國立臺灣大學工學院機械工程學系

碩士論文

Department of Mechanical Engineering

College of Engineering

National Taiwan University

Master Thesis

以規則為基礎的遞歸選擇性拆卸序列規劃的綠色設計

Rule-Based Recursive Selective Disassembly Sequence Planning for Green Design

陳維祥

Chen, Wei-Hsiang

指導教授：陳湘鳳 博士

Advisor: Shana Smith, Ph.D.

中華民國 98 年 6 月

June, 2009

國立臺灣大學碩士學位論文
口試委員會審定書

以規則為基礎的遞歸選擇性拆卸序列規劃的綠色設計
Rule-Based Recursive Selective Disassembly Sequence Planning for Green Design

本論文係陳維祥君（R95522626）在國立臺灣大學機械學系完成之碩士學位論文，於民國 98 年 06 月 23 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

陳湘鳳 陳湘鳳（簽名）

（指導教授）

陳湘鳳 陳湘鳳

尤春風 尤春風

劉　霆 劉霆

系 主 任 張所鋐 張所鋐（簽名）

（是否須簽章依各院系所規定）

摘要

拆卸序列計劃不僅減少產品的生命週期費用，而且還大大地影響環境衝擊。 所以，許多先前的綠色設計研究，集中於廢棄產品的完全拆卸再利用、回收和重新生產有用或貴重的組成成分。要減少環境的影響，許多國家設定某些法規避免進口對環境不友好的產品。考慮綠色設計在拆卸序列計畫階段，參考環境法規是重要的。然而，如果一個特定產品僅回收部分零件或成分，完全拆卸則不實用或不符合經濟效益。選擇性的拆卸序列計劃通常被用於，從產品中拆卸一個或多個成分以重複利用、回收、恢復、和重新製造，以減少對環境的影響。

多數先前的方法列舉所有解答或使用一個隨機方法產生任意解。列舉或隨機方法經常要求巨大計算資源，同時，他們經常可能無法找到符合現實或最佳的方案。這份論文提出一個規則性的遞歸方法以優化在綠色設計中的選擇性拆卸序列。並且，本論文建立某些拆卸規則，其可除去不可行或不切實際的解答。利用這些拆卸規則，能夠減少需要用到的計算資源，並且有效地找出高品質的解。

基於已訂定的規則，在拆除任何零件之前，必須先將限制零件移動的緊固件拆除。然而，在拆除緊固件之前，可能有其它的零件或緊固件也須要移除。在這篇論文裡，設計三個主要的運算功能，對零件和緊固件進行遞歸的移除動作。另外，並不是就每一個單一零件對所有

其它零件作幾何限制，本論文只考慮該零件和它的相鄰零件之間的幾何關係。如果取出的零件可以被拆卸，它與相鄰零件之間的幾何關係將被刪除並且更新。最後，此方法可以有效地找出一個在綠色設計中優化選擇性的拆卸序列，並且大大地減少計算時間和空間。

關鍵字：選擇性拆卸序列計劃，規則為基礎，遞歸，優化，綠色設計.

Abstract

Disassembly sequence planning not only reduces product lifecycle cost, but also greatly influences environmental impact. Therefore, many prior green design research studies have focused on complete disassembly of an end-of-life product to reuse, recycle, recovery, and remanufacturing useful or valuable components. To reduce environmental impact, many countries set up certain regulations to avoid importing environmental unfriendly products. In green design, it is important to consider environmental regulations during the disassembly sequence planning stages. However, complete disassembly is often not practical or cost effective if only a few components will be recovered and recycled from a given product. Selective disassembly sequence planning is usually used to only disassemble one or more components from a product to reuse, recycle, recovery and remanufacturing to reduce environmental impact.

Most prior methods either enumerate all solutions or use a stochastic method to generate random solutions. Enumerative or stochastic methods often require tremendous computational resources while, at the same time, they often fail to find realistic or optimal solutions. This thesis presents a rule-based recursive method for finding an optimal heuristic selective disassembly sequence for green design. Based on certain heuristic disassembly rules, the proposed method can eliminate uncommon or unrealistic solutions. Thus, it can greatly reduce computational resources and find high-quality solutions effectively.

Based on the defined rules, before any component can be removed, its attached fasteners need to be removed first. However, before the fasteners can be removed, other components or fasteners might need to be removed. In this research, three major functions are developed to handle the recursive removal of components and fasteners. In addition, rather than considering geometric constraints for each pair of components, the developed method only considers geometric relationships between a part and its neighboring parts. If a retrieved part can be disassembled, its geometric relationships with the neighboring parts will dynamically be deleted and updated. As a result, the developed method can effectively find an optimal heuristic selective disassembly sequence while greatly reducing computational time and space.

Keywords. Selective disassembly sequence planning, rule-based, recursive, optimal, green design.

Contents

Figure of Contents

Table of Contents

Chapter 1 Introduction

In early study, many researchers focus on optimal assembly or disassembly sequence planning in manufacturing. The

purpose is to reduce the manufacturing cost and increase the product value. However, having an optimal assembly/disassembly sequence is not the only way to reduce the product cost. Other research focuses on reusing the useful components or valuable materials. Thus, disassembly sequence designers should consider not only finding an optimal disassembly sequence but also reusing, recycling, and remanufacturing useful parts.

There is no clear definition to define what “Green Design” is. However, many studies consider planning optimal disassembly sequences, reducing environment impacts, and reusing, recycling, recovery and remanufacturing of end-of-life products is “Green Design”. In order to reduce the environmental impacts, many countries set up certain regulations to avoid importing environmentally harmful products. There are three general regulations: Waste Electrical and Electronic Equipment Directive (WEEE), the restriction of the use of certain hazardous substances in electrical and electronic equipment (Restriction of Hazardous Substances Directive 2002/95/EC, RoHS) and Energy Using Products Directive (EuP), which is used widely in electrical and electronic equipments and energy-using products.

WEEE proposes that products which can be easily disassembled and be easily removed can be easily reused, recovered, recycled and remanufactured [24]. It prevents manufacturers from producing special design products so that the end-of-life products can be easily disassembled for reuse, recycle, recovery and remanufacturing, unless the special designed products are good, efficient and useful for reducing the environment impacts [24]. RoHS restricts producers to include parts with hazardous substances above the maximum concentration values and the producers have to provide a documentation to show that the product is compliant RoHS. EuP requires manufacturers to consider life cycle design and ecodesign requirements in product design [26]. The life cycle means the consecutive and interlinked stages of an energy-using product from raw material to final disposal [26]. The ecodesign requirement means any requirement in relation to an energy-using product, or the design of an energy-using product, intended to improve its environmental performance, or any requirement for the supply of information with regard to the environmental aspects of an energy-using product [26].

Selective disassembly sequence planning is especially important for green design because selective disassembly sequence planning usually disassembles one or more components from a product to reuse, recycle, recovery, or remanufacturing to reduce the environmental impacts. Selective disassembly sequence planning is a powerful and an efficient tool for solving de-manufacturing (DM) problem. DM involves separating certain components and materials from a product for reuse, recycle, replace, and maintenance to increase product life cycle cost [6]. However, finding an optimal selective disassembly sequence is a very difficult and complex problem when multiple factors are involved, e.g., disassembly time, cost, reorientations, tools, and environmental regulations.

Some prior studies have utilized advanced searching algorithms to find optimal selective disassembly sequences. Srinivasan et al. applied a wave propagation method to solve selective disassembly problems [19, 20, 21]. They used the geometric and topological information from the CAD models of components to determine the selective disassembly sequences. Their method does not only focus on disassembling one component but also multiple components, or even the total selective disassembly [19]. They assume four conditions for their selective disassembly sequence planning: 1) the relative motions of the components are determined without considering the tools, fixtures, or robots; 2) assemblies are assumed to be frictionless and nominal geometry; 3) components are removed from an assembly by single linear motions and they are removable after removing one of their adjacent components. Fastener are not considered to be components; 4) the disassembly sequences remove one component at a time, and all components can be removed by a non-destructive disassembly method [19, 20]. They evaluate each selective disassembly sequence by the number of removed components.

They consider the optimal selective disassembly sequence to be the sequence with the minimum number of removals [19, 20, 21]. Although their evaluation function is simple, it might not satisfy the demand of certain product designs. Therefore, Chung and Peng [7] added more evaluation criteria when using wave propagation method in selective disassembly sequence planning, e.g., disassembly time, cost, and tool changes.

Ant colony optimization (ACO) algorithms are also used in the optimal selective disassembly sequence planning problem [23]. Most ACO-based selective disassembly sequence planning consider the geometric constrains of the assembly and evaluate each selective disassembly sequence with the number of reorientations and the number of removed components to get the optimal solutions [23, 24, 25, 27].

Other than the wave propagation method and ACO, some other methods are also used in the selective disassembly sequence planning problems, e.g., Kara et al. [14] reversed and modified assembly sequences to obtain disassembly sequences. They used a liaison diagram to show the geometric connections, Chung and Peng [7] used heuristic methods, e.g., genetic algorithms, to solve the selective disassembly planning problem, Aguinaga et al. [2] used a rapid-growing random tree method to solve the selective disassembly planning problem. However, their method generates too many paths, and, thus, it takes a significant amount of time to find optimal sequences. In addition, their results might not be consistent.

Shyamsundar and Gadh [18] developed a recursive method which analyzes the geometric information and considers both separation directions and disassembly directions to remove a target component from an assembly. It is not easy to set up the input information for the separation directions and disassembly directions. Srinivasan and Gadh also developed a global selective disassembly method which considers non-interfering (collision free) geometrical constraints, including the spatial constraints and user-defined constraints [22].

Most searching methods use specific information in their searching processes: geometric constraints [2, 11, 15, 22, 23, 26], topological positions [7, 15, 22, 26], liaison relationships [14], AND/OR graphs [2, 11], precedence graphs [11], fastener accessibility [5], and component accessibility [22]. Criteria used to evaluate disassembly sequences include the number of removed components [1, 7, 11, 15, 22, 23, 24, 26], disassembly time [2, 6, 13, 14, 25], reorientations [23, 25, 27], and tool changes [7, 25, 27]. Usually, "cost" is a controlling evaluation factor. To reduce disassembly cost, most selective disassembly sequence planning methods focus on minimizing the number of removed components, disassembly time, and reorientation time. Table 1 shows the existing selective disassembly sequence planning methods and their approaches and evaluation methods.

Table 1. Approaches of the existing selective disassembly sequence planning methods.

Author(s)	Methodology	Input Information	Evaluation
Srinivasan et al. [19, 20, 21]	Wave propagation	Geometric and topological information	Minimal removals
Garcia et al. [11]	Wave propagation	Geometric information, precedence graph, and AND/OR graph	Minimal removals
Mascle and Balasoiu [15]	Wave propagation	Tool or components accessibility and topological information	Minimal removals
Chung and Peng [5,6]	Wave propagation	Topological information, tool accessibility, and fastener accessibility	Time
Yi et al. [26]	Wave propagation	Geometric and topological information	Minimal removals
Wang et al. [23,24]	Ant colony optimization	Geometric information, pro toolkit, and Pro/E CAD models	Minimal removals and reorientation
Xue et al. [25]	Ant colony optimization	Disassembly hierarchy information graph (DHIG) and disassembly precedence constrain matrix (DPM)	Time, reorientation, and tool changes
Zhan et al. [27]	Ant colony optimization	Hybrid graph	Reorientation and tool changes
Shyamsunder and Gadh [18]	Recursive method	Geometric information and virtual prototype	Minimal removals
Aguinaga et al. [1,2]	Rapid-growing random tree (RRT)	Geometric information , AND/OR graph, and CAD	Time
Srinivasan and Gadh [22]	Global selective disassembly	Geometric and topological	Minimal removals

	algorithm	Information, and component accessibility	
Kara et al. [13,14]	Reversing and modifying a methodology developed by nevins and whitney (1989)	Liaison relationship	Time
Chung and Peng [7]	Genetic algorithms (GAs)	Topological information	Removals, time, tool changes, and weights

According to Table 1, most selective disassembly sequence planning methods focus on minimizing the removal of components, the disassembly time, and the reorientation times to reduce the cost of the selective disassembly process. The purpose is to find optimal selective disassembly sequences and increase the value of EOL products and reduce the environmental impacts through the effective DM methods. Selective disassembly sequence planning research aims to find optimal solutions to the selective disassembly planning problem. However, finding an optimal solution is a difficult problem. Most prior methods either enumerate all solutions or use stochastic methods to generate random solutions. Methods which enumerate all solutions can find optimal solutions. However, they might require a tremendous amount of computational resources. Therefore, they are generally not practical for solving realistic product design problems. As a result, most recent methods aim to find near-optimal or heuristic solutions. Stochastic random methods, such as ACO and GAs, might generate solutions which meet geometric and topological constraints. However, the given solutions might not be practical for use in reality.

This paper presents a rule-based recursive method for obtaining optimal heuristic selective disassembly sequences. The method uses certain disassembly rules to eliminate uncommon or unrealistic solutions. The geometric and topological information and fastener accessibility of a product will be examined from inward to outward to set up any possible disassembly sequences until the target component is removed. Use the proposed recursive techniques, only the geometric relationship of a component with its neighboring components and fasteners need to be considered. It greatly reduces the searching complexity which considers the geometric constraints between each pair of components.

