generation of the disassembly sequence needs to satisfy the disassembly of the parts. It is necessary to generate three feasible initial disassembly sequences according to the disassembly sequence generation method proposed in Section 2.1;

(2) Initialize the attacker and the defender
The solution to the SEO optimization problem is a vector, called “person”; each bit of the vector represents a variable of the objective function, called the “trait” of “person”. The algorithm generates two random initial solutions. Unlike the disassembly sequence, the swap sequence can directly perform operations, such as exchange and displacement, so the attacker and defender in the proposed method are composed of swap sequences. The value of the objective function in the middle is selected as the standard solution. Use the basic swap sequence generation method given in Section 3.2 to generate two. In order to make the length of the swap sequence consistent, all swap operators are reserved.

The swap sequence that transforms the standard solution into the optimal solution acts as the attacker, while the other acts as the defender, as shown in Figure 2;

\[ S_1 = \{1, 4, 2, 3, 5, 6\} \]

**Figure 2.** Generate the attacker and the defender.

(3) Train and retrain

In this step, a random integer \(\alpha\) between 1 and \(n\) is set in advance. Then, randomly select \(\alpha\) bits in defender vector and replace them with corresponding positions in the attacker vector, respectively, to generate a feasible solutions. If the optimal solution among feasible solutions is better than the defender, the defender is replaced with this solution.

For example, use the swap sequence shown in Figure 2. let \(\alpha = 3\). The example is displayed to clarify this step in Figure 3;

**Figure 3.** Training and retraining process.

\[ S_2 = \{3, 1, 4, 2, 5, 6\} \]

**Figure 3.** Training and retraining process.

\[ S_3 = \{4, 1, 3, 5, 2, 6\} \]

(4) Spot an attack

This step randomly selects a method for the attacker to approach the defender in the search space. There are four methods to spot attacks in SEO: obtaining, phishing, diversion theft and pretext. However, these cannot be directly used to solve DSP problems. Therefore, some special operators are proposed to meet the requirements.
Addition operator $\oplus$: 
The addition operator has been mentioned in Section 3.2 and is expressed as 
\[ SS = SS_1 \oplus SS_2, \]
where $SS$, $SS_1$ and $SS_2$ are swap sequences. Assuming that the length of $SS_1$ and $SS_2$ is $n$, 
the length of $SS$ is $2n$.

Figure 4 shows the process of using the addition operator on swap sequences consisting of three swap operators.

\[
\begin{array}{c|c|c|c}
SS_1 & SO(1, 2) & SO(2, 3) & SO(3, 4) \\
SS_2 & SO(1, 3) & SO(2, 2) & SO(3, 3) \\
SS & SO(1, 2) & SO(2, 3) & SO(3, 4) & SO(1, 3) & SO(2, 2) & SO(3, 3) \\
\end{array}
\]

Figure 4. Addition operator.

Derange operator $\otimes$: 
Suppose $SS$ is a swap sequence of length $n$, $\beta \in [0, 1]$ is the input variable of search engine. Let $\mu = \text{ceil}(n \times U(0, 1))$, $\nu = \text{ceil}(n \times \sin(\beta))$, the derange operator is to rearrange the 
$\mu$th position to the $(\mu + \nu)$th position of the sequence, if $\mu + \nu \geq n$, then rearrange the $\mu$th 
presentation to the last position. In particular, for the case of $(1 - U(0, 1) \times \sin(\beta))$, rearrange the positions that are not between $\mu$th and $(\mu + \nu)$th position. The derange operator is expressed as 
\[
SS' = SS \otimes (U(0, 1) \times \sin(\beta)), \\
SS'' = SS \otimes (1 - U(0, 1) \times \sin(\beta)).
\]

Let $\mu = 3, \nu = 2$, Figure 5 shows the process of using the derange operator on swap sequences consisting of six swap operators.

\[
\begin{array}{c|c|c|c|c|c|c}
SS & SO(1, 2) & SO(2, 3) & SO(3, 4) & SO(1, 3) & SO(2, 2) & SO(3, 3) \\
SS' & SO(1, 2) & SO(2, 3) & SO(2, 2) & SO(3, 4) & SO(1, 3) & SO(3, 3) \\
SS'' & SO(3, 3) & SO(2, 3) & SO(2, 2) & SO(3, 4) & SO(1, 3) & SO(1, 2) \\
\end{array}
\]

Figure 5. Derange operator.