Although the prior methods which use wave propagation and recursion also analyze the geometric information from the inward target component to the outward to generate disassembly sequences, prior methods require some conditions for an assembly and such conditions limit general application of their methods [18, 19, 20, 21].

Our method can handle both single-component and multiple-component disassembly problems. The evaluation function includes disassembly time, reorientations, number of removed components and fasteners, environmental impacts, and green design regulations. Disassembly time refers to the time required to remove a fastener or a component from an assembly. Reorientation refers to changing the next disassembly direction to remove a component from an assembly. Minimizing the number of removed components and fasteners can reduce the reorientation times, tool changes and disassembly time to increase the effectiveness of the DM process. Environmental impacts and green design regulations are used to reduce pollutant goods and increase the EOL product value. Three major regulations are used: Waste Electrical and Electronic Equipment Directive (WEEE) [9], the restriction of the use of certain hazardous substances in electrical and electronic equipment (Restriction of Hazardous Substances Directive 2002/95/EC, RoHS) [8] and the Energy Using Products Directive (EuP) [10].

Chapter 2 Definitions of Geometric Relationships

In this thesis, a component means a non-fastener element in a product. A part means either a component or a fastener in a product. Here, we consider parts are removed from an assembly by single linear motions.

2.1 Disassembly Parameters for Fasteners

We define a disassembly parameter matrix for fasteners, *DF*, which records the disassembly directions of each fastener. If a fastener can be disassembled out of an assembly along its axial direction without any collision, the corresponding tuple in *DF* is set to be *0*; otherwise, it is set to be *-1*. It is because we set positive integrate for components and fasteners identification. For example, in Figure 1, there are four fasteners, *5*, *6*, *7*, and *8*, and four components, *1*, *2*, *3*, and *4*. In Figure 1, the fastener *5* can be removed along +y direction, but not in other directions. Thus, DF_5 (+x : –x : +y : -y) = (*-1* : *-1* : *0* : *-1*). Likewise, DF_6 (+x : –x : +y : -y) = (*-1* : *0* : *-1* : *-1*), DF_7 (+x : –x : +y : -y) = (0 : *-1* : *-1* : *-1*), and DF_8 (+x : –x : +y : -y) = (*-1* : *-1* : *0* : *-1*). A fastener disassembly parameter matrix *DF* is composed of the disassembly parameters for all the fasteners. Thus, $DF = [DF_5\ DF_6\ DF_7\ DF_8]^T$:

$$DF = \begin{bmatrix} DF_5 \\ DF_6 \\ DF_7 \\ DF_8 \end{bmatrix} = \begin{bmatrix} -1 & -1 & 0 & -1 \\ -1 & 0 & -1 & -1 \\ 0 & -1 & -1 & -1 \\ -1 & -1 & 0 & -1 \end{bmatrix}$$

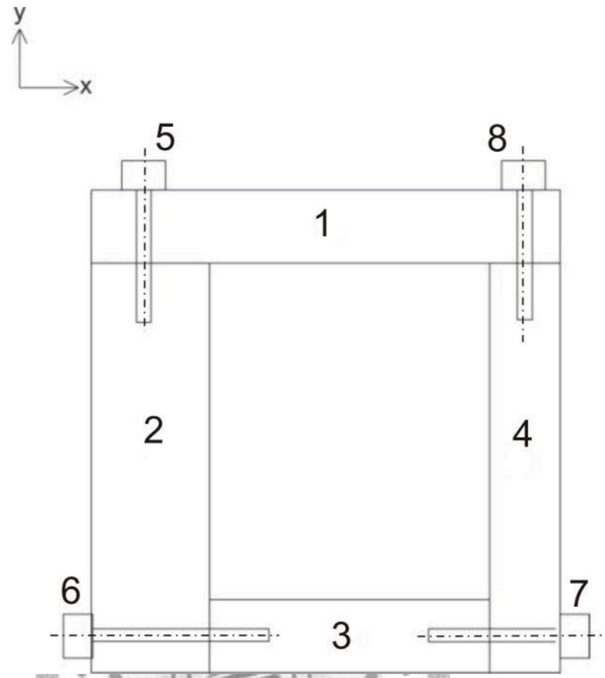


Figure 1. Example assembly 1.

In this study, we assume all fasteners can only be disassembled in one direction. If a fastener is welded to a component, it becomes an integral part of that component. Thus, in this study, the fastener will not has parameters like (+x : –x : +y : –y) = (*-1* : *-1* : *-1* : *-1*).

2.2 Disassembly Parameters for Components

We define a disassembly parameter matrix for components, *DC*, which records the immediately touched components and fasteners which constrain the motion of a target component in only one direction of a principal axis. We set the parameter value *0* if there are not any components and fasteners constraining the target component to disassemble. Parts constrain the motion of a target component in both directions of a principal axis will be considered in the next session.

If a component will collide with any fasteners or components when moving along a principal direction, the corresponding tuple in *DC* will record the immediate collided fasteners and components. Here, we use lower case letters to represent fasteners and upper case letters to represent components. For example, in Figure 1, DC_1 (+x : –x : +y : -y) =(0 : 0 : *5,8* : *2,4*). Likewise, DC_2 (+x : –x : +y : -y) = (*3* : *6* : *1,5* : 0), DC_3 (+x : –x : +y : -y) = (*4,7* : *2,6* : 0 : 0), and DC_4 (+x : –x : +y : -y) = (7 : *3* : *1,8* : 0). A component disassembly parameter matrix *DC* is composed of the disassembly parameters for all the components. Thus, $DC = [DC_1\ DC_2\ DC_3\ DC_4]^T$:

$$DC = \begin{bmatrix} DC_1 \\ DC_2 \\ DC_3 \\ DC_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 5,8 & 2,4 \\ 3 & 6 & 1,5 & 0 \\ 4,7 & 2,6 & 0 & 0 \\ 7 & 3 & 1,8 & 0 \end{bmatrix}$$

2.3 Motion Constraint Parameters

We define two motion constraint parameter matrices, one records motion constraints for fasteners (*MF*), and the other records motion constraints for components (*MC*). Before *MF* and *MC* can be defined, "first-level parts" needs to be defined. First-level parts are parts which do not immediately touch the target components or fasteners but which are the first parts beyond the immediately touching parts which would block movement of fasteners or target components in given moving directions.

The *MF* matrix records both the first-level parts of a fastener and any immediately touching components of the fastener in a given disassembly direction. For example, in Figure 2, there are six fasteners, *7*, *8*, *9*, *10*, *11* and *12*, and six components, *1*, *2*, *3*, *4*, *5*, and *6*. The fastener *12* can only be disassembled along the *+y* direction. However, since component *1* is the first component which fastener *12* would collide with, in the given disassembly direction, component *1* is a first-level component of fastener *12*. Thus, $MF_{12}(+x : -x : +y : -y) = (0 : 0 : 1 : 0)$.

Recall that the disassembly parameter matrix for components, *DC*, only records immediately touching components and fasteners which constrain motion of a target component in one direction of a principal axis. In contrast, *MC* records only the first-level parts of a target component, omits fasteners, and includes components and fasteners which constrain motion of the target component in both directions of a principal axis. For example, in Figure 2, the first-level part of component *6* is component *5*, and component *6* is also constrained by fastener *12* in both directions of the x-axis. Thus, MC_6 ($+x$: $-x$: $+y$: $-y$) = (*12* : *12* : *5* : 0). In the two parameters we set the parameter value *0* if it is not satisfying with the two conditions, “first-level parts” and the constraint includes in both directions of a principal axis. If we set *-1* in the two parameters, it means to constrain the target component to disassemble in the corresponding direction.

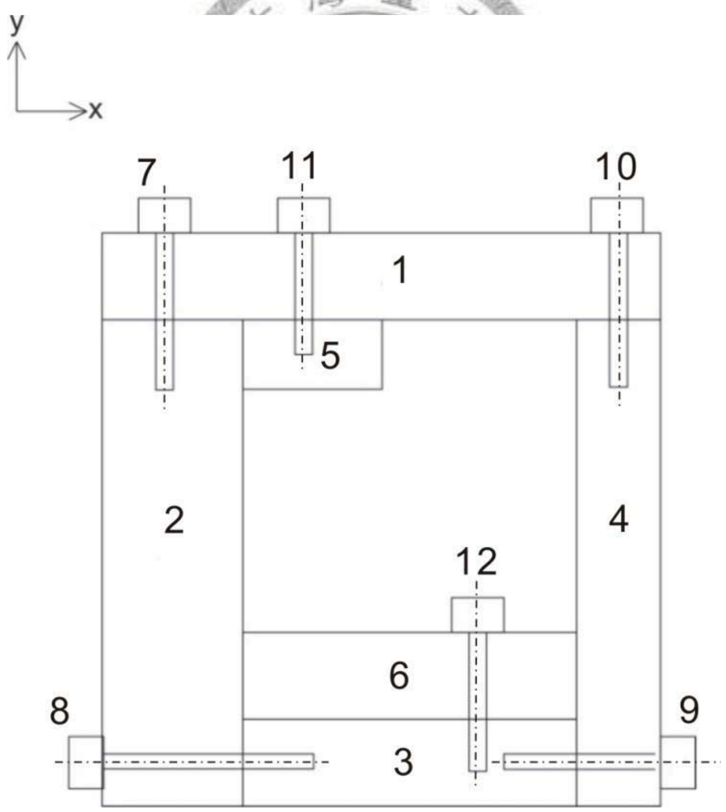


Figure 2. Example assembly 2.

Figure 3 shows another example. The assembly in Figure 3 includes two components, *1* and *2*, and one fastener *3*. For the given assembly, DC_1 (+*x* : –*x* : +*y* : -*y*) = (0 : 0 : *2,3* : 0), MC_1 (+*x* : –*x* : +*y* : -*y*) = (*2,3* : *2,3* : 0 : 0), DC_2 (+*x* : –*x* : +*y* : -*y*) = (0 : 0 : *3* : *1*), and MC_2 (+*x* : –*x* : +*y* : -*y*) = (*1,3* : *1,3* : 0 : 0).

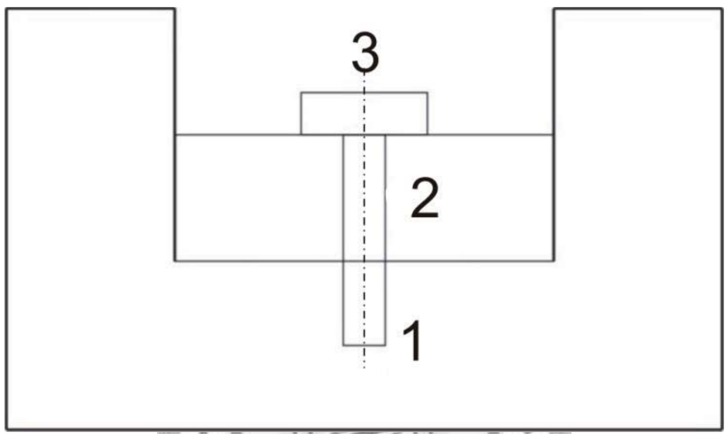


Figure 3. Example assembly 3.

Chapter 3 Selective Disassembly Sequence Planning

We define six rules for our recursive selective disassembly planning processes. In the rules below a parent part is a part which has already been selected for disassembly. The Sequence_Store is a storage space to store an incomplete or a complete disassembly sequence.

Rule 1: IF (there is any fastener attached to a component) THEN (the fastener needs to be disassembled first)

Rule 2: IF (there are corresponding tuples in both DF_i and MF_i which are 0) THEN (disassemble fastener *i* along the direction associated with the tuples and store fastener *i* in Sequence_Store)

Rule 3: IF (there are tuples in DF_i which are 0 but the corresponding tuples in MF_i which are not 0) THEN (remove the parts in the corresponding tuples in MF_i but which are not parent parts of fastener *i*, before *i* can be disassembled)

Rule 4: IF (there are corresponding tuples in both DC_n and MC_n which are 0) THEN (disassemble component *n* along the direction associated with the tuples and store component *n* in Sequence_Store)

Rule 5: IF (there are no corresponding tuples in both DC_n and MC_n which are 0) THEN (remove the parts which are in DC_n and the first-level components which are in MC_n, but not the parent components of *n*, before component *n* can be disassembled)

Rule 6: IF (the motion of component *n* is constraint by the same parts in both directions of a principal axis in MC_n) THEN (the both directions of the principal axis cannot be chosen as disassembly directions)

As shown in Figures 4-6, our recursive selective disassembly planning method includes three basic functions: Func_Component(), Func_Remove_Component(), and Func_Remove_Fastener(). The first function, Func_Component(), checks if a component *n* is fixed by any fasteners or components. If component *n* is fixed by any fasteners, according to Rule 1, all the fasteners need to be disassembled first, and function Func_Remove_Fastener() is called. If component *n* is not fixed by any fasteners but fixed by some other components, according to Rule 5, all the components need to be disassembled first, and function Func_Remove_Component() is called. If component *n* is not fixed by any fasteners or any components, according to Rule 4, it can be disassembled and stored in Sequence_Store.

Func_Remove_Fastener() will check if a fastener *i* is fixed by any fasteners or components. If fastener *i* is fixed by any other fasteners, according to Rule 3, all the fasteners need to be disassembled first, and function Func_Remove_Fastener() itself is called. If fastener *i* is not fixed by any fasteners but fixed by some other components, according to Rule 3, all the components need to be disassembled first, and function

Func_Remove_Component() is called. If fastener *i* is not fixed by any fasteners or any components, according to Rule 2, it can be disassembled and stored in Sequence_Store. If a fastener or a component is disassembled, its index will be deleted from the *DC, MC,* and *MF* matrices.

For example, in Figure 2, if component *2* is a target component, since DC_2 (+*x* : –*x* : +*y* : -*y*) = (*3,5,6* : *8* : *1,7* : 0), fasteners *7* and *8* need to be disassembled first. Since DF_7 (+*y*) = MF_7 (+*y*) = 0, fastener *7* can be disassembled in the +y direction. After fastener *7* is removed, DC_2 (+*x* : –*x* : +*y* : -*y*) = (*3,5,6* : *8* : *1* : 0). Since DF_8 (-x) = MF_8 (-x) = 0, fastener *8* can be disassembled in the -x direction. After fastener *8* is removed, DC_2 (+*x* : –*x* : +*y* : -*y*) = (*3,5,6* : *0* : *1* : 0), and all the fasteners attached to component *2* have been removed. After fasteners *7* and *8* are removed, DC_2 (-x) = MC_2 (-x) = 0, Thus, component *2* can be disassembled in the -x direction. Component *2* is then deleted from *DC, MC,* and *MF*.

The process is rule-based and recursive. Thus, not all possible solutions are generated and checked. However, the method generates reasonable and near-optimal heuristic solutions both efficiently and effectively. The given rules reduce searching time by eliminating unrealistic and uncommon solutions.

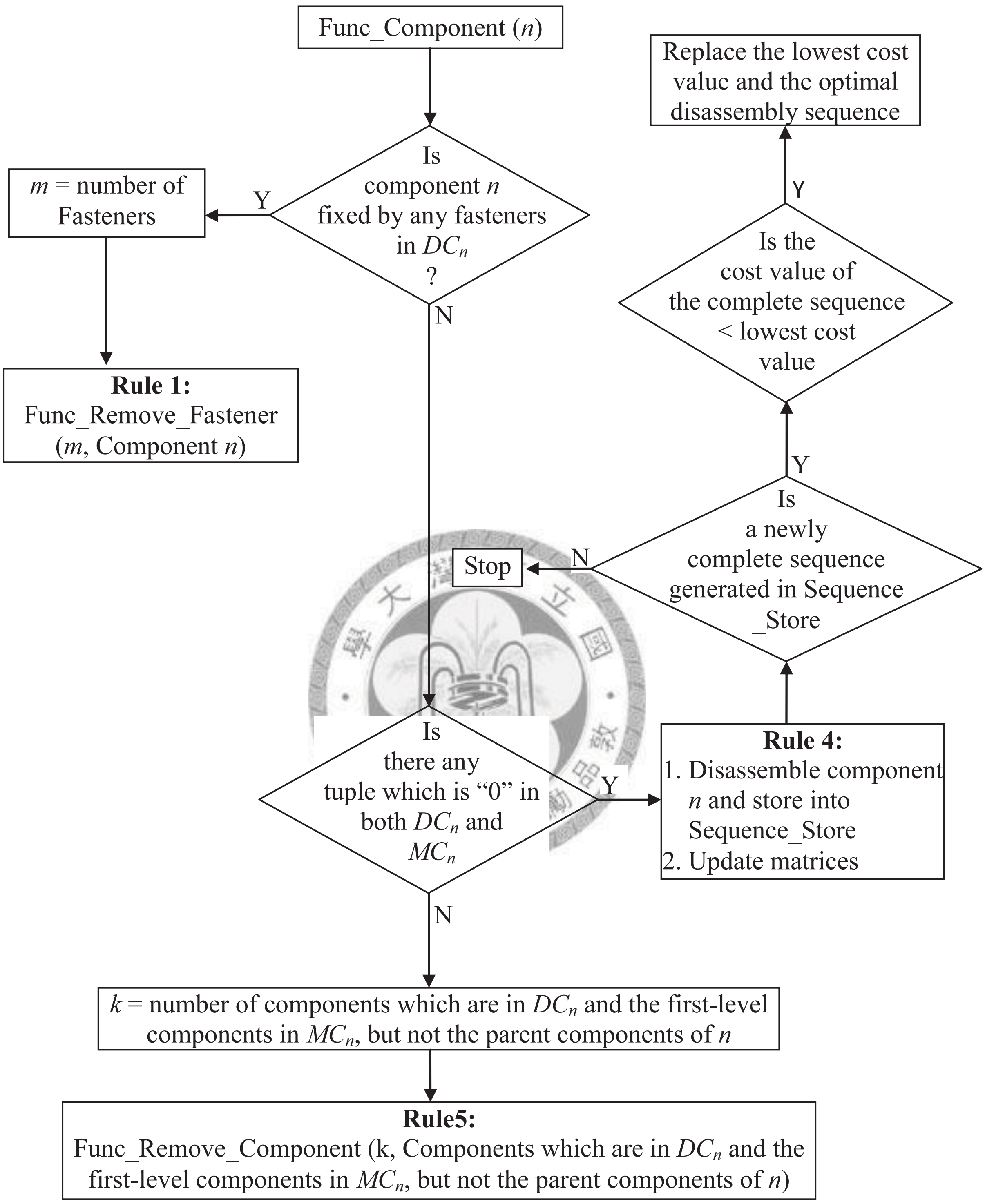


Figure 4. Main function for testing the disassemblability of a component

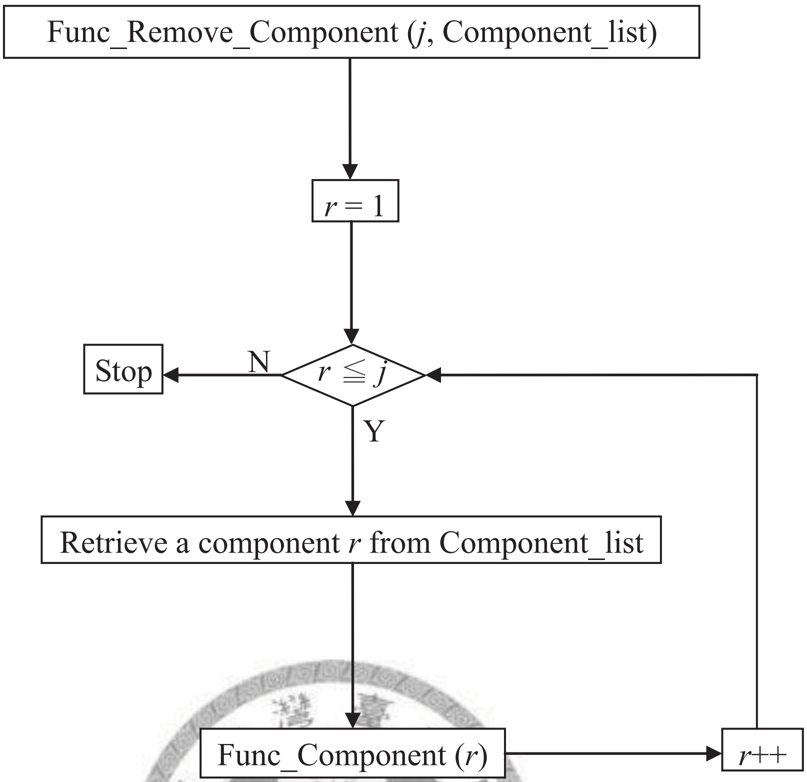


Figure 5. Function for removing a component.

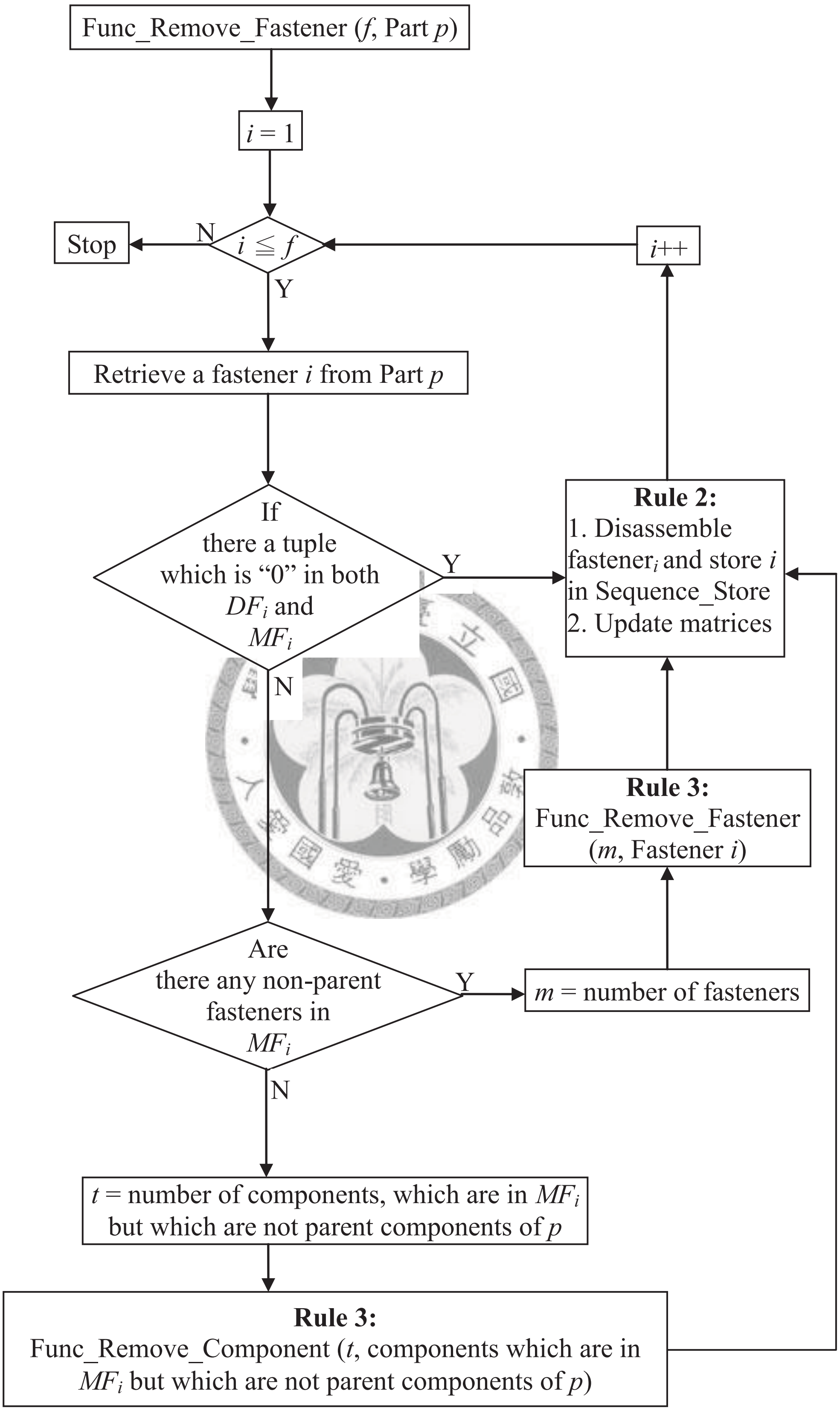


Figure 6. Function for removing a fastener.

Chapter 4 Cost Function

Our cost function for evaluating disassembly sequences includes disassembly time, reorientations, and number of components and fasteners removed.

$$Cost\ value = w_1 \times time + w_2 \times reorientations + w_3 \times parts \qquad \text{Eq. (1)}$$

In Equation 1, we can choose weight values w_1, w_2, and w_3 to establish the weighted importance of each of the cost parameters in determining the outcome of the search process. The final heuristic optimal selective disassembly sequence will have the lowest cost value.

4.1 Time

Some prior studies use time as an evaluation parameter. However, the time values cannot be easily verified. Here, we use experimental time values from Boothroyd et al. [3]. It considers the effect of part symmetry, grasping or manipulating with hands or with the aid of grasping tools, and part inserted with no secured immediately or secured immediately by screw fastening with power tool [27].

The parts that can be grasped and manipulated with hands or the aid of grasping tools include the parts that can be handled by one hand without the aid of grasping tools or can be handled by one hand but require two hands because they severely nest, tangle, flexible, or require forming etc [27]. The part that inserted with no secured immediately or secured immediately by screw fastening with power tool include the part that can be inserted but not secured immediately or secured by snap fit and part inserted and secured immediately by screw fastening with power tool [27]. Boothroyd et al. show the handling time in seconds in Table 2 and Table 3.

The two Tables are designed for assembly, not for disassembly. There are some differences between assembly and disassembly. In assembly, handling a component from a box and manipulating it into a correct direction and position requires more time than disassembling the component. In disassembly, it does not need to spend time to check if the component is in right direction or position because it is fixed on a product. We just need to remove it from the product. However, since there are no documented disassembly times, here, we use the recorded assembly times as reference to estimate the disassembly times. The data is more credible than the time defined arbitrary by designers.