Average operator $\odot$: 
Suppose $SS_1$ and $SS_2$ are two swap sequences. The averaging operator starts from the 
first position of the sequence and randomly selects the content in $SS_1$ or $SS_2$ as the new 
sequence with a 50% probability. The average operator is expressed as 
\[ SS = SS_1 \odot SS_2. \]

Figure 6 shows the process of using the average operator on swap sequences consisting of six swap operators.
Therefore, four methods to spot an attack is expressed as:

Obtaining
\[ \text{def}_{\text{new}} = (\text{def}_{\text{old}} \odot (1 - \sin \beta \times U(0,1))) \oplus ((\text{def}_{\text{old}} \odot (\sin \beta \times U(0,1))); \]  \hfill (6)

Phishing
\[ \text{def}_{\text{new}}^{1} = (\text{att} \odot (1 - \sin \beta \times U(0,1))) \oplus ((\text{def}_{\text{old}} \odot (\sin \beta \times U(0,1))); \]  \hfill (7)
\[ \text{def}_{\text{new}}^{2} = (\text{def}_{\text{old}} \odot (1 - \sin(\frac{\pi}{2} - \beta) \times U(0,1))) \oplus ((\text{def}_{\text{old}} \odot (\sin(\frac{\pi}{2} - \beta) \times U(0,1))); \]  \hfill (8)

Diversion theft
\[ \text{def}_{\text{new}} = (\text{def}_{\text{old}} \odot (1 - \sin \beta \times U(0,1))) \oplus ((\text{def}_{\text{old}} \odot (\sin(\frac{\pi}{2} - \beta) \times U(0,1))) \odot (\sin \beta \times U(0,1))); \]  \hfill (9)

Pretext
\[ \text{def}_{\text{new}} = (\text{def}_{\text{old}} \odot (\sin(\frac{\pi}{2} - \beta) \times U(0,1)) \odot (1 - \sin \beta \times U(0,1))) \oplus (((\text{def}_{\text{old}} \odot (\sin(\frac{\pi}{2} - \beta) \times U(0,1)))) \odot (\sin \beta \times U(0,1))); \]  \hfill (10)

where \( \beta \in [0, 1] \) is the input variable of search engine.

(5) Check the disassembly sequence

The defender generated through the above steps needs to verify its dismountability. Starting from the first position of the sequence, if the part represented by this position is dismountable, check the dismountability of the next part; if it is not dismountable, randomly select one of the dismountable parts to replace, and then check the dismountability of the next part. Repeat the above operations until the last part meets the dismountability requirements;

(6) Respond to attack and create a new defender

Let \( m \) be the number of attack times. After each attack, evaluate the new fitness of the defender and compare it with the old one, then choose a better fitness as the defender. If the fitness of the defender is better than attacker, the identities of the two are exchanged. When the attack lasts \( m \) times, the attack stops. Then, a new feasible solution vector is generated as the defender. Repeat the above steps until the maximum number of iterations is reached or the algorithm stops.

The optimization procedure of the proposed method is shown in Figure 7.
where $\beta \in [0, 1]$ is the input variable of search engine;

(5) Check the disassembly sequence

The defender generated through the above steps needs to verify its dismountability. Starting from the first position of the sequence, if the part represented by this position is dismountable, check the dismountability of the next part; if it is not dismountable, randomly select one of the dismountable parts to replace, and then check the dismountability of the next part. Repeat the above operations until the last part meets the dismountability requirements;

(6) Respond to attack and create a new defender

Let $m$ be the number of attack times. After each attack, evaluate the new fitness of the defender and compare it with the old one, then choose a better fitness as the defender. If the fitness of the defender is better than attacker, the identities of the two are exchanged. When the attack lasts $m$ times, the attack stops. Then, a new feasible solution vector is generated as the defender. Repeat the above steps until the maximum number of iterations is reached or the algorithm stops.

The optimization procedure of the proposed method is shown in Figure 7.

Figure 7. The flowchart of improved social engineering optimizer (SEO).

4. Results and Discussion

4.1. Problem Description

In order to verify the effectiveness, an example is used taken from [45]. A turbine reducer consists of 25 components, as shown in Figure 8. According to the product assembly relationship and spatial position constraints, the disassembly hybrid graph of the turbine reducer can be obtained, as shown in Figure 9.