Tables 2-4 are used to calculate disassembly time. For part handling, we only consider if any grasping tools are used, parts need two hands to handle, and the thickness of part. For the inverse of part insertion (disassembly), we only consider if the parts need holding down or need a power tool. Table 4 gives an example of part symmetry for calculating part handling time.

Table 2. Selected manual handling time standards, seconds (parts are within easy reach, are no smaller than 6mm, do not stick together, and are not fragile or sharp) [3].

(a) The parts can be grasped and manipulated with one hand without any grasping tools

		no handling difficulties			part nests or tangles		
		thickness > 2mm		< 2mm	thickness > 2mm		< 2mm
sym (deg) = (alpha+ beta)		size > 15mm	6mm < size < 15mm	size > 6mm	size > 15mm	6mm < size < 15mm	size > 6mm
		0	1	2	3	4	5
sym < 360	0	1.13	1.43	1.69	1.84	2.17	2.45
360 <= sym < 540	1	1.5	1.8	2.06	2.25	2.57	3.0
540 <= sym < 720	2	1.8	2.1	2.36	2.57	2.9	3.18
sym = 720	3	1.95	2.25	2.51	2.73	3.06	3.34

(b) The parts are severely nest or tangle, are flexible or require forming etc. require two hands to handle

	alpha <= 180		alpha = 360
	size > 15mm	6mm < size <15mm	size > 6mm
	0	1	2
4	4.1	4.5	5.6

Table 2. (*continued*)

(c) Part thickness, handling difficulties and part nests

no handling difficulties			part nests or tangles		
thickness > 2mm		< 2mm	thickness > 2mm		< 2mm
size > 15mm	6mm < size < 15mm	size > 6mm	size > 15mm	6mm < size < 15mm	size > 6mm
0	1	2	3	4	5
1.13	1.43	1.69	1.84	2.17	2.45

Table 3. Selected manual insertion time standards, seconds (parts are small and there is no resistance to insertion) [3].

(a) Part inserted but not secured immediately or secured by snap fit

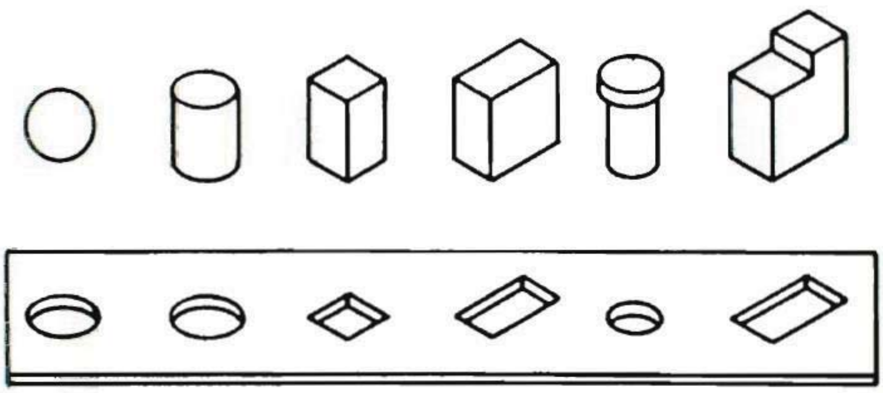
		secured by separate operation or part				secured on insertion by snap fit	
		no holding down required		holding down required			
		easy to align	not easy to align	easy to align	not easy to align	easy to align	not easy to align
		0	1	2	3	4	5
no access or vision difficulties	0	1.5	3.0	2.6	5.2	1.8	3.3
obstructed access or restricted vision	1	3.7	5.2	4.8	7.4	4.0	5.5
obstructed access and restricted vision	2	5.9	7.4	7.0	9.6	7.7	7.7

(b) Part inserted and secured immediately by screw fastening with power tool (rimes are for 5 revs or less and do not include a tool acquisition time of 2.9s)

		easy to align	not easy to align
		0	1
no access or vision difficulties	3	3.6	5.3
restricted vision only	4	6.3	8.0
obstructed access only	5	9.0	10.7

	screw tighten with power tool	manipulation, reorientation or adjustment	addition of non solids
	0	1	2
6	5.2	4.5	7

Table 4. Various parts illustrate the alpha and beta rotational symmetries [3].



α	0	180	180	90	360	360
β	0	0	90	180	0	360

In Table 2, alpha is the rotational symmetry of a part about an axis perpendicular to its axis of insertion. For parts with one axis of insertion, end-to-end orientation is necessary when alpha equals 360 degrees, otherwise alpha equals 180 degrees. Beta is the rotational symmetry of a part about its axis of insertion. The magnitude of rotational symmetry is the smallest angle through which the part can be rotated and repeat its orientation. For a cylinder inserted into a circular hole, beta equals zero.

In Table 3, holding down required means that the part will require gripping, realignment, or holding down before it is finally secured. Easy to align and position means that insertion is facilitated by well designed chamfers or similar features.

Obstructed access means that the space available for the assembly operation causes a significant increase in the assembly time. Restricted vision means that the operator has to rely mainly on tactile sensing during the assembly process. Based on the two Tables, we can calculate the time parameter.

Table 4 illustrates the alpha and beta rotational symmetries for various parts. Thickness is the length of the shortest side of the smallest rectangular prism that encloses the part. However, if the part is cylindrical, or has a regular polygonal cross-section with five or more sides, and the diameter is less than the length, then thickness is defined as the radius of the smallest cylinder which can enclose the part. Size is the length of the longest side of the smallest rectangular prism that can enclose the part.

4.2 Reorientation

During disassembly, if the number of disassembly direction reorientations is reduced, disassembly time is also reduced [23, 24, 25, 27]. Since we only consider principal disassembly directions, each reorientation requires either a 90-degree or a 180-degree direction change. For example, if the disassembly direction changes from $+x$ to $+y$, $-y$, $+z$, or $-z$, the reorientation requires a 90-degree direction change, for which we set the reorientations cost function parameter to *1*. However, if the disassembly direction changes from $+x$ to $-x$, the reorientation requires a 180-degree direction change, for which we set the cost function parameter to *2*. When no reorientations are needed, we set the reorientations cost function parameter to *0*.

4.3 Parts

Many research studies consider the problem of reducing the number of components to be removed. It is a basic criterion in evaluating the quality of a disassembly sequence [2, 11, 15, 21, 22, 23, 24, 26]. If fewer parts are removed in disassembling a target component or multiple-target components, lower cost and less time are required in the disassembly process. Therefore, the optimal selective disassembly sequences will be the ones with the least cost value.

Chapter 5 Examples and Discussions

5.1 Example 1

We used three examples to test our rule-based recursive selective disassembly method. Figure 7 shows the first example. Figure 8 shows the corresponding *DC* and *MC* matrices. For target component *3*, there are no fasteners. Since DC_3= (0 : 0 : *4* : *2*) and MC_3= (*1* : *1* : 0 : 0), there is no tuple which is *0* in both DC_3 and MC_3, Therefore, Rule 5 is executed. Since component *1* is not a first-level component of *3*, only components *4* and *2* are passed into Func_Remove_Component().

If component *4* is retrieved, there is no tuple which is *0* in both DC_4 and MC_4. Therefore, Rule 5 is executed. Since components *3* is a parent component of component *4*, only component *5* is passed into Func_Remove_Component(). If component *5* is retrieved next, DC_5= (0 : 0 : 0 : *4*) and MC_5= (*1* : *1* : 0 : 0). Therefore, component *5* can be removed in the +y direction. After component *5* is removed, DC_4 is updated to (0 : 0 : 0 : *3*), and MC_4 is updated to (*1* : *1* : 0 : 0). Thus, component *4* can be removed in the +y direction. Finally, one disassembly sequence, *5-4-3*, can be generated. Similarly, a second disassembly sequence, *1-2-3*, can also be found. The two disassembly sequences are shown in Figure 8.

The assembly in Figure 7 has five components and, therefore, there are 5! = 120 possible disassembly sequence possibilities. However, our rule-based recursive selective disassembly planning can eliminate considering many unrealistic or uncommon solutions. The developed method can find optimal heuristic selective disassembly sequences quickly and effectively.

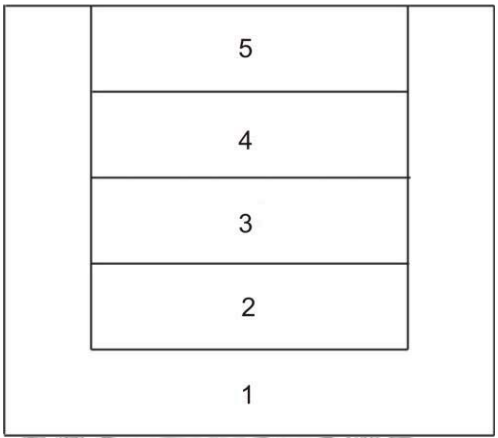


Figure 7. Example assembly.

$$DC = \begin{bmatrix} DC_1 \\ DC_2 \\ DC_3 \\ DC_4 \\ DC_5 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 2 & 0 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 4 & 2 \\ 0 & 0 & 5 & 3 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

$$MC = \begin{bmatrix} MC_1 \\ MC_2 \\ MC_3 \\ MC_4 \\ MC_5 \end{bmatrix} = \begin{bmatrix} 2,3,4,5 & 2,3,4,5 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

Figure 8. The *DC* and *MC* matrices for the example assembly in Figure 7.

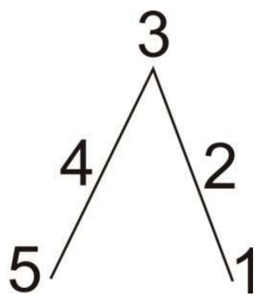


Figure 9. Disassembly sequences.

5.2 Example 2

Figure 10 shows an example of power brake, given by Mascle and Hong (2008). The material list is given in Table 5. We use website http://www.fastener-world.com.tw/giga to define which parts are fasteners and which parts are components. In this example, single-component selective disassembly for component *17* is shown.

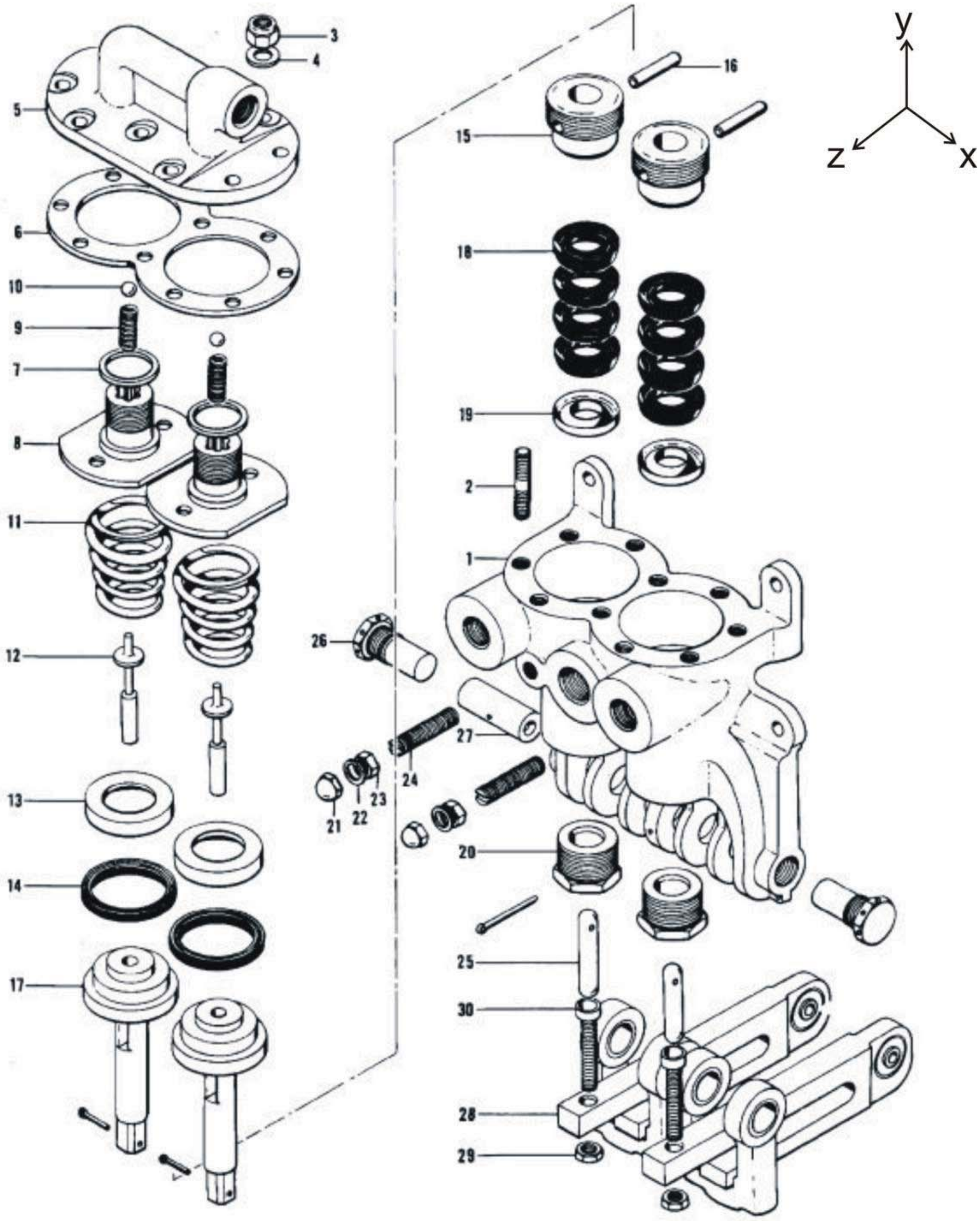


Figure 10. Exploded view of power brake [16] .