The original disassembly information of components is shown in Table 1. According to the steps of the developed method presented in Section 4, disassembly sequence planning can be carried out.
Table 1. Parts of turbine reducer.

| Order | Name                                | Quantity | Tool                     | Direction |
|-------|-------------------------------------|----------|--------------------------|-----------|
| 1     | Shell (non-removable)               | 1        | —                        | —         |
| 2     | Grease fitting                      | 1        | Wrench (T₁)              | +z        |
| 3     | Turbine shaft shim end cover        | 1        | Special tool (T₂)        | −y        |
| 4     | Hexagon socket head cap screws      | 4        | Allen wrench (T₃)        | +y        |
| 5     | Turbine shaft end cover 1           | 1        | Hand (T₀)                | −y        |
| 6     | Skeleton oil seal 1                 | 1        | Hammer (T₄)              | −y        |
| 7     | Turbine shaft bearing 1             | 1        | Hammer (T₄)              | −y        |
| 8     | Turbine                            | 1        | Special tool (T₅)        | +y        |
| 9     | Turbine shaft                       | 1        | Hammer (T₄)              | −y        |
| 10    | Slotted set screws with flat point  | 3        | Screwdriver (T₆)         | −y        |
| 11    | Turbine shaft bearing 2             | 1        | Hammer (T₄)              | −y        |
| 12    | Skeleton oil seal 2                 | 1        | Hammer (T₄)              | −y        |
| 13    | Turbine shaft end cover 2           | 1        | Hand (T₀)                | −y        |
| 14    | Hexagon socket head cap screws      | 4        | Allen wrench (T₃)        | −x        |
| 15    | Hexagon socket head cap screws      | 4        | Allen wrench (T₃)        | −x        |
| 16    | Worm shaft end cover 1              | 1        | Hand (T₀)                | −x        |
| 17    | Oil seal 1                          | 1        | Tong (T₇)                | −x        |
| 18    | Worm shaft bearing 1                | 1        | Hammer (T₄)              | −x        |
| 19    | Bearing cap gasket 1                | 1        | Special tool (T₂)        | −x        |
| 20    | Worm                                | 1        | Special tool (T₅)        | −x        |
| 21    | Bearing cap gasket 2                | 1        | Special tool (T₂)        | +x        |
| 22    | Worm shaft bearing 2                | 1        | Hammer (T₄)              | +x        |
| 23    | Oil seal 2                          | 1        | Tong (T₇)                | +x        |
| 24    | Worm shaft end cover 2              | 1        | Hand (T₀)                | +x        |
| 25    | Hexagon socket head cap screws      | 4        | Allen wrench (T₃)        | +x        |

Figure 8. Drawing of turbine reducer.
A turbine reducer consists of 25 components, as shown in Figure 8. According to the disassembly sequence planning can be carried out.

Table 1. Parts of turbine reducer.

| Order | Name                        | Quantity | Tool          | Direction |
|-------|-----------------------------|----------|---------------|-----------|
| 1     | 25 Hexagon socket head cap screws | 4        | Allen wrench (T 3) | -y        |
| 2     | 17 Oil seal                 | 1        | 1 Tong (T 7)  | -y        |
| 3     | 24 Worm shaft end cover     | 2        | 1 Hand (T 0)  | -y        |
| 4     | 18 Worm shaft bearing       | 1        | 1 Hammer (T 4) | -y        |
| 5     | 14 Hexagon socket head cap screws | 4        | Allen wrench (T 3) | -y        |
| 6     | 22 Worm shaft bearing       | 2        | 1 Hammer (T 4) | -y        |
| 7     | 21 Bearing cap gasket       | 2        | 1 Special tool (T 2) | -y    |
| 8     | 20 Worm                     | 1        | Special tool (T 5) | -y        |
| 9     | 23 Oil seal                 | 2        | 1 Tong (T 7)  | -y        |
| 10    | 19 Bearing cap gasket       | 1        | 1 Special tool (T 2) | -y    |
| 11    | 11 Turbine shaft bearing    | 2        | 1 Hammer (T 4) | -y        |
| 12    | 13 Turbine shaft end cover  | 2        | 1 Hand (T 0)  | -y        |
| 13    | 10 Slotted set screws with flat point | 3 | Screwdriver (T 6) | -y      |
| 14    | Grease fitting              | 1        | Wrench (T 1)  | -y        |
| 15    | Shell (non-removable)       | 1        |               | -y        |
| 16    | 9 Turbine                   | 1        | Hammer (T 4)  | -y        |
| 17    | 4 Hexagon socket head cap screws | 4        | Allen wrench (T 3) | -y        |
| 18    | 6 Skeleton oil seal         | 1        | 1 Hammer (T 4) | -y        |
| 19    | 5 Turbine shaft end cover   | 1        | 1 Hand (T 0)  | -y        |
| 20    | 3 Turbine shaft shim end cover | 1 | Special tool (T 2) | -y    |
| 21    | 2 Turbine                   | 1        | Special tool (T 5) | -y        |
| 22    | 7 Turbine shaft bearing     | 1        | 1 Hammer (T 4) | -y        |