Table 5. Components and fasteners lists of power brake in figure 10 [16] .

Fig. and Index No.	Component (*C*) or Fastener (*f*)	Nomenclature	Units per Assy.
1	*C*	Housing	1
2	*f*	Stud 1/4”	10
3	*f*	Nut	10
4	*f*	Washer	10
5	*C*	Cover	1
6	*C*	Gasket	1
7	*f*	Washer	2
8	*C*	Seat	2
9	*f*	Spring	2
10	*f*	Ball $^{1}/_{4}$” Dia.	2
11	*f*	Spring	2
12	*f*	Pin	2
13	*f*	Spacer $1^{1}/_{2}$ Dia.	2
14	*C*	Packing - Neoprene	2
15	*f*	Nut 1 1/8	2

16	*f*	Pin 3/8” Dia.	2
17	*C*	Piston	2
18	*C*	Packing - neoprene	8
19	*f*	Spacer 1 1/8”	2
20	*f*	Nut 7/8” 14 NF	2
21	*f*	Capnut	2
22	*f*	Washer	2
23	*f*	Nut 10 – 32 NF	2
24	*f*	Screw 1 – 32 NF	2
25	*C*	Link	2
26	*f*	Shaft 5/8” Nickel Steel	2
27	*f*	Shaft 9/11” Dia. Nickel Steel	2
28	*C*	Lever - Assembly	1
29	*f*	Nut 5/16	2
30	*f*	Screw 3/8 Dia. 5/16	2

$$DC=\begin{bmatrix}1\\5\\6\\8\\14\\17\\18\\25\\28\end{bmatrix}=\begin{bmatrix}26,27 & 26,27 & 2,8 & 20,28 & 24 & 0\\ 0 & 0 & 2,4 & 10,6 & 0 & 1\\ 0 & 0 & 2,5 & 8 & 0 & 0\\ 0 & 0 & 2,6,7 & 1,11 & 0 & 0\\ 0 & 0 & 13 & 17 & 0 & 0\\ 0 & 16 & 11,12,13,14 & 15 & 0 & 0\\ 0 & 0 & 15 & 19 & 0 & 0\\ 0 & 0 & 20 & 0 & 0 & 0\\ 0 & 0 & 1,30 & 29 & 0 & 0\end{bmatrix}$$

(a)

$$MC=\begin{bmatrix}1\\5\\6\\8\\14\\17\\18\\25\\28\end{bmatrix}=\begin{bmatrix}-1 & -1 & -1 & -1 & -1 & -1\\ -1 & -1 & 0 & 0 & -1 & -1\\ -1 & -1 & 0 & 0 & -1 & -1\\ 5,6,7 & 5,6,7 & 0 & 0 & 5,6,7 & 5,6,7\\ 1,17 & 1,17 & 0 & 0 & 1,17 & 1,17\\ 1,14,15 & 1,14,15 & 16 & 16 & 1,14,15 & 1,14,15\\ 1 & 1 & 0 & 0 & 1 & 1\\ 0 & 0 & 0 & 30 & 0 & 1\\ 1 & 1 & 27 & 27 & 27 & 27\end{bmatrix}$$

(b)

Figure 11. (*continued*)

$$DF = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 7 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 15 \\ 16 \\ 19 \\ 20 \\ 21 \\ 22 \\ 23 \\ 24 \\ 26 \\ 27 \\ 29 \\ 30 \end{bmatrix} = \begin{bmatrix} -1 & -1 & 0 & -1 & -1 & -1 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ -1 & -1 & 0 & 0 & -1 & -1 \\ -1 & -1 & 0 & 0 & -1 & -1 \\ -1 & -1 & 0 & 0 & -1 & -1 \\ -1 & -1 & 0 & 0 & -1 & -1 \\ -1 & -1 & 0 & 0 & -1 & -1 \\ -1 & -1 & 0 & 0 & -1 & -1 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 & -1 \\ -1 & -1 & 0 & 0 & -1 & -1 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & -1 & 0 & -1 \\ 0 & 0 & -1 & -1 & -1 & -1 \\ 0 & 0 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 0 & -1 & -1 \\ -1 & -1 & 0 & -1 & -1 & -1 \end{bmatrix}$$

(c)

Figure 11. (*continued*)

$$
MF=\begin{bmatrix}2\\3\\4\\7\\9\\10\\11\\12\\13\\15\\16\\19\\20\\21\\22\\23\\24\\26\\27\\29\\30\end{bmatrix}=\begin{bmatrix}
-1 & -1 & 4 & -1 & -1 & -1\\
-1 & -1 & 0 & -1 & -1 & -1\\
-1 & -1 & 3 & -1 & -1 & -1\\
-1 & -1 & 9 & 8 & -1 & -1\\
-1 & -1 & 10 & 7 & -1 & -1\\
-1 & -1 & 5 & 9 & -1 & -1\\
-1 & -1 & 8 & 12 & -1 & -1\\
-1 & -1 & 11 & 13 & -1 & -1\\
-1 & -1 & 12 & 14 & -1 & -1\\
-1 & -1 & 16,17 & -1 & -1 & -1\\
-1 & 0 & -1 & -1 & -1 & -1\\
-1 & -1 & 18 & 20 & -1 & -1\\
-1 & -1 & -1 & 25 & -1 & -1\\
-1 & -1 & -1 & -1 & 0 & -1\\
-1 & -1 & -1 & -1 & 21 & -1\\
-1 & -1 & -1 & -1 & 22 & -1\\
-1 & -1 & -1 & -1 & 23 & -1\\
0 & 0 & -1 & -1 & -1 & -1\\
26 & 26 & -1 & -1 & -1 & -1\\
-1 & -1 & -1 & 0 & -1 & -1\\
-1 & -1 & 25 & -1 & -1 & -1
\end{bmatrix}
$$

(d)

Figure 11. Illustrate the *DC*, *MC*, *DF*, and *MF* of power brake.

Using the developed method, here component *17* is chosen as a target component. Component *17* is input to the Func_Component() function. From the *DC* of the target component *17*, we can see that it requires disassemble some fasteners before component *17* can be disassembled. Thus, function Func_Remove_Fastener() is called. If fastener *11* is firstly chosen, it can be removed along +y or –y directions. From MF_{11}, we know that , before disassembling *11, 8* needs to be disassembled along the +y direction or *12* needs to be disassembled along the -y direction. However, *12* cannot be selected, because *12* is a parent component of *11*. Therefore, the next part to be chosen is component *8*, and Func_Remove_Component() is called.

Inside of Func_Remove_Component(), Func_Component() is calledCheck the *DC* of *8*, there are three fasteners, *2*, *7*, and *11* require to be disassembled. Thus, Func_Remove_Fastener() is called. However, since *11* is a parent part of *8*, *11* will not to be considered here. If *2* is selected to remove, check *DF* and *MF* of *2.* We found that *2* cannot be removed unless *4* is removed first. Thus, Func_Remove_Fastener() is called again. However, before *4* can be removed, *3* needs to be removed first. Thus, Func_Remove_Fastener() is called again. Finally, fastener *3* can be removed from the +y direction. After that, fasteners *4*, and *2* can be removed and the incomplete disassembly sequence (*3*-*4*-*2*) is stored in the Sequence_Stack, and we need to go back to select fastener *7* to disassemble.

Check the *DF* and *MF* of *7*, there are parts *8* and *9* which have to be disassembled before *7*. However, since *8* is a parent component of *7*, we cannot choose *8* to disassemble. Therefore, the next part to be chosen is fastener *9*, and function Func_Remove_Fastener() is called. Check the *DF* and *MF* of *9*, we found that fastener *10* needs to be disassembled before *9*. Thus, Func_Remove_Fastener() is called.

However, *10* cannot be disassembled along +y direction before part *5* is disassembled. Thus, function Func_Remove_Component() is called, and inside of which, function Func_Component is called. Check the *DC* of *5*, we found that component *5* has no fasteners attached to it. The parameters in the +y direction in *DC* and *MC* of component *5* both are *0*. Thus, component *5* can be disassembled along the +y direction. Therefore, disassemble *5*, *10*, *9* and *7* and store the incomplete disassembly sequence (*5*-*10*-*9*-*7*) in the Sequence_Stack and go back to check the *DC* and *MC* of *8*.

Now, we need to check the *DC* and *MC* of component *8*. We found that component *6* needs to be disassembled first so that function Func_Remove_Component() is called, and inside of which, function Func_Component() is called. Since at this moment, all parts blocking the motion of component *6* have been removed, component *6* can be removed along the +y direction. After component *6* is disassembled, component *8* and fastener *11* can be disassembled afterward, and the incomplete disassembly sequence (*6*-*8*-*11*) is stored in the Sequence_Stack. Now, fasteners *12* and *13* need to be disassembled in the +y direction. If we select *12* as the next part to be disassembled. Check the *DF* and *MF* of *12*, since *11* has been removed, *12* can be removed along the +y direction. The incomplete disassembly sequence (*12*) is stored in the Sequence_Stack. Follow the same process, fastener *13* can also be removed directly along the +y direction. The incomplete disassembly sequence (*13*) is stored in the

Sequence_Stack. Finally, check the *DC* and *MC* of *17*, we found that fasteners *16* and *15* need to be removed. Check *DF* and *MF* of *16*, we found that *16* can be removed in the -x direction. The incomplete sequence (*16*) is stored in the Sequence_Stack. After that, check *DF* and *MF* of *15*, since *16* has been removed and *17* is a parent of *15*, *15* is not considered here.

After that, all the fasteners of *17* have been checked, and +x, -x, +z, -z are all 0 in DC_{17} so that they are possible disassembly directions for component *17*. However, according to Rule 6, component *17* is blocked by *1*, *14*, *15* in both directions of the x principal axis and is blocked by *1*, *14*, *15* in both directions of the principal z axis, so both x and z axis are not considered as the disassembly directions. Therefore, the possible disassembly directions will be +y and –y. If we choose the –y direction, fastener *15* needs to be removed, and component *17* needs to be removed before *15*. However, since *17* is a parent of *15*, we will not consider the –y direction. Now, we chose +y direction as the next disassembly direction, so component *14* needs to be removed. Function Func_Remove_Component() is called, and inside of which, function Func_Component() is called. Check the *DC* and *MC* of *14*, we found that *14* can be removed in the +y direction. The incomplete sequence (*14-17*) is stored in the Sequence_Stack. After that, we found a selective disassembly sequence planning for disassembling component *17*: *3*, *4*, *2*, *5*, *10*, *9*, *7*, *6*, *8*, *11*, *12*, *13*, *16, 14* and *17*. The complete disassembly sequence can be shown in Figure 12.

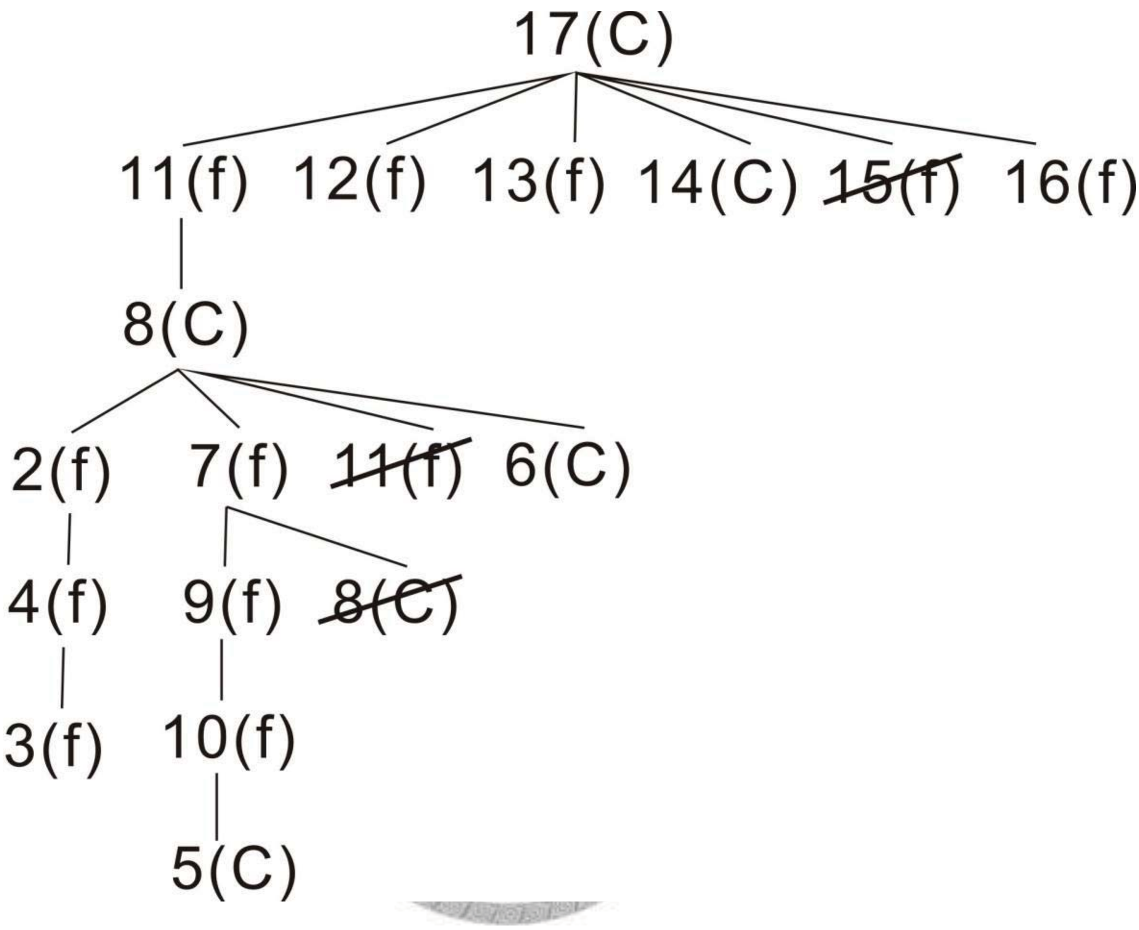


Figure 12. Disassembly sequence for example 2.