Figure 9. Hybrid graph model of turbine reducer.

4.2. Influence of Parameters

According to Section 3, a and β are two parameters that affect the results of SEO. To analyze the sensitive of parameters in SEO, three different values of α and β are taken and substituted into the program. Let the number of iterations be 300, \( t_e = 8 \), \( t_d = 4 \), \( \omega_l = \omega_e = \omega_d = 1 \), \( m = 15 \). Run this 20 times to get nine sets of data, as shown in Figure 10.

![Boxplot graph of disassembly time under different parameters](image)

Figure 10. Boxplot graph of disassembly time under different α and β. (a) α = 0.2; (b) α = 0.5; (c) α = 0.8.

It can be seen from the figure that the larger α is, the closer the result is to the optimal solution and the more concentrated it is. In a certain range, the increase in β will improve the optimization accuracy of the program. However, if β is too large, the optimization accuracy will decrease and the dispersion of the results will increase.

4.3. Comparison with Other Algorithms

In order to verify the effectiveness of SEO, use the models of genetic algorithm (GA), ant colony optimization (ACO), artificial bee colony (ABC) to solve the disassembly sequence of the turbine reducer and compare with the result of SEO. Let the number of attacks \( m \) (population number in other algorithms) be 5, 15 and 30, respectively, and let...
the number of iterations be 300, $\alpha = 0.5$, $\beta = 5\pi/18$, $t_e = 8$ s, $t_d = 4$ s, $\omega_1 = \omega_2 = \omega_d = 1$. Each algorithm runs 20 times. Figure 11 shows the boxplot graph of the four algorithms under different population numbers. “+” indicates an outlier in data.

Figure 11. Boxplot graph of disassembly time for genetic algorithm (GA), artificial bee colony (ABC), ant colony optimization (ACO), and social engineering optimizer (SEO). (a) $m = 5$; (b) $m = 15$; (c) $m = 30$.

It can be seen that when the number of attacks (population number) reaches a certain level, the optimization result of ACO is the best and most stable. However, when the number of attacks (population number) is low, the results of GA, ABC, and ACO are more dispersed, while SEO can maintain better accuracy. In actual production, when dismantling a group of institutions with similar but different structures, or when random damage occurs in the institutions and the disassembly sequence needs to be re-planned, the operating efficiency of the program must also be taken into consideration. The proposed method can effectively balance efficiency and accuracy.

Let $m = 50$, and the number of iterations be 500; other parameters are the same. For the illustration in Section 4.1, the convergence curve of a run is shown in Figure 12; the disassembly sequences of the four algorithms are shown in Table 2.

It can be seen that the proposed SEO algorithm has good convergence efficiency compared to other algorithms.
5. Conclusions

This paper proposes a novel DSP method based on social engineering optimizer. In order to solve the problem that the social engineering optimizer cannot solve the discrete combinatorial optimization, the swap operator and the swap sequence are introduced in SEO, and the swap sequence of the disassembly sequence is used as the optimization variable of the algorithm. On this basis, the calculation rule of exchange order and the new step of social engineering optimizer are defined. In order to generate a feasible disassembly sequence, a method of generating and correcting a disassembly sequence based on a constraint matrix is proposed. Finally, a turbo reducer is used to verify the proposed algorithm. Compared with known algorithms, this algorithm can better solve the disassembly sequence planning problem.

In actual production, the research on the disassembly line balancing problem, that is, the efficiency optimization of the disassembly process, is more extensive. As a way to improve disassembly efficiency, disassembly sequence planning is an important part of the research into the balance of disassembly lines. However, the condition of the internal parts of EOL products is unknown, and the preset disassembly sequence generally cannot meet actual requirements. Setting the internal conditions of EOL products as random variables and performing partial destructive disassembly is a solution. At the same time, parallel disassembly sequence planning is also a means to improve disassembly efficiency. However, the swap sequence is very effective in dealing with the sequence of the same length, but it cannot handle the partial disassembly and parallel disassembly. In the following research, we will focus on combining the proposed method with this.
Author Contributions: Conceptualization, A.M.F.-F., C.Z.; Methodology, A.M.F.-F., C.Z., T.Z.; Software, A.M.F.-F., C.Z.; Supervision, J.L., G.T.; writing—original draft, C.Z.; writing—review and editing, all coauthors. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China, grant number 52075303 and 51775238, and supported by State Key Laboratory of Robotics and Systems (HIT) (SKLRS-2021-KF-09), and State Key Laboratory of Fluid Power & Mechatronic Systems, grant number GZKF-202012.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available in a publicly accessible repository.

Acknowledgments: The authors are thankful to the technical and financial support of National Natural Science Foundation of China.

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

References
1. Zhu, X.T.; Zhang, Z.Q.; Zhu, X.M. An ant colony optimization algorithm for multi-objective disassembly line balancing problem. China Mech. Eng. 2014, 25, 1075–1079.
2. Guo, L.; Zhang, X.F. Remanufacturing parallel disassembly sequence planning method driven by multiple failures. J. Zhejiang Univ. Eng. Sci. 2020, 54, 2233–2246.
3. Henrioud, J.M.; Bourjault, A.L. A Computer-Aided Generator of Assembly Plans; Springer: Boston, MA, USA, 1991; pp. 191–215.
4. Homem, D.M.L.S.; Sanderson, A.C. AND/OR graph representation of assembly plans. IEEE Trans. Robot. Autom. 1990, 6, 188–199. [CrossRef]
5. Li, J.R.; Khoo, L.P.; Tor, S.B. A novel representation scheme for disassembly sequence planning. Int. J. Adv. Manuf. Tech. 2002, 20, 621–630. [CrossRef]
6. Huang, Y.M.; Huang, C.T. Disassembly matrix for disassembly processes of products. Int. J. Prod. Res. 2002, 40, 255–273. [CrossRef]
7. Mitrouchev, P.; Wang, C.G.; Lu, L.X.; Li, G.Q. Selective disassembly sequence generation based on lowest level disassembly graph method. Int. J. Adv. Manuf. Tech. 2015, 80, 141–159. [CrossRef]
8. Hu, Y.; Srinivasan, R.; Spoll, J.; Ameta, G. Graph based method and tool for complete and selective disassembly time estimation in early design. J. Comput. Inf. Sci. Eng. 2015, 15, 031005. [CrossRef]
9. Edmunds, R.; Kobayashi, M.; Higashi, M. Using constraint-satisfaction to optimize disassembly sequences generated from and/or information. Int. J. Prod. Res. 2012, 50, 4105–4126. [CrossRef]
10. Smith, S.S.; Chen, W.H. Rule-based selective disassembly sequence planning for green design. Adv. Eng. Inform. 2011, 25, 77–87. [CrossRef]
11. Issaoui, L.; Aïfaoui, N.; Benamara, A. Model of mobility state of parts, the automation of feasibility test in disassembly sequence generation. Proc. Inst. Mech. Eng. C J. Mech. Eng. Sci. 2017, 231, 3702–3714. [CrossRef]
12. Tian, G.D.; Ren, Y.P.; Feng, Y.X.; Zhou, M.C.; Zhang, H.H.; Tan, J.R. Modeling and planning for dual-objective selective disassembly using AND/OR graph and discrete artificial bee colony. IEEE Trans. Ind. Inform. 2019, 15, 2456–2468. [CrossRef]
13. Wang, W.J.; Tian, G.D.; Zhang, T.Z.; Jabarullah, N.H.; Li, F.Y.; Fathollahi-Fard, A.M.; Wang, D.Q.; Li, Z.W. Scheme selection of design for disassembly (DFD) based on sustainability: A novel hybrid of interval 2-tuple linguistic intuitionistic fuzzy numbers and regret theory. J. Clean. Prod. 2020, 281, 124724. [CrossRef]
14. Feng, Y.X.; Gao, Y.C.; Tian, G.D.; Li, Z.W.; Hu, H.S.; Zheng, H. Flexible process planning and end-of-life decision-making for product recovery optimization based on hybrid disassembly. IEEE Trans. Autom. Sci. Eng. 2019, 16, 311–326. [CrossRef]
15. Yuan, G.; Yang, Y.S.; Tian, G.D.; Zhuang, Q.W. Comprehensive evaluation of disassembly performance based on the ultimate cross-efficiency and extension-gray correlation degree. J. Clean. Prod. 2020, 245, 118800. [CrossRef]
16. Jiao, Q.L.; Xu, D.; Li, C. Product disassembly sequence planning based on flower pollination algorithm. Comput. Integr. Manuf. Syst. 2016, 22, 2791–2799.
17. Zhang, X.F.; Wei, G.; Wang, L. Parallel disassembly sequence planning for complex products based on genetic algorithm. J. Comput. Aided Des. Comput. Graph. 2015, 27, 1327–1333.
18. Jin, L.L.; Zhang, C.Y.; Fei, X.J. Realizing energy savings in integrated process planning and scheduling. Processes 2020, 7, 120. [CrossRef]
19. Hu, Z.B.; He, D.F.; Song, W.; Feng, K. Model and algorithm for planning hot-rolled batch processing under time-of-use electricity pricing. Processes 2020, 8, 42. [CrossRef]
20. Wang, X.; Du, Z.Z.; Zhang, Y.L.; Wang, J.D.; Wang, J.H.; Sun, W. Optimization of distillation sequences with nonsharp separation columns. Processes 2019, 7, 323. [CrossRef]
22. Zhang, X.F.; Zhang, S.Y. Product disassembly sequence planning based on particle swarm optimization algorithm. *Comput. Integr. Manuf. Syst.* 2009, 15, 508–514.