There are fifteen parts to be disassembled and one reorientation, *13* to *16* and *16* to *14*. In equation 1, *Cost Value* = w_1**Time* + w_2**Reorientation* + w_3**Parts*. The cost value = 0 + 2+ 15 = 17. The time parameter used here is the same as Mascle and Hong (2008). To compare with the results by Mascle and Hong, we set w_1 to be "0" to ignore the time parameter, and w_2 and w_3 are "1". The cost value of the results by Mascle and Hong is also 0 + 2 + 15 = 17. However, the results by Mascle and Hong (2008) needs to remove fastener *2* first. From Figure 10, we can see that fastener *2* cannot be removed unless fasteners *3* and *4* are removed first. Therefore, our method can provide a better solution.

Table 6. The selective disassembly sequence planning of power brake by Mascle and Hong (2008) [16] .

Wave No. 1 :	2
Wave No. 2 :	3
Wave No. 3 :	4
Wave No. 4 :	5
Wave No. 5 :	6 7 10
Wave No. 6 :	9
Wave No. 7 :	8
Wave No. 8 :	11
Wave No. 9 :	12 16
Wave No. 10 :	13
Wave No. 11 :	14
Wave No. 12 :	17

5.3 Example 3

A gear reducer assembly from Srinivasan and Gadh (2000), as shown in Figure 13, is used to test the developed selective disassembly method for single-target-component and multiple-target-component disassembly. To simplify the problem, we only consider the disassembly direction in the x direction. Thus, in this case, the following components can be ignored in *DC*, *DF*, *MC*, and *MF*: *9*, *10*, *11*, *12*, *13*, *14*, *15*, *16*, *17*, *18*, *27*, *28*, *33*, *34*, *35*, and *36*, because they do not directly interfere other parts removing along the x direction. Figure 14 shows the corresponding *DC*, *DF*, *MC*, and *MF*.

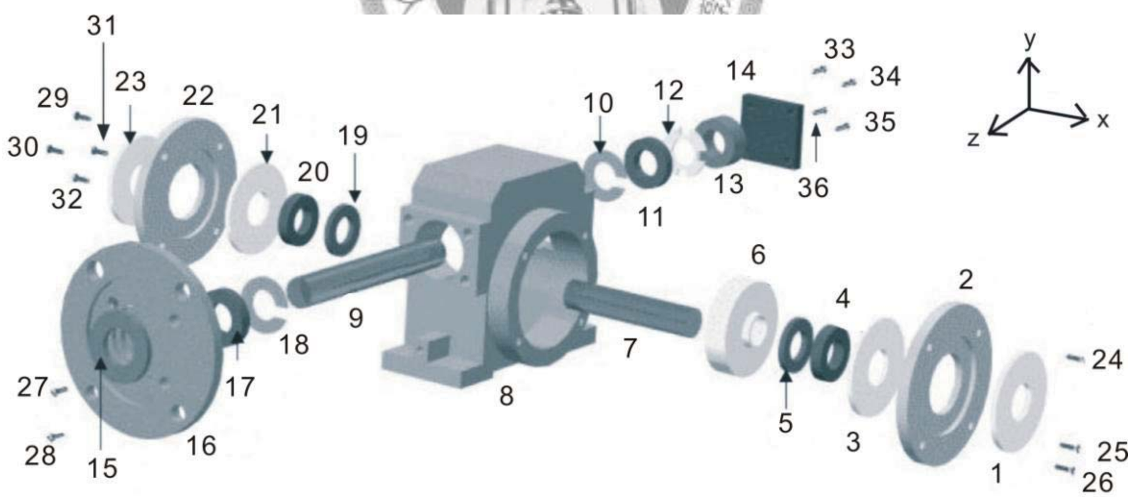


Figure 13. Gear reducer assembly [19].

$$DC = \begin{bmatrix} DC_1 \\ DC_2 \\ DC_3 \\ DC_4 \\ DC_5 \\ DC_6 \\ \vdots \\ DC_{19} \\ DC_{20} \\ DC_{21} \\ DC_{22} \\ DC_{23} \end{bmatrix} = \begin{bmatrix} 0 & 2 & 0 & 0 & 0 & 0 \\ 1,24,25,26 & 3,8 & 0 & 0 & 0 & 0 \\ 2 & 4 & 0 & 0 & 0 & 0 \\ 3 & 5 & 0 & 0 & 0 & 0 \\ 4 & 6 & 0 & 0 & 0 & 0 \\ 5 & 19 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 6 & 20 & 0 & 0 & 0 & 0 \\ 19 & 21 & 0 & 0 & 0 & 0 \\ 20 & 22 & 0 & 0 & 0 & 0 \\ 21 & 23,29,30,31,32 & 0 & 0 & 0 & 0 \\ 22 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(a)

$$MC = \begin{bmatrix} MC_1 \\ MC_2 \\ MC_3 \\ MC_4 \\ MC_5 \\ MC_6 \\ \vdots \\ MC_{19} \\ MC_{20} \\ MC_{21} \\ MC_{22} \\ MC_{23} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 2,7,8 & 2,7,8 & 2,7,8 & 2,7,8 \\ 0 & 0 & 1,3,7,24,25,26 & 1,3,7,24,25,26 & 1,3,7,24,25,26 & 1,3,7,24,25,26 \\ 0 & 0 & 2,7,8 & 2,7,8 & 2,7,8 & 2,7,8 \\ 0 & 0 & 7,8 & 7,8 & 7,8 & 7,8 \\ 0 & 0 & 6,7,8 & 6,7,8 & 6,7,8 & 6,7,8 \\ 0 & 0 & 5,7,8,19 & 5,7,8,19 & 5,7,8,19 & 5,7,8,19 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 6,7,8 & 6,7,8 & 6,7,8 & 6,7,8 \\ 0 & 0 & 7,8 & 7,8 & 7,8 & 7,8 \\ 0 & 0 & 7,8,22 & 7,8,22 & 7,8,22 & 7,8,22 \\ 0 & 0 & 7,21,23,29,30,31,32 & 7,21,23,29,30,31,32 & 7,21,23,29,30,31,32 & 7,21,23,29,30,31,32 \\ 0 & 0 & 7,8,22 & 7,8,22 & 7,8,22 & 7,8,22 \end{bmatrix}$$

(b)

Figure 14. (*continued*)

$$DF = \begin{bmatrix} DF_{24} \\ DF_{25} \\ DF_{26} \\ \vdots \\ DF_{29} \\ DF_{30} \\ DF_{31} \\ DF_{32} \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 & -1 & -1 & -1 & -1 & -1 \\ 0 & -1 & -1 & -1 & -1 & -1 \\ 0 & -1 & -1 & -1 & -1 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -1 & 0 & -1 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

(c)

$$MF = \begin{bmatrix} MF_{24} \\ MF_{25} \\ MF_{26} \\ \vdots \\ MF_{29} \\ MF_{30} \\ MF_{31} \\ MF_{32} \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 & -1 & -1 & -1 & -1 & -1 \\ 0 & -1 & -1 & -1 & -1 & -1 \\ 0 & -1 & -1 & -1 & -1 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -1 & 0 & -1 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 & -1 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

(d)

Figure 14. *DC, MC, DF*, and *MF* for the gear reducer assembly.

5.3.1 Single Target Component

Following the process shown in Figure 4 to disassemble target component *5*. From *DC* we know that *5* is not fixed by any fasteners. Thus, choose any direction parameter which is *0* in the *DC*. Suppose +y direction is chosen, then the corresponding tuples in *MC* will be the constraints for *5* moving in the y direction, which are *6*, *7*, and *8*. Same constraints exist in the z direction. Thus, component *5* has to move along the x axis direction. However, in *DC*, we can see that component *4* blocks component *5* in moving in the +x direction and component *6* blocks component *5* in moving in the –x direction. Thus, component *4* and *6* are the subparts of *5*. Now, component *4* is first taken as a subpart to be a new target part.

Now check *DC* of *4* to see if there are any fasteners. Follow the same procedure, find that component *4* must be removed in the x direction. Component *4* is constrained by *5* in the –x direction and *3* in the +x direction. Since *5* is a "parent" component of *4*, the next component to be selected as the subpart of *4* is *3*. *3* is thus a new target part to be considered. Repeat the same procedure and find out the subpart of *3* is *2*. Up to now, since none of the fasteners and components are disassembled, none of the tuple values are changed.

Finally, there are three fasteners, *24*, *25*, and *26* in the *DC* of component *2*, and DC_2 = (*1,24,25,26* : *3,8* : 0 : 0 : 0 : 0). Thus, fasteners, *24*, *25*, and *26* will be the next candidate to be removed. According to *DF*, *24*, *25*, and *26* can only be disassembled along the +x direction. In addition, since the corresponding tuple (+x) in *MF* is *0*, fasteners *24*, *25*, and *26* can be disassembled along the +x direction without any collision. After the three fasteners, *24*, *25*, and *26* are removed, DC_2 is be updated to (*1* : *3,8* : 0: 0 : 0 : 0), and MC_2 is updated to (0 : 0 : *1,3,7* : *1,3,7* : *1,3,7* : *1,3,7*).

After all the fasteners are removed from *2*, follow the same procedure and find that

2 must be removed along the x axis. Since *3* is a parent component of *2*, the subpart of *2* will be *1*. *1* thus become the new target component. Now check *DC* of *1* to see if there are any fasteners. Follow the same procedure, find that both +x tuple in DC_1 and MC_1 are *0*. Thus, *1* can be disassembled along the +x direction without any collision. After *1* is removed, all the tuples which involve *1* will subtract it from them. Thus, DC_2 will be updated to be (0 : *3,8* : 0 : 0 : 0 : 0), and MC_2 will be updated to be (0 : 0 : *3,7* : *3,7* : *3,7* : *3,7*). Since both the +x tuple in DC_2 and MC_2 are *0*, *2* can be disassembled from the +x direction.

By continuing to follow the process diagram, one disassembly sequence *24*, *25*, *26*, *1*, *2*, *3*, *4*, *5*, can be found. The disassembly sequence includes 3 fasteners and 5 components. Since Srinivasan and Gadh (2000) did not consider time and reorientation in their study, in order to compare with their results, in example 2, we set our cost function for time and reorientation parameters to *0*. Use the Equation 1 (Cost Value) to calculate the cost of the sequence: $Cost\ Value = w_1 * Time + w_2 * Reorientation + w_3 * Parts$; $w_1 = w_2 = 0$, $w_3 = 1$. Therefore, the *Cost Value* = 0 + 0 + 8 = 8.

If another direction is chosen, i.e., component *6* is chosen instead of *4*, to disassemble component *5*, The disassembly sequence will be: *29*, *30*, *31*, *32*, *23*, *22*, *21*, *20*, *19*, *6*, and *5*. There are 4 fasteners and 7 components. The $Cost\ Value = w_1 * Time + w_2 * Reorientation + w_3 * Parts$; $w_1 = w_2 = 0$, $w_3 = 1$. Therefore, the *Cost Value* = 0 + 0 +

11 = 11.

Compare the two disassembly sequences, the first sequence is better than the second one because the first one only removes 8 objects and the second one removes 11 objects. Thus, based on the evaluation results, the best disassembly sequence to disassemble *5* is: *24*, *25*, *26*, *1*, *2*, *3*, *4* and *5*. For single-component disassembly, in their example, Srinivasan and Gadh (2000) chose component *3* as the target component, and they did not consider disassembly of fasteners, in which case, *1*, *2*, and *3* is the obvious best disassembly sequence solution.