23. Xing, Y.F.; Wang, C.E.; Liu, Q. Disassembly sequence planning based on pareto ant colony algorithm. *J. Mech. Eng.* 2012, 48, 186–192. [CrossRef]

24. Wu, H.; Zuo, H.F. Product disassembly sequence planning based on improved genetic algorithm. *China Mech. Eng.* 2009, 20, 699–703.

25. Kheder, M.; Trigui, M.; Aifaoui, N. Disassembly sequence planning based on a genetic algorithm. *Proc. Inst. Mech. Eng. C J. Mech. Eng. Sci.* 2015, 229, 2281–2290. [CrossRef]

26. Tseng, H.E.; Chang, C.C.; Lee, S.C.; Huang, Y.M. A block-based genetic algorithm for disassembly sequence planning. *Expert Syst. Appl.* 2018, 96, 492–505. [CrossRef]

27. Zhang, L.; Peng, H.W.; Bian, B.Y. Parallel disassembly modeling and planning method of complex products. *China Mech. Eng.* 2014, 25, 937–943. [CrossRef]

28. Ren, Y.P.; Tian, G.D.; Zhao, F.; Yu, D.Y.; Zhang, C.Y. Selective cooperative disassembly planning based on multiobjective discrete artificial bee colony algorithm. *Eng. Appl. Artif. Intell.* 2017, 64, 415–431.

29. Yeh, W.C. Simplified swarm optimization in disassembly sequencing problems with learning effects. *Comput. Oper. Res.* 2012, 39, 2168–2177. [CrossRef]

30. Xia, K.; Gao, L.; Li, W.D.; Chao, K.M. Disassembly sequence planning using a simplified teaching learning-based optimization algorithm. *Adv. Eng. Inform.* 2014, 28, 518–527. [CrossRef]

31. Tao, F.; Bi, L.N.; Zuo, Y.; Nee, A.Y.C. Partial/parallel disassembly sequence planning for complex products. *J. Manuf. Sci. Eng.* 2018, 140, 011016. [CrossRef]

32. Ren, Y.P.; Zhang, C.Y.; Zhao, F.; Xiao, H.J.; Tian, G.D. An asynchronous parallel disassembly planning based on genetic algorithm. *Eur. J. Oper. Res.* 2018, 269, 647–660. [CrossRef]

33. Guo, X.W.; Liu, S.X.; Zhou, M.C.; Tian, G.D. Dual-objective program and scatter search for the optimization of disassembly sequences subject to multiresourse constraints. *IEEE Trans. Autom. Sci. Eng.* 2018, 15, 1098–1103. [CrossRef]

34. Tian, G.D.; Zhou, M.C.; Li, P.G. Disassembly sequence planning considering fuzzy component quality and varying operational cost. *IEEE Trans. Autom. Sci. Eng.* 2018, 15, 748–760. [CrossRef]

35. Yang, Y.S.; Yuan, G.; Zhuang, Q.W.; Tian, G.D. Multi-objective low-carbon disassembly line balancing for agricultural machinery using MDFOA and fuzzy AHP. *J. Clean. Prod.* 2019, 233, 1465–1474. [CrossRef]

36. Stefanini, A.; Aloini, D.; Benevento, E.; Dulmin, R.; Mininno, V. A data-driven methodology for supporting resource planning of health services. *Socio-Econ. Plan. Sci.* 2020, 70, 100744. [CrossRef]

37. Parsa, S.; Saadat, M. Intelligent selective disassembly planning based on disassemblability characteristics of product components. *Int. J. Adv. Manuf. Tech.* 2019, 104, 1769–1783. [CrossRef]

38. Bentaha, M.L.; Voisin, A.; Marangé, P. A decision tool for disassembly process planning under end-of-life product quality. *Int. J. Prod. Econ.* 2020, 219, 386–401. [CrossRef]

39. Babbitt, C.W.; Madaka, H.; Althaf, S.; Kasulaitis, B.; Ryan, E.G. Disassembly-based bill of materials data for consumer electronic products. *Sci. Data 2020*, 7, 251. [CrossRef]

40. Tian, G.D.; Hao, N.N.; Zhou, M.C.; Pedrycz, W.; Zhang, C.Y.; Ma, F.W.; Li, Z.W. Fuzzy grey Choquet integral for evaluation of multicriteria decision making problems with interactive and qualitative indices. *IEEE Trans. Syst. Man Cybern. Syst.* 2021, 5, 1855–1868. [CrossRef]

41. Tian, G.D.; Zhang, H.H.; Feng, Y.X.; Jia, H.F.; Zhang, C.Y.; Jiang, Z.G.; Li, Z.W.; Li, P.G. Operation patterns analysis of automotive components remanufacturing industry development in China. *J. Clean. Prod.* 2017, 146, 1363–1375. [CrossRef]

42. Wang, W.J.; Tian, G.D.; Chen, M.N.; Tao, F.; Zhang, C.Y.; Ai-Ahmari, A.; Li, Z.W.; Jiang, Z.G. Dual-objective program and improved artificial bee colony for the optimization of energy-conscious milling parameters subject to multiple constraints. *J. Clean. Prod.* 2020, 245, 118714. [CrossRef]

43. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evolut. Comput.* 1997, 1, 67–82. [CrossRef]

44. Amir, M.F.F.; Mostafa, H.K.; Reza, T.M. The social engineering optimizer (SEO). *Eng. Appl. Artif. Intell.* 2018, 72, 267–293.

45. Yang, D.Y.; Xu, Z.G.; Zhu, J.F.; Su, K.Y.; Liu, W.M. Objective selective disassembly sequence planning considered product fault features. *J. Harbin Inst. Technol.* 2019, 51, 160–170.

46. Chen, Y.W.; Chen, W.D. Product disassembly sequence optimization based on profit-probability under uncertain environment. *Comput. Integr. Manuf. Syst.* 2014, 20, 793–798.

47. Krombholz, K.; Hobel, H.; Huber, M.; Weippel, E. Advanced social engineering attacks. *J. Inf. Secur. Appl.* 2015, 22, 113–122. [CrossRef]

48. Huang, L.; Wang, K.P.; Zhou, C.G.; Pang, W.; Dong, L.J.; Peng, L. Particle swarm optimization for traveling salesman problems. *J. Jilin Univ. Sci. Ed.* 2003, 41, 477–480.