5.3.2 Multiple- Target Components

In this multiple-target component disassembly, two situations are considered. One situation is to disassemble component *5* first and *19* later, the other situation is to disassemble component *19* first and *5* later.

5.3.2.1 The First Target Component is *5*

For the single target component case, *5* can be disassembled in either the +*x* or -*x* direction. Likewise, *19* can be disassembled in either the +*x* or –*x* direction, too. However, since it is not valid to remove *5* in the –*x* direction and *19* in the +*x* direction, or to remove both *5* and *19* the -*x* direction, there are only 2 valid sequences for the multiple-target component case. Following the same disassembly process to find each sequence and calculating the corresponding cost values, if both *5* and *19* are both disassembled in the +*x* direction, the best selective disassembly sequence is: *24*, *25*, *26*, *1*, *2*, *3*, *4*, *5*, *6*, *19*. There are three fasteners and seven components. The *Cost Value* = w_1**Time* + w_2**Reorientation* + w_3**Parts*; w_1 =0, w_2 = w_3 =1. The cost value = 0 + 0 + 10 = 10. If *5* is disassembled in the +*x* direction and *19* is disassembled in the -*x* direction, the best selective disassembly sequence is *24*, *25*, *26*, *1*, *2*, *3*, *4*, *5*, *29*, *30*, *31*, *32*, *23*, *22*, *21*, *20*, *19*. There are seven fasteners, ten components and one reorientation (from *5* to *29*). The *Cost Value* = 0 + 2 + 17 = 19. Based upon the two cost values, the best disassembly sequence is *24*, *25*, *26*, *1*, *2*, *3*, *4*, *5*, *6*, *19*. The cost value is 10 and both *5* and *19* must be disassembled in the +*x* direction.

5.3.2.2 The First Target is *19*

Similarly, if component *19* is the first target component, there are two valid disassembly sequences. If *19* and *5* are both disassembled in the -*x* direction, the selective disassembly sequence for the multiple-target component disassembly is *29*, *30*, *31*, *32*, *23*, *22*, *21*, *20*, *19*, *6*, *5*. The cost value is 11. If *19* is disassembled in the -*x* direction and *5* is disassembled in the +*x* direction, the sequence is *29*, *30*, *31*, *32*, *23*, *22*, *21*, *20*, *19*, *24*, *25*, *26*, *1*, *2*, *3*, *4*, *5*. The cost value is 19. Thus, if component *19* is the first component which is disassembled, the best selective disassembly sequence plan is *29*, *30*, *31*, *32*, *23*, *22*, *21*, *20*, *19*, *6*, *5*, for which both *19* and *5* are disassembled along the -*x* direction.

Srinivasan and Gadh(2000) only evaluated selective disassembly sequences by number of removed components. They did not consider disassembly time, number of fasteners removed, and reorientations. For multiple-component selective disassembly, they determined that *1*, *2*, *3*, *4*, *5*, *6*, *19* and *23*, *22*, *21*, *20*, *19*, *6*, *5* are the two possible and equivalent disassembly sequences for removing *5* and *19*. Without considering number of fastener removes and reorientations, the results given by the developed method in this paper is the same as Srinivasan and Gadh both their results. However, if we consider number of removed fasteners, number of removed components, and reorientations, we find that there is only one best solution, *24*, *25*, *26*, *1*, *2*, *3*, *4*, *5*, *6*, *19*, with a cost value of 10.

Chapter 6 Graphical User Interface

Figure 15 shows the user interface of the program. The program allows the users to upload figures of the products. The button, Start, executes the algorithm. In the middle of the program, users can input *DC*, *MC*, *DF*, and *MF*. The cost value and the resulting disassembly sequence are shown at the right side of the program.

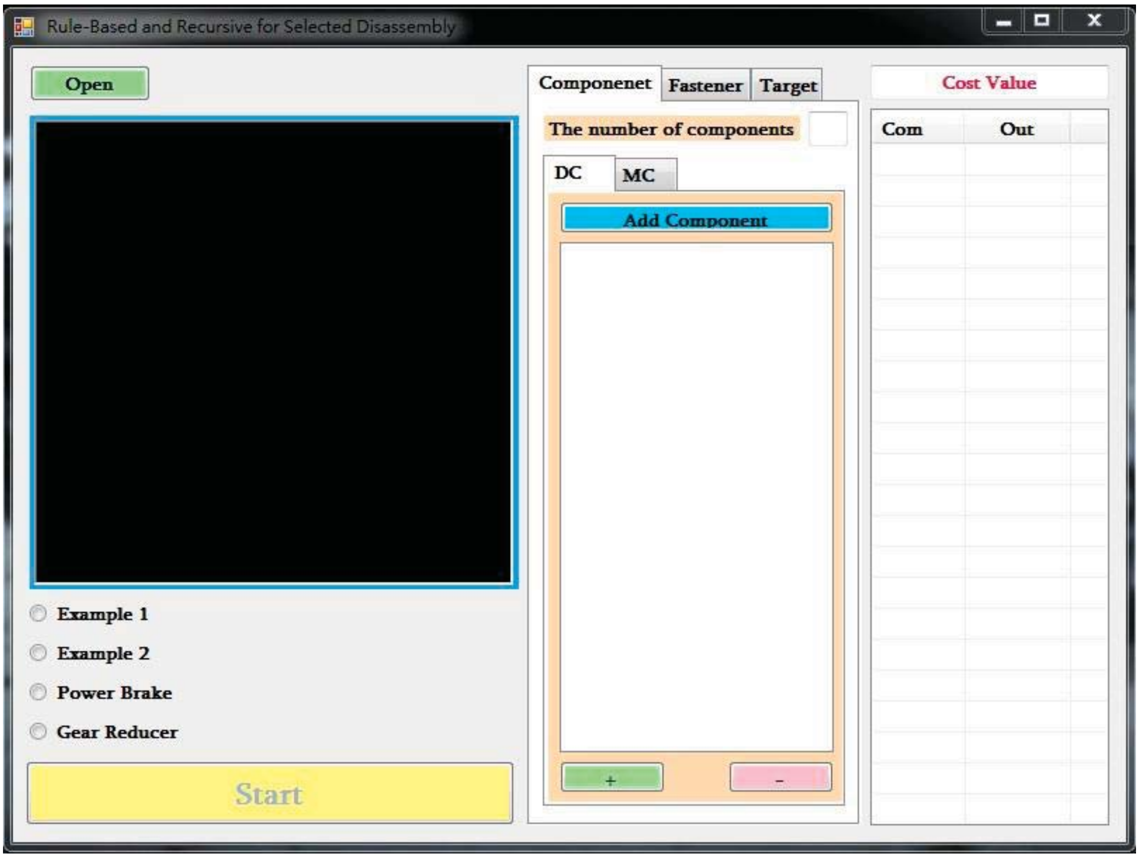


Figure 15. The user interface of the program.

6.1 Example 1

The first step is to open the picture of example 1 in the program, as shown in Figure 16. The second step is to set up the parameters of the geometrical and topological information in *DC*, *MC*, *DF*, and *MF,* as shown in Figure 17 (a)-(d). Since there is no fastener in this example, *DF* and *MF* both are empty in this program in Figure 17(c) and 17(d). Finally, we need to input a target component, which is 3 in this example, as shown in Figure 18. The disassembly directions for 2D products are 4, which includes +x, -x, +y, and –y. The disassembly directions for 3D products are 6, which includes +x, -x, +y, –y, +z, -z.

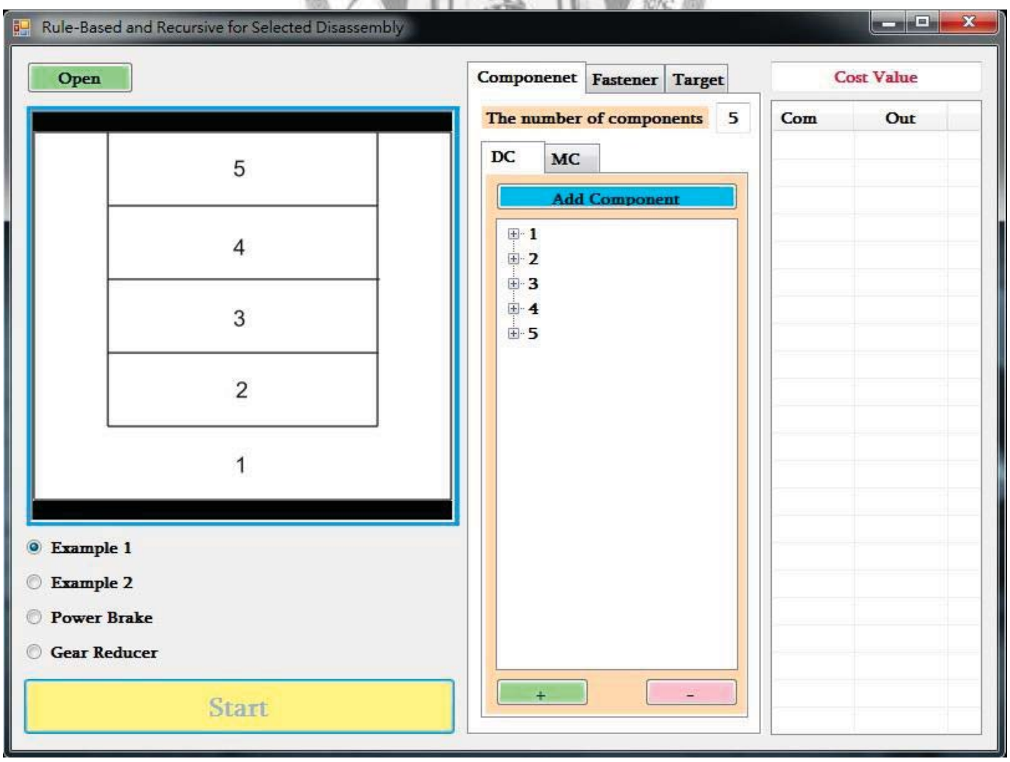
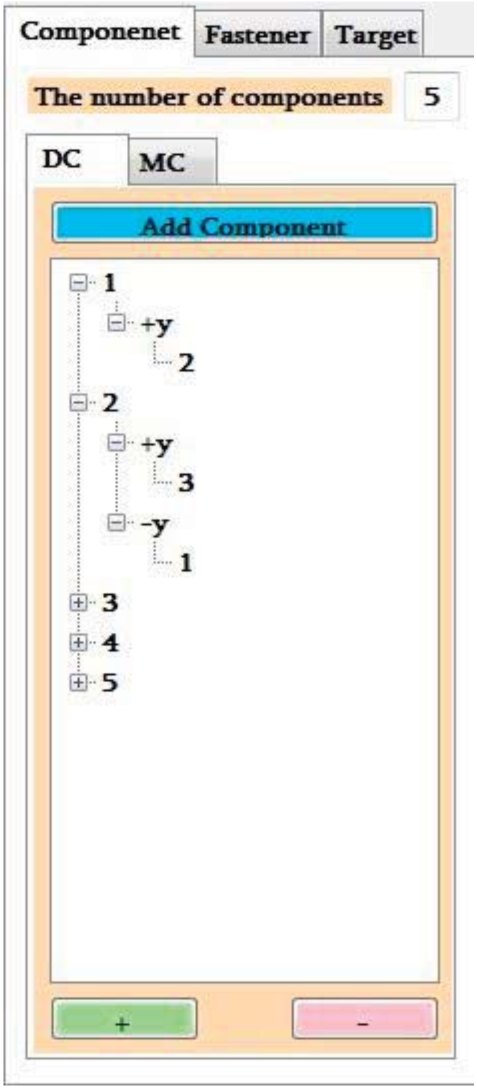
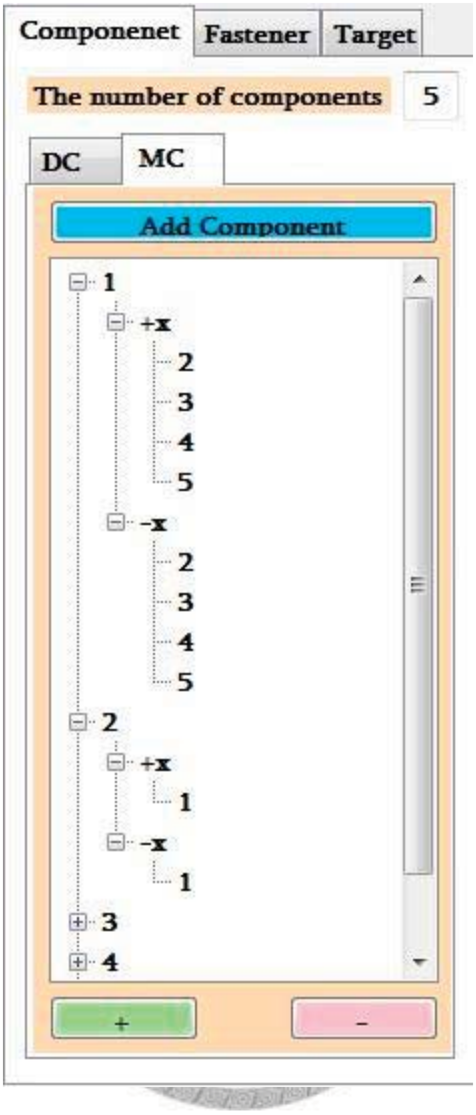


Figure 16. The graphical user interface.



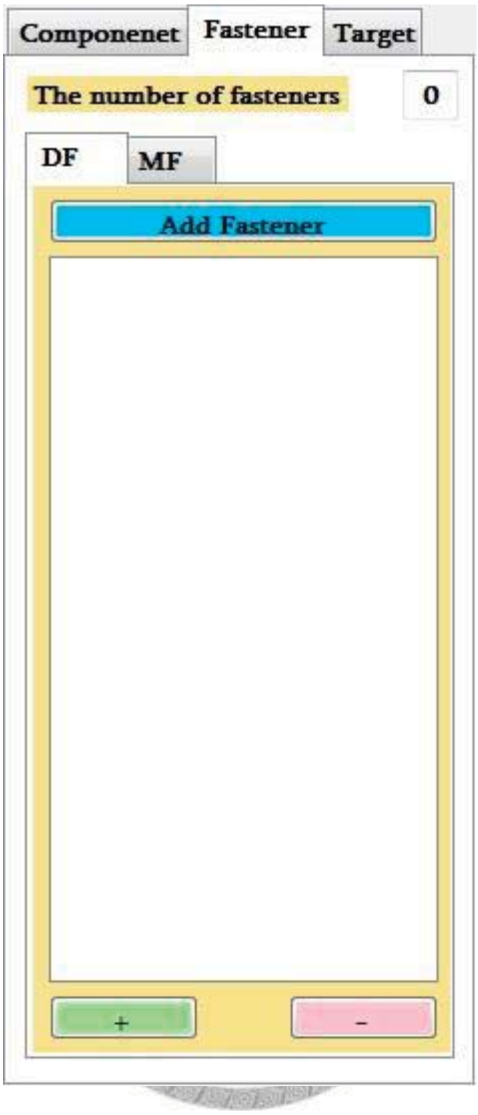
(a)

Figure 17. (*continued*)



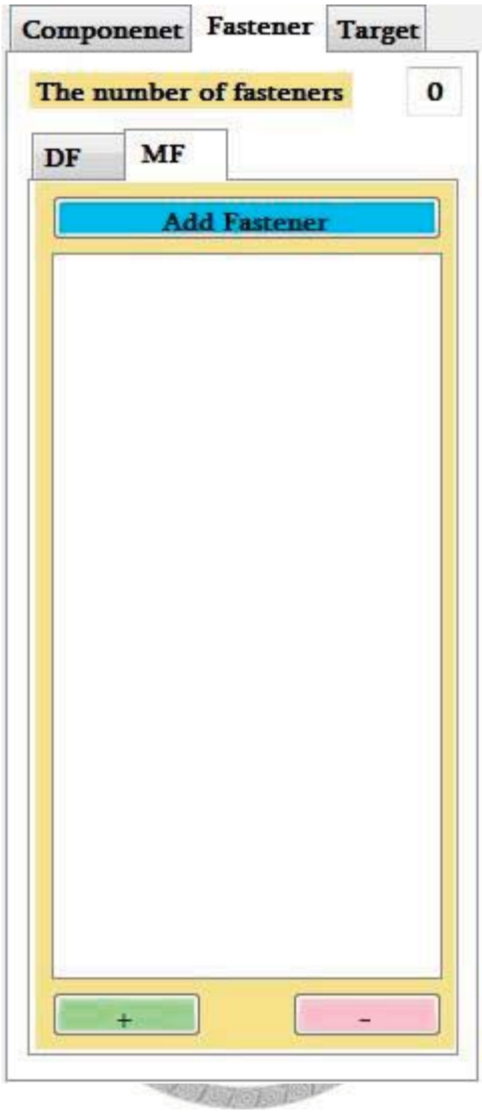
(b)

Figure 17. (*continued*)



(c)

Figure 17. (*continued*)



(d)

Figure 17. Input parameters for (a) *DC*, (b) *MC*, (c) *DF*, and (d) *MF* for example 1.

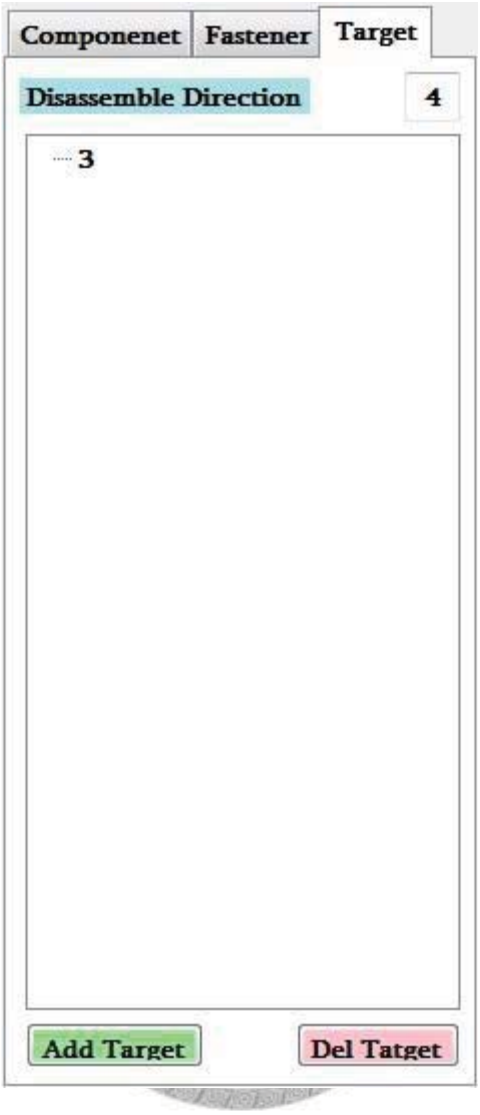


Figure 18. Input a target component.

After the program is executed, it finds two selective disassembly sequences: one is 5-4-3 in the +y direction; the other one is 1-2-3 in the –y direction, as shown in Figures 17 (a) and (b). The cost value of the two sequences both are 3. The two sequences are the optimal solutions for this example. Using the developed method to solve example 1 for selective disassembly sequence planning, it only creates two possible sequences. It eliminates other unrealistic or uncommon solutions. If an exhaustive method is used to generate all possible solutions, it will have 5! (5 x 4 x 3 x 2 x 1 = 120) different results. If other randomized method is used to solve the problem, e.g., GAs, it might create many uncommon and unrealistic solutions. Thus, our rule-based recursive method can greatly reduce the searching time and find a heuristic solution effectively.

(a)

Figure 19. (*continued*)

3	
Com	Out
1	-y
2	-y
3	-y

(b)

Figure 19. Two solutions: (a) *E*, *D*, and *C* (b) *A*, *B*, and *C*.

Chapter 7 Conclusions

In this thesis, we present a rule-based recursive selective disassembly sequence planning method. The method can be used to solve selective disassembly sequence planning problems with only simple geometric and topological information supplied by a user. The method is based upon six disassembly rules which are used to eliminate unrealistic and uncommon disassembly sequences to increase the credibility of the solutions. With the given rules, the searching process can effectively and efficiently find reasonable and near-optimal selective disassembly sequence solutions for complex disassembly problems. With the proposed rule-based recursive approach, users only need to supply information concerning the geometric constraints of a component with respect to its neighbouring components and fasteners. Compared to methods which consider geometric constraints between each pair of components, the developed method greatly reduces required information storage space and searching complexity. The method can solve both single-target component and multiple-target component selective disassembly sequence planning problems. Compared to most existing methods, our method is much easier to implement for general products.

In the future, method for defining and disassembling subassemblies needs to be investigated. Therefore, how to modulate parts become important. Sometimes disassembling of a target component from an assembly requires dividing the original assembly into two or more subassemblies. To reduce the time and cost, we just need to treat the subassembly which contains the target component as a target module (or a target integrated component) and to disassemble the module. Therefore, the concept of modules in selective disassembly planning is another important issue. The other issue is that engineers require a more credible database to set up the time or cost parameters.

References

[1] Aguinaga, I., Borro, D., and Matey, L., Automatic selective disassembly and path planning for the simulation of maintenance operations. IEEE Virtual Reality Conference 2007 March 10-14; 283-284.

[2] Aguinaga, I., Borro, D., and Matey, L., Parallel RRT-based path planning for selective disassembly planning. Int. J. Adv. Manuf. Technol. 2008; 36: 1221-1233.

[3] Boothroyd, G., Dewhurst, P., and Knight, W., Product design for manufacture and assembly second edition revised and expanded. Published by CRC Press Taylor & Francis Group. Boca RatonLondon New York, 2002; 96-103.

[4] Chen, S. F., Oliver, J. H., Chuo, S. Y., and Chen, L. L., Parallel disassembly by onion peeling. Transactions of The ASME, Journal of Mechanical Design 1997; 119: 267-274.

[5] Chung, C., and Peng, Q., An integrated approach to selective-disassembly sequence planning. Robotics and Computer Integrated Manufacturing 2005; 21: 475-485.

[6] Chung, C., and Peng, Q., A hybrid approach to selective-disassembly sequence planning for de-manufacturing and its implementation on the Internet. Int. J. Adv. Manuf. Technol. 2006; 30: 521-529.

[7] Chung, C., and Peng, Q., Sequence planning for selective disassembly in

de-manufacturing. International Journal of Computer Integrated Manufacturing 2006; 19: 3: 278-286.

[8] Directive 2002/95/EC of the European Parliament and of the Council of 27 January 2003 on the restriction of the use of certain hazardous substances in electrical and electronic equipment. Official Journal of the European Union, 2003.

[9] Directive 2002/96/EC of the Europoean Parliament and of the Council of 27 January 2003 on waste electrical and electronic equipment. Official Journal of the European Union, 2003.

[10] Directive 2005/32/EC of the European Parliament and of the Council of 6 July 2005. Official Journal of the European Union, 2005.

[11] García, M. A., Larré, A., López, B., and Oller, A., Reducing the complexity of geometric selective disassembly. Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems. 1474-1479.

[12] Giudice, F., and Fargione, G., Disassembly planning of mechanical systems for service and recovery: a genetic algorithms based approach. J. Intell. Manuf. 2007; 18:313–329.

[13] Kara, S., Pornprasitpol, P., and Kaebernick, H., A selective disassembly methodology for end-of-life products. Assembly Automation 2005; 25(2) 124–134.

[14] Kara, S., Pornprasitpol, P., and Kaebernick, H., Selective disassembly sequencing:

A methodology for the disassembly of end-of-life products. Annals of the CIRP 2006; 55(1): 37-40.

[15] Mascle, C., and Balasoiu, B. A., Algorithmic selection of a disassembly sequence of a component by a wave propagation method. Robotics and Computer Integrated Manufacturing 2003; 19: 439-448.

[16] Mascle, C., and Hong, P Z., Integrating environmental consciousness in product/process development based on life-cycle thinking. Int. J. Production Economics 2008; 112: 5-17.

[17] Pitipong, V., and Surendra M., G., Optimal analysis of lot-size balancing for multiproducts selective disassembly. International Journal of Flexible Automation and Integrated Manufacturing 1998; 6(3): 245-269.

[18] Shyamsundar, N., and Gadh, R., Selective disassembly of virtual prototypes. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics 1996; 4: 3159-3164.

[19] Srinivasan, H., and Gadh, R., Selective disassembly: Representation and comparative analysis of wave propagation abstractions in sequence planning. Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning Porto, Portugal, 1999; 129-134.

[20] Srinivasan, H., Figueroa, R., and Gadh, R., Selective disassembly for virtual

prototyping as applied to de-manufacturing. Robotics and Computer Integrated Manufacturing 1999; 15: 231-245.

[21] Srinivasan, H., and Gadh, R., Efficient geometric disassembly of multiple components from an assembly using wave propagation. Journal of Mechanical Design 2000; 122: 179-184.

[22] Srinivasan, H., and Gadh, R., A non-interfering selective disassembly sequence for components with geometric constraints. IIE Transactions 2002; 34: 349-361.

[23] Wang, J, F., Liu, J. H., Li S. Q., and Zhong, Y. F., Intelligent selective disassembly using the ant colony algorithm. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 2003; 17: 325–333.

[24] Wang, J. F., Li, S. Q., and Liu, J. H., Selective disassembly planning for product green manufacturing. Computer Integrated Manufacturing Systems 2007 June; 13(6): 1097-1102.

[25] Xue, J. F., Qiu, C. H., and Xiang, D., Planning of selective disassembly sequence based on ant colony optimization algorithm. Journal of Computer Aided Design & Computer Graphics 2007; 1119(16): 742-747.

[26] Yi, J., Yu, B., Du, L., Li, C., and Hu, D., Research on the selectable disassembly strategy of mechanical parts based on the generalized CAD model. Int. J. Adv. Manuf. Technol. 2008; 37: 599-604.

[27] Zhang, X. H., Li, S. Q., Wang, J. F., and Li, X., Single object selective disassembly sequence planning based on ant colony algorithm. Computer Integrated Manufacturing Systems 2007; 13(6): 1110-1114